BRAINSTROKE DETECTION USING MEDICAL IMAGES

BY

CHEW XIN RU

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONOURS)

DIGITAL ECONOMY TECHNOLOGY

Faculty of Information and Communication Technology

(Kampar Campus)

FEBRUARY 2025

COPYRIGHT STATEMENT

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Puan Nur Lyana Shahfiqa Binti Albashah and my moderator, Mr Yew King Tak who have given me this bright opportunity to engage in a brain stroke detection project. It is my first step to establish a career in machine learning field. A million thanks to my supervisor and moderator.

To my parents and family, for their unwavering love, support, and continuous encouragement throughout the course. I must also extend my deepest gratitude to Yong Chung Wei, a very special person in my life, for his patience, unconditional support, and for standing by my side during hard times.

ABSTRACT

The main goal of the proposed project, "Brain Stroke Detection Using Medical Images," is to improve clot detection capabilities by analysing medical images, which will help patients and medical professionals as well. The goal of this project is to offer a complete solution that gives medical professionals the knowledge and resources they need to recognize brain strokes quickly and accurately. By combining machine learning algorithms with image processing techniques, the project introduces innovation and provides a fresh way to improve the diagnostic procedure. The technology application entails converting SWI-MRI scans from DICOM format to PNG. With a focus on feature extraction techniques, SVM the project seeks to provide a reliable brain clot detection system to assist medical professionals in their diagnostic pursuits. In order to enable automated brain clot detection and adapt to various image variations, machine learning algorithms are integrated for classification. A brain clot detection system is one of the project's outputs; it offers medical professionals automated image analysis, enabling prompt and precise diagnosis. The project aims to improve patient care by providing cutting-edge technology to healthcare professionals, enabling them to perform better neuroimaging diagnostics. This project contributes to improvements in patient outcomes and healthcare by putting cutting-edge technologies to use in the field of medical image analysis.

Area of Study (Minimum 1 and Maximum 2): Medical Image Processing, Deep Learning

Keywords (Minimum 5 and Maximum 10): Blood Clot Detection, Neurotechnology, AI Diagnostics, Medical Imaging, Machine Learning, Mobile Application

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

EEG	Electroencephalography
GRU	Gated Recurrent Unit
biLSTM	Bidirectional LSTM
LSTM	Long Short-Term Memory
FFNN	Feed-forward Neural Network
MRI	Magnetic Resonance Imaging
3D-DRNN	Three-Dimensional Deep Regression Neural Network
MWT	Microwave Tomography
MI	Microwave Imaging
FDFD	Frequency Domain Finite Difference
ADC	Average Dielectric Constant
AMM	Average Brain Configuration Map
RM	Representative Method
CT	Computed Tomography
DICOM	Digital Imaging and Communications in Medicine
FSL	Few-Shot Learning
ROC	Receiver Operating Characteristic

Chapter 1
Introduction

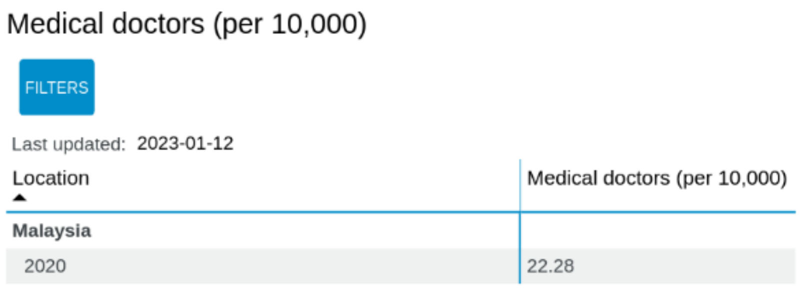
In this chapter, we present the background and motivation of the research, contributions to the field, and the outline of the thesis. We hope that this methodical approach will offer a thorough understanding of our work, its contributions, and its possible influence on the larger domains of neurology and medical imaging.

1.1 Problem Statement and Motivation

Brain Stroke, a crippling neurological condition, continues to be one of the world's leading causes of death and permanent disability. It is critical to diagnose strokes promptly and accurately to improve patient outcomes and implement effective interventions. Medical imaging is essential to the diagnosis process because it provides a non-invasive way to see brain structures and spot abnormalities linked to strokes. The use of artificial intelligence (AI) and sophisticated image analysis methods to improve the precision and effectiveness of clot detection has gained popularity in recent years.

The goal of the proposed study, "Brain Stroke Detection Using Medical Images," is to use artificial intelligence and advanced medical imaging technologies to address the pressing need for a more accurate and timely stroke diagnosis. Stroke early detection is critical because strokes represent a significant global health burden. The goal of this research is to further the ongoing efforts to transform the diagnosis of stroke, which will ultimately result in earlier interventions and better patient outcomes.

According to the World Health Organization (WHO), [1] stroke is a common cause of death. As the third most common cause of death in Malaysia, brain stroke or cerebrovascular disease resulted in 47,911 incident cases, 19,928 deaths, 443,995 prevalent cases, and 512,726 DALYs lost in 2019 [2]. Since 2006, there has been a **steady increase in the prevalence of risk factors** like diabetes, hyperlipidemia, and obesity, which has led to a higher incidence of stroke in people under 65. This is especially evident in men and women, whose rates have increased by 53.3% and 50.4%, respectively, from the 35–39 age group.[2] In 85% of cases, the loss of life and serious brain damage can be avoided if a stroke is identified or diagnosed early.



Medical doctors (per 10,000)

Last updated: 2023-01-12

Location	Medical doctors (per 10,000)
Malaysia	
2020	22.28

Figure 1.2 Medical doctors per 10,000 population in Malaysia

Particularly in Malaysia, one of the major issues in the area of brain stroke care is the **lack of access to specialized healthcare.** Based on data from the World Health Organization (WHO), [3] the availability of specialized doctors in Malaysia is notably constrained, with only 22.28 specialized doctors per 10,000 population. The lack of qualified medical personnel, such as neurologists and stroke care specialists, is a major obstacle to prompt and thorough stroke diagnosis and treatment. Access to neurologists and other stroke specialists is restricted in rural and underserved areas, where there is a particularly acute shortage of specialized healthcare providers. This geographic discrepancy makes it more difficult for people living in isolated areas to get timely and professional care in the event of an emergency.

Based on the Malaysian Journal of Medical Science, the results of the study showed that every participant (n=343) was able to identify at least one stroke risk factor. However, only 36.44% of respondents were able to properly identify all symptoms, indicating a lack of complete understanding about stroke symptoms. While most participants said they were aware with the term "stroke," less than half acknowledged that the proper course of action when stroke symptoms are noticed is to contact emergency medical services. These findings point to serious gaps in the general public's knowledge on how to recognize stroke symptoms and what to do in an emergency [4].

The aim of the thesis is to **improve brain stroke cases' early detection and treatment**. Better diagnostic tools are essential as strokes are one of the world's leading causes of death and disability. The use of medical imaging presents an appealing way of quickly and accurately identifying anomalies related to stroke. However, to advance the field of brain clot detection using medical images, dedicated research is required due to the existing challenges, which include the need for precise identification, differentiation of stroke types, and integration into clinical workflows.

The critical need for **immediate action** serves as the main driving force behind research on brain clot detection using medical imaging. The manual interpretation-based diagnostic techniques used today might not be as effective as they need to be for quick decision-making. By incorporating advanced medical imaging, healthcare providers can identify stroke indicators more quickly and take immediate action to save lives. Furthermore, the use of medical images offers an advanced technique for differentiating between different strokes, **allowing the generation of customized therapies,** and providing a **complete understanding of the extent and location of brain injury**.

In conclusion, the motivation is to transform stroke care by **increasing diagnosis speed, accuracy, and versatility.** This will improve patient outcomes, lessen the workload for medical professionals, and increase our comprehension of the complex dynamics related to this serious medical condition.

1.2 Objectives

- **Provide brain stroke - clot detection with SWI-MRI images and Few Shot models.**

  The creation of a specific model for brain clot detection that will analyze the SWI-MRI (Susceptibility weighted imaging- magnetic resonance imaging) images by using sophisticated classification models as input. The emphasis is on applying machine learning algorithms to correctly recognize and categorize various kinds of brain strokes from MRI scans. By seamlessly integrating modern image processing and classification technologies, the outcome seeks to offer a user-friendly interface that is easy to use for both healthcare professionals and users, enabling prompt and reliable brain clot detection.

- **Enhance Accessibility and Public Awareness of Stroke Diagnosis and Treatment**
  This project enhances preparedness by integrating a map displaying all hospitals within Malaysia with dedicated stroke centres, ensuring rapid access to specialized treatment. Furthermore, it features an educational resource centre providing users with knowledge on: stroke symptom recognition using the FAST acronym, understanding the rehabilitation journey with relevant exercises, and facilitating emotional recovery through stress management techniques and support contacts.

1.3 Project Scope and Direction

The scopes of the project include creates a sophisticated system that uses medical imaging to identify brain strokes early on, with a particular emphasis on addressing the challenges prevalent in Malaysia. The scope includes investigating cutting-edge technologies to improve the accuracy and effectiveness of stroke diagnosis, such as classifying SWI-MRI images. The project will take into account the barriers to specialized healthcare in Malaysia, particularly in underserved and rural areas where a lack of neurologists and stroke care specialists makes it difficult to diagnose and treat patients in a timely and competent manner. Additionally, the project will consider the integration of existing research findings for enhancing the diagnostic capabilities of this proposed system. This app will empower individuals with crucial information about stroke, including symptom recognition via the FAST method, and guide them to the nearest stroke-ready hospitals identified through an integrated map feature.

1.4 Contributions

This study advances the area of medical image processing in a number of important ways, most notably in the identification of brain strokes. First, it tackles the problem of sparse labeled data in medical datasets by applying few-shot learning algorithms to categorize SWI-MRI images into clot and no-clot categories. Furthermore, the study improves image preprocessing by integrating median and Gaussian filters, which raise the accuracy and reliability of stroke identification by enhancing the quality of MRI scans. In addition, the system's intuitive interface makes it possible for people to upload

MRI pictures and get real-time analysis, making it available to medical professionals and speeding up diagnosis. Besides, by integrating a thorough database of all stroke facilities in Malaysia, the Hospital Mapping System enables users to find the closest specialized treatment facility in an emergency. Lastly, evidence-based information about stroke symptoms, risk factors, preventative techniques, and available treatments is available at the Educational Resource Center.

1.5 Report Organization

This report is organized into 6 chapters: The Chapter 1 Introduction present the specifics of the study. Chapter 2 Literature Review, which covered a few relevant background topics. Chapter 3 then offers a system design, which discussing overall design of the system. Chapter 4 then goes on System Implementation and Testing of the project. In addition, Chapter 5 summarizes the project's System Outcome and Discussion. Lastly, Chapter 6 which is Conclusion of the project.

Chapter 2
Literature Review

2.1 Previous works on Brain Stroke Detection Input

2.1.1 EEG (Electroencephalography) Bioelectrical Signals with Time Series-based prediction module

Figure 2.1 EEG bioelectrical signals

The revised brain stroke detection model represents a pioneering approach to stroke prediction, utilizing EEG bioelectrical signals and advanced deep learning models [5] The goal of this methodology is to create a non-invasive, effective method for early stroke detection. This will enable medical professionals to identify patients who are at risk and start treatments on time to stop further brain damage. This strategy has the potential to save lives and reduce the impact of strokes on patients and society at large by utilizing deep learning techniques. This approach makes use of deep learning models (GRU, biLSTM, LSTM, and FFNN) for predictive analysis, smart sensing devices for data collection, and a graphical user interface for result presentation.

The methodology is an organized, step-by-step procedure that begins with the use of smart sensing devices to gather raw EEG data [6]. Subsequent to the data collection process, bioelectrical signals are obtained in a non-invasive and economical manner by means of sensors collecting EEG data. Data preprocessing comes next. After pre-processing the raw EEG data, features are taken out of it with a focus on the signals' frequency characteristics. Five different signal types are used to classify EEG signals, including the alpha, beta, gamma, theta, and delta. It is very important to use these categories of the signals to detect normal people and stroke affected people. [7]

Next, using the processed EEG data, four distinct deep learning models are applied to predict strokes: LSTM, biLSTM, GRU, and FFNN. Subsequently, the models undergo training using offline data and validation using online data to guarantee precise forecasts of early stroke detection. Ultimately, the test findings demonstrate that while all deep learning models are effective at predicting strokes based on biosignals, GRU and biLSTM perform better than FFNN and traditional LSTM in terms of accuracy and prediction error rates.

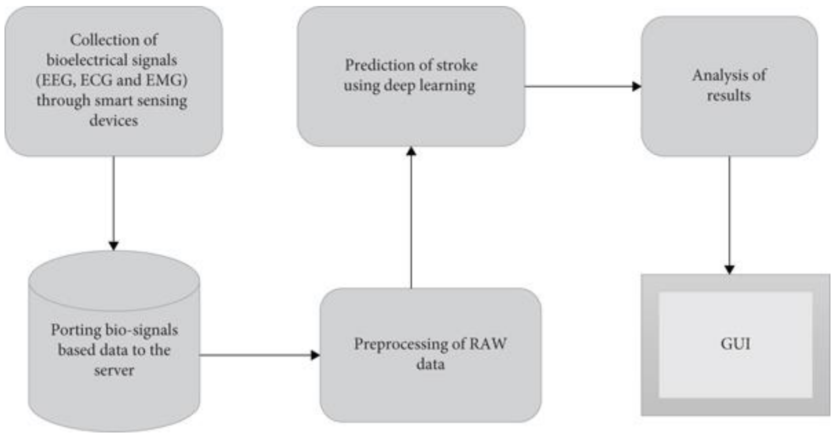

Collection of bioelectrical signals (EEG, ECG and EMG) through smart sensing devices
Prediction of stroke using deep learning
Analysis of results
Porting bio-signals based data to the server
Preprocessing of RAW data
GUI

Figure 2.2 Architecture diagram of proposed methodology

Strengths

One significant advantage of the suggested methodology is the use of EEG signals as a **nonsurgical method for stroke prediction**. This method is more affordable and accessible since it avoids the problems related to expensive and radiation-exposing imaging techniques [8]. The suggested methodology, which makes use of EEG signals, presents a viable noninvasive method for the early diagnosis of strokes. This method has the potential to save lives by greatly increasing the accessibility of stroke detection.

Weaknesses

One of the limitations of the proposed methodology is the **assumption of linearity in EEG signals.** The deep learning models—such as GRU, biLSTM, LSTM, and FFNN—are efficient tools for identifying complex patterns, but they are predicated on the idea that there is a linear relationship between the input (EEG signals) and the output (stroke predictions). The complexities of EEG signals, which can display non-linear patterns and relationships, may be beyond the scope of this assumption [9].

EEG signals frequently involve complex interactions and nonlinear dynamics because they are a reflection of the complex electrical activity that occurs within the brain. The complexity of EEG signals is influenced by neural processes, interactions between neurons, and other neurophysiological phenomena. Therefore, it could be oversimplifying the underlying mechanisms to assume a linear relationship between these signals and the frequency of strokes. By the resolution of the limitation, this method can be enhanced to raise the standard of brain stroke detection through the use of deep learning and medical imaging.

2.1.2 MRI (Magnetic Resonance Imaging) with Deep Learning

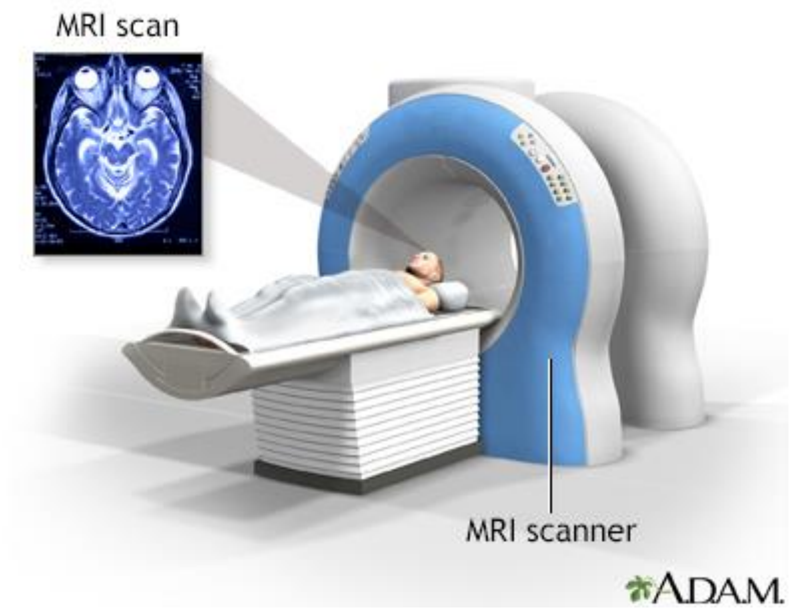


Figure 2.3 Magnetic Resonance Imaging

The second reviewed brain stroke detection methodology involves the integration of advanced imaging modalities, Magnetic Resonance Imaging (MRI), and cutting-edge deep learning techniques. The advanced MR modality-based methods for brain stroke detection with integration of MRI and deep learning primarily prioritize on MR perfusion imaging [10], an advanced MRI modality that has many advantages in stroke detection. With these techniques, the effectiveness of automated computer-aided systems for stroke diagnosis is intended to be increased.

One noteworthy example of an advanced MR modality-based technique is the three-dimensional residual framework, or BrainSegNet, created by Hu and colleagues [11].

The purpose of this framework is to automatically segment lesions from perfusion magnetic resonance images. The technique makes use of the perfusion MR images' whole-brain coverage and their easier post-processing, which can together carry out diffusion imaging.

The research of To et al., who developed a three-dimensional Deep Regression Neural Network (3D-DRNN) for efficient segmentation of lesion regions in the brain, is another cutting-edge MR modality-based technique [12] . For semantic segmentation, the 3D-DRNN employs an encoder-decoder framework. The encoder uses high-dimensional feature representation, and the decoder reconstructs the output so that it matches the ground truth for validation.

A Deep CNN network, consisting of multiple convolutional layers, fully-connected layers, pooling layers, and a Softmax classifier, was proposed by Liu et al. based on advanced MRI modality [13]. Using only the original pre-treatment perfusion images, this model automatically picks up centralized imaging features.

In short, the use of MR perfusion imaging and other advanced MRI modalities is the main focus of advanced MR modality-based techniques for brain stroke detection using MRI and deep learning. These approaches increase the effectiveness and precision of automated systems for stroke diagnosis by utilizing a variety of deep learning architectures and techniques.

Strength

Whole-brain coverage and simultaneous diffusion imaging are two benefits of MR perfusion imaging. This all-inclusive method gives deep learning models access to a rich dataset, improving their capacity to accurately identify and categorize stroke lesions.

Also, the diagnostic workflow is streamlined by the ease of post-processing, which includes the quick creation of perfusion maps. These processed maps can be used by deep learning models to efficiently and automatically diagnose strokes, which will expedite clinical decision-making.

Weakness

Many healthcare facilities find MRI equipment to be financially burdensome due to the **high initial costs of acquisition and maintenance**, as well as the requirement for **specialized personnel**. This limits the availability of MRI-based diagnostics, making them available only in a small number of healthcare facilities. The lack of MRI machines in underdeveloped or underdeveloped regions of the world keeps a large portion of the population from taking advantage of this cutting-edge diagnostic tool [14]. Significant resources are needed for MRI machine upkeep and operation. This entails the requirement for specialized facilities, frequent equipment maintenance, and technician training. MRI requires a lot of resources, which can be problematic in healthcare settings where resources are already limited.

2.1.3 MI (Microwave Imaging) using prior clinical database

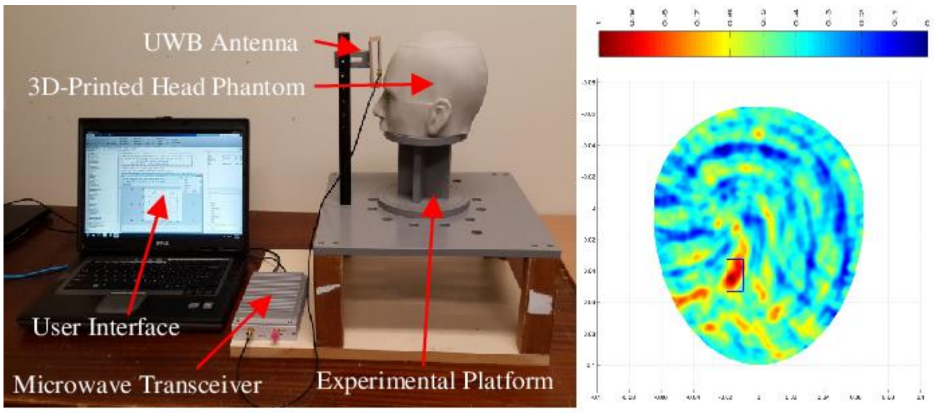


Figure 2.4 Microwave Imaging

The third reviewed brain stroke detection makes use of microwave imaging (MI), more especially microwave tomography (MWT) [15]. Tissues are imaged using microwave imaging according to variations in their dielectric characteristics. The suggested method uses a multifrequency approach along with Maxwell's equations to implement a Frequency-Domain Finite-Difference (FDFD) method. [16] This methodology employs a multifrequency approach and two well-known regularization-like strategies to reconstruct the brain image: the use of prior information, such as brain structure data from an MRI database and known dielectric properties of brain tissues, and the level set technique for shape reconstruction.

The purpose of the numerical experiments was to validate a stroke detection technique based on microwaves. To recreate real-world conditions, a circular array of antennas emitted microwaves (0.3–1 Hz) in a 1 m² domain. Average Dielectric Constant (ADC), Average Brain Configuration Map (AMM), and Representative Method (RM) were the three initial guess strategies used. Using a database of 100 MRI images, AMM and RM showed faster and more accurate stroke detection than ADC. For computational efficiency, the study focused on a 2D reconstruction model and used a stopping criterion based on the variance of residuals over the last three iterations. The method's potential for quick, noninvasive stroke diagnosis was shown by the results, and AMM and RM provided useful advantages over conventional methods.

Strength

Rapid stroke detection is made possible by AMM and RM initial guesses, which is crucial for prompt clinical diagnosis. Besides, this model can identify the accurate location of strokes, even in cases where lesions are positioned deeply.

Weakness

It's conceivable that the **stroke sizes cannot be accurately estimated** using this stroke detection methodology. This restriction may result from a number of things, including the **imaging technique's resolution** or difficulties telling apart particular tissue properties. As a result, the accuracy of the size approximation becomes a significant factor in the validity of the diagnostic results. There may be associated costs for microwave imaging, particularly if it involves more recent and specialized technologies. **These costs will vary based on the equipment that is used** and the particular medical facility that offers the service.

2.1.4 Few Shot Model

Machine learning has a subfield called few-shot learning. When there are limited training samples with supervised data, it entails classifying new data. Even with little training samples, a computer vision model can achieve reasonable performance [17].

In a simpler examples, consider kid discovering colour. The kid is first shown hundreds of red objects, then hundreds of blue objects, and so on in traditional guided learning.

With few-shot inference, a kid learns by seeing only a few number of red and blue objects, but with practice, they are able to accurately identify other red and blue objects [18].

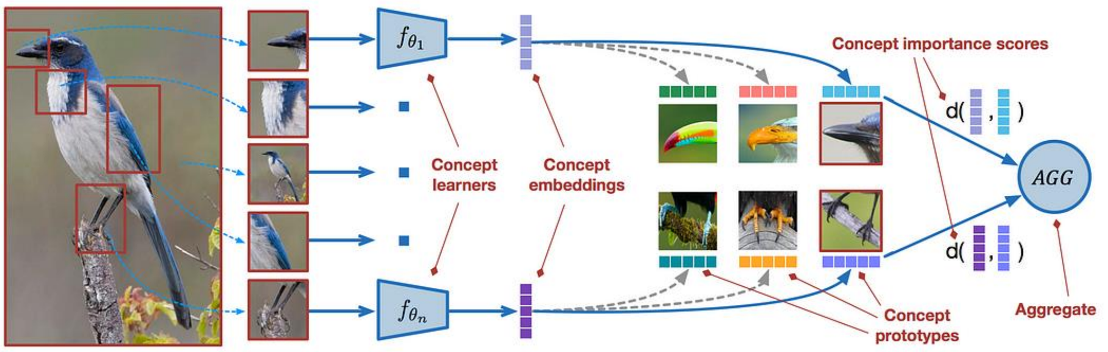


Figure 2.5 Concepts of Few Shot Model

From the figure above, COMET uses independent concept learners to learn concept embeddings along each concept dimension, then compares the results to concept prototypes. After that, COMET efficiently compiles data from all idea dimensions and gives each dimension a concept relevance score [19].

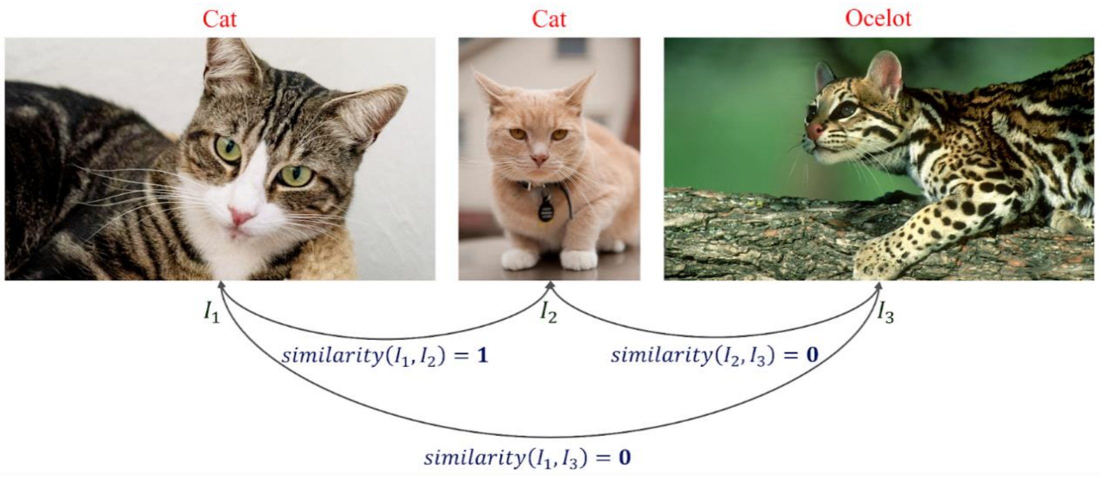

Cat
Cat
Ocelot
I_1
I_2
I_3
$similarity(I_1, I_2) = \mathbf{1}$
$similarity(I_2, I_3) = \mathbf{0}$
$similarity(I_1, I_3) = \mathbf{0}$

Figure 2.6 How does Few-Shot Learning Work?

In conventional few-shot systems, learning a similarity function capable of mapping the similarities between the classes in the query and support sets is the main objective. Similarity functions return a probability value for the similarity [20].

For instance, comparing the two cat photos in the image above should result in a perfect similarity function with a value of 1.0. The similarity output for the other two scenarios, in which cat images are contrasted with ocelot images, ought to be 0.0. Still, this is the perfect situation. In actuality, the numbers may be 0.95 for I1 and I2, and for the other two situations (such as 0.02 and 0.03), a minor amount greater than 0.

Since few-shot learning (FSL) can quickly adjust models to uncommon and unexplored data classes, it is especially well-suited for the medical domain. Obtaining massive volumes of labeled data is a big difficulty in the healthcare industry, particularly for sophisticated medical imaging activities like echocardiography or MRIs, or unusual illnesses. These illnesses are frequently uncommon, and making accurate annotations of medical photographs requires specialized knowledge, which makes matters more challenging. Therefore, compiling a large dataset of labeled samples can be unnecessarily difficult.

By allowing models to generalize from a small number of labeled samples, FSL solves this problem. This capacity is especially important in the medical domain, where large labeled datasets are hard to come by due to the rarity of some illnesses. Through the use of a pre-trained model and limited data particular to the new illness, FSL is able to learn to recognize and diagnose uncommon diseases or anomalies. This method not only lessens the requirement for extensive data collecting, but it also expedites the creation of decision support systems and diagnostic tools for the medical field. As a result, FSL is a useful tool for improving patient care and medical technology by providing a workable answer to some of the most important problems with diagnostic accuracy and medical data processing [21].

Strengths

Few-shot learning (FSL) aims to generalize from a small number of labeled samples, therefore it **does not require a lot of expensive labeled data**. Through the use of a pre-trained model—one that has been trained on large datasets like ImageNet—FSL saves a significant amount of computational resources by **avoiding the need to start from scratch**. With this strategy, models can be trained to learn about rare or limited categories of data, such newly found or endangered species, even in situations where the amount of available data is limited. Furthermore, if the data in the support and query sets are coherent, **FSL models can adapt to new data domains even if the pre-training data distribution varies** dramatically [21].

Weakness

Despite the fact that FSL is made to function with very little labeled data, there **may be difficulties in obtaining high accuracy due to the lack of adequate examples**. As presented with complicated or highly variable data, the model might find it difficult to generalize, which would lead to poorer performance as compared to models trained on bigger datasets.

2.2 Comparison between existing model and proposed model

Proposed Method: **MRI Scan Images with Few-Shot Model**

	EEG with time series-based module	**MRI with Deep Learning**	**MI with prior clinical database**	**Proposed Method**
Noninvasive Method	**Yes**	**Yes**	**Yes**	**Yes**
Whole Brain Coverage	**Yes**	**Yes**	**Yes**	**Yes**
Cost	**Affordable**	**Moderate**	**Vary Based on Equipment Used**	**Low**
Accessibility	**Yes**	**No**	**No**	**Yes**
Able to Predict and Detect Brain Clot	**Yes**	**Yes**	**Yes**	**Yes**

Table 2.1 Comparison between existing model and proposed model

Table 2.1 show that the comparison between existing and proposed methodology. The existing methodology is EEG with time series-based module, MRI with deep learning, and MI with prior clinical database.

All of the techniques are non-invasive, cover the entire brain, and show promise in the identification and prediction of brain strokes. Still, there are some significant differences in terms of price, usability, and extra diagnostic features.

On the other hand, the proposed method offers whole brain coverage, non-invasiveness, affordability in comparison to others, and improved accessibility.

In conclusion, the proposed method which is SWI-MRI images with Few-Shots Learning For the machine learning has been used in this project. For **clot detection**, dataset of MRI pictures which is ischemic stroke type, with a particular emphasis on pictures that show clots either present or not was gathered. Prototypical Networks and Matching Networks are two examples of few-shot learning models that function well for classification problems with little data. These are the models used. Based on the support set, the model architecture was created to learn a prototype, or representative feature, for each class ("clot" and "no clot").

The model was trained on a variety of tasks that required it to use the limited number of labeled samples in the support set to distinguish between clots and no clots. Several episodes were used in the training procedure, each of which replicated the testing circumstances to help the model adjust and become more proficient at classifying new photos with little information.

2.3 System Review

2.3.1 Brainomix 360 Mobile

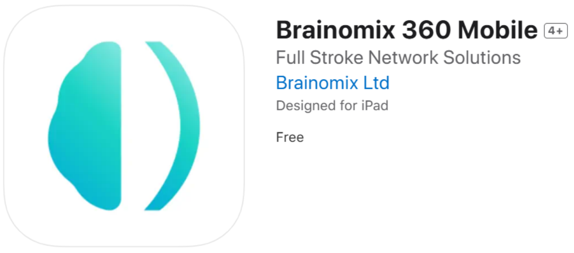


Figure 2.7 Brainomix 360 Mobile

The Brainomix 360 Mobile app is a sophisticated tool designed to empower physicians with quick and easy access to critical results. Its primary focus is on pseudonymized scan results, which strengthen the connection between clinicians and facilitate faster treatment decisions. Utilizing state-of-the-art AI algorithms, Brainomix 360 Mobile provides real-time interpretation of brain scans, guiding treatment and transfer decisions for stroke patients. The app aims to ensure that more patients receive the right treatment at the right place and time. It consists of three modules – e-ASPECTS, e-CTA, and e-CTP – that cover the full range of stroke imaging needs, from simple NCCT scans to advanced CT Perfusion assessments [22].

Strengths

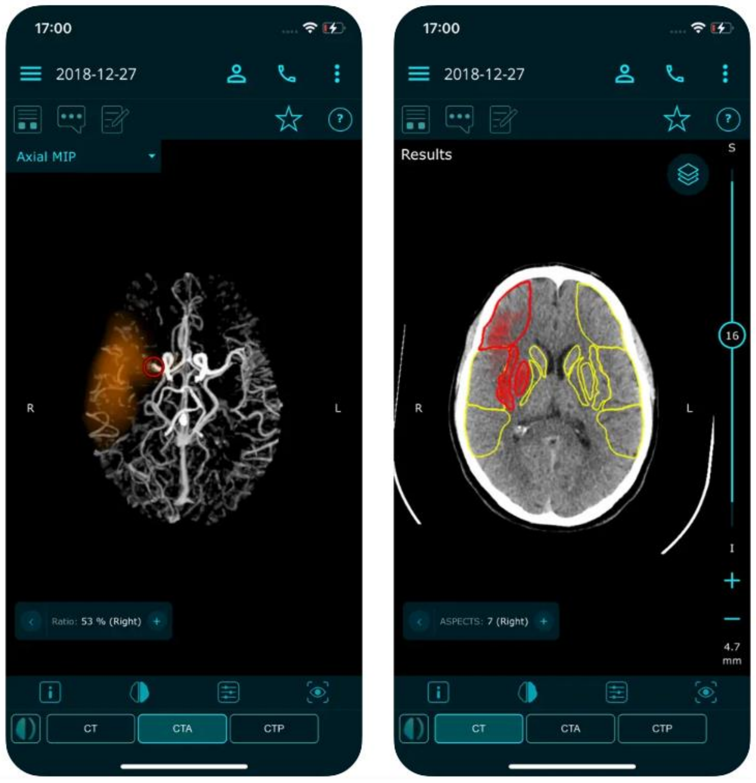


Figure 2.8 Screenshot of Brainomix 360 Mobile Internal Function

First, the app provides **AI-Powered Real-Time Interpretation**. The app employs advanced AI algorithms to analyze brain scans in real-time, providing reliable and timely results to aid clinical decision-making. Besides, it has a **comprehensive imaging support**. With its three modules (e-ASPECTS, e-CTA, and e-CTP), Brainomix 360 Mobile supports a wide range of stroke imaging needs, catering to both basic and advanced diagnostic requirements.

Moreover, the app **enhanced collaboration**. By sharing pseudonymized scan results across a network of clinicians, the app fosters better communication and collaboration, which can expedite treatment decisions.

Weaknesses

First, **limited diagnostic functionality**. The app is explicitly stated to be for informational purposes only and cannot be used for diagnostic decision-making. Physicians must rely on PACS or approved radiology displays for formal diagnostics.

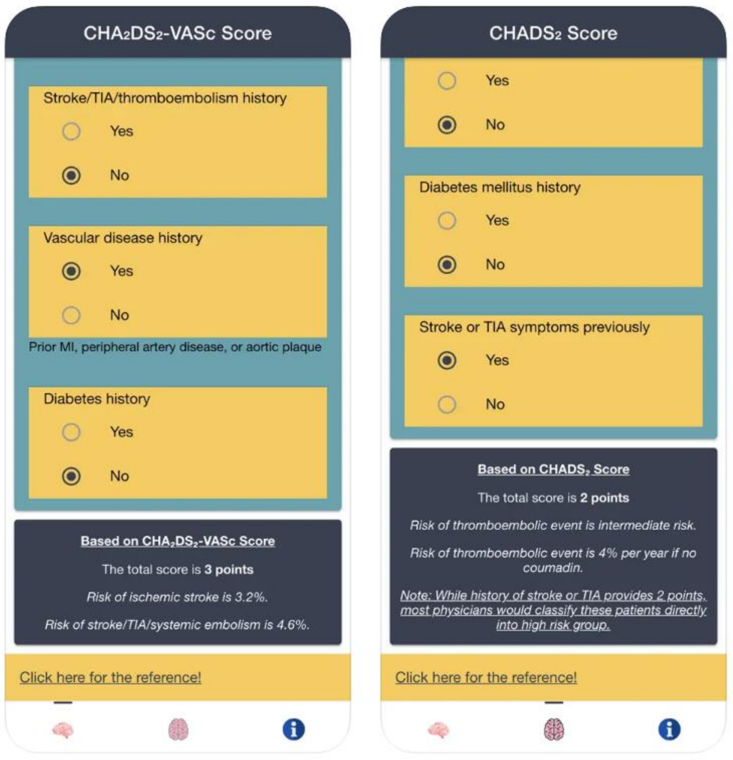
Also, **regional availability**. Some products and features of Brainomix 360 Mobile are not available in all regions, which may limit its accessibility and adoption globally. Besides, there is a weakness in **training and usability**. While AI-powered tools can enhance decision-making, there may be a learning curve for clinicians unfamiliar with the app's interface or specific functionalities.

2.3.2 AF Stroke Risk

Figure 2.9 AF Stroke Risk

AF Stroke Risk is a mobile application specifically designed to assist medical practitioners in predicting the risk of stroke, transient ischemic attack (TIA), or embolism in patients with atrial fibrillation (AF). Utilizing well-established clinical scoring systems, such as CHA_2DS_2-VASc and $CHADS_2$, the app provides a straightforward and highly accessible approach to evaluating a patient's risk for thromboembolic events. By offering features like precise calculations and a user-friendly interface, the app aims to improve stroke risk stratification and support decision-making regarding anticoagulant or antiplatelet therapy. However, it emphasizes that all calculations should be re-checked and not replace clinical judgment [23].

Strengths



CHA$_2$DS$_2$-VASc Score

Stroke/TIA/thromboembolism history

Yes [✓ No]

Vascular disease history

[✓ Yes] No

Prior MI, peripheral artery disease, or aortic plaque

Diabetes history

Yes [✓ No]

Based on CHA$_2$DS$_2$-VASc Score

The total score is **3 points**

Risk of ischemic stroke is 3.2%.

Risk of stroke/TIA/systemic embolism is 4.6%.

Click here for the reference!

CHADS$_2$ Score

Yes [✓ No]

Diabetes mellitus history

Yes [✓ No]

Stroke or TIA symptoms previously

[✓ Yes] No

Based on CHADS$_2$ Score

The total score is **2 points**

Risk of thromboembolic event is intermediate risk.

Risk of thromboembolic event is 4% per year if no coumadin.

Note: While history of stroke or TIA provides 2 points, most physicians would classify these patients directly into high risk group.

Click here for the reference!

Figure 2.10 Screenshot of AF Stroke Risk Internal Function

First, the app is **easy to use**. The app is simple and highly intuitive, making it accessible for medical practitioners with varying levels of technical expertise. Besides, the app has a **scientific basis**. Its reliance on validated scoring systems (CHA$_2$DS$_2$-VASc and CHADS$_2$) ensures a robust and evidence-based approach to risk prediction. It also has a **comprehensive feature**. The app not only calculates stroke risk but also evaluates risks for other complications, such as transient ischemic attack and embolism.

Also, it has an **educational value**. It provides detailed explanations and references for the scoring systems, which can serve as a useful educational tool for practitioners and trainees.

Weaknesses

First, it **depends on clinical judgment**. The app explicitly states that its calculations should not substitute for clinical expertise or local guidelines, limiting its standalone utility. Next, **potential for misuse**. Without adequate training, there is a risk of misuse or over-reliance on the app, which could impact patient outcomes.

Also, **lack of diagnostic integration**. Although it calculates risk, it does not integrate with diagnostic tools or patient records, reducing its functionality as part of a broader clinical workflow. Next, **limited scope**. The app focuses solely on risk estimation and does not provide detailed guidance on subsequent management or treatment options.

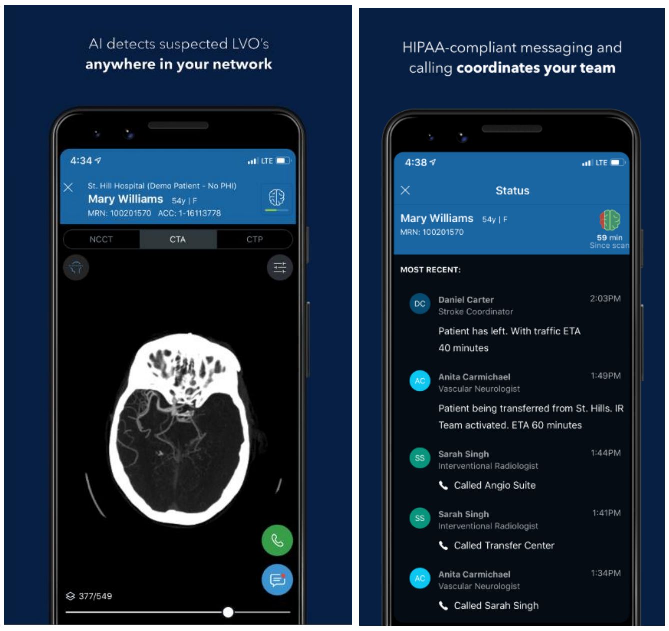
2.3.3 Viz Training

Figure 2.11 Viz Training

Viz Training is a mobile application designed to streamline stroke workflows in hospitals by leveraging artificial intelligence (AI). The app automatically detects suspected strokes on brain imaging and triages cases directly to a clinician's phone within minutes. Its goal is to reduce time to physician notification, improve access to care, and enhance overall healthcare delivery.

Viz claims to alert neurovascular specialists to suspected large vessel occlusion (LVO) strokes earlier than standard care in 95% of true positive cases, with an average time-saving of 52 minutes. Beyond notifications, the app allows users to securely view patient imaging, access clinical history, communicate with referral hospitals, and send HIPAA-compliant texts, making it a comprehensive tool for stroke care [24].

Strengths


AI detects suspected LVO's
anywhere in your network
4:34
St. Hill Hospital (Demo Patient - No PHI)
Mary Williams 54y | F
MRN: 100201570 ACC: 1-16113778
NCCT
CTA
CTP
377/549
HIPAA-compliant messaging and
calling coordinates your team
4:38
Status
Mary Williams 54y | F
MRN: 100201570
59 min
Since scan
MOST RECENT:

DC Daniel Carter, Stroke Coordinator, 2:03PM
Patient has left. With traffic ETA 40 minutes

AC Anita Carmichael, Vascular Neurologist, 1:49PM
Patient being transferred from St. Hills. IR Team activated. ETA 60 minutes

SS Sarah Singh, Interventional Radiologist, 1:44PM
Called Angio Suite

SS Sarah Singh, Interventional Radiologist, 1:41PM
Called Transfer Center

AC Anita Carmichael, Vascular Neurologist, 1:34PM
Called Sarah Singh

Figure 2.12 Screenshot of Viz Training Internal Function

First, **AI-Powered stroke detection**. The use of artificial intelligence enables the app to automatically detect suspected strokes on brain imaging, significantly improving the speed and accuracy of diagnosis. Next, **time efficiency**. The app reduces the time to physician notification by saving an average of 52 minutes in suspected LVO stroke cases, which is critical in stroke treatment.

Besides, **comprehensive workflow integration**. Viz Training supports a range of functionalities, including secure imaging review, clinical history access, and communication with referral hospitals, consolidating multiple stroke care processes into a single platform.

Weaknesses

First, **non-medical device limitation**. As the app is intended for informational purposes only and is not a medical device, it cannot be relied on for diagnostic or treatment decisions. Next weakness is **learning curve**. Adopting the app may require training for clinicians unfamiliar with its features or workflow integration.

2.4 Comparison between existing system and proposed system

	Brainomix 360 Mobile	AF Stroke Risk	Viz Training	StrokeSense (Proposed System)
AI Powered	Yes	No	Yes	Yes
Stroke Imaging Support	Yes	No	Yes	Yes
Workflow Integration	No	No	Yes	Yes
Accessibility	Yes	No	No	Yes
Able to Predict and Detect Brain Clot	Yes	No	No	Yes
Primary Purpose	**Real-time Ai-based interpret brain scan for stroke patients.**	**Stroke risk prediction for atrial fibrillation patient.**	**Ai-power stroke workflow synchronization and triage.**	**Comprehensive stroke care app combining clot detection, and workflow synchronization**

Table 2.2 Comparison between existing system and proposed system

The comparison presented in Table 2.2 highlights key feature differences between existing systems, including Brainomix 360 Mobile, AF Stroke Risk, Viz Training, and the proposed system, StrokeSense.

Brainomix 360 Mobile leverages artificial intelligence (AI) for real-time brain scan interpretation, specifically for stroke patients. It supports stroke imaging and is accessible in certain regions, but it lacks workflow integration and is limited to predictive capabilities for brain scans rather than broader clot detection. AF Stroke Risk, on the other hand, focuses on stroke risk prediction for atrial fibrillation patients using scoring systems like CHA_2DS_2-VASc and $CHADS_2$. However, it does not incorporate AI, imaging support, or workflow integration, and it is not accessible globally.

Viz Training utilizes AI to streamline stroke workflow synchronization and triage, providing time-saving features and enhancing hospital communication. It also supports stroke imaging but lacks broader accessibility and the ability to predict and detect brain clots.

In contrast, the proposed system, StrokeSense, combines the strengths of all these systems and introduces unique features. It integrates AI for advanced functionalities, including imaging support and workflow synchronization. StrokeSense is globally accessible for the clot prediction function, however the hospital mapping function is currently Malaysia ready only.

Chapter 3
System Methodology

3.1 Methodology

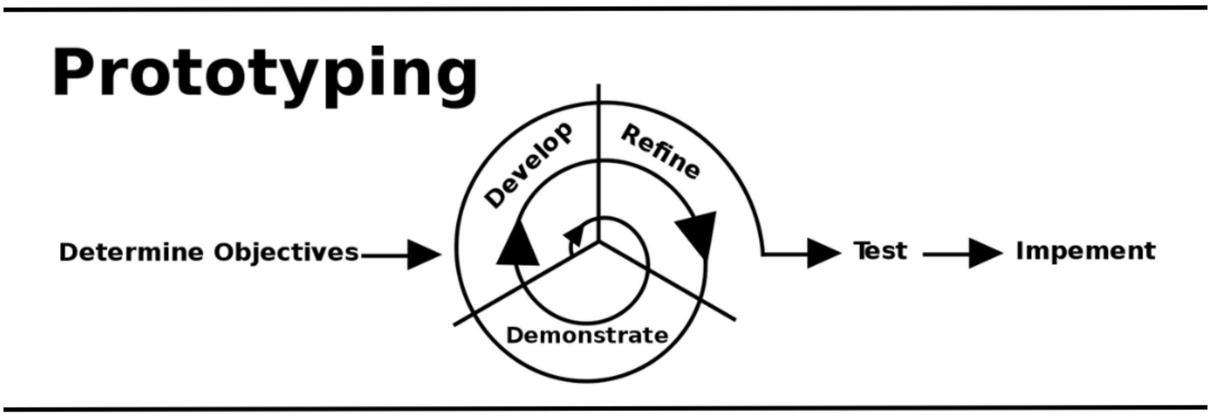


Figure 3.1 Prototyping Methodology

Introduction to Prototyping Methodology

This chapter outlines the methodological approach employed in the development of the brain stroke detection application. The project adopts a prototyping methodology, which allows for iterative development and continuous refinement based on feedback. This approach is particularly suitable for medical applications where user interaction and accuracy are paramount. This methodology allows for incremental development, enabling early feedback and iterative improvements to align the app with the needs of users and healthcare professionals.

3.2 Prototyping Approach

The development of the blood clot detection application followed a structured prototyping methodology. As defined by David Walker, Ph.D., Executive Director of Learning Technologies at McMaster University, this approach involves refining earlier versions, with the aim of convergence on the desired product [25]. This evolutionary design methodology was selected due to its suitability for complex systems that require significant user feedback and technical refinement.

Unlike the waterfall model, which completes each phase sequentially and makes modifications difficult after the concept stage, the prototyping approach allowed for continuous improvements and adaptations throughout the development process. This was crucial for incorporating medical expertise and user feedback into the application design.

The app development follows an evolutionary prototyping model, where each iteration builds upon earlier versions. This approach ensures continuous refinement based on user feedback and technical evaluations. The steps involved include:

1. **Define Objectives**

   Establish clear goals for the app, including early clot detection, accessibility, and usability for non-specialist healthcare providers.

2. **Develop Initial Prototype**

   Create a basic version of the app with limited functionality, focusing on core features such as MRI image preprocessing and analysis, map feature to locate nearby stroke-ready hospitals and educational resources on stroke symptoms and recovery.

3. **Refine Through Feedback**

   Test the prototype with mri images and get feedback from users to identify area of improvements.

4. **Demonstrate and Test**

   Validate the prototype's performance in detecting stroke indicators using real-world medical images and scenarios.

5. **Iterate and Expand**

   Incorporate additional features, refine existing functionality, and address user concerns in subsequent iterations.

3.3 Development Process

First step of development process is **Requirement Analysis**. Define the key requirements of the app, such as accuracy in clot detection, ease of use, and integration with existing clinical workflows. Besides, identify the target user base, including healthcare providers in underserved areas and patients seeking educational resources.

Secondly, **Design and Prototyping**. From the paper prototyping which including sketch the initial design concepts, including the user interface (UI) and user experience (UX) to the digital prototyping which translate paper designs into digital mockups using prototyping tools. Functional prototypes also been included which is develop a working prototype with key features, including AI-based clot detection, google maps feature and educational modules.

Third step goes to **Development.** Prototype is implemented using agile sprints. Each sprint focuses on a specific set of features, such as image preprocessing algorithms, map integration, or educational content delivery. Also, integration of TensorFlow Lite for AI model inference also been done for ensuring compatibility with mobile platforms.

Then, it comes with **Testing and Validation.** Conduct extensive testing with real-world medical images to validate the accuracy and efficiency of the clot detection algorithms. Usability testing also been performed with users to ensure the app is intuitive and meets user expectations.

Finally, **Deployment and Maintenance**. Deploy the app using android studio on mobile emulator. Also, continuously monitor app performance and update features based on user feedback and advancements in medical imaging technology.

3.4 Key Features of the Prototype

The StrokeSense app leverages advanced machine learning algorithms trained on extensive medical imaging datasets to **identify stroke** indicators quickly and accurately. Designed with mobile optimization in mind, the model is converted into TensorFlow Lite format, allowing for smooth real-time inference directly on users' devices. This ensures immediate analysis of SWI-MRI scans or images, offering users faster insights.

To further support users in urgent situations, the app features a **built-in map that displays the stroke-ready hospitals**. This tool not only shows directions but also provides essential contact information, allowing users to connect with specialized healthcare providers without delay. This feature is especially valuable for individuals in underserved or rural areas who may not have easy access to stroke-specific medical care.

Beyond detection and emergency support, the app also serves as an educational platform. Its knowledge hub offers **comprehensive resources on stroke awareness**, including how to recognize symptoms using the FAST method, understand recovery protocols, and manage emotional well-being. To make learning more engaging, interactive elements such as quizzes and videos are included, helping users retain important information while encouraging proactive health practices.

Advantages of the Prototyping Methodology

Prototyping plays a crucial role in app development by enabling **early error detection**, allowing developers to identify and resolve issues before they become costly or time-consuming. By creating tangible models of the app, prototypes also **improve communication** with target users, ensuring everyone shares a common understanding of the app's intended functionality and features. Moreover, the iterative nature of prototyping supports a **user-centered design approach**, as continuous feedback from users helps shape the app into a product that truly aligns with their needs and expectations.

Challenges and Mitigation Strategies

One of the main challenges in developing an AI-powered mobile application using iterative methods is the **increased complexity** that comes with **continuous refinement and feature addition**. As new elements are introduced in each iteration, managing code dependencies and ensuring overall system stability can become difficult. Additionally, there's a risk of **scope creep**, where the project expands beyond its original goals due to constant improvements. Another significant challenge is achieving a **balance between model accuracy and performance**, particularly on mobile devices with limited processing power and memory.

To mitigate these challenges, it's essential to **prioritize core features** in the initial development stages to build a solid foundation before adding advanced functionalities. To maintain good performance on mobile platforms, **AI models should be optimized** using methods like pruning, quantization, or by choosing lightweight architectures. This approach ensures a smooth user experience without compromising too much on the model's accuracy.

3.5 Concluding Remarks

The methodology for developing the StrokeSense App combines the strengths of prototyping and iterative design. By focusing on early error detection, user feedback, and incremental improvements, this approach ensures the app is both accurate and user-friendly. The incorporation of advanced medical imaging and educational resources further enhances its value, addressing critical gaps in stroke diagnosis and treatment accessibility.

The prototyping methodology provided an effective framework for developing a complex medical application that integrates advanced image processing techniques with practical clinical utility. By emphasizing iterative development and stakeholder involvement, this approach ensured that the final application would meet the needs of healthcare professionals while incorporating state-of-the-art few-shot learning algorithms for clot detection.

Chapter 4

System Design

4.1 System Block Diagram

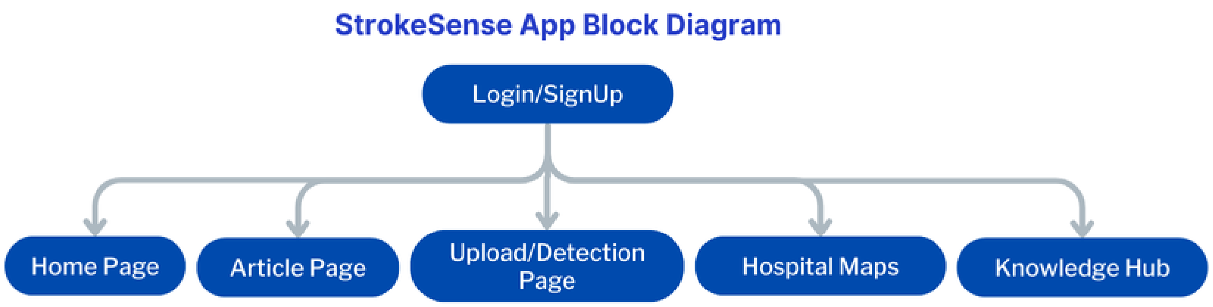


Figure 4.1 StrokeSense App Block Diagram

The Brain Stroke Detection (StrokeSense) Application is structured as a brain stroke solution incorporating multiple interconnected components designed to facilitate stroke diagnosis and patient care. At its foundation, the application implements a secure authentication system consisting of Login and Signup pages that verify user credentials and establish appropriate access permissions before granting entry to the system's core functionalities.

The **Home Page** serves as the central hub of the application, providing intuitive overview of the apps functionalities. This dashboard-style interface presents users with a streamlined overview of available features while maintaining a clean, uncluttered user experience optimized for healthcare environments where efficiency is paramount.

Complementing these clinical components, the **Article Page** serves as a comprehensive educational resource. This module contains evidence-based information regarding stroke symptoms, risk factors, prevention strategies, and treatment protocols. The content is structured to serve both healthcare professionals seeking reference materials and patients requiring accessible explanations of stroke-related concepts.

Connected to the profile is the **Profile Page** module, which maintains user-specific information and preferences. This component stores FAQ sections which let user browse on frequently asked questions, About Us which let user know more about the app and also Feedback for user feedback collection.

The **Upload/Detection Page** represents the primary diagnostic component of the application. This sophisticated module leverages few-shot learning algorithms to analyze Susceptibility-Weighted Imaging (SWI) MRI scans for potential stroke indicators. The detection system implements advanced image preprocessing techniques, including median and Gaussian filters, to enhance image quality before applying machine learning classification. This approach addresses the challenge of limited labeled medical datasets while maintaining high diagnostic accuracy, making it particularly valuable in resource-constrained healthcare environments.

Lastly, **Hospital Map** interface which being added after first round of user testing. This geolocation-enabled component displays all stroke treatment centers across Malaysia, automatically identifying the nearest facilities based on the user's current location. The map provides critical decision support during medical emergencies by offering facility-specific information including specialized treatment capabilities, contact information, and estimated travel times.

The application's block diagram illustrates the logical connections between these components, demonstrating how users progress from authentication through diagnosis to treatment location. This architecture supports the critical time-sensitive nature of stroke care by minimizing navigation complexity and prioritizing efficient information flow. The integration of diagnostic capabilities with immediate access to treatment facility information represents a significant advancement in streamlining the stroke care continuum, potentially reducing critical time-to-treatment intervals that directly impact patient outcomes.

4.2 Use Case Diagram

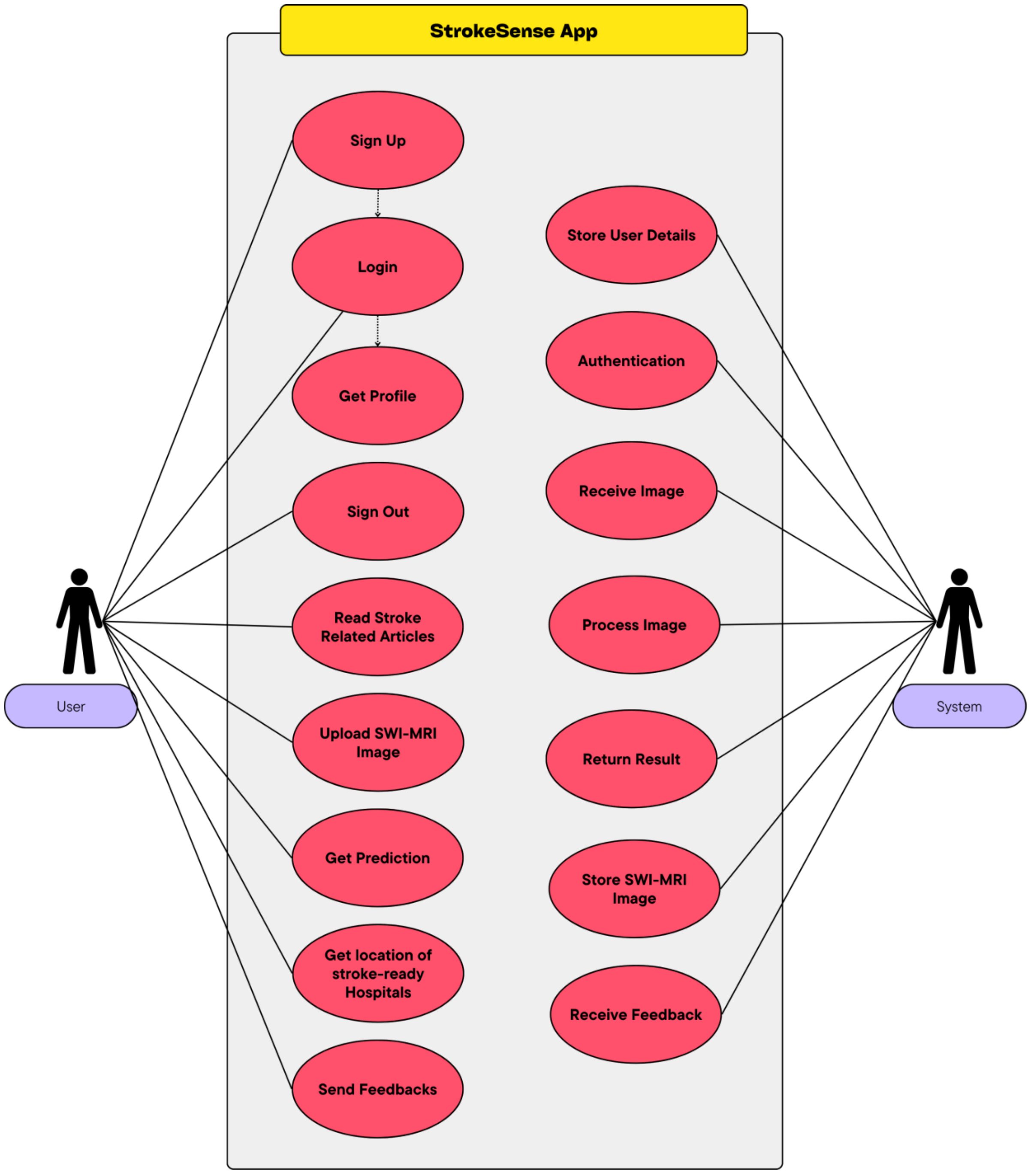


Figure 4.2 Use Case Diagram

The StrokeSense App is a comprehensive solution designed to assist users in clot detection, education, and healthcare access. This report elaborates on the use case diagram provided, illustrating the interaction between the user and the system within the app. The diagram highlights the functionalities offered by the app and their corresponding system operations. The main tasks that users can take are indicated in the diagram, along with how the system facilitates them.

At its core, the app allows users to create and manage their accounts securely through a sign-up and login process, enabling them to access personalized features such as their profile and past activities. The system supports these interactions by handling secure user data storage and authentication. Once logged in, users can navigate the app’s functionalities including image uploading for stroke prediction, educational content browsing, and locating nearby hospitals equipped for stroke care. The use case diagram visualizes how the user interacts with these features, while the system operates in the background to ensure seamless, efficient processing of each task.

One of the app’s key components is its AI-powered stroke prediction feature, where users can upload SWI-MRI brain images. The system receives and processes these images using machine learning algorithms, providing quick and accurate prediction results to the user. Additionally, the app empowers users with stroke-related knowledge through a curated knowledge hub, helping raise awareness and promote prevention. The healthcare navigation feature enhances user support by offering location-based access to stroke-ready hospitals. Finally, the app includes a feedback mechanism, allowing users to share their experiences and suggestions. All these use cases are supported by system operations like data storage, image analysis, result generation, and feedback management, ensuring the app runs smoothly. Overall, the StrokeSense App demonstrates a well-structured and purpose-driven approach to integrating technology into stroke prevention and education.

4.3 Activity Diagram

Sign-Up/ Login

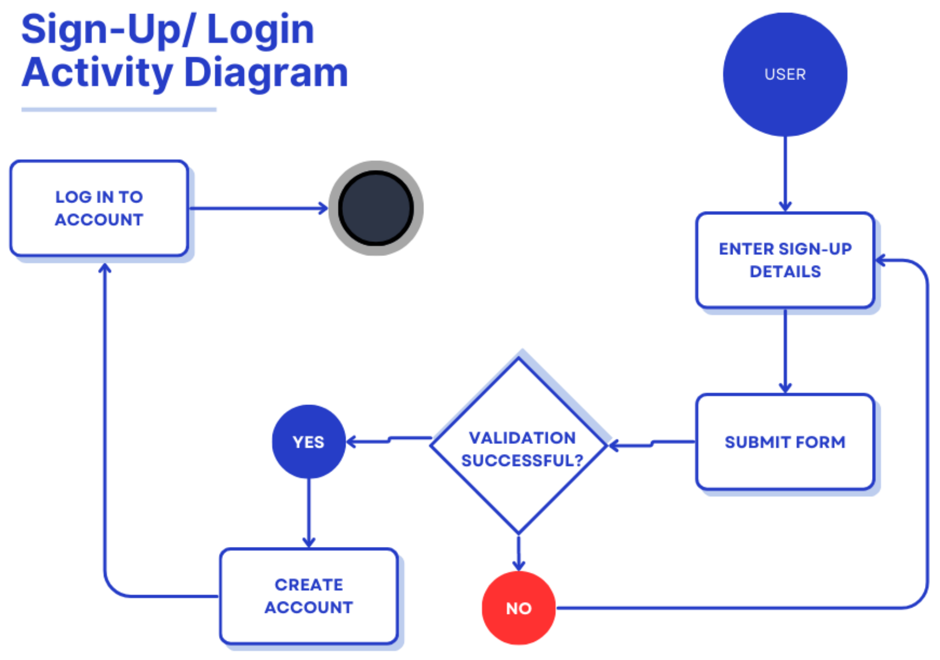


Figure 4.3 Sign-Up/ Login Activity Diagram

The activity diagram in the brain clot detection model shows the process from user engagement to result presentation. The StrokeSense app is launched by the user to start the process, which initiates the system. Initially, the user needs to sign up an account as logs onto the program. The procedure moves on to the next stage if the login or sign-up is successful; if not, the system asks the user to try again. The sign-up or login activity ends here.

Brain Clot Detection

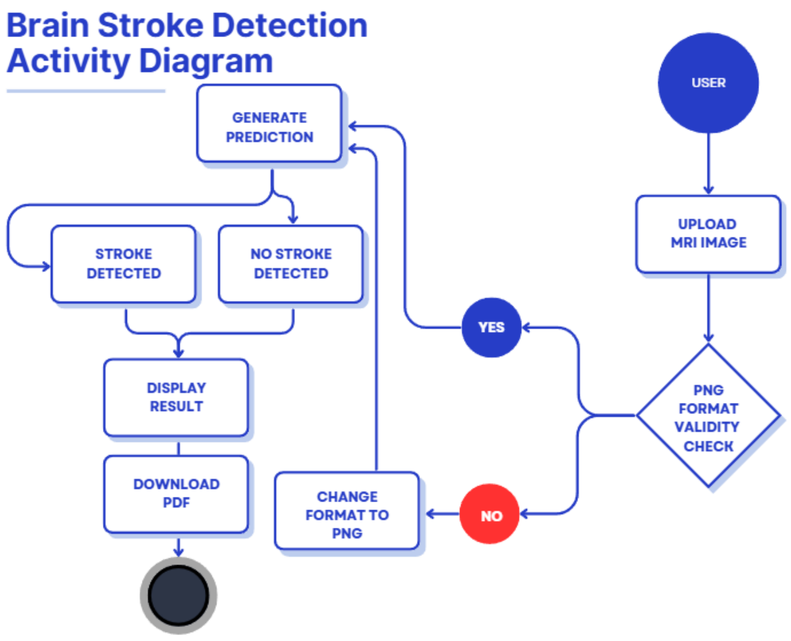


Figure 4.4 Brain Clot Detection Activity Diagram

The user uploads a brain MRI scan after logging in. The image format is then verified by the system. Preprocessing is the next step if the image is in the right format, at which point it is resized and normalized. The system will change the format to PNG if the file format found different.

Following preprocessing, the brain clot detection model is used by the StrokeSense app to analyze the image. It verifies whether the model processing has finished successfully. If it is successful, the system uses the analysis to create a prediction; if not, it logs the problem and shows an error message.

Project pre-development, data pre-processing, model training architecture construction and data training, and test dataset prediction were the activities that made up the project's development phases.

Next, the prediction is applied to see if a stroke is identified. The system produces a result showing the existence of a stroke if one is identified, and an absence of a stroke if none is. Ultimately, the user sees the results displayed by the system. Once the user has reviewed the results, the activity ends.

Read Stroke-Related Articles

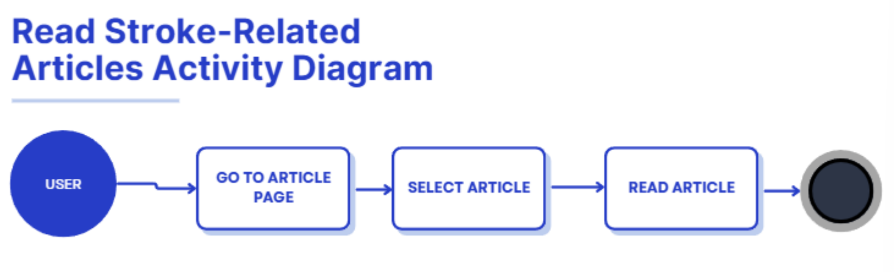


Figure 4.5 Read Stroke-Related Articles Activity Diagram

The activity diagram for reading stroke-related articles outlines a simple and user-friendly process. The user begins the interaction by navigating to the Article Page within the StrokeSense app. Once on the page, the user can browse through the available resources and select an article of interest. After selecting the desired article, the user proceeds to read the article, gaining valuable insights into stroke-related topics such as prevention, symptoms, and recovery. This streamlined flow ensures that users can easily access and consume educational content to enhance their understanding of stroke-related issues.

Get Maps

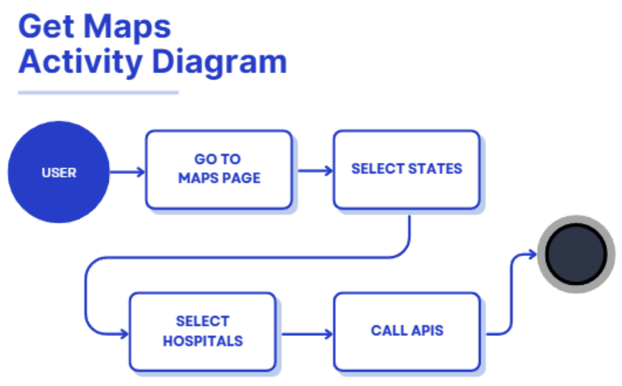


Figure 4.6 Get Maps Activity Diagram

The activity diagram for accessing maps within the application outlines a structured process for locating stroke-ready hospitals. The user initiates the process by navigating to the Maps Page. Then, the user selects a state to narrow down the search area. Following this, the user can select a hospital from the available options listed for the chosen state. The system then calls relevant APIs to fetch and display detailed information about the selected hospital, including its address. This flow ensures a seamless and efficient experience for users seeking critical healthcare facilities.

Submit Feedback

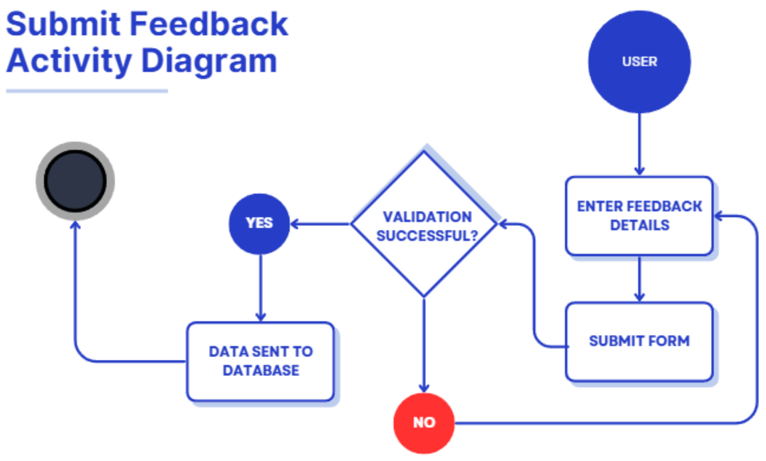


Figure 4.7 Submit Feedback Activity Diagram

The activity diagram for submitting feedback outlines the process through which users can provide their input to the system. The user begins the process by entering feedback details, such as suggestions, complaints, or comments, into the designated form. Once the feedback is filled out, the user proceeds to submit the form.

The system then validates the submitted feedback details. If the validation is successful, the data is sent to the database for storage and further analysis. If the validation fails (e.g., due to incomplete or invalid information), the user is prompted to correct the errors and resubmit the form. This ensures that only valid and meaningful feedback is stored, helping maintain the integrity of the feedback system while improving user experience.

4.4 Wireframe

The wireframe serves as a foundational blueprint for the StrokeSense app, providing a visual representation of the app's intended user interface and functionality. It outlines the core structure and flow of the app, ensuring that all key features are accounted for and logically designed before development begins. The wireframe offers a simplified, low-fidelity view of the app's layout, focusing on basic design elements and interactions rather than aesthetic details. It may differ slightly from the final design, as it is created during the pre-development phase and is subject to refinements during the development process.

The project will 2 two versions of the wireframe, reflecting the iterative development process. The first version represents the initial design, incorporating essential features such as user authentication, article browsing, image upload for clot detection, and profile management. After gathering user feedback and conducting thorough testing, the second version, or Prototype 2, will be developed. This iteration will include enhancements based on user responses and the addition of a new feature: the hospital maps functionality. The hospital maps feature will allow users to locate nearby stroke-ready hospitals, addressing a critical need for accessible healthcare information. Also, the navigation bar changed from 4 buttons to 5.

By creating and refining the wireframe across 2 versions, the design process ensures that user needs are met effectively, and the app's usability is maximized. This approach allows flexibility for adjustments and improvements, resulting in a user-centered design that aligns with both functional requirements and user expectations.

4.4.1 Wireframe Version One

Login and Sign-Up Page

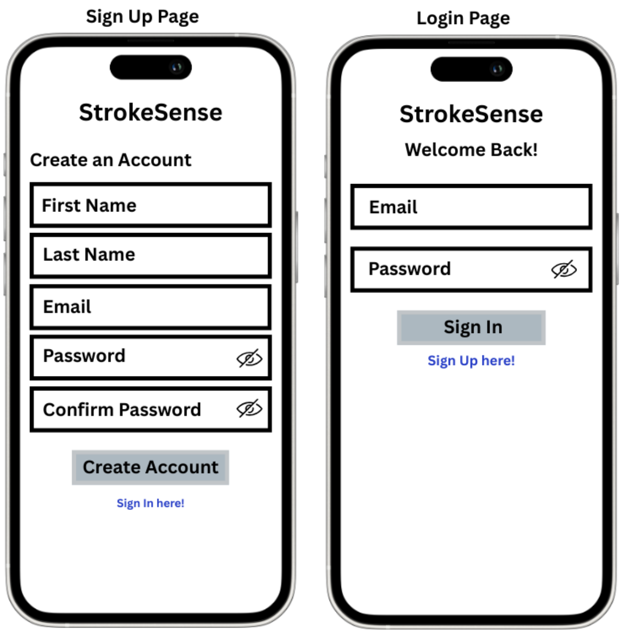


Figure 4.8 Wireframe V1 – Login and Sign Up Page

The wireframe depicts two essential authentication interfaces for the StrokeSense app: the Sign-Up Page and the Login Page. These screens are designed to facilitate user account creation and secure access to the app's features.

On the Sign-Up Page, users are prompted to create a new account by providing their First Name, Last Name, Email, and Password, with an additional field to Confirm Password for validation. Each input field is clearly labeled, and the password fields feature a visibility toggle to enhance usability and prevent errors during input. At the bottom of the screen, a prominent Create Account button allows users to submit their details and complete the registration process. Additionally, a link labeled Sign In here is provided for users who already have an account, enhancing navigation and user experience.

The Login Page is designed for returning users to access their accounts. It includes fields for Email and Password, both of which are clearly labeled for easy identification. Similar to the Sign-Up Page, the password field includes a visibility toggle for convenience. A Sign In button is positioned centrally for users to log in efficiently.

Below the button, a link labeled Sign Up here is included, allowing new users to navigate to the registration page if needed.

Both pages follow a simple, clean design with intuitive layouts, ensuring accessibility and ease of use for users of all experience levels. The consistent placement of navigation links promotes seamless transitions between the pages.

Home Page

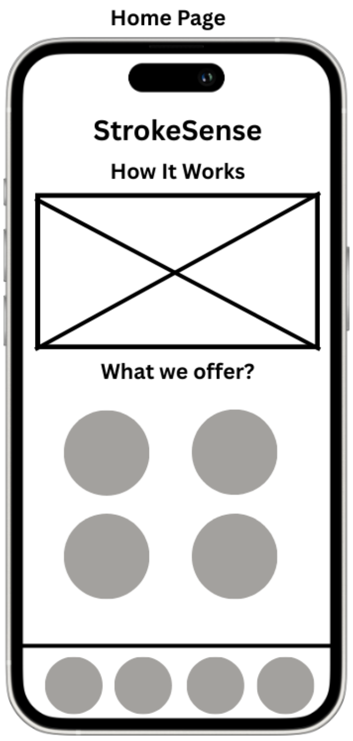


Figure 4.9 Wireframe V1 – Home Page

The wireframe represents the home page of the StrokeSense app, designed to provide users with a concise overview of the app's functionality and offerings. At the top of the page, the app's name, StrokeSense, is prominently displayed to reinforce branding. Below this, there is a section titled "How It Works," which is accompanied by a placeholder image or graphic that visually explains the app's core functionality.

Further down, a section labeled "What we offer?" showcases multiple circular icons arranged in a grid format. These icons represent the app's primary features or services, giving users a quick glimpse of what they can access. At the bottom of the page, there is a navigation bar with five circular buttons, likely corresponding to the app's main sections such as home, articles, upload/detection, maps, and profile. The layout is clean and minimalistic, focusing on simplicity and ease of navigation for users.

Upload Page

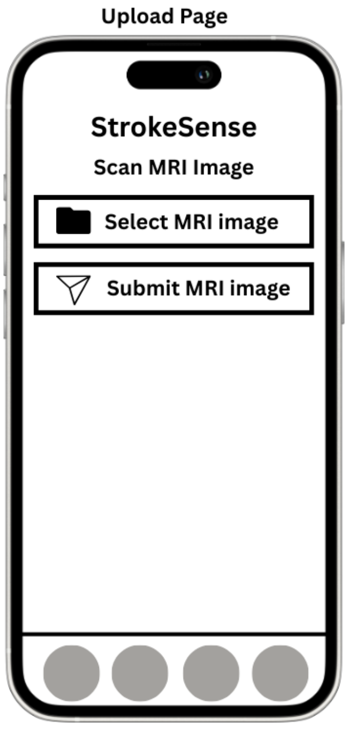


Figure 4.10 Wireframe V1 – Upload Page

The upload page of the StrokeSense app is designed to facilitate the process of scanning and submitting MRI images for analysis. At the top of the page, the app's name, StrokeSense, is displayed along with the title "Scan MRI Image" to indicate the page's purpose.

The interface contains two primary buttons. The first button, labeled "Select MRI image," allows users to browse and choose an MRI image from their device. The second button, labeled "Submit MRI image," enables users to upload the selected image for processing. Both buttons are accompanied by minimalistic icons to enhance user understanding and visual appeal.

Clot Detection Result Page

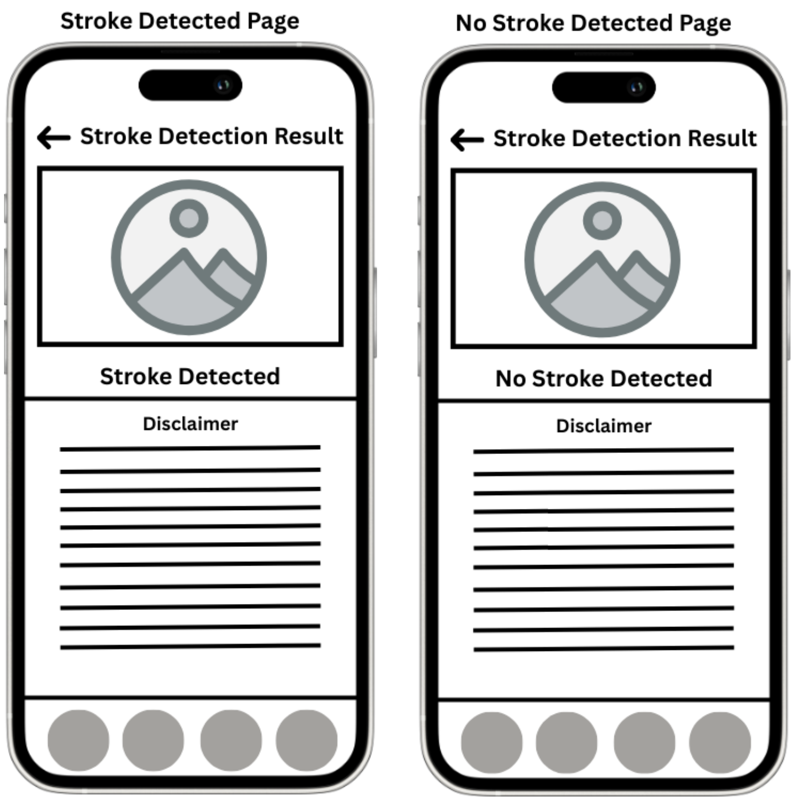


Figure 4.11 Wireframe V1 – Clot Detection Result Page

The clot detection result page displays the outcome of the clot detection process, with two variations depending on the result: stroke detected, or no stroke detected. At the top of both pages, there is a back arrow for navigation, followed by the title "Stroke Detection Result" to clearly indicate the purpose of the page.

The center of the page features an image placeholder, likely intended to display a visual representation or result related to the uploaded MRI image. Below the image, the detection result is shown in text form, either as "Stroke Detected" or "No Stroke Detected," depending on the outcome of the analysis.

A disclaimer section is positioned beneath the result, containing additional information or warnings relevant to the detection process and its limitations. This ensures users are aware of the app's purpose and any necessary precautions. The design is kept simple and direct, focusing on delivering the critical result information clearly and efficiently.

Article and Content Page

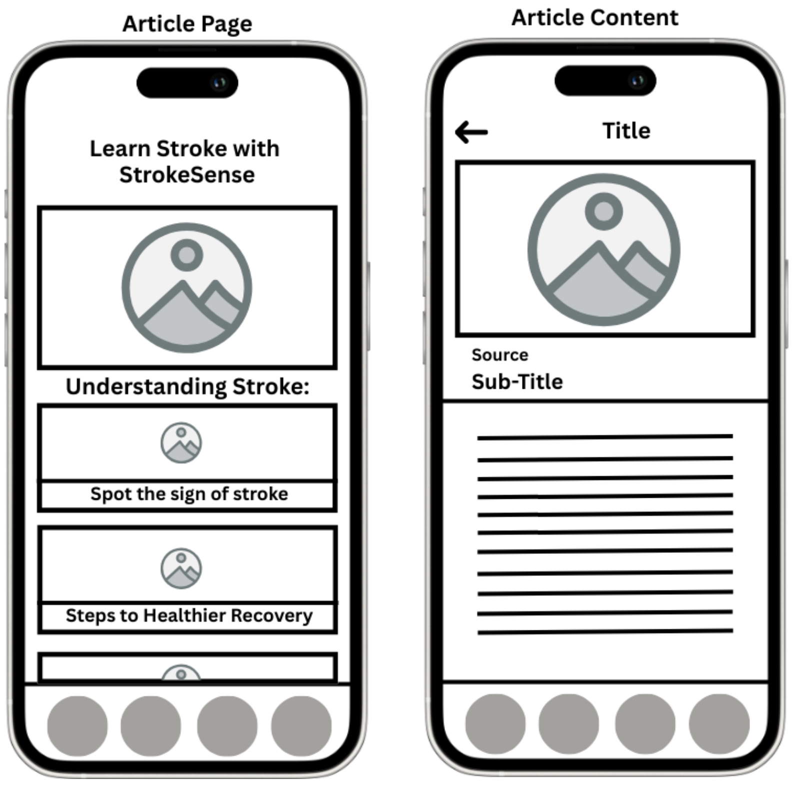


Figure 4.12 Wireframe V1 – Article and Content Page

The article page and article content page of the StrokeSense app are designed to educate users about strokes through accessible and structured information. The article page serves as an entry point for users to explore educational content. At the top, the header "Learn Stroke with StrokeSense" is displayed, followed by a placeholder image that represents the theme of the articles. Below this, a section titled "Understanding Stroke" introduces a list of articles represented as buttons with accompanying icons. Each button includes a title, such as "Spot the sign of stroke" or "Steps to Healthier Recovery," allowing users to navigate to specific articles for more detailed information.

The article content page opens when a user selects an article from the previous screen. At the top, there is a back arrow for navigation, followed by the article's title. A placeholder image is displayed below the title, providing visual context for the content. Under the image, the source of the article and a subtitle are displayed to give users additional context. The main content of the article is presented in a scrollable text area, allowing users to read the complete information at their own pace.

Profile Page

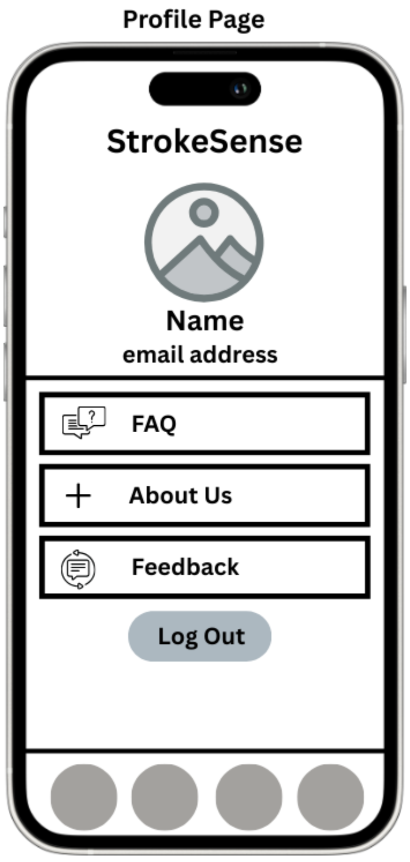


Figure 4.13 Wireframe V1 – Profile Page

The profile page of the StrokeSense app provides users with access to their account details and various informational and support features. At the top of the page, the app name StrokeSense is displayed, followed by a placeholder image representing the user's profile picture. Below the image, the user's name and email address are shown, allowing users to verify their account information at a glance.

The page includes three buttons for additional options. The FAQ button provides access to frequently asked questions, offering users quick answers to common queries. The About Us button gives information about the app and its purpose. The Feedback button allows users to share their thoughts or report issues directly to the developers.

At the bottom of the section, a prominent Log Out button is provided for users to securely exit their accounts. The overall design is clean and user-friendly, making it easy for users to navigate and access key features.

FAQ and Single Page

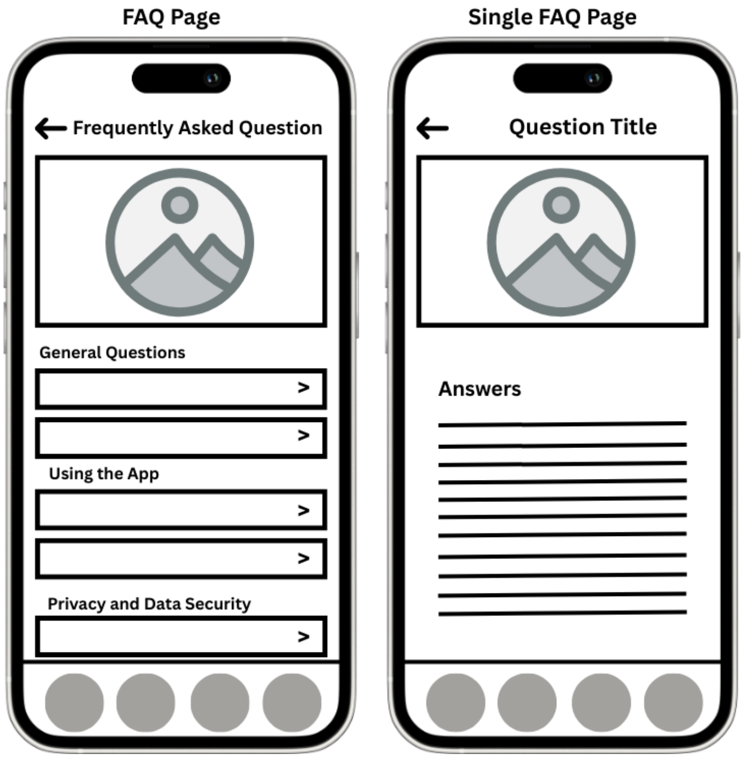


Figure 4.14 Wireframe V1 – FAQ and Single Page

The FAQ page and single FAQ page of the StrokeSense app are designed to provide users with quick and accessible answers to common questions. The FAQ page serves as the main directory for frequently asked questions. At the top of the page, a back arrow allows users to navigate to the previous screen, and the title "Frequently Asked Question" is displayed. Below the title, there is a placeholder image, likely representing the FAQ section visually. The page is divided into categories, such as "General Questions," "Using the App," and "Privacy and Data Security," each represented as buttons with a forward arrow. These buttons allow users to select a specific category or question to view more detailed answers.

The single FAQ page provides detailed answers to a specific question selected from the FAQ page. At the top, a back arrow enables navigation back to the FAQ directory, while the title of the selected question is displayed prominently. A placeholder image is positioned below the title, followed by a text area containing the detailed answer to the question. The layout is designed to be clear and user-friendly, ensuring that users can easily find and read the information they need.

About Us Page

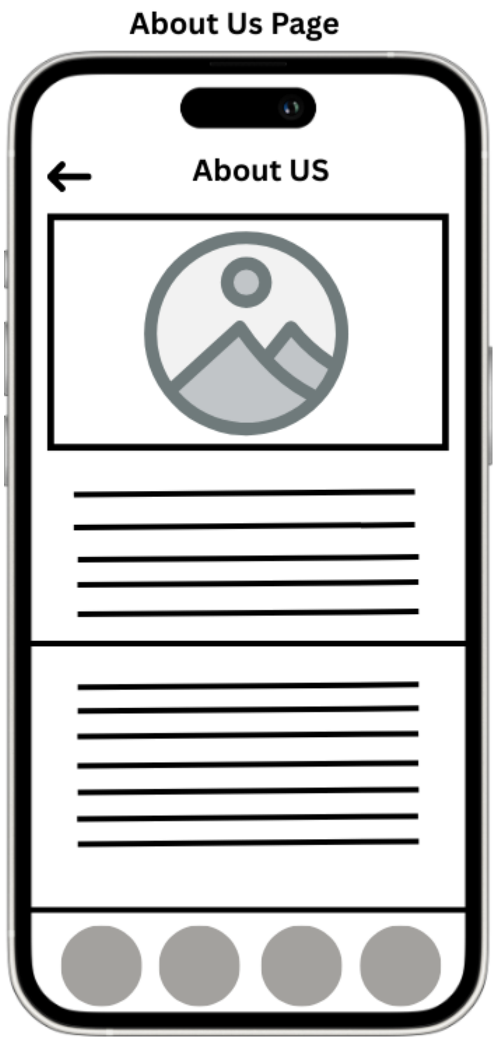


Figure 4.15 Wireframe V1 – About Us Page

The about us page of the StrokeSense app provides users with detailed information about the app and its purpose. At the top of the page, a back arrow allows users to return to the previous screen, followed by the title "About Us" to clearly indicate the purpose of the page. Below the title, a placeholder image is displayed. The main content is divided into two sections of text. The layout is straightforward and designed to effectively communicate the app's story and objectives to the users.

Feedback Page

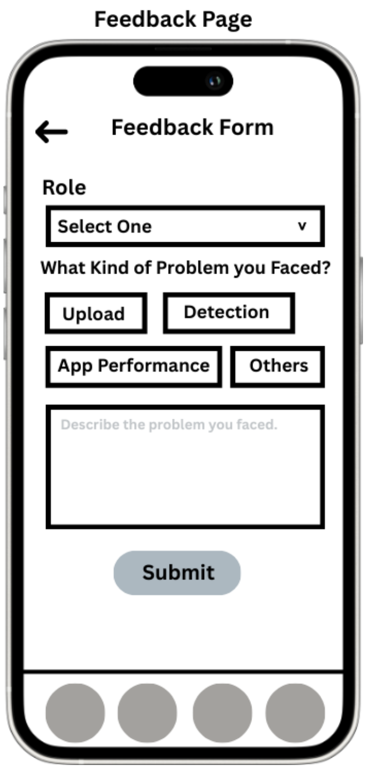


Figure 4.16 Wireframe V1 – Feedback Page

The feedback page of the StrokeSense app allows users to provide details about any issues or concerns they encounter while using the app. At the top of the page, a back arrow enables navigation to the previous screen, followed by the title "Feedback Form" to indicate the purpose of the page.

The form begins with a dropdown menu labeled "Role," where users can select their role or association with the app, such as a patient or healthcare provider. Below this, the section "What Kind of Problem you Faced?" provides multiple selectable options for users to categorize their issue. The available categories include "Upload," "Detection," "App Performance," and "Others," helping users specify the area of concern.

A text box is provided for users to describe the problem in detail, offering additional context about the issue they faced. At the bottom of the form, a "Submit" button allows users to send their feedback directly to the app's developers. The interface is simple and user-friendly, enabling users to submit their feedback quickly and efficiently.

Component – Navigation Bar

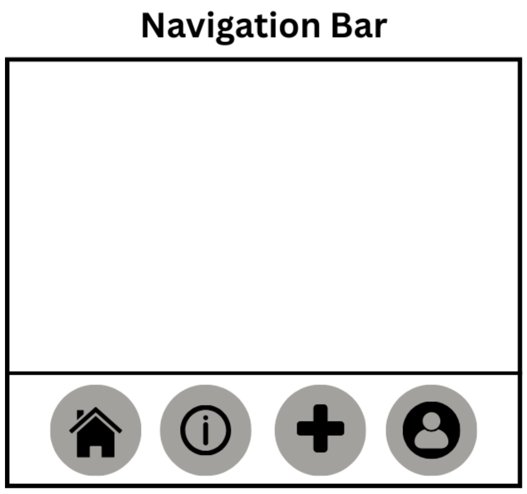


Figure 4.17 Wireframe V1 – Navigation Bar

The navigation bar component in the StrokeSense app provides users with quick access to the app's main sections. It consists of four circular icons, each representing a different functional area. The first icon, shaped like a house, likely serves as a shortcut to the home page. The second icon, represented by an information symbol, appears to lead to an informational section, such as the about us or FAQ page. The third icon, depicted with a plus sign, likely corresponds to a feature for uploading or adding new content, such as MRI scans. Finally, the fourth icon, in the shape of a user profile, provides access to the user's profile page.

The navigation bar is positioned at the bottom of the screen, ensuring consistent and easy access across the app. Its design is minimalistic, focusing on clarity and simplicity to enhance the user experience.

4.4.2 Wireframe Version Two

Based on the feedback gathered, users emphasized the importance of a centralized map that lists all hospitals in Malaysia with stroke-ready centers. This feature is particularly valuable for patients and caregivers who need quick access to hospital information during emergencies. The map allows users to sort hospitals by state, making it convenient to locate nearby facilities. Additionally, providing general details about each hospital, such as contact information and address, ensures that users can make informed decisions when seeking care.

I have created the second version of the wireframe after incorporating feedback from the first round of user testing. The updated wireframe reflects the changes and improvements suggested, and I will present it in the report below for further evaluation and feedback.

Login and Sign-Up Page

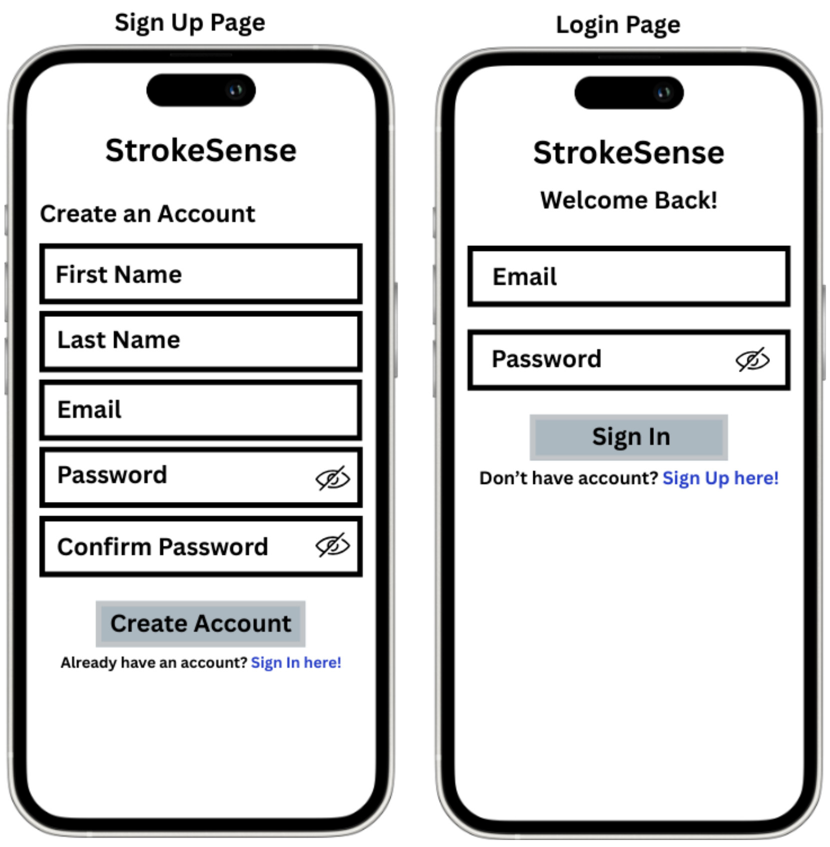


Figure 4.18 Wireframe V2 – Login and Sign Up Page

The second version of the signup and login pages for StrokeSense retains a clean and user-friendly layout. The Sign-Up Page includes fields for first name, last name, email, password, and confirm password, with a Create Account button and a link to sign in for existing users. The Login Page features fields for email and password, a Sign In button, and a link to navigate to the sign-up page. Both pages include an eye icon for toggling password visibility for user convenience. The design elements and structure remain consistent with version 1.

Home Page

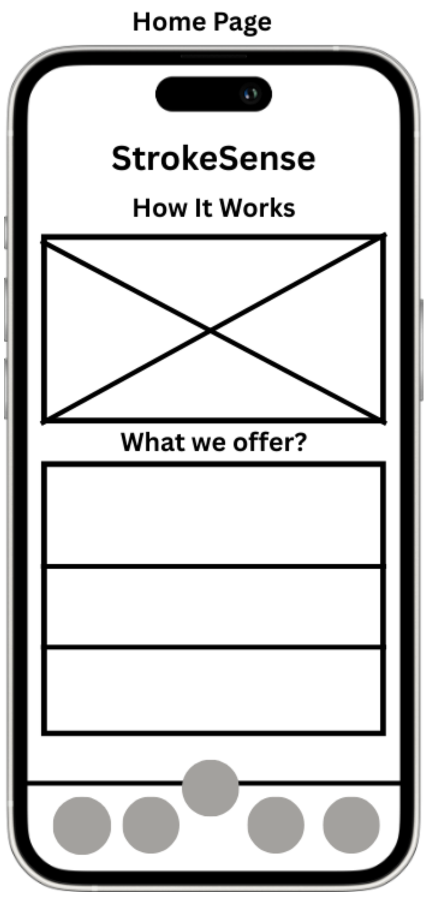


Figure 4.19 Wireframe V2 – Home Page

The second version of the home page for StrokeSense introduces slight differences compared to version 1. The primary change is the transition from a circular layout to a grid design for displaying content under the "What we offer?" section. This grid layout creates a more organized and tidy appearance, making it easier for users to navigate and visually process the information. While maintaining the core functionality, the updated design enhances clarity and user experience by adopting a more structured format.

Upload Page

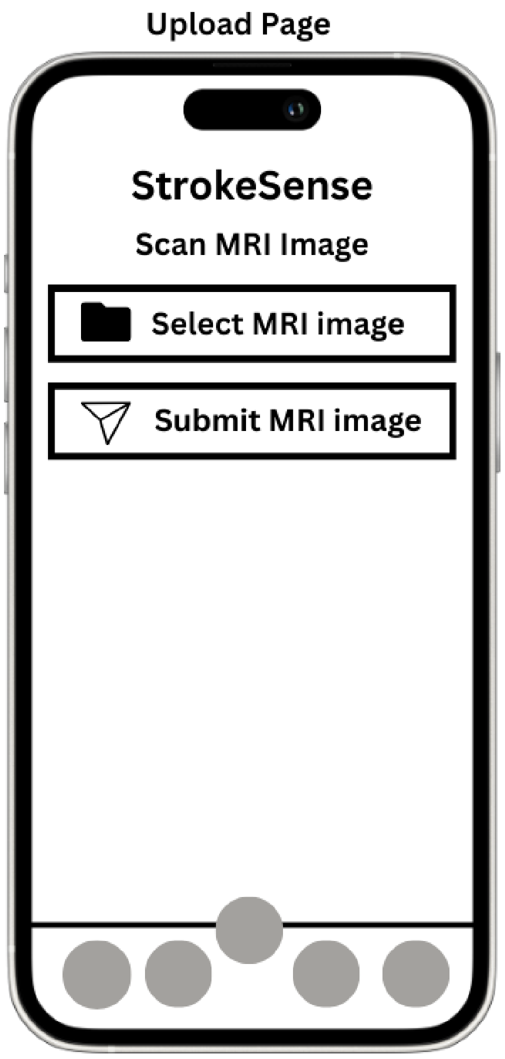


Figure 4.20 Wireframe V2 – Upload Page

The second version of the upload page for StrokeSense retains the same design and functionality as version 1. It includes two primary buttons: one for selecting an MRI image and another for submitting the selected image. The layout is simple and intuitive, ensuring ease of use for users while maintaining consistency with the previous version.

Detection Result Page

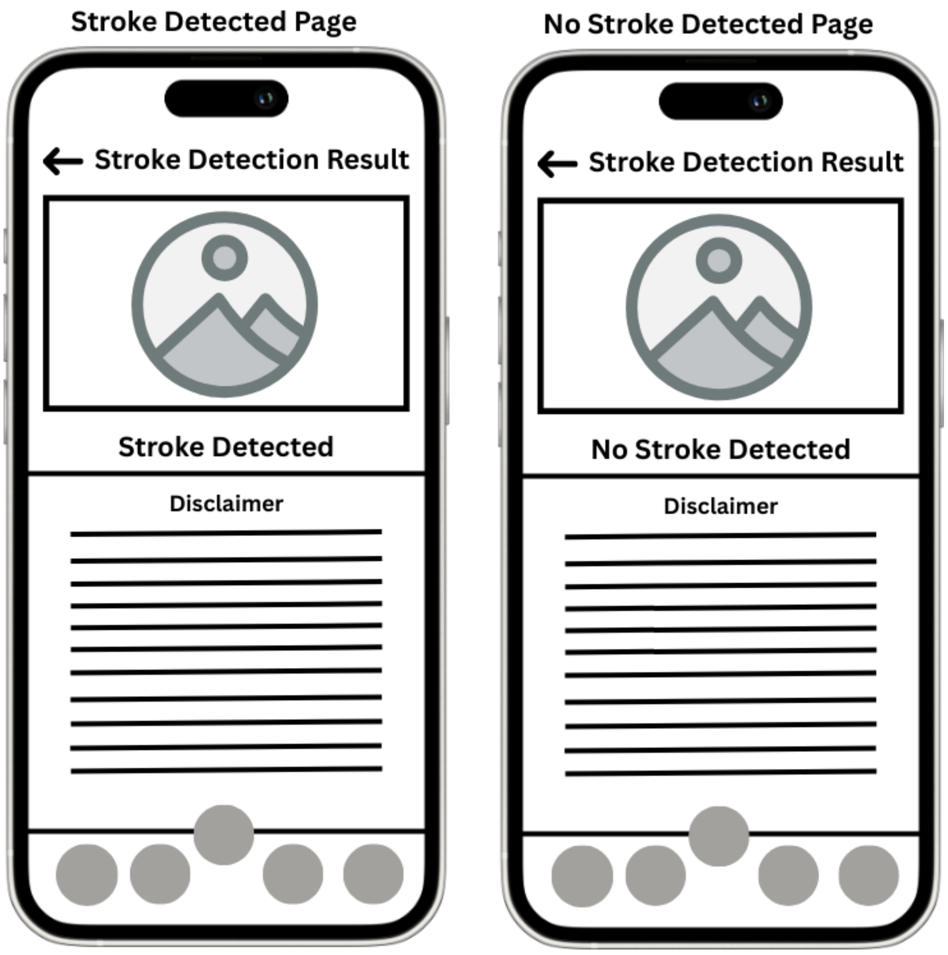


Figure 4.21 Wireframe V2 – Clot Detection Result Page

The second version of the Stroke Detection Result pages retains the same design and functionality as version 1. Both the "Clot Detected" and "No Clot Detected" pages feature a clear and simple layout with an image placeholder, a bold detection result message, and a disclaimer section below. The design ensures consistency and ease of understanding for users without introducing any changes from the first version.

Article and Single Page

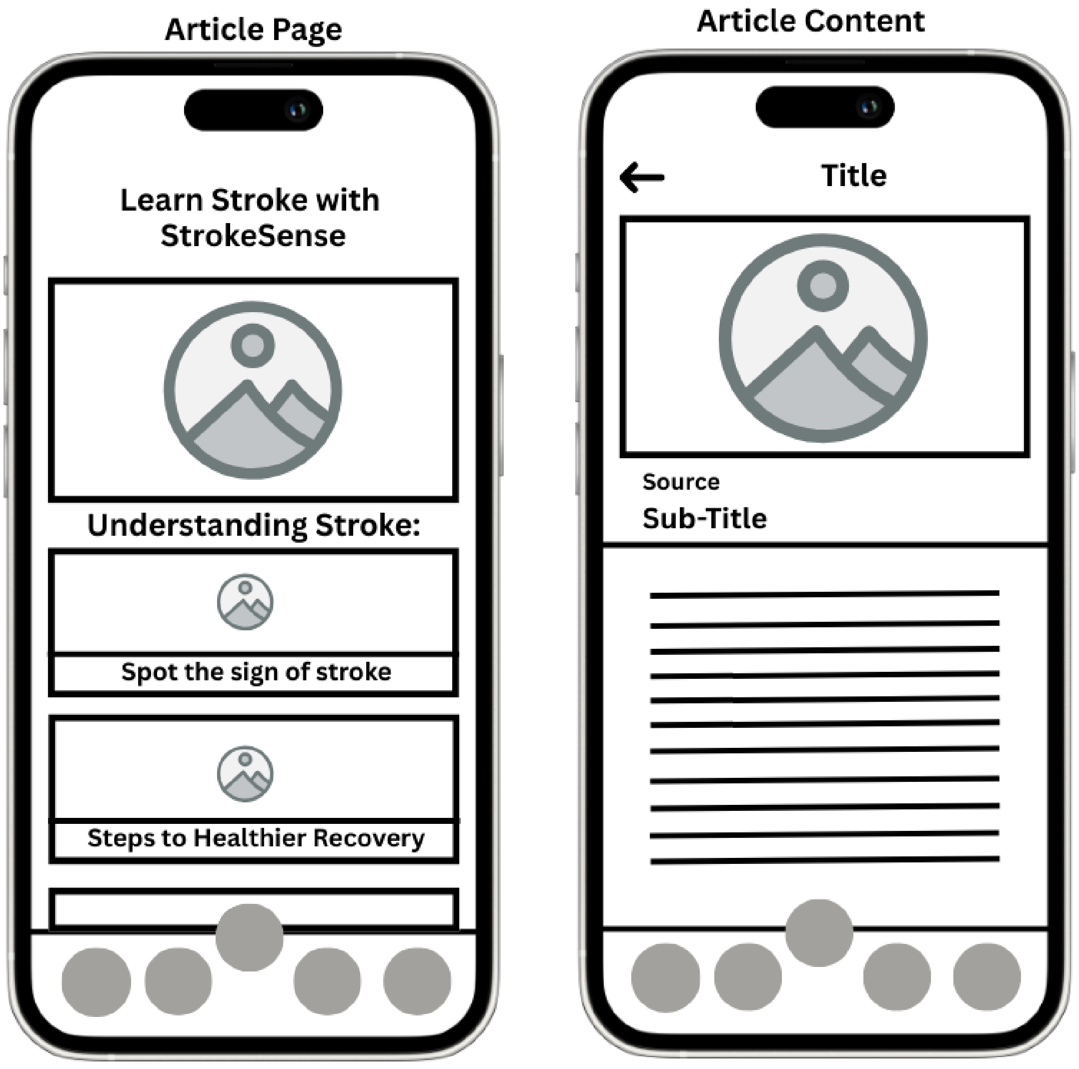


Figure 4.22 Wireframe V2 – Article and Content Page

The second version of the article pages retains the same design and functionality as version 1. The Article Page presents a visually structured layout with a featured section titled "Understanding Stroke" and clickable items like "Spot the sign of stroke" and "Steps to Healthier Recovery." The Article Content Page includes a title, source, subtitle, and detailed content below the image placeholder. The design remains consistent, ensuring familiarity and ease of use for the end user.

Profile Page

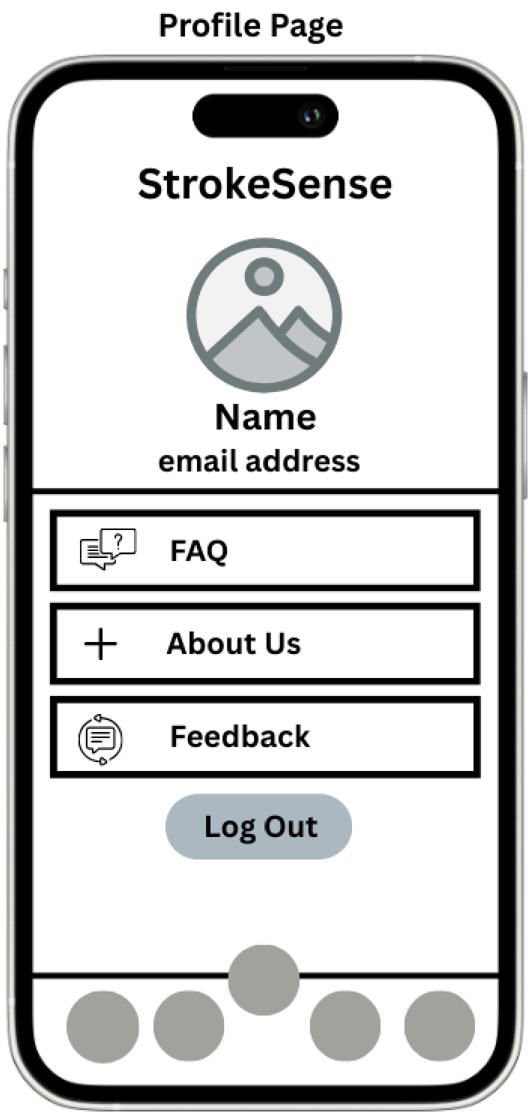


Figure 4.23 Wireframe V2 – Profile Page

The second version of the profile page remains the same as version 1, featuring a simple and intuitive layout. It displays the user's name and email address at the top, followed by buttons for accessing FAQ, About Us, and Feedback sections. A clearly visible Log Out button is included at the bottom for user convenience. The design is consistent and user-friendly, ensuring a seamless experience.

FAQ and Single Page

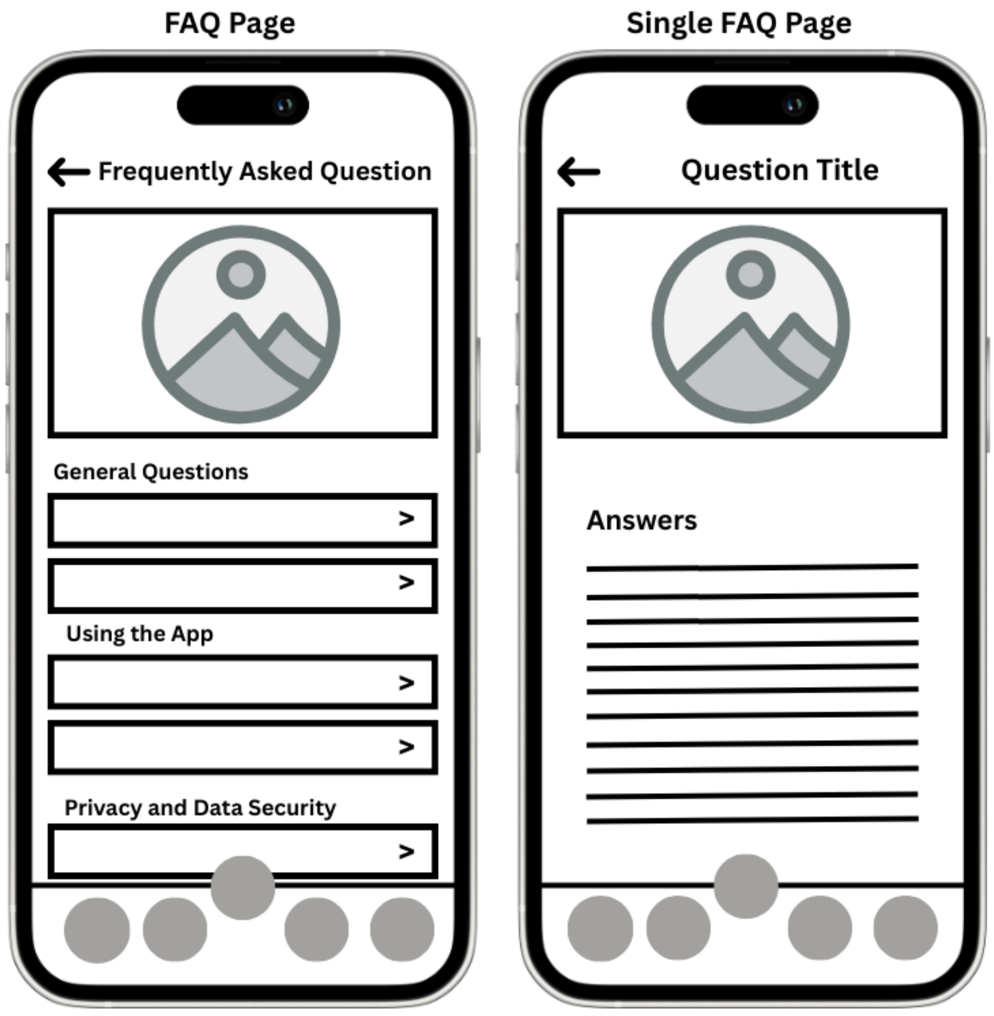


Figure 4.24 Wireframe V2 – FAQ and Single Page

The second version of the FAQ pages remains the same as version 1, maintaining a straightforward and user-friendly design. The FAQ Page displays categorized questions such as "General Questions," "Using the App," and "Privacy and Data Security," each with a clickable arrow for navigation. The Single FAQ Page provides the title of the selected question at the top, followed by detailed answers. The layout ensures ease of navigation and clarity for users seeking information.

About Us Page

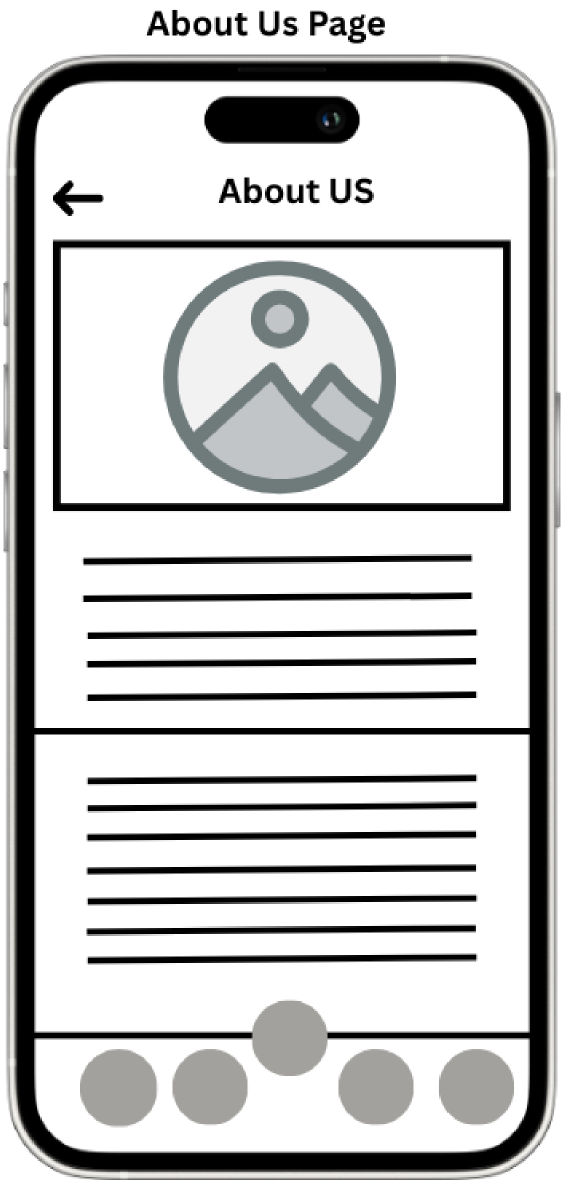


Figure 4.25 Wireframe V2 – About Us Page

The second version of the About Us page retains the same design and structure as version 1. It features a prominent image at the top, followed by sections of descriptive text providing information about the app or organization. The layout remains consistent, ensuring familiarity and ease of understanding for users.

Feedback Page

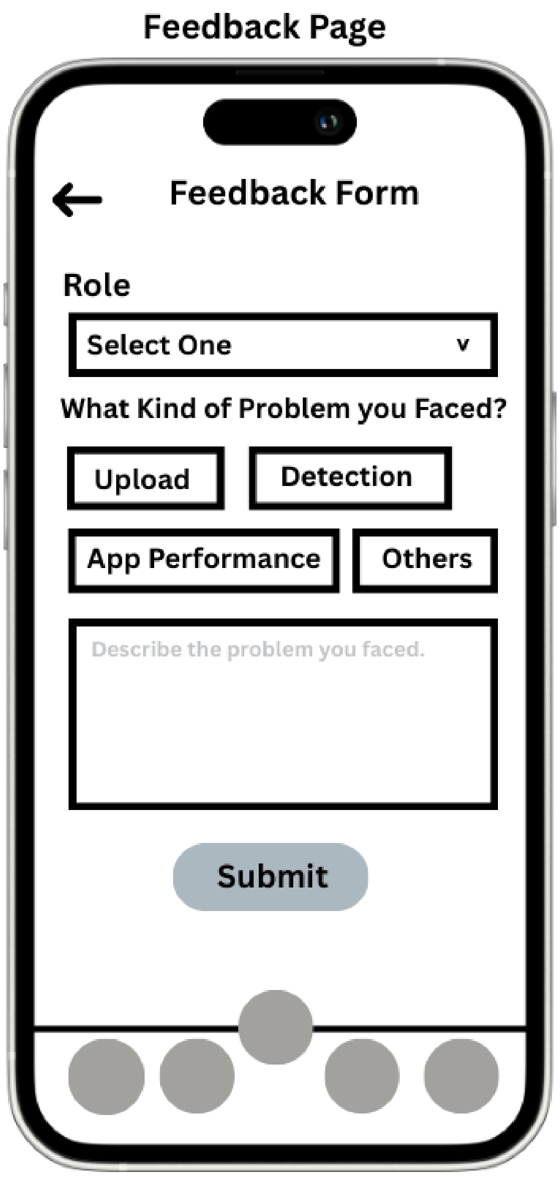


Figure 4.26 Wireframe V2 – Feedback Page

The second version of the Feedback page remains the same as version 1. It includes options to select the user's role, choose the type of problem faced like Upload, Detection, App Performance, Others), and a text box for describing the issue in detail. A "Submit" button is provided for users to send their feedback. The design is simple, user-friendly, and consistent with the previous version.

Maps Page

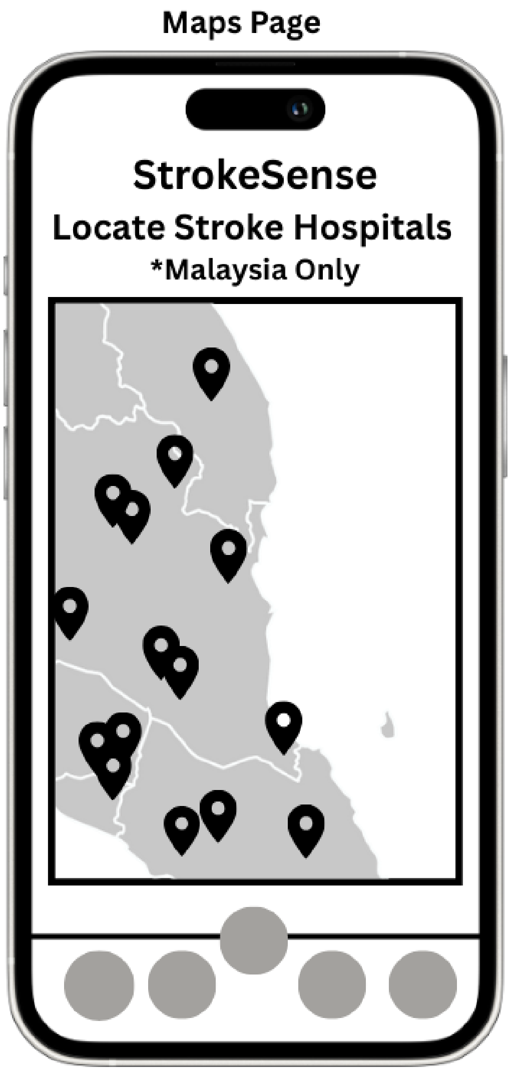


Figure 4.27 Wireframe V2 – Maps Page

The Maps Page is a new addition to StrokeSense, implemented based on user feedback to enhance accessibility to healthcare services. This page allows users to locate hospitals specializing in stroke treatment within Malaysia. It features an interactive map with marked locations pinpointing stroke hospitals across the Malaysia. The design ensures that users can easily identify nearby facilities for urgent medical assistance. This new feature reflects StrokeSense's commitment to improving user experience and addressing critical needs for timely stroke care.

Component - Navigation Bar

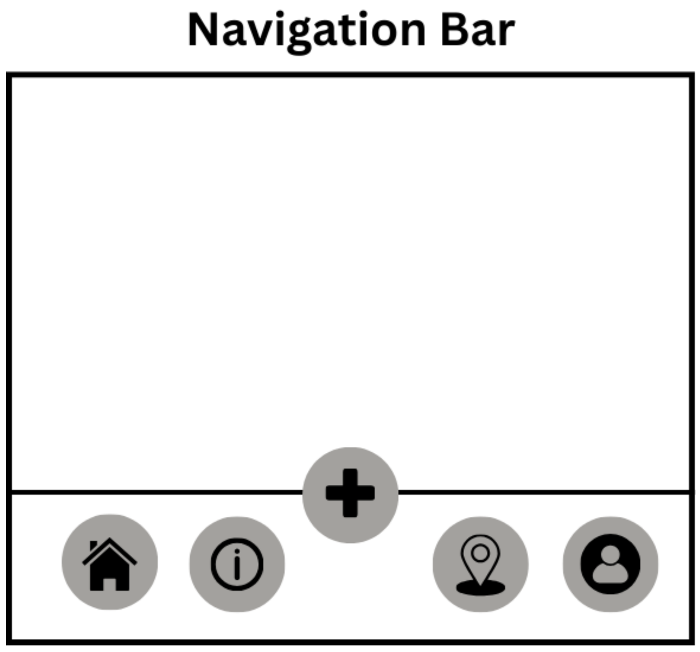


Figure 4.28 Wireframe V2 – Navigation Bar

The updated Navigation Bar now includes a new feature, the Maps Page, represented by a map pin icon. This update enhances navigation and user experience by providing quick access to the feature that helps users locate stroke hospitals. The navigation bar consists of five icons: Home (for the main dashboard), Information (for accessing FAQs and general details), Add/Create (for initiating new actions like orders or feedback), Maps (the newly-added feature for hospital locations), and Profile (for managing user accounts). This design ensures a user-friendly and streamlined experience.

Chapter 5

System Implementation

5.1 Hardware Setup

Laptop

Model	HP ZBook Firefly 14 inch G10
System Type	64-bit operating system, x64-based processor
Processor	AMD Ryzen 7 PRO 7840HS w/ Radeon 780M Graphics 3.80 GHz
Graphic Processor	4GB NVIDIA Quadro® GPU.
Graphic Card	AMD Raedon ™ Graphics
RAM	16 GB
Operating System	Microsoft Windows 11 Pro

Demo Mobile Device

Model	Xiaomi 11 Lite 5G NE
Version	MIUI Global 12.5.1 Stable
Processor	Qualcomm® Snapdragon™ 778G CPU: Qualcomm® Kryo™ 670, octa-core CPU, up to 2.4GHz GPU: Qualcomm® Adreno™ 642L GPU
RAM	8 GB
Storage	256 GB
Operating System	MIUI 12.5, Android 11

Table 5.1 List of Hardware

5.2 Software Setup

Software	**Description**
Google Colab	Google colab is acloud-based development environment to write and run Jupyter Notebook that containing Python code. This platform is used for training brain clot detection model.
Flutter	Flutter is a Dart framework, and it is cross-platform to develop mobile application such as IOS and Android. It uses for develop StrokeSense app.
Firebase	Firebase is a cloud-based platform developed by Google that offers a variety of tools and services to help developers build, improve, and scale apps quickly. It use for authentication and data storing.
Visual Studio Code	Visual Studio Code provides coding environment to edit code for StrokeSense app.
Google Maps API	Google Maps API is to integrate the maps functionality and locate stroke-ready hospitals.
Android Studio	Android Studio is an IDE for Android app development, it will integrate ARCore and OpenCV in Android Studio to develop application
Node.js	Node.js is an open-source, cross-platform JavaScript runtime environment. It executes JavaScript code of brain clot detection app on server side.
Python	Python is a versatile, high-level programming language. It handles backend logic and TensorFlow lite model.
Flask	Flask is a lightweight and flexible web framework written in Python that is used to build web applications and APIs quickly and easily.
Postman	Postman is a API development and testing tool that allows developers to easily create, send, and analyze HTTP requests and responses.

Table 5.2 List of Software

Google Colab

Figure 5.1 Google Colab

Google Colab is a cloud based Jupyter notebook environment provided by Google. It allows users to write and execute Python code directly in the browser, with no setup required. Colab supports many popular machine learning libraries such as TensorFlow, PyTorch, and OpenCV, making it ideal for data science and AI projects. Since it runs in the cloud, it provides access to high-performance computing resources and allows users to save their work directly to Google Drive, which enhances productivity and accessibility from any device.

In this project, Google Colab is used as the main platform for developing and running the AI-based clot detection model. It allows for the training, testing, and evaluation of the machine learning model using medical images (SWI-MRI scans). The model processes uploaded brain images to detect the presence of a stroke-related clot or no clot condition. Google Colab is particularly useful for this project because it supports powerful GPU and TPU acceleration, which significantly speeds up the training and prediction processes. It also enables easy collaboration and access to cloud storage, which helps in managing datasets and code efficiently.

Flutter

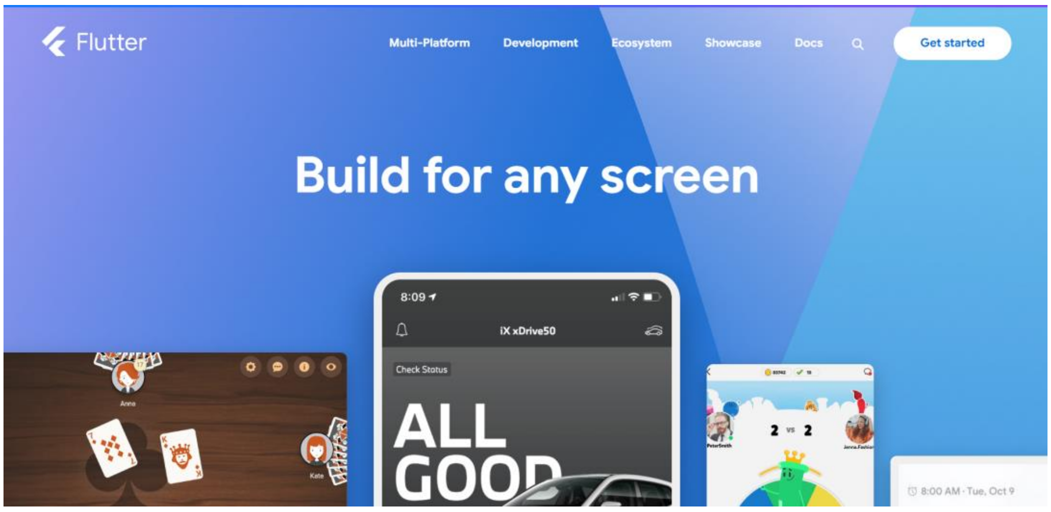


Figure 5.2 Flutter

Flutter is an open-source UI software development kit (SDK) created by Google. It uses the **Dart** programming language and provides a rich set of pre-designed widgets and tools for building visually attractive and responsive interfaces. One of Flutter’s key strengths is its hot reload feature, which allows developers to see changes in real-time without restarting the app. Flutter is especially known for its fast performance and customizable UI, making it a popular choice for building high-quality mobile apps in a short amount of time.

In this project, Flutter is not used directly, but it is worth noting as an alternative tool for building cross-platform mobile applications. Flutter can be considered for future development phases or similar app projects due to its high performance and flexibility. It enables developers to create beautiful, natively compiled applications for both Android and iOS from a single codebase, which can help reduce development time and ensure a consistent user experience across devices.

Firebase

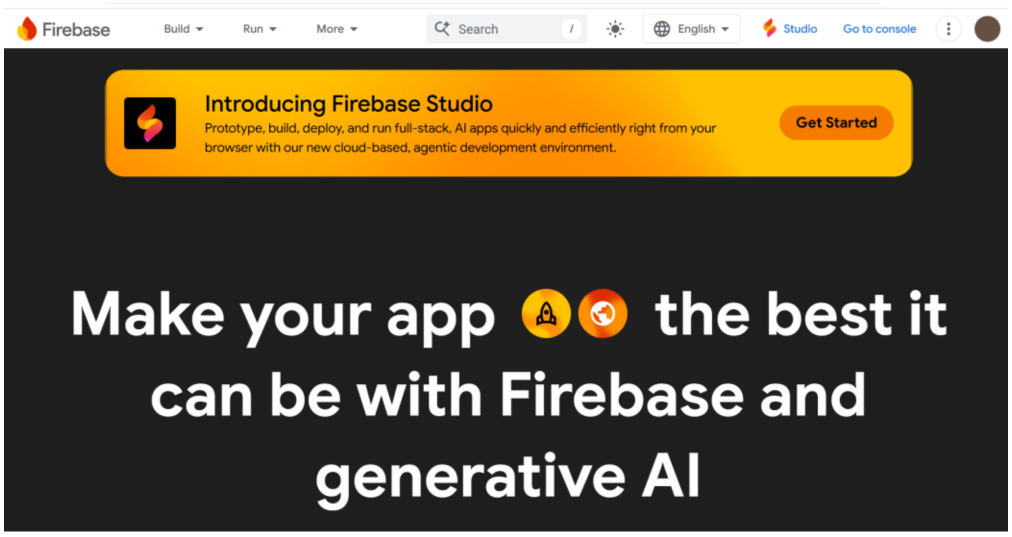

Firebase
Build
Run
More
Search
English
Studio
Go to console
Introducing Firebase Studio
Prototype, build, deploy, and run full-stack, AI apps quickly and efficiently right from your browser with our new cloud-based, agentic development environment.
Get Started
Make your app the best it can be with Firebase and generative AI

Figure 5.3 Firebase

Firebase is a platform developed by Google that offers a wide range of cloud-based services to support app development, including authentication, databases, storage, analytics, and notifications. It allows developers to build high-quality apps without needing to manage servers or build backend infrastructure from scratch. Key services include Firebase Authentication, Cloud Firestore, Cloud Storage, and Firebase Hosting.

In the StrokeSense App project, Firebase is primarily used for user authentication which is managing secure user login and signup processes. Also, real-time database management, storing and syncing user information and activities. Cloud Storage, saving uploaded medical images temporarily or permanently. Security management, protecting user data with authentication and database rules. Firebase helps ensure that users can safely create accounts, log in, and access their personalized profiles and uploaded images securely within the app.

Firebase Firestore

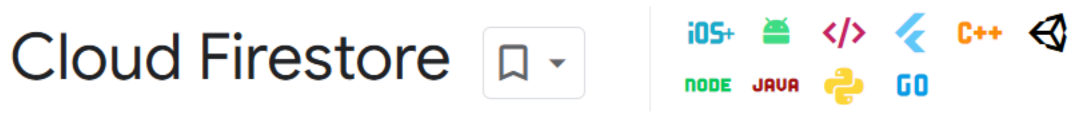
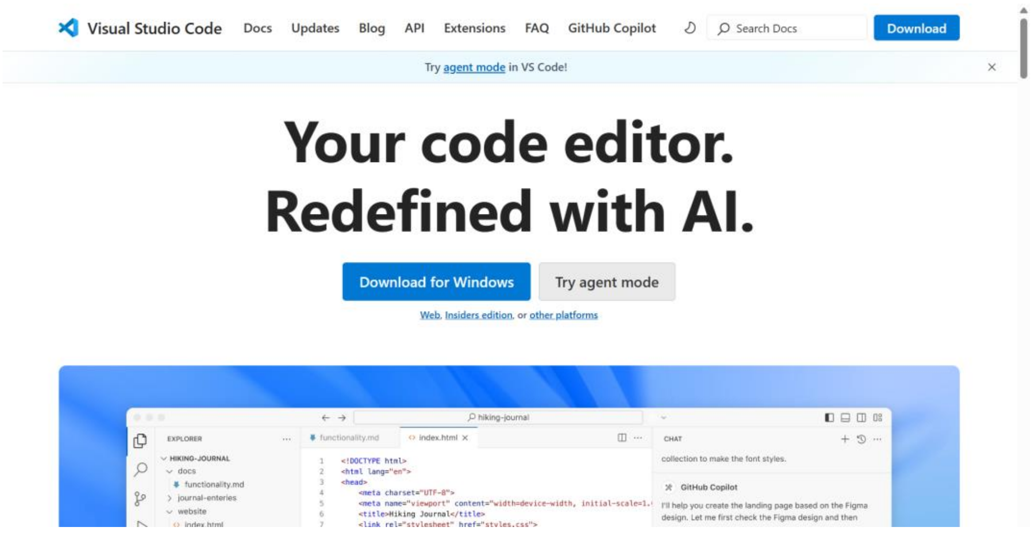


Figure 5.4 Firebase Firestore

Firebase Firestore is a cloud-based NoSQL document database that stores data in a flexible, scalable format. Data is organized into collections and documents, making it easy to structure complex data hierarchies. Firestore supports real-time data synchronization, offline capabilities, and strong security rules, making it a popular choice for mobile and web apps that require dynamic, responsive data management.

In the StrokeSense App project, Firebase Firestore is used for user profile storage, saving user account information like name, email, and profile details. It also used for uploaded image metadata, storing information about uploaded SWI-MRI images, such as upload time and associated prediction results. Feedback collection, storing user feedback about the app for future improvements. Firestore ensures that the app's data is structured, easily retrievable, and synced in real-time, improving the responsiveness and reliability of the app for users.

Visual Studio Code



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.
    <title>Hiking Journal</title>
    <link rel="stylesheet" href="styles.css">
```

collection to make the font styles.

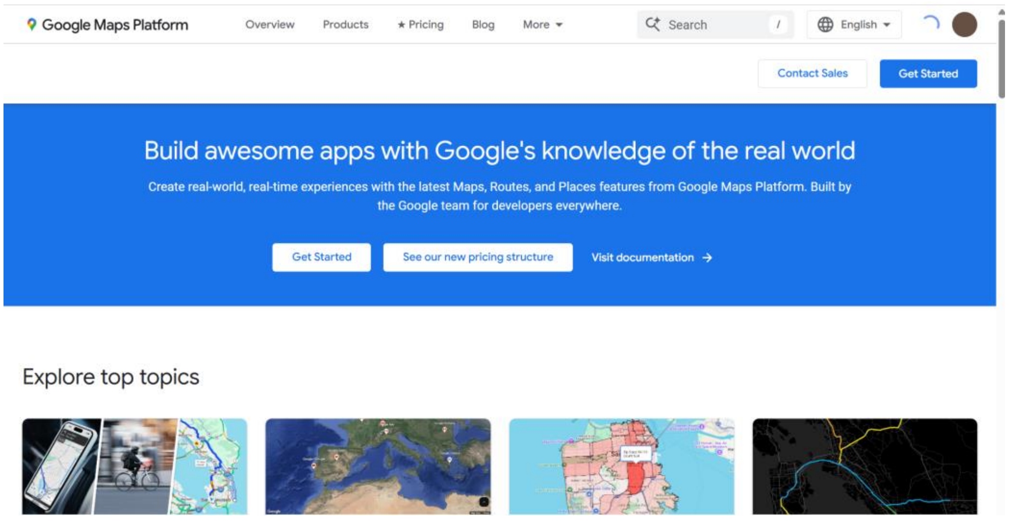
I'll help you create the landing page based on the Figma design. Let me first check the Figma design and then

Figure 5.5 Visual Studio Code

Visual Studio Code (VS Code) is a free, open-source code editor developed by Microsoft. It supports a wide range of programming languages such as JavaScript, Python, Java, C#, and more. VS Code is known for its lightweight performance, intelligent code completion (IntelliSense), built-in debugging, version control (Git) integration, and a wide range of extensions that enhance development productivity. It offers a highly customizable environment, allowing developers to tailor the editor to fit their project needs.

In the StrokeSense App project, Visual Studio Code is used for writing and managing code, developing the front-end layout and navigation for the app's pages like the Home Page, Profile Page, Upload Page, and Article Page. Connecting to firebase services like implementing Firebase authentication and Firestore database management within the app. Also, it involves in managing machine learning API integration, which including writing scripts that allow the app to send uploaded images to the machine learning model and receive prediction results. Besides, testing and debugging, identifying, and resolving code errors to ensure smooth app functionality. In short, it is all about project organization which managing multiple files and folders efficiently using the editor's workspace features.

Google Maps API



Build awesome apps with Google's knowledge of the real world

Create real-world, real-time experiences with the latest Maps, Routes, and Places features from Google Maps Platform. Built by the Google team for developers everywhere.

Explore top topics

Figure 5.6 Google Maps API

The Google Maps API is a set of services and tools provided by Google that allows developers to embed and customize maps in mobile applications, websites, and software systems. It offers features such as displaying interactive maps, adding markers, generating directions, calculating distances, and accessing place information. The API supports location-based functionality that enhances user experience by delivering real-world navigation and mapping services directly inside an app.

In the StrokeSense App project, the Google Maps API is used for locating stroke-ready hospitals, which can help users quickly find hospitals that specialize in stroke care based on their desired location. Also, providing directions, offering a way for users to view routes and navigate easily to healthcare facilities. Besides, displaying interactive maps, it allows users to interact with the map, zoom in/out, and select different hospitals or healthcare points. In short, it enhances user trust and experience, which giving users reliable, real-time location data integrated seamlessly into the app's interface.

Android Studio

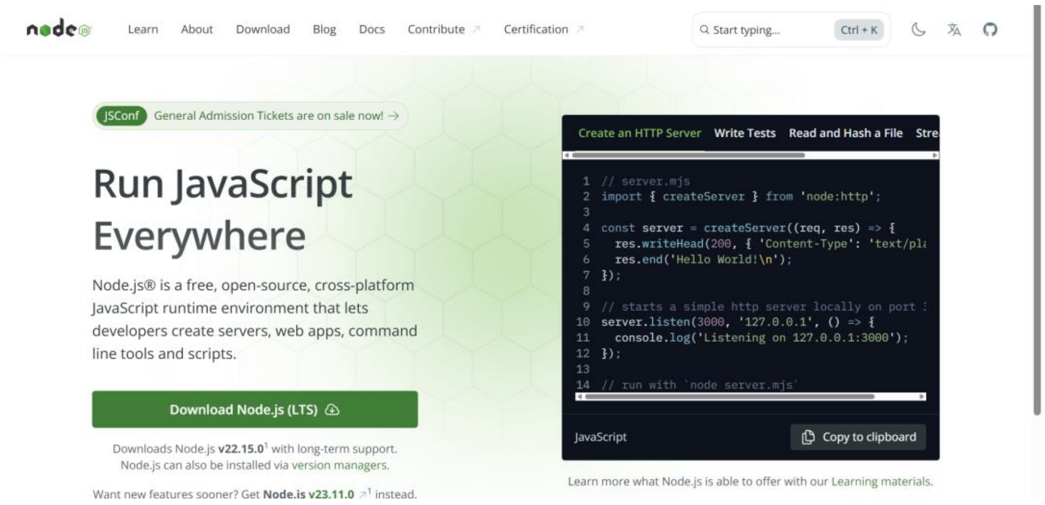
Figure 5.7 Android Studio

Android Studio is the official integrated development environment (IDE) for Android application development, provided by Google. It offers a complete set of tools for coding, designing, testing, and debugging Android apps. Android Studio supports programming languages such as Java, Kotlin, and XML, and includes features like an intelligent code editor, a visual layout editor, an emulator for testing apps, and powerful debugging and performance tools. It helps streamline the app development process by providing a unified platform that integrates all necessary development functions. There are several features include versatile Gradle-based build system, powerful and quick emulator that support user can test their apps in Android Studio when developing their apps, a single environment where you can work on Android projects for all devices, Firebase integration and others.

In the StrokeSense App project, Android Studio is used specifically for running the Android Emulator to test the user interface and navigation flow of the app. Besides, it also simulating app behaviour across different Android devices and screen sizes. Also, it debugs and observed performance to ensure the app functions correctly before actual deployment.

The emulator provides a safe and controlled environment to examine how different pages of the app (such as the Home Page, Upload Page, Profile Page, etc.) appear and interact, without relying on physical hardware.

Node.js



General Admission Tickets are on sale now! →

Run JavaScript Everywhere

Node.js® is a free, open-source, cross-platform JavaScript runtime environment that lets developers create servers, web apps, command line tools and scripts.

```
// server.mjs
import { createServer } from 'node:http';

const server = createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/pl
  res.end('Hello World!\n');
});

// starts a simple http server locally on port
server.listen(3000, '127.0.0.1', () => {
  console.log('Listening on 127.0.0.1:3000');
});

// run with `node server.mjs`
```

Downloads Node.js **v22.15.0**[1] with long-term support.
Node.js can also be installed via version managers.

Want new features sooner? Get **Node.js v23.11.0**[1] instead.

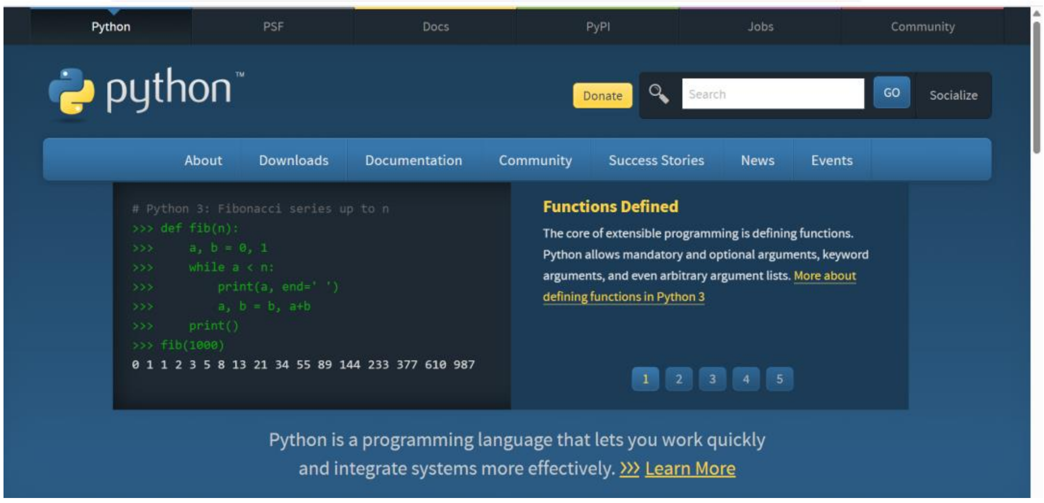
Learn more what Node.js is able to offer with our Learning materials.

Figure 5.8 Node.js

Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to run JavaScript code outside of a web browser. It is built on the V8 JavaScript engine developed by Google. Node.js is well known for its event-driven, non-blocking I/O architecture, which makes it highly efficient and scalable for building server-side applications, RESTful APIs, and real-time services. It also has a rich ecosystem of libraries available through npm (Node Package Manager).

In the StrokeSense App project, Node.js could serve in the following ways. First, API Backend Server, managing the interaction between the mobile app and the AI model hosted on Google Colab or cloud platforms. Next, image upload and processing management, including manage the upload of SWI-MRI images from the mobile app to cloud storage and processing services.

Python



```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

Functions Defined

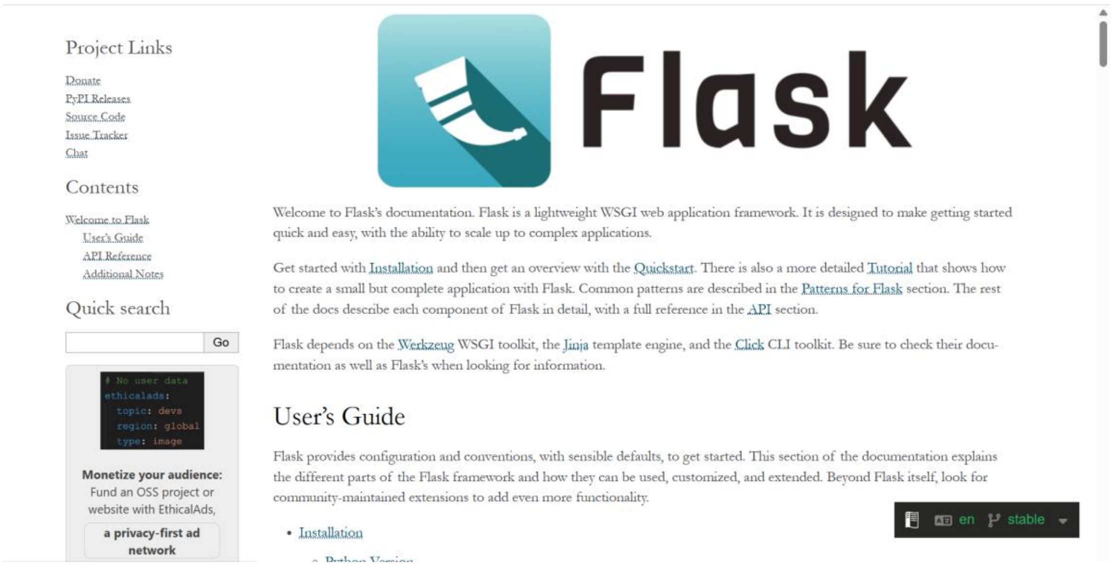
The core of extensible programming is defining functions. Python allows mandatory and optional arguments, keyword arguments, and even arbitrary argument lists. More about defining functions in Python 3

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

Figure 5.9 Python

Python is a versatile, high-level programming language widely used for its simplicity and readability, making it ideal for both beginners and experienced developers. In the StrokeSense app, Python plays a critical role in developing the machine learning components, particularly for integrating TensorFlow Lite, a lightweight version of the TensorFlow framework designed for mobile and embedded devices. Python is used to train, optimize, and deploy the brain clot detection models, enabling real-time predictions from MRI images. Additionally, Python handles the backend logic of the app, ensuring smooth data processing and communication with other system components, thus enhancing the app's functionality and user experience in detecting stroke conditions.

Flask



Project Links

Donate
PyPI Releases
Source Code
Issue Tracker
Chat

Contents

Welcome to Flask
- User's Guide
- API Reference
- Additional Notes

Quick search

Go

```
# No user data
ethicalads:
  topic: devs
  region: global
  type: image
```

Monetize your audience: Fund an OSS project or website with EthicalAds, **a privacy-first ad network**

Flask

Welcome to Flask's documentation. Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications.

Get started with Installation and then get an overview with the Quickstart. There is also a more detailed Tutorial that shows how to create a small but complete application with Flask. Common patterns are described in the Patterns for Flask section. The rest of the docs describe each component of Flask in detail, with a full reference in the API section.

Flask depends on the Werkzeug WSGI toolkit, the Jinja template engine, and the Click CLI toolkit. Be sure to check their documentation as well as Flask's when looking for information.

User's Guide

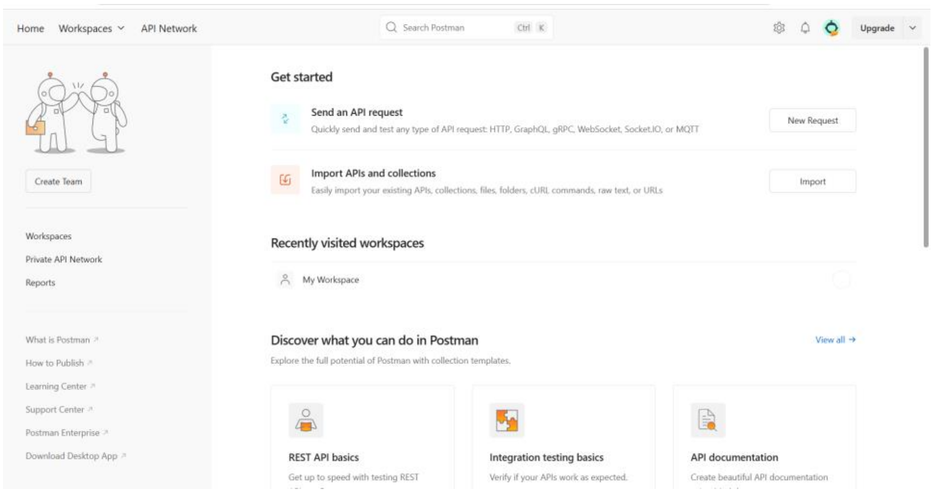
Flask provides configuration and conventions, with sensible defaults, to get started. This section of the documentation explains the different parts of the Flask framework and how they can be used, customized, and extended. Beyond Flask itself, look for community-maintained extensions to add even more functionality.

- Installation
  - Python Version

Figure 5.10 Flask

Flask is a lightweight and flexible web framework written in Python that is used to build web applications and APIs quickly and easily. It provides the essential tools and libraries needed to create server-side functionality without imposing too many restrictions, allowing developers to structure their projects freely. In the context of the StrokeSense app, Flask is used to create a RESTful API that connects the machine learning model to the mobile application. This API handles requests from the app, processes uploaded MRI images, runs the clot detection model, and sends back the prediction results. By using Flask, the app ensures fast, efficient, and scalable communication between the front end and the machine learning backend.

Postman



Get started

Send an API request
Quickly send and test any type of API request: HTTP, GraphQL, gRPC, WebSocket, Socket.IO, or MQTT

Import APIs and collections
Easily import your existing APIs, collections, files, folders, cURL commands, raw text, or URLs

Recently visited workspaces

My Workspace

Discover what you can do in Postman
Explore the full potential of Postman with collection templates.

Get up to speed with testing REST

Verify if your APIs work as expected.

Create beautiful API documentation

Figure 5.11 Postman

Postman is a popular API development and testing tool that allows developers to easily create, send, and analyze HTTP requests and responses. It provides a user-friendly interface for testing RESTful APIs without needing to write code, making it especially useful during backend development. In the StrokeSense app, Postman is used to test the Flask API that handles MRI image uploads and returns clot detection results. By using Postman, developers can simulate client requests to ensure the API correctly receives images, processes them through the machine learning model, and returns accurate predictions. It also helps in debugging and verifying API endpoints, checking response times, and ensuring the backend functions properly before it is connected to the mobile app or deployed to platforms like Heroku or AWS.

5.3 Brain Clot Detection Model Training Process

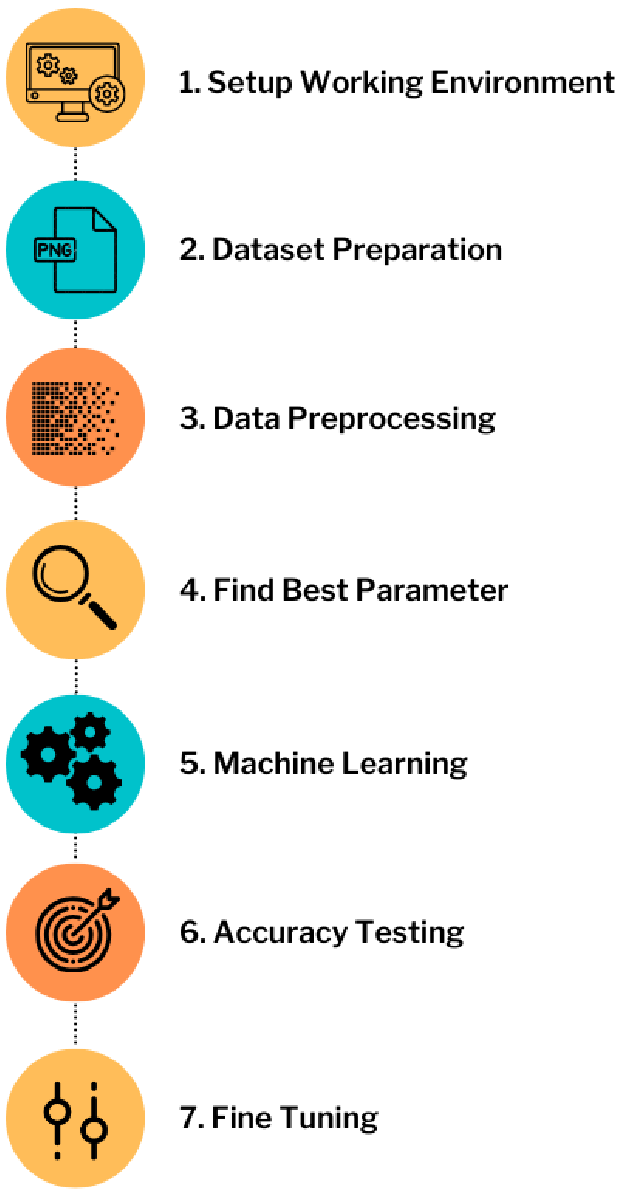


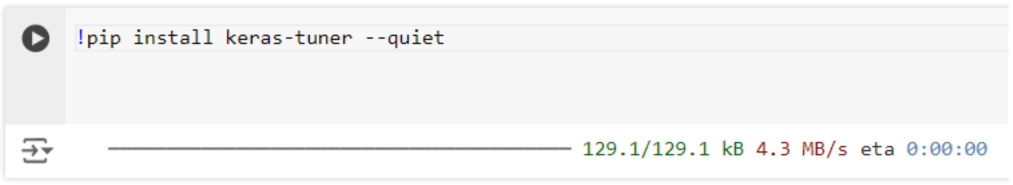
Figure 5.12 Brain Clot Detection Model Training Process

5.3.1 Setting up for Working Environment, Google Colab

Before starting to develop the Brain Clot Detection Model, Google Colab has been set up. It was crucial to set up a stable and adaptable environment for machine learning algorithm testing and experimentation before beginning work on the brain clot detection model. The main reason for selecting Google Colab as the development environment was its user-friendliness, robust GPU resources, and smooth integration with Python-based machine learning frameworks [26]. Larger computations are needed for Deep Learning when a CPU is insufficient. Consequently one must also rely on a GPU.

In order to use Google Colab and its related services, such as Google Drive, the setup process started with making a new Google account or using existing Google account. In this case, new Google Account was created for separation of FYP from personal work. After logging in, a web browser was used to access the Google Colab interface, and a GPU-supporting runtime was linked to manage the processing demands of deep learning model training.

The necessary libraries for Python were then installed, which is Keras for managing neural networks. Keras utilizes the TensorFlow framework to improve performance and visualization [24].

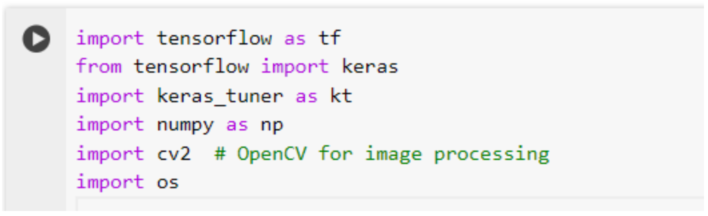


```
!pip install keras-tuner --quiet
```

```
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 129.1/129.1 kB 4.3 MB/s eta 0:00:00
```

Figure 5.13 Install Keras Libraries

After install libraries, others required libraries imported included NumPy for numeric operations, OpenCV for image processing, and TensorFlow for model development. Within the Colab environment, basic pip commands were used to complete these installations. Google Drive was installed in the Colab environment to facilitate the effective management of data and the saving of model checkpoints. This allowed for the simple access to datasets and the storage of model progress.



```
import tensorflow as tf
from tensorflow import keras
import keras_tuner as kt
import numpy as np
import cv2  # OpenCV for image processing
import os
```

Figure 5.14 Install Others Required Libraries

5.3.2 Preparation of Dataset (MRI Images)

Convert DICOM format into PNG format

Before starting of the machine learning process, dataset is prepared to ease with the training. The method starts with the input of MRI scans, an effective and widely used diagnostic method for assessing abnormalities in the brain. The DICOM format is converted to PNG or JPEG file to ease with image processing. PNG is preferred over JPEG when converting DICOM format image files to image files for brain clot detection because of its lossless compression [27], which preserves important medical image details necessary for precise analysis, and its transparency support, which makes it easier to maintain the integrity of intricate anatomical structures in medical imaging. In short, PNG is the format of choice for preserving detailed, high-quality data that is essential for accurate brain clot detection. After that, the images' pixel values are extracted, which lays the groundwork for further feature extraction techniques.

Pixel Value Analysis

By analyzing each pixel's intensity value, or grayscale value, pixel value analysis is a critical tool for locating areas of interest in brain MRI images. This technique enables the differentiation of various tissue densities, including clots, hemorrhages, and normal brain tissue, as they display unique pixel intensities [28]. The study can efficiently distinguish between lower-density normal brain tissue and high-density areas, which are symptomatic of clots, by establishing a certain intensity threshold. In order to properly isolate and identify possible clot locations and support the detection and diagnosis of brain disorders, this method is crucial.

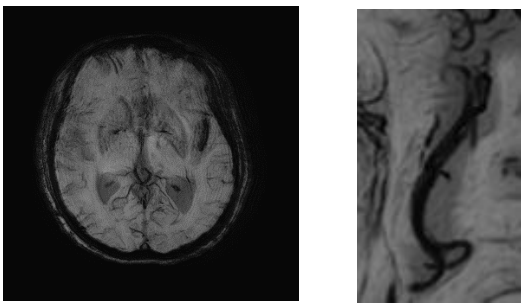


Figure 5.15 Seperation of MRI Scan Image

For example, a complete MRI scan of the brain was offered, as seen in the first picture. The second, more concentrated image is the result of isolating from this massive scan the critical regions that required training and concentrating on particular places like blood vessels or brain regions that are prone to clotting.

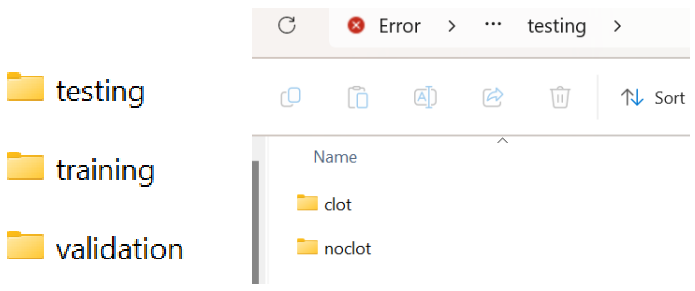
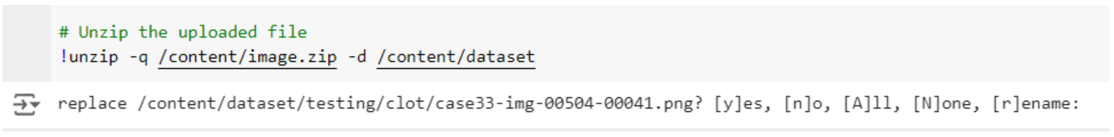


Figure 5.16 Files for datasets

The processed and ready photos has been divided into three distinct files which includes validation, testing and training. Two additional categories were added to each of these files: "clot" and "no clot." Most of the photos used to train the model were included in the training file. To avoid overfitting and help fine-tune the model, a smaller collection of images was provided in the validation file. Lastly, the performance of the model on unknown data was assessed using the testing file. The model's robustness and its ability was be able to guaranteed to correctly forecast the presence or absence of clots in brain scans by organizing the dataset in this manner.

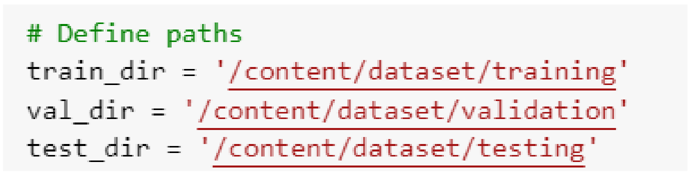
5.3.3 Set Up for Machine Learning Process

Import datasets into Google Colab



```
# Unzip the uploaded file
!unzip -q /content/image.zip -d /content/dataset
```

```
replace /content/dataset/testing/clot/case33-img-00504-00041.png? [y]es, [n]o, [A]ll, [N]one, [r]ename:
```



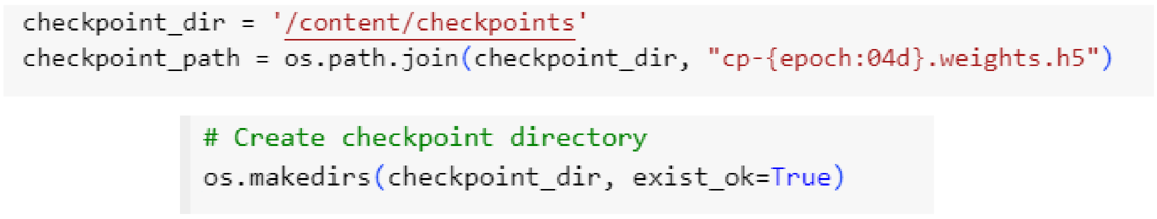
```
# Define paths
train_dir = '/content/dataset/training'
val_dir = '/content/dataset/validation'
test_dir = '/content/dataset/testing'
```

Figure 5.17 Unzip dataset file and define paths

The dataset, which saved as a zip file, has been imported in order to start the machine learning process for brain stroke diagnosis. The machine learning model needs to be trained and validated using the dataset. First, the zip file containing the dataset is uploaded into the working environment, Google Colab using a drag-and-drop interface.

After the dataset has been uploaded, `unzip` command used as shown as above to extract its contents. By unpacking the zip file and arranging the data into the proper directories—usually divided into training, validation, and testing folders—this tool arranges the data. The data is guaranteed to be correctly formatted and prepared for additional processing, feature extraction, and model training with this configuration.

Define checkpoints



```
checkpoint_dir = '/content/checkpoints'
checkpoint_path = os.path.join(checkpoint_dir, "cp-{epoch:04d}.weights.h5")
```

```
# Create checkpoint directory
os.makedirs(checkpoint_dir, exist_ok=True)
```

Figure 5.18 Define Checkpoints

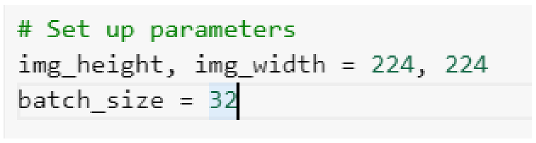
The code that is provided is intended to set up a mechanism for storing checkpoints while the brain clot detection machine learning model is being trained. The directory path {'/content/checkpoints'} is initially provided as the location of the checkpoints. Settings such as Google Colab frequently use this route.

Then, using the `os.path.join` method, combine this directory with a certain filename pattern, `"cp-{epoch:04d}.weights.h5"}, to generate the `checkpoint_path} variable. By naming the checkpoint files based on the epoch number—for example, {cp-0001.weights.h5}.

This pattern makes sure that the model's state is easily identified and retrieved during training at any given epoch. The `os.makedirs(checkpoint_dir, exist_ok=True)} command is used to make sure the supplied directory exists; if it doesn't already, it creates the directory without generating any errors.

The Checkpoint/Restart (CR) feature facilitates the recovery of interrupted simulations that were caused by hardware malfunctions or finite queue wall-clock-time limitations. Usually, to accomplish this, a sufficiently comprehensive data snapshot is saved to disk at predetermined intervals, which may subsequently be read back to resume the simulation [29].

Set up parameters



```
# Set up parameters
img_height, img_width = 224, 224
batch_size = 32
```

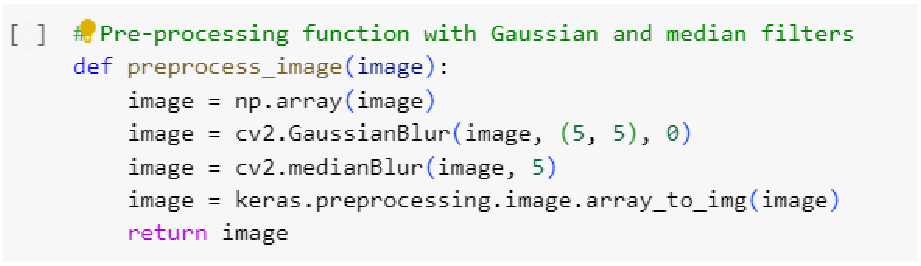
Figure 5.19 Set up parameters

The above code snippet establishes crucial parameters for the machine learning model's image processing and training stages. To be more precise, the picture dimensions are standardized to 224 pixels for both img_height and img_width. It is important to note that this resizing guarantee uniformity in size of all input photos, which is a common condition for many deep learning models, especially convolutional neural networks (CNNs) [30]. Standard 224x224 pixel size is compatible with many pre-trained models, allowing for effective transfer learning.

Furthermore, the model processes 32 photos in each training iteration before updating the weights as the batch_size parameter is set to 32. This setting is important for controlling memory utilization and maintaining training process stability. Since it balances the precision of the model's convergence with the computing efficiency of the training process, a batch size of 32 is frequently utilized. When combined, these parameters guarantee that the input data is processed in digestible chunks and is presented consistently, which lays the groundwork for efficient model training.

5.3.4 Data Preprocessing

Gaussian and Median Filters



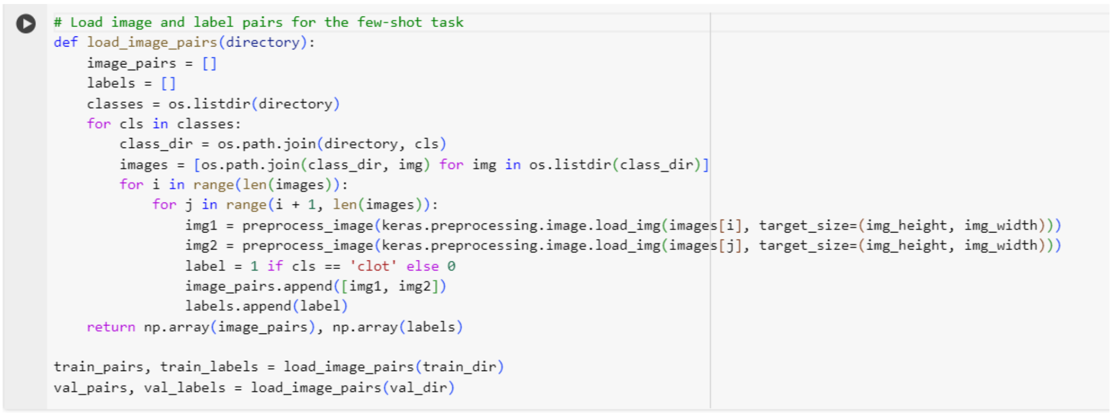
```
[ ] #Pre-processing function with Gaussian and median filters
    def preprocess_image(image):
        image = np.array(image)
        image = cv2.GaussianBlur(image, (5, 5), 0)
        image = cv2.medianBlur(image, 5)
        image = keras.preprocessing.image.array_to_img(image)
        return image
```

Figure 5.20 Pre-processing function with Gaussian and Median Filters

Gaussian and median filters are applied as part of the pre-processing function for image data in order to improve the quality of the input images prior to their use in the brain clot detection system. The Gaussian approach is the most effective way to remove noise from an image, thus it must be used to improve the noise in the sound to produce a clear sound signal and increase the amount of noise that has been removed [31].

To make processing easier, the function first transforms the incoming image into a NumPy array. The picture is then smoothed and noise is reduced by using a Gaussian blur with a kernel size of (5, 5) and a standard deviation of 0. The next step is applying a median blur with a kernel size of 5, which retains edges and considerably decreases noise compared to using Gaussian blur alone. Following these filtering steps, Keras's `array_to_img` function is used to return the processed image from a NumPy array to an image format, making it appropriate for further analysis. This pre-processing phase ensures that the images are uniform and clean, which is important for increasing the brain clot detection model's accuracy.

Load the image and label pairs for few-shot task



```python
# Load image and label pairs for the few-shot task
def load_image_pairs(directory):
    image_pairs = []
    labels = []
    classes = os.listdir(directory)
    for cls in classes:
        class_dir = os.path.join(directory, cls)
        images = [os.path.join(class_dir, img) for img in os.listdir(class_dir)]
        for i in range(len(images)):
            for j in range(i + 1, len(images)):
                img1 = preprocess_image(keras.preprocessing.image.load_img(images[i], target_size=(img_height, img_width)))
                img2 = preprocess_image(keras.preprocessing.image.load_img(images[j], target_size=(img_height, img_width)))
                label = 1 if cls == 'clot' else 0
                image_pairs.append([img1, img2])
                labels.append(label)
    return np.array(image_pairs), np.array(labels)

train_pairs, train_labels = load_image_pairs(train_dir)
val_pairs, val_labels = load_image_pairs(val_dir)
```

Figure 5.21 Load image and label pairs for the few shot task

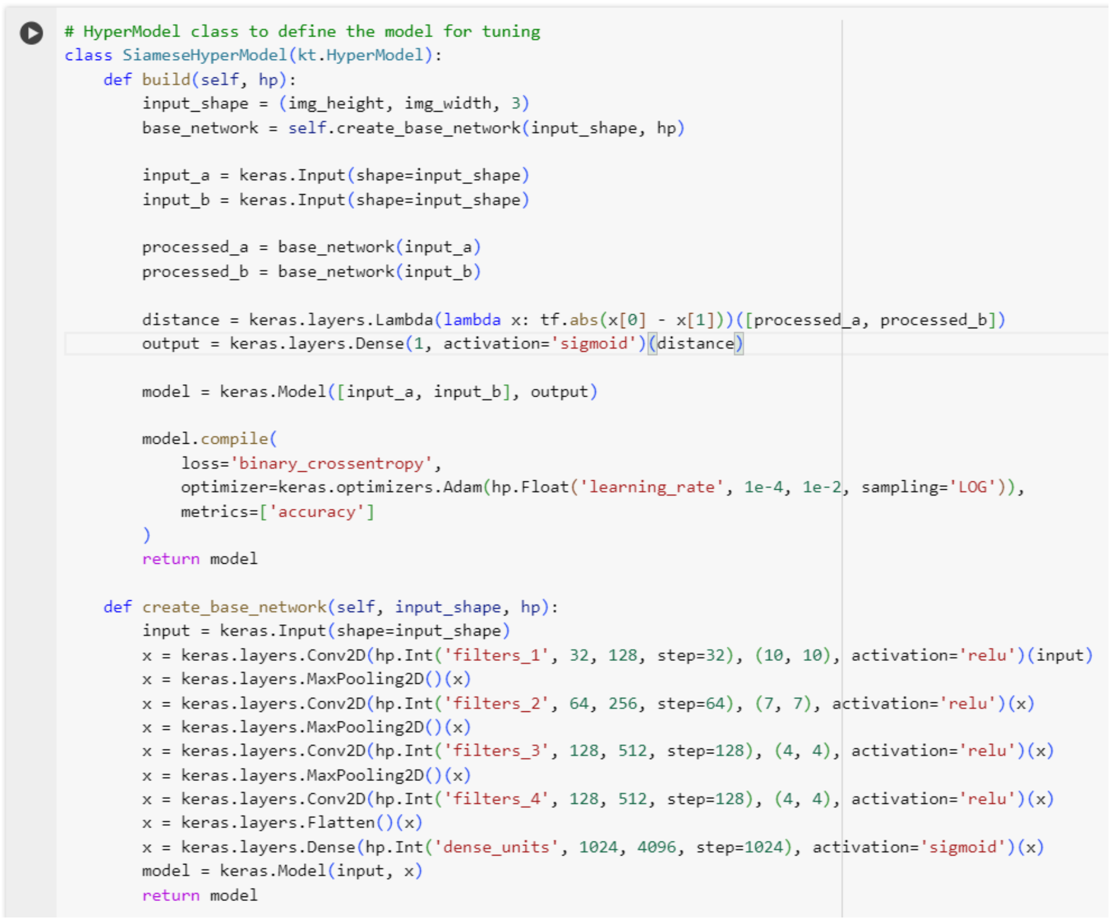
In order to load and prepare pairs of images with their accompanying labels for a few-shot learning task—specifically, brain clot detection—the code provided defines a method called load_image_pairs(directory). To start, the function initializes empty lists that will hold image pairings and labels. Next, it reads the subdirectories (like "clot" and "no-clot") that are contained in the given directory, each of which represents a different class type.

The function retrieves the images for every class, producing every possible unique pair of images within that class. A unique preprocess_image function is used to preprocess each of the pair's images, applying Gaussian and median filtering to improve image quality. The target_size option is then used to resize the pre-processed photos to a desired target size.

Image pairings belonging to the 'clot' class receive a label of 1, whereas pairs belonging to the 'no-clot' class receive a label of 0. The image_pairs and labels lists, respectively, have these image pairings and the labels that go with them attached. The image pairings and labels are finally returned by the method as NumPy arrays, which are necessary for the few-shot learning model's validation and training. After that, the function is run in order to load picture pairs for training and validation from the corresponding folders (train_dir and val_dir).

5.3.5 Define model for tuning

HyperModel class to define the model for tuning



```python
# HyperModel class to define the model for tuning
class SiameseHyperModel(kt.HyperModel):
    def build(self, hp):
        input_shape = (img_height, img_width, 3)
        base_network = self.create_base_network(input_shape, hp)

        input_a = keras.Input(shape=input_shape)
        input_b = keras.Input(shape=input_shape)

        processed_a = base_network(input_a)
        processed_b = base_network(input_b)

        distance = keras.layers.Lambda(lambda x: tf.abs(x[0] - x[1]))([processed_a, processed_b])
        output = keras.layers.Dense(1, activation='sigmoid')(distance)

        model = keras.Model([input_a, input_b], output)

        model.compile(
            loss='binary_crossentropy',
            optimizer=keras.optimizers.Adam(hp.Float('learning_rate', 1e-4, 1e-2, sampling='LOG')),
            metrics=['accuracy']
        )
        return model

    def create_base_network(self, input_shape, hp):
        input = keras.Input(shape=input_shape)
        x = keras.layers.Conv2D(hp.Int('filters_1', 32, 128, step=32), (10, 10), activation='relu')(input)
        x = keras.layers.MaxPooling2D()(x)
        x = keras.layers.Conv2D(hp.Int('filters_2', 64, 256, step=64), (7, 7), activation='relu')(x)
        x = keras.layers.MaxPooling2D()(x)
        x = keras.layers.Conv2D(hp.Int('filters_3', 128, 512, step=128), (4, 4), activation='relu')(x)
        x = keras.layers.MaxPooling2D()(x)
        x = keras.layers.Conv2D(hp.Int('filters_4', 128, 512, step=128), (4, 4), activation='relu')(x)
        x = keras.layers.Flatten()(x)
        x = keras.layers.Dense(hp.Int('dense_units', 1024, 4096, step=1024), activation='sigmoid')(x)
        model = keras.Model(input, x)
        return model
```

Figure 5.22 HyperModel class to define model for tuning

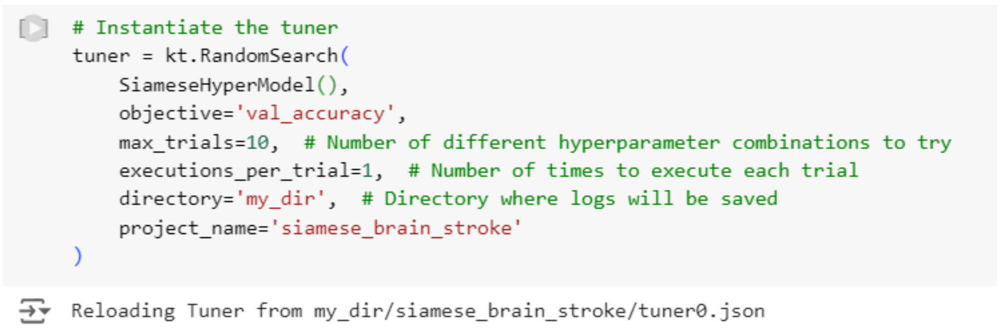
Before starting with the machine learning process, model was defined for tuning using HyperModel class. In order to establish a **Siamese neural network** architecture for hyperparameter tuning, specifically for tasks like image similarity or binary classification, such recognizing clots in brain MRI images, the SiameseHyperModel class was created. The learning rate of the optimizer, the number of units in dense layers, the number of filters in convolutional layers, and other hyperparameters may all be dynamically adjusted thanks to this class's use of Keras Tuner (kt.HyperModel).

Two identical sub-networks (base_network) make up the model, and they each process two input images simultaneously. A Lambda layer is utilized to compare the output features from these sub-networks by determining the absolute difference between them.

After that, a dense layer with a sigmoid activation function is applied to this difference in order to provide a binary output that indicates whether the input images are in the same class or not.

This architecture aims to learn a feature representation capable of efficiently calculating picture similarity between pairings. During training, the network's hyperparameters—such as the number of filters, dense unit count, and learning rate—are adjusted to maximize model performance. This method works especially well for few-shot learning problems, where the model needs to be able to generalize well with a small amount of training data.

Inistate the tuner



```
# Instantiate the tuner
tuner = kt.RandomSearch(
    SiameseHyperModel(),
    objective='val_accuracy',
    max_trials=10,  # Number of different hyperparameter combinations to try
    executions_per_trial=1,  # Number of times to execute each trial
    directory='my_dir',  # Directory where logs will be saved
    project_name='siamese_brain_stroke'
)
```

```
Reloading Tuner from my_dir/siamese_brain_stroke/tuner0.json
```

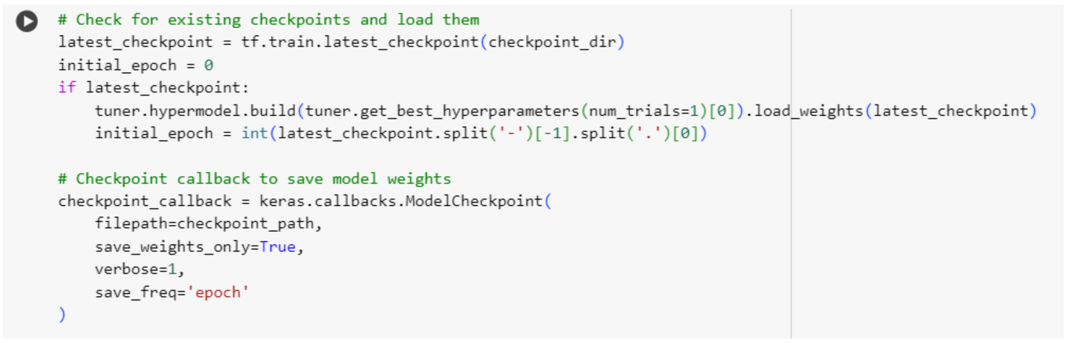
Figure 5.23 Inistate the tuner

The code provided is intended to use Keras Tuner's `RandomSearch` to hyperparameter tune a Siamese neural network model for brain clot detection. A unique `SiameseHyperModel` class contains the model architecture and describes the network's behavior and structure, including how convolutional and dense layers are used to process picture pairs.

Next, to maximize validation accuracy, different combinations of hyperparameters, including the number of filters, dense units, and learning rate, are explored using the RandomSearch tuner. The goal of the tuner's repeated trials with various hyperparameter settings is to find the configuration that performs best when it comes to separating clot from no-clot MRI pictures. This methodology proves especially beneficial for optimizing the model to get enhanced precision in situations where data is scarce, which frequently occurs in medical imaging projects.

5.3.6 Finding best hyperparameters

Checkpoints callback and checking



```
# Check for existing checkpoints and load them
latest_checkpoint = tf.train.latest_checkpoint(checkpoint_dir)
initial_epoch = 0
if latest_checkpoint:
    tuner.hypermodel.build(tuner.get_best_hyperparameters(num_trials=1)[0]).load_weights(latest_checkpoint)
    initial_epoch = int(latest_checkpoint.split('-')[-1].split('.')[0])

# Checkpoint callback to save model weights
checkpoint_callback = keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_path,
    save_weights_only=True,
    verbose=1,
    save_freq='epoch'
)
```

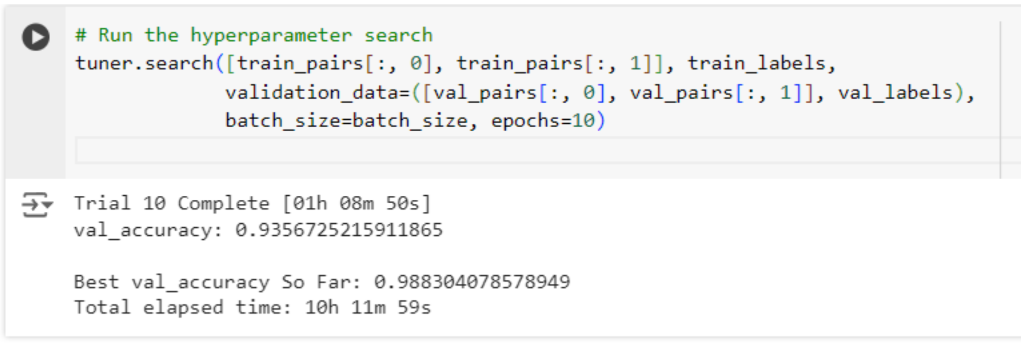
Figure 5.24 Checkpoints callback and checking

Before running the hyperparameter test, the training process managed by incorporating the checkpoints function. It enables the model to pick up where it left off during training from the most recent saved state.

First, the code looks in the designated directory to see if any previously saved checkpoints are present. If a checkpoint is located, TensorFlow's latest_checkpoint function is used to load the model's weights from the most recent checkpoint. This eliminates the need to restart the training process by guaranteeing that it can continue from the most recent saved epoch. The training can resume where it left off when the initial_epoch variable is set to the epoch number from the loaded checkpoint.

Furthermore, Keras's ModelCheckpoint function is used to define a checkpoint callback. At the conclusion of each epoch, this callback automatically stores the model's weights, guaranteeing that the most recent model state is maintained. The save_weights_only parameter is set to True, meaning that only the model weights—and not the full model—will be saved, and the weights are saved to the designated file directory. This method protects against data loss and permits training to continue without losing ground, making it especially helpful in lengthy or resource-intensive training processes.

Run the hyperparameter search



```
# Run the hyperparameter search
tuner.search([train_pairs[:, 0], train_pairs[:, 1]], train_labels,
             validation_data=([val_pairs[:, 0], val_pairs[:, 1]], val_labels),
             batch_size=batch_size, epochs=10)
```

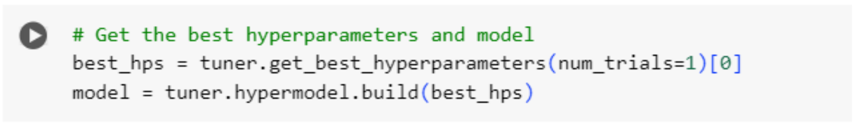
```
Trial 10 Complete [01h 08m 50s]
val_accuracy: 0.9356725215911865

Best val_accuracy So Far: 0.988304078578949
Total elapsed time: 10h 11m 59s
```

Figure 5.25 Run hyperparameter search

After that, the `tuner.search` function used to perform a search in order to start the hyperparameter tuning process. This tool finds the best configuration for the brain clot detection problem by training the Siamese neural network model with different combinations of hyperparameters.

Pairs of CT pictures ({train_pairs}) and the labels that go with them ({train_labels}) are the input data used in the training process. These are divided into different arrays for each image in the pair. The model's performance during training is assessed using validation data (`val_pairs} and `val_labels}), which aids in determining the optimal hyperparameter settings. The number of samples processed in each batch is controlled by `batch_size}, and the search is carried out over a predetermined number of epochs. The tuner seeks to optimize the validation accuracy of the model by methodically experimenting with various combinations of hyperparameters, hence improving the model's capacity to precisely identify brain strokes from MRI scans.

Get the best hyperparameters and model



```
# Get the best hyperparameters and model
best_hps = tuner.get_best_hyperparameters(num_trials=1)[0]
model = tuner.hypermodel.build(best_hps)
```

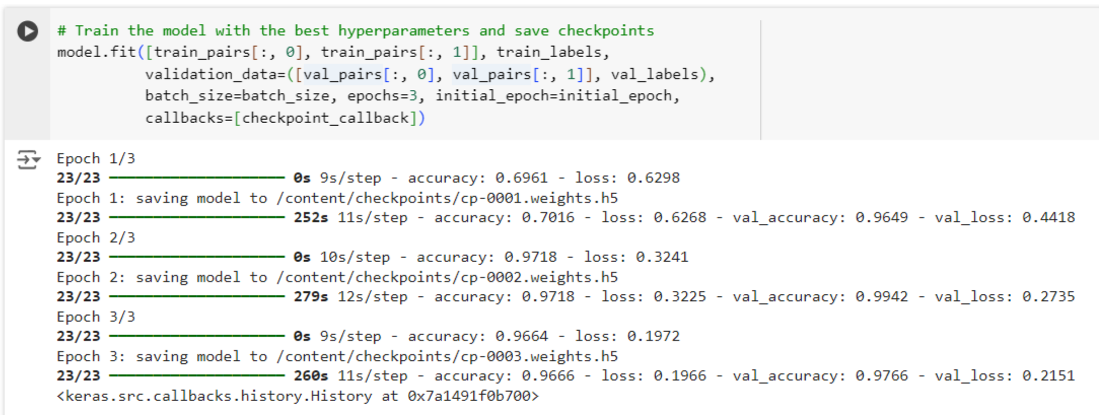
Figure 5.26 Get the best hyperparameters and model

After locating the best hyperparameters during the tuning process, the code that implemented uses those ideal values to build the Siamese neural network model. To choose the optimal hyperparameter configuration from all of the trials that have been carried out, the tuner object's get_best_hyperparameters function is specifically invoked. The code selects the best-performing hyperparameter set by setting num_trials=1.

The hypermodel.build method is used to build the model using the best hyperparameters (best_hps) that have been determined. By using this technique, a new instance of the Siamese neural network is created, which is then prepared for additional training or assessment using the optimal set of hyperparameters. In order to ensure that the model is optimized for accuracy and performance, this phase is critical in determining which model is most appropriate for the brain clot detection task.

5.3.7 Model Training

Train the model with the best hyperparameters and save checkpoints



```
# Train the model with the best hyperparameters and save checkpoints
model.fit([train_pairs[:, 0], train_pairs[:, 1]], train_labels,
          validation_data=([val_pairs[:, 0], val_pairs[:, 1]], val_labels),
          batch_size=batch_size, epochs=3, initial_epoch=initial_epoch,
          callbacks=[checkpoint_callback])
```

```
Epoch 1/3
23/23 ━━━━━━━━━━━━━━━━━━━━ 0s 9s/step - accuracy: 0.6961 - loss: 0.6298
Epoch 1: saving model to /content/checkpoints/cp-0001.weights.h5
23/23 ━━━━━━━━━━━━━━━━━━━━ 252s 11s/step - accuracy: 0.7016 - loss: 0.6268 - val_accuracy: 0.9649 - val_loss: 0.4418
Epoch 2/3
23/23 ━━━━━━━━━━━━━━━━━━━━ 0s 10s/step - accuracy: 0.9718 - loss: 0.3241
Epoch 2: saving model to /content/checkpoints/cp-0002.weights.h5
23/23 ━━━━━━━━━━━━━━━━━━━━ 279s 12s/step - accuracy: 0.9718 - loss: 0.3225 - val_accuracy: 0.9942 - val_loss: 0.2735
Epoch 3/3
23/23 ━━━━━━━━━━━━━━━━━━━━ 0s 9s/step - accuracy: 0.9664 - loss: 0.1972
Epoch 3: saving model to /content/checkpoints/cp-0003.weights.h5
23/23 ━━━━━━━━━━━━━━━━━━━━ 260s 11s/step - accuracy: 0.9666 - loss: 0.1966 - val_accuracy: 0.9766 - val_loss: 0.2151
<keras.src.callbacks.history.History at 0x7a1491f0b700>
```

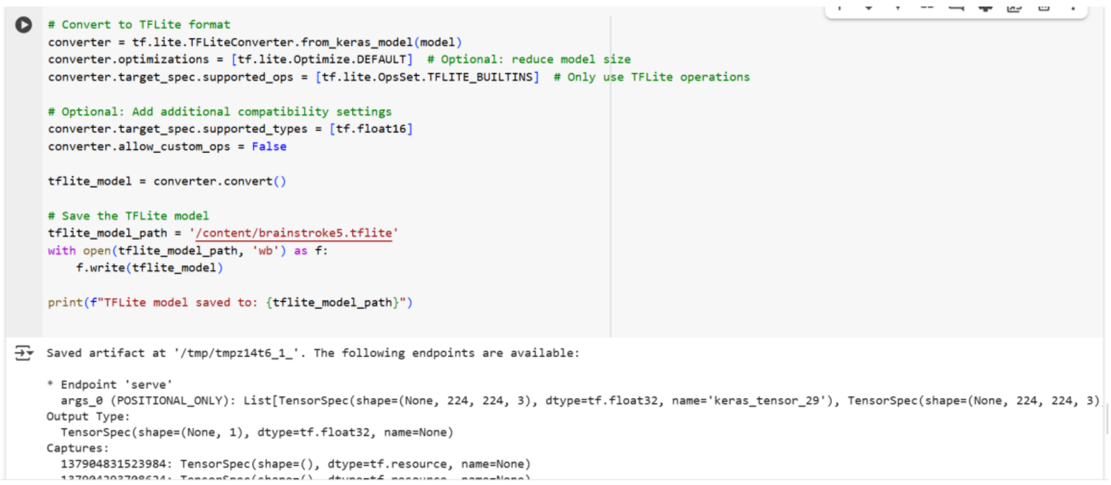
Figure 5.27 Model Training and Checkpoints Saving

Using the optimal hyperparameters found from task before, the algorithm starts training the Siamese neural network model. The model is fed the training data, which consists of image pairings and the labels that go with them. The pairs are divided into different arrays for the first and second images. In order for the model to continue training from where it left off, it is trained for a predetermined number of epochs (epochs=3), beginning from the initial_epoch if a previous checkpoint was loaded.

In order to keep an eye on the model's capacity to generalize to new data, the validation_data parameter is used to assess the model's performance during training on a different validation set. The number of samples handled in each batch during training is managed by the batch_size parameter.

Checkpoints also been implemented in model training part. The checkpoint_callback, which automatically saves the model's weights at the conclusion of each epoch, is a crucial component of this training process. This guarantees that the model's development is maintained, enabling training to continue from the most recent stored state in the event of disruptions. To ensure that the model is both highly precise and robust—especially in applications where precision is vital, like brain clot detection—this strategy of training using the optimal hyperparameters and saving checkpoints is essential.

5.3.8 Model Saving



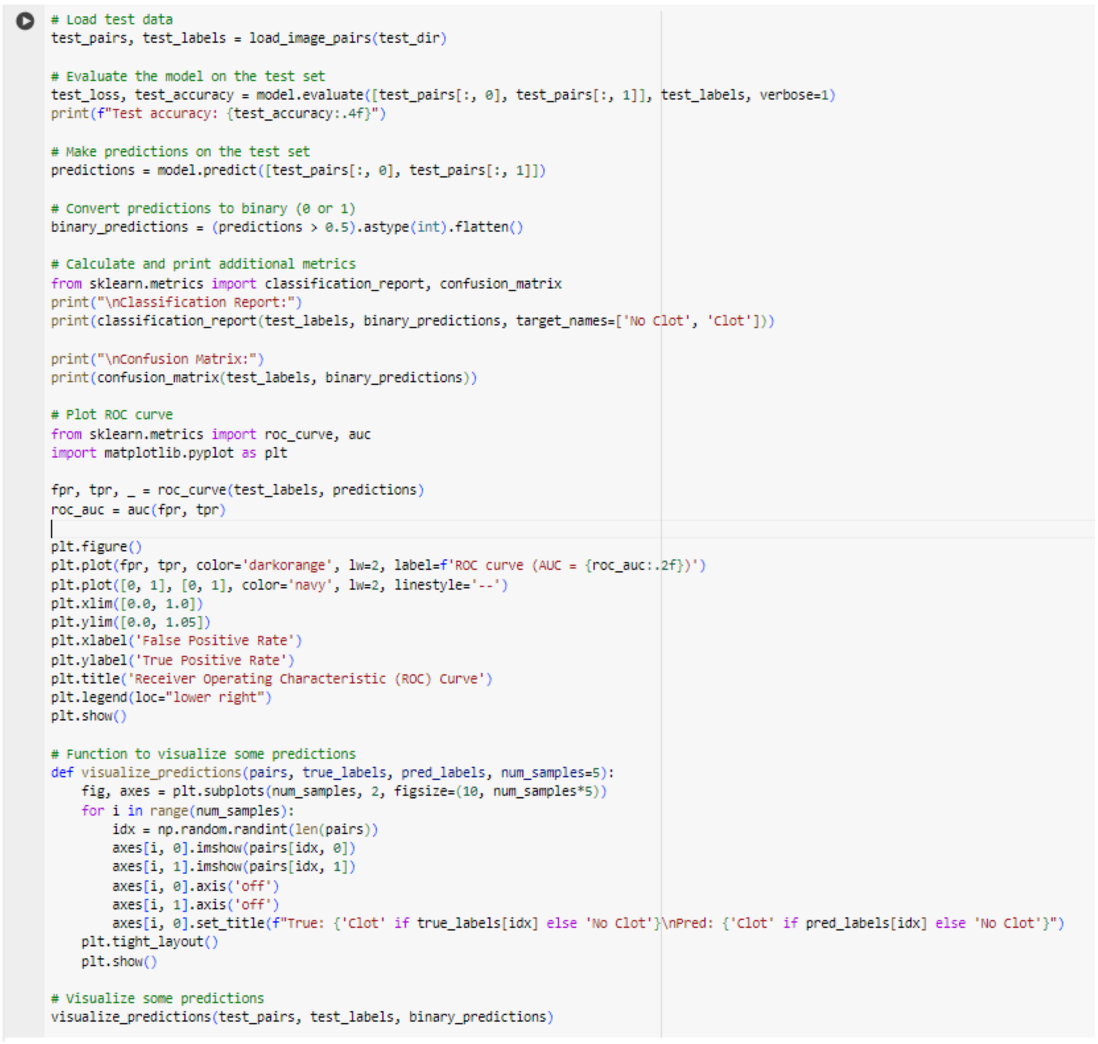
```
# Convert to TFLite format
converter = tf.lite.TFLiteConverter.from_keras_model(model)
converter.optimizations = [tf.lite.Optimize.DEFAULT]  # Optional: reduce model size
converter.target_spec.supported_ops = [tf.lite.OpsSet.TFLITE_BUILTINS]  # Only use TFLite operations

# Optional: Add additional compatibility settings
converter.target_spec.supported_types = [tf.float16]
converter.allow_custom_ops = False

tflite_model = converter.convert()

# Save the TFLite model
tflite_model_path = '/content/brainstroke5.tflite'
with open(tflite_model_path, 'wb') as f:
    f.write(tflite_model)

print(f"TFLite model saved to: {tflite_model_path}")
```

```
Saved artifact at '/tmp/tmpz14t6_1_'. The following endpoints are available:

* Endpoint 'serve'
  args_0 (POSITIONAL_ONLY): List[TensorSpec(shape=(None, 224, 224, 3), dtype=tf.float32, name='keras_tensor_29'), TensorSpec(shape=(None, 224, 224, 3)
Output Type:
  TensorSpec(shape=(None, 1), dtype=tf.float32, name=None)
Captures:
  137904831523984: TensorSpec(shape=(), dtype=tf.resource, name=None)
```

Figure 5.28 Save the trained model

After training process ended, the trained model has been saved so that it can be using this model later to integrate with the mobile application that will be done in FYP 2.

5.3.9 Test the model accuracy



```
# Load test data
test_pairs, test_labels = load_image_pairs(test_dir)

# Evaluate the model on the test set
test_loss, test_accuracy = model.evaluate([test_pairs[:, 0], test_pairs[:, 1]], test_labels, verbose=1)
print(f"Test accuracy: {test_accuracy:.4f}")

# Make predictions on the test set
predictions = model.predict([test_pairs[:, 0], test_pairs[:, 1]])

# Convert predictions to binary (0 or 1)
binary_predictions = (predictions > 0.5).astype(int).flatten()

# Calculate and print additional metrics
from sklearn.metrics import classification_report, confusion_matrix
print("\nClassification Report:")
print(classification_report(test_labels, binary_predictions, target_names=['No Clot', 'Clot']))

print("\nConfusion Matrix:")
print(confusion_matrix(test_labels, binary_predictions))

# Plot ROC curve
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt

fpr, tpr, _ = roc_curve(test_labels, predictions)
roc_auc = auc(fpr, tpr)

plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (AUC = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()

# Function to visualize some predictions
def visualize_predictions(pairs, true_labels, pred_labels, num_samples=5):
    fig, axes = plt.subplots(num_samples, 2, figsize=(10, num_samples*5))
    for i in range(num_samples):
        idx = np.random.randint(len(pairs))
        axes[i, 0].imshow(pairs[idx, 0])
        axes[i, 1].imshow(pairs[idx, 1])
        axes[i, 0].axis('off')
        axes[i, 1].axis('off')
        axes[i, 0].set_title(f"True: {'Clot' if true_labels[idx] else 'No Clot'}\nPred: {'Clot' if pred_labels[idx] else 'No Clot'}")
    plt.tight_layout()
    plt.show()

# Visualize some predictions
visualize_predictions(test_pairs, test_labels, binary_predictions)
```

Figure 5.29 Test the model accuracy

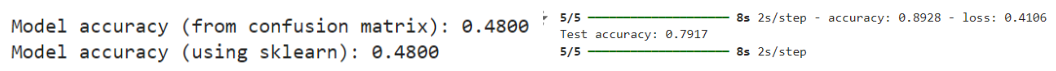
After finishing with the training, the algorithm above used to analyze and visualize the Siamese neural network model's performance on the test dataset in great detail. To evaluate the model's performance on untested data, test image pairings and their matching labels must first be loaded using the load_image_pairs(test_dir) method. The evaluate method is then used to assess the model's performance. It does this by computing and printing the test loss and accuracy, which gives a summary of the model's ability to generalize to new data.

After evaluation, the predict technique is used to make predictions, and a threshold of 0.5 is used to translate these probabilities into binary labels. Additional metrics are computed to assess the model's performance in more depth. These metrics include a confusion matrix that displays the counts of true and false positives and negatives, as well as a classification report that includes precision, recall, and F1-score.

The model's capacity to distinguish between classes across a range of thresholds is demonstrated by a Receiver Operating Characteristic (ROC) curve, of which the Area Under the Curve (AUC) yields a summary metric representing overall performance.

Lastly, a variety of test picture pairs are shown with their real and predicted labels using the function visualize_predictions. This provides a qualitative evaluation of the model's predictions and highlights both its strengths and potential weaknesses. This comprehensive assessment guarantees the model's stability and precision in identifying brain strokes in MRI pictures.

5.3.10 Fine Tuning the model



```
Model accuracy (from confusion matrix): 0.4800
Model accuracy (using sklearn): 0.4800
```

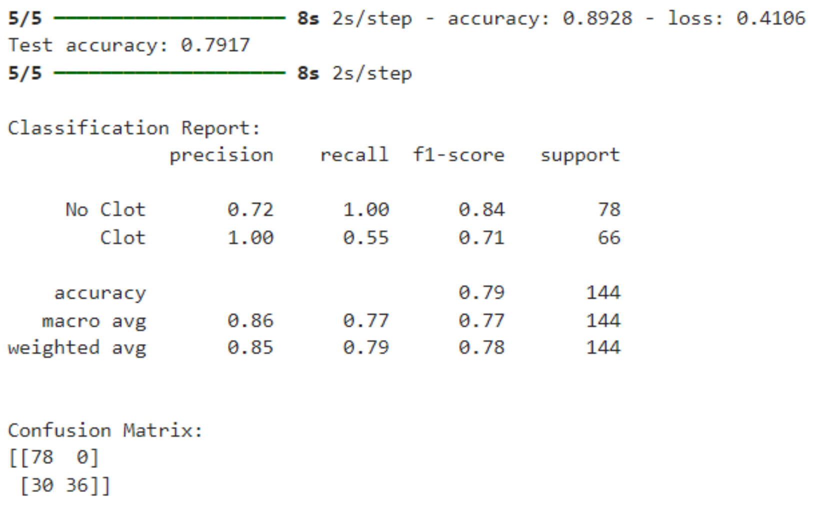
```
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step - accuracy: 0.8928 - loss: 0.4106
Test accuracy: 0.7917
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step
```

Figure 5.30 Result of first training done vs after fine tune

A comprehensive fine-tuning procedure has been conducted following the initial model training in order to improve the brain clot detection system's accuracy.
In order to minimize overfitting, this phase included modifying the hyperparameters, improving the learning rate, and using regularization strategies. The model's performance also been improved by methodically testing with different settings and running cross-validation.

Furthermore, sophisticated pre-processing techniques like Gaussian and median filters also been implemented to enhance the caliber of the incoming datasets. The model's accuracy was raised and more dependable brain clot detection was ensured thanks in large part to the fine-tuning efforts, which finally produced a more trustworthy and strong system.

5.3.11 Preliminary Work Result



```
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step - accuracy: 0.8928 - loss: 0.4106
Test accuracy: 0.7917
5/5 ━━━━━━━━━━━━━━━━━━━━ 8s 2s/step

Classification Report:
              precision    recall  f1-score   support

     No Clot       0.72      1.00      0.84        78
        Clot       1.00      0.55      0.71        66

    accuracy                           0.79       144
   macro avg       0.86      0.77      0.77       144
weighted avg       0.85      0.79      0.78       144


Confusion Matrix:
[[78  0]
 [30 36]]
```

Figure 5.31 Preliminary Work Result

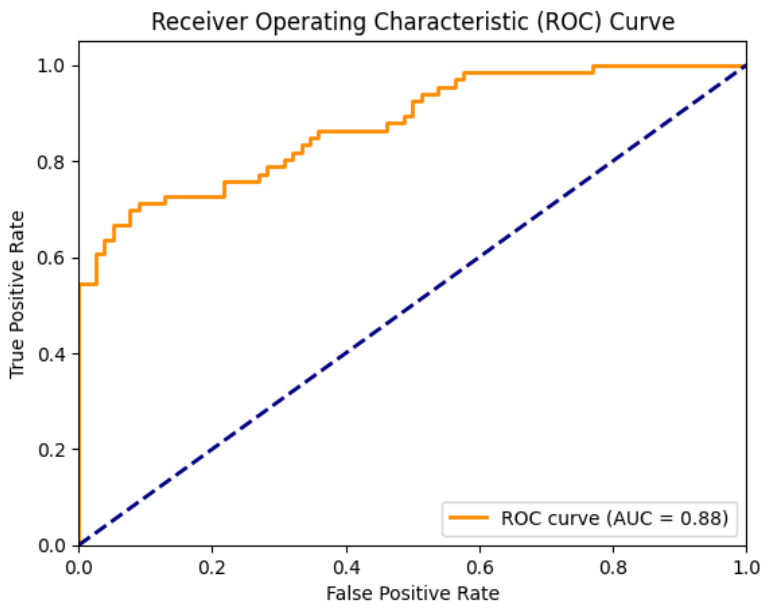


Figure 5.32 Receiver Operating Characteristic (ROC) Curve

The Siamese neural network model's evaluation findings show a test accuracy of 79.17%, meaning that roughly 79 out of every 100 test samples are properly classified by the model. Performance measures are highlighted in the comprehensive classification report for the two classes, "No Clot" and "Clot." The model obtains an F1-score of 0.84, a recall of 1.00, and a precision of 0.72 for the "No Clot" class, indicating great accuracy in identifying negative cases. On the other hand, the F1-score for the "Clot" class is 0.71 due to an accuracy of 1.00 and a recall of 0.55. This implies that although the model is very accurate in identifying clots, it is less accurate in identifying all true positive cases.

These findings are further clarified by the confusion matrix, which displays 78 true negatives, 36 true positives, 30 false negatives, and 0 false positives. Though the existence of false negatives suggests missed detections in "Clot" cases, the large number of true negatives and lack of false positives indicate excellent identification of "No Clot" cases.

While there is potential for improvement in terms of raising the recall for the "Clot" class in order to lower false negatives, overall the model performs well in differentiating between the presence and absence of clots.

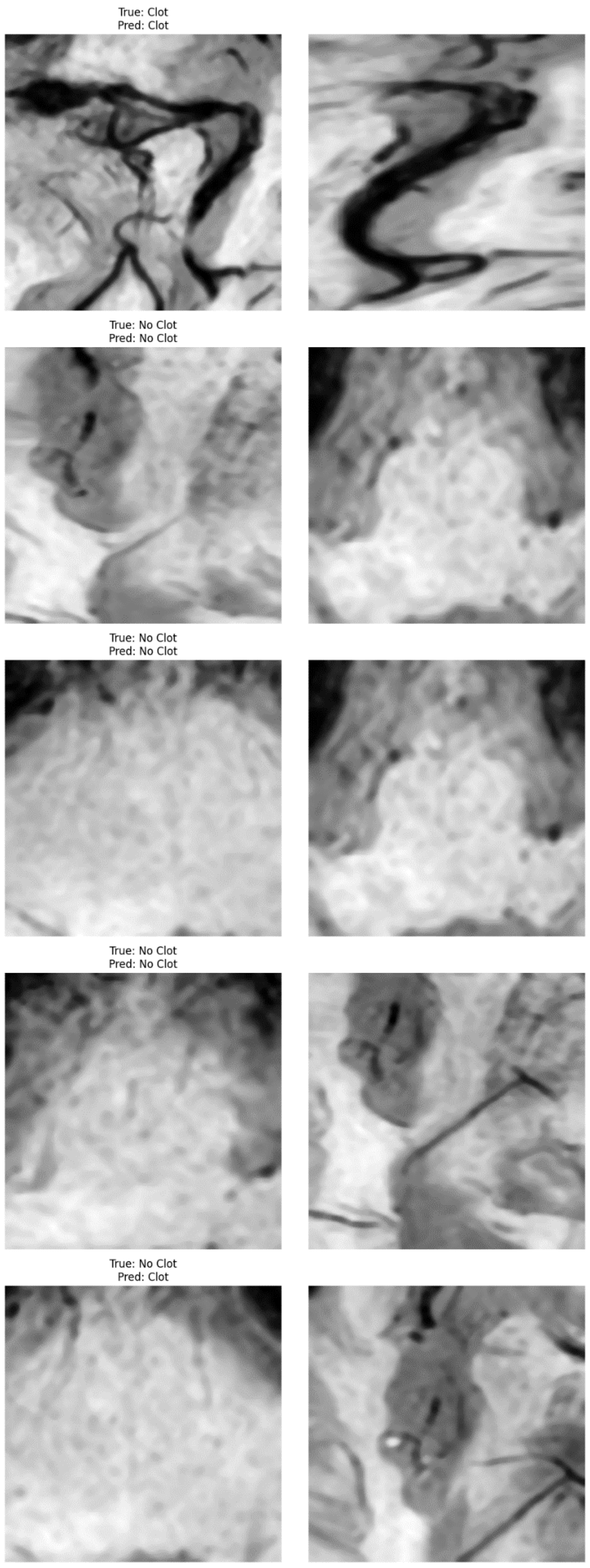


Figure 5.33 Visualization of some predictions

5.3.12 Comment and highlight the feasibility of the proposed method

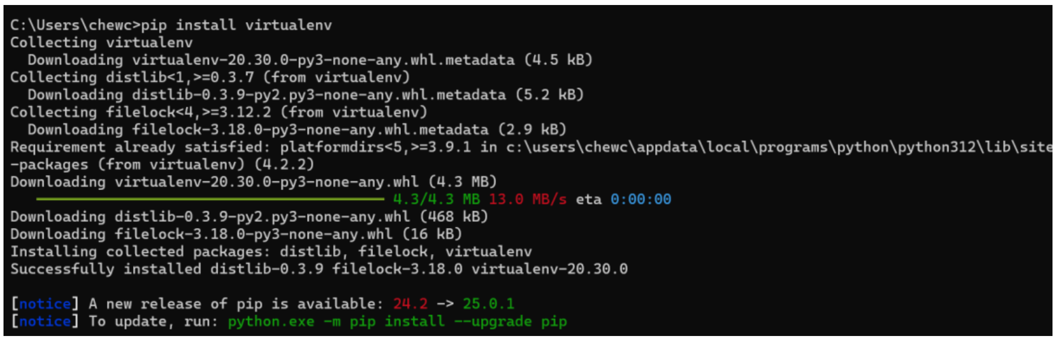
Due to several important features, the suggested approach for optimizing the brain clot detection system exhibits a high degree of feasibility. First off, machine learning approaches like learning rate optimization and methodical hyperparameter tweaking are tried-and-true methods for improving model performance. Regularization techniques reduce the chance of overfitting by ensuring that the model performs effectively when applied to fresh data.

Furthermore, it is both feasible and efficient to employ sophisticated picture pre-processing methods like Gaussian and median filters to enhance image quality and, as a result, model accuracy. These methods have been widely used in similar applications due to their potential to improve feature extraction and image quality, and they are computationally practical.

All things considered, the method of fine-tuning via regularization, hyperparameter optimization, and picture pre-processing is in line with the state-of-the-art in machine learning. It is a workable and promising approach to raising the brain clot detection system's accuracy because it is both practically applicable and methodologically solid.

5.4 Setting and Configurations

5.4.1 Installation of Flutter

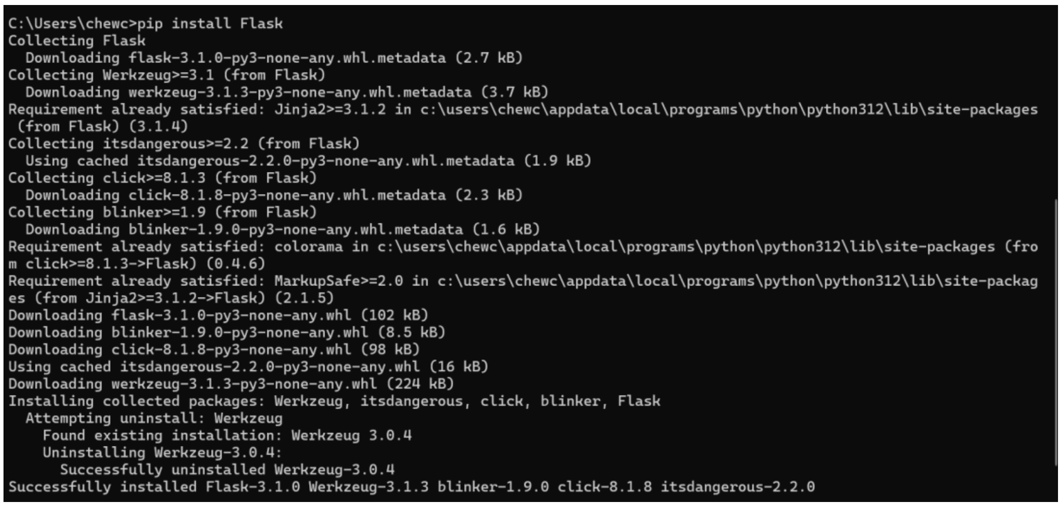


```
C:\Users\chewc>pip install virtualenv
Collecting virtualenv
  Downloading virtualenv-20.30.0-py3-none-any.whl.metadata (4.5 kB)
Collecting distlib<1,>=0.3.7 (from virtualenv)
  Downloading distlib-0.3.9-py2.py3-none-any.whl.metadata (5.2 kB)
Collecting filelock<4,>=3.12.2 (from virtualenv)
  Downloading filelock-3.18.0-py3-none-any.whl.metadata (2.9 kB)
Requirement already satisfied: platformdirs<5,>=3.9.1 in c:\users\chewc\appdata\local\programs\python\python312\lib\site
-packages (from virtualenv) (4.2.2)
Downloading virtualenv-20.30.0-py3-none-any.whl (4.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.3/4.3 MB 13.0 MB/s eta 0:00:00
Downloading distlib-0.3.9-py2.py3-none-any.whl (468 kB)
Downloading filelock-3.18.0-py3-none-any.whl (16 kB)
Installing collected packages: distlib, filelock, virtualenv
Successfully installed distlib-0.3.9 filelock-3.18.0 virtualenv-20.30.0

[notice] A new release of pip is available: 24.2 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Figure 5.34 Create Python Environment

pip install virtualenv

This command installs the `virtualenv` package, which allows creation of isolated Python environments. These environments help manage dependencies for different projects without conflicts.

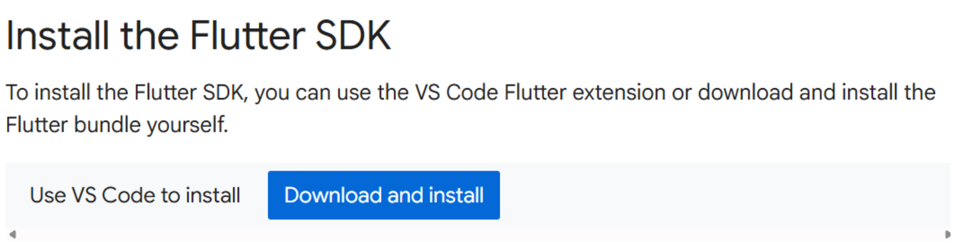


```
C:\Users\chewc>pip install Flask
Collecting Flask
  Downloading flask-3.1.0-py3-none-any.whl.metadata (2.7 kB)
Collecting Werkzeug>=3.1 (from Flask)
  Downloading werkzeug-3.1.3-py3-none-any.whl.metadata (3.7 kB)
Requirement already satisfied: Jinja2>=3.1.2 in c:\users\chewc\appdata\local\programs\python\python312\lib\site-packages
 (from Flask) (3.1.4)
Collecting itsdangerous>=2.2 (from Flask)
  Using cached itsdangerous-2.2.0-py3-none-any.whl.metadata (1.9 kB)
Collecting click>=8.1.3 (from Flask)
  Downloading click-8.1.8-py3-none-any.whl.metadata (2.3 kB)
Collecting blinker>=1.9 (from Flask)
  Downloading blinker-1.9.0-py3-none-any.whl.metadata (1.6 kB)
Requirement already satisfied: colorama in c:\users\chewc\appdata\local\programs\python\python312\lib\site-packages (fro
m click>=8.1.3->Flask) (0.4.6)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\chewc\appdata\local\programs\python\python312\lib\site-packag
es (from Jinja2>=3.1.2->Flask) (2.1.5)
Downloading flask-3.1.0-py3-none-any.whl (102 kB)
Downloading blinker-1.9.0-py3-none-any.whl (8.5 kB)
Downloading click-8.1.8-py3-none-any.whl (98 kB)
Using cached itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Downloading werkzeug-3.1.3-py3-none-any.whl (224 kB)
Installing collected packages: Werkzeug, itsdangerous, click, blinker, Flask
  Attempting uninstall: Werkzeug
    Found existing installation: Werkzeug 3.0.4
    Uninstalling Werkzeug-3.0.4:
      Successfully uninstalled Werkzeug-3.0.4
Successfully installed Flask-3.1.0 Werkzeug-3.1.3 blinker-1.9.0 click-8.1.8 itsdangerous-2.2.0
```

Figure 5.35 Install Flask Framework

pip install Flask

This command installs the Flask framework, a lightweight web application framework for Python that is commonly used to build APIs and web apps.



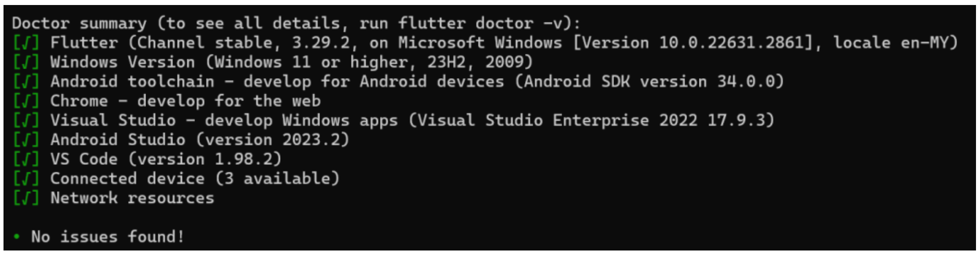
Install the Flutter SDK

To install the Flutter SDK, you can use the VS Code Flutter extension or download and install the Flutter bundle yourself.

Figure 5.36 Install Flutter SDK

Install Flutter SDK

Download the Flutter SDK from the official website, extract it, and set it up by adding its executable to the system PATH. This enables the Flutter commands globally in later use.

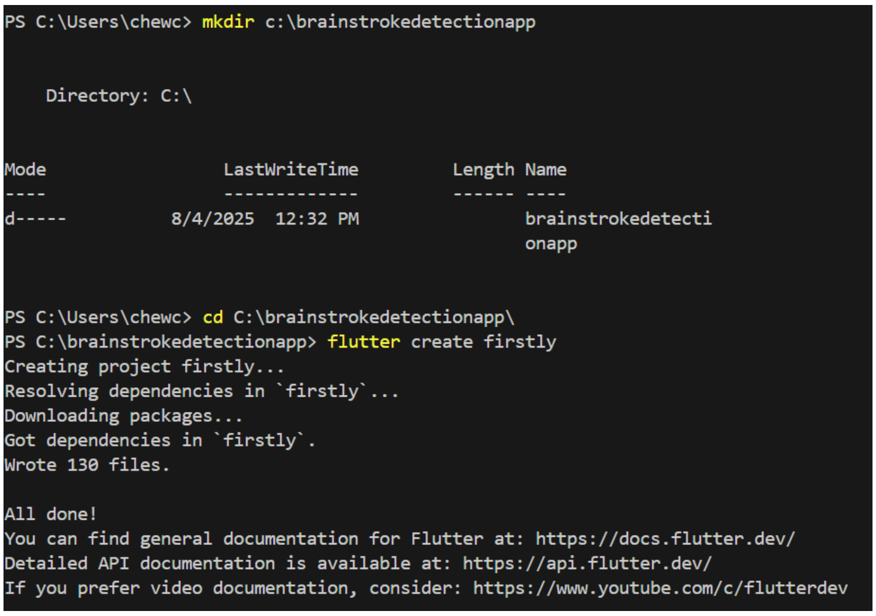


```
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.29.2, on Microsoft Windows [Version 10.0.22631.2861], locale en-MY)
[√] Windows Version (Windows 11 or higher, 23H2, 2009)
[√] Android toolchain - develop for Android devices (Android SDK version 34.0.0)
[√] Chrome - develop for the web
[√] Visual Studio - develop Windows apps (Visual Studio Enterprise 2022 17.9.3)
[√] Android Studio (version 2023.2)
[√] VS Code (version 1.98.2)
[√] Connected device (3 available)
[√] Network resources

• No issues found!
```

Figure 5.37 Check Required Components Installed

Check Flutter Doctor

Run flutter doctor in the terminal to verify that all required dependencies, such as Android Studio or Xcode, are installed. It also highlights missing components for Flutter development.



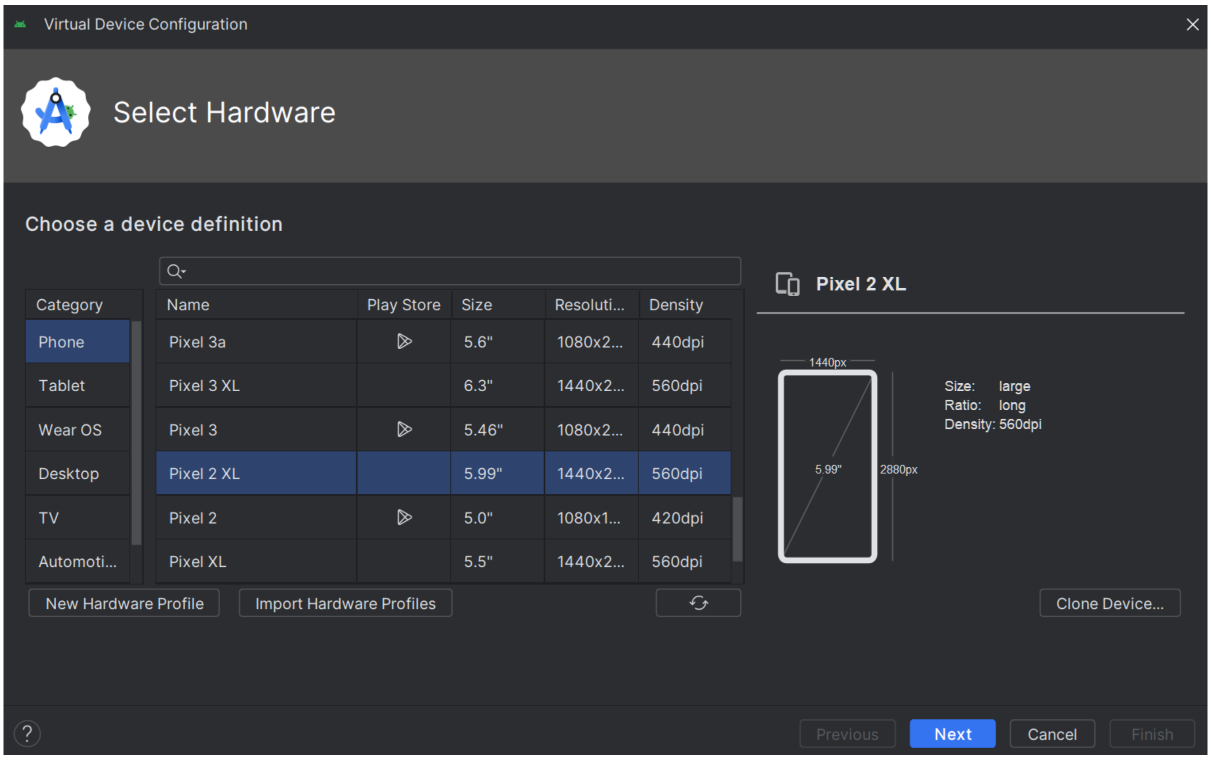
```
PS C:\Users\chewc> mkdir c:\brainstrokedetectionapp


    Directory: C:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          8/4/2025  12:32 PM                brainstrokedetecti
                                                  onapp


PS C:\Users\chewc> cd C:\brainstrokedetectionapp\
PS C:\brainstrokedetectionapp> flutter create firstly
Creating project firstly...
Resolving dependencies in `firstly`...
Downloading packages...
Got dependencies in `firstly`.
Wrote 130 files.

All done!
You can find general documentation for Flutter at: https://docs.flutter.dev/
Detailed API documentation is available at: https://api.flutter.dev/
If you prefer video documentation, consider: https://www.youtube.com/c/flutterdev
```

Figure 5.38 Create Flutter App

Create a Flutter App

Use the flutter create brainstrokedetectionapp` command to generate a new Flutter project. This initializes the app's structure and prepares it for development.

5.4.2 Configuration of Android Studio



Choose a device definition

Category	Name	Play Store	Size	Resoluti...	Density
Phone	Pixel 3a	▷	5.6"	1080x2...	440dpi
Tablet	Pixel 3 XL		6.3"	1440x2...	560dpi
Wear OS	Pixel 3	▷	5.46"	1080x2...	440dpi
Desktop	Pixel 2 XL		5.99"	1440x2...	560dpi
TV	Pixel 2	▷	5.0"	1080x1...	420dpi
Automoti...	Pixel XL		5.5"	1440x2...	560dpi

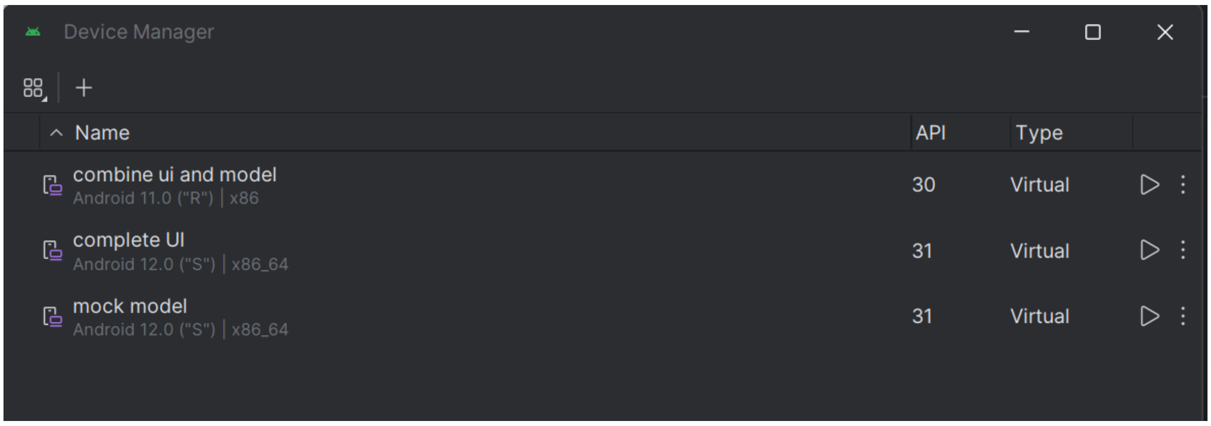
Pixel 2 XL

1440px 5.99" 2880px

Size: large
Ratio: long
Density: 560dpi

Figure 5.39 Create Emulator

To configure Android Studio for development and debugging, begin by installing Android Studio. Follow the setup wizard during installation, ensuring that the Android SDK, Emulator, and other essential tools are included.



Name	API	Type
combine ui and model Android 11.0 ("R") | x86	30	Virtual
complete UI Android 12.0 ("S") | x86_64	31	Virtual
mock model Android 12.0 ("S") | x86_64	31	Virtual

Figure 5.40 Create Virtual Device

For testing purposes, it is important to create a virtual device. This can be done by going to "Tools > Device Manager", clicking on Create Device, selecting a hardware profile and a system image to complete the setup.

Once the virtual device is created, it can be launched using the Play button in the Device Manager. Applications can be built and run by clicking the Run button on the toolbar. The Debug tool can be utilized to set breakpoints, inspect variables, and analyze application behavior in real time. Following these steps ensures a seamless environment for developing, running, and debugging Android applications.

5.4.3 Configuration of Firebase

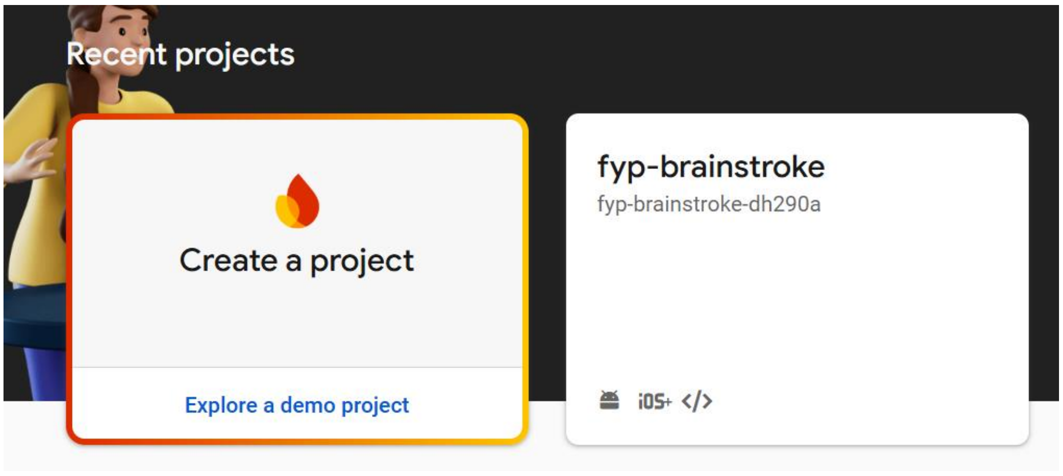
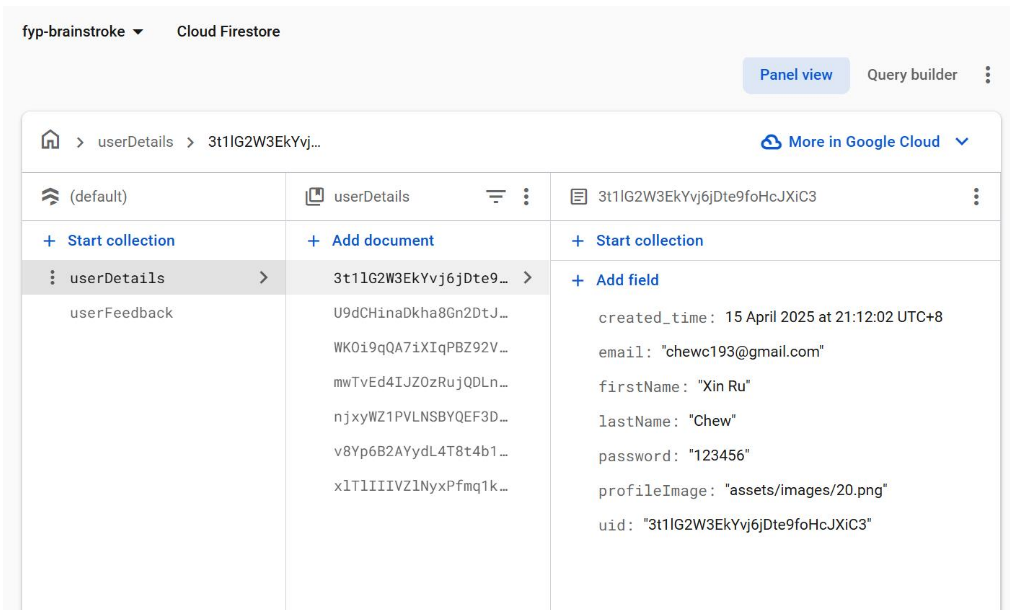


Figure 5.41 Create Firebase Project

Creation of New Project

The first step in configuring Firebase is to create a new project. Navigate to the Firebase Console and create a project. Enter a project name (fyp-brainstroke) and follow the setup wizard. During the setup, disable or enable Google Analytics based on project requirements. Once the project is created, the Firebase console will redirect to the project dashboard.

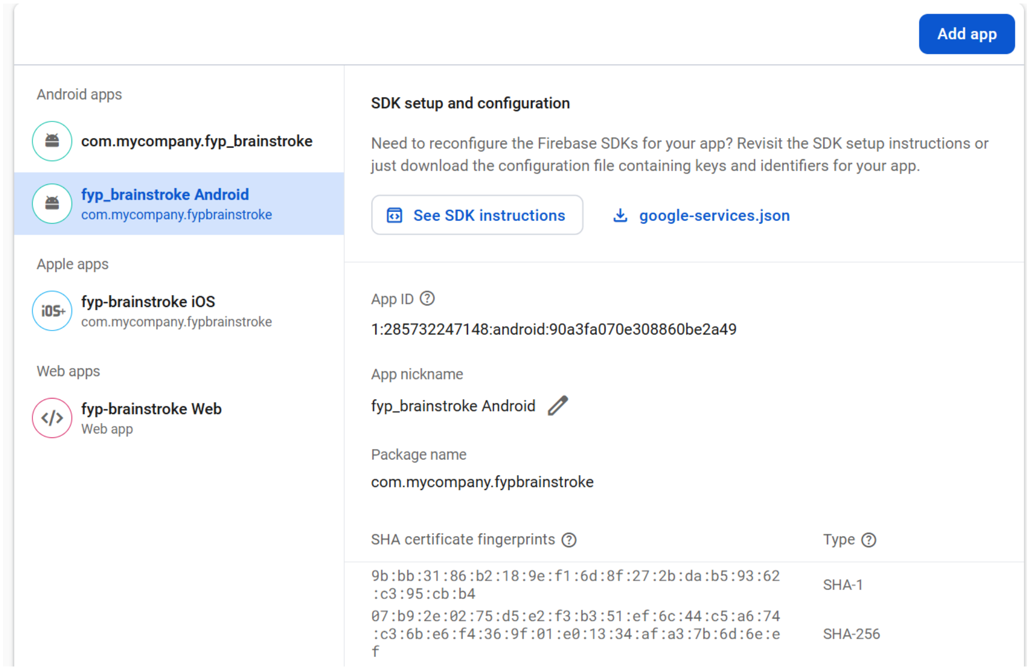


```
created_time: 15 April 2025 at 21:12:02 UTC+8
email: "chewc193@gmail.com"
firstName: "Xin Ru"
lastName: "Chew"
password: "123456"
profileImage: "assets/images/20.png"
uid: "3t1lG2W3EkYvj6jDte9foHcJXiC3"
```

Figure 5.42 Start Firebase Collection

Start Collection and Add Data

After creating the project, set up the Firestore database to store user data. The UserDetails, UserFeedback collections are created to store user-specific information for authentication and store feedback provided by users. Also, adding a sample document with fields and values according to the application's data model.



SDK setup and configuration

Need to reconfigure the Firebase SDKs for your app? Revisit the SDK setup instructions or just download the configuration file containing keys and identifiers for your app.

App ID
1:285732247148:android:90a3fa070e308860be2a49

App nickname
fyp_brainstroke Android

Package name
com.mycompany.fypbrainstroke

SHA certificate fingerprints	Type
9b:bb:31:86:b2:18:9e:f1:6d:8f:27:2b:da:b5:93:62:c3:95:cb:b4	SHA-1
07:b9:2e:02:75:d5:e2:f3:b3:51:ef:6c:44:c5:a6:74:c3:6b:e6:f4:36:9f:01:e0:13:34:af:a3:7b:6d:6e:ef	SHA-256

Figure 5.43 Enable Apps and Web

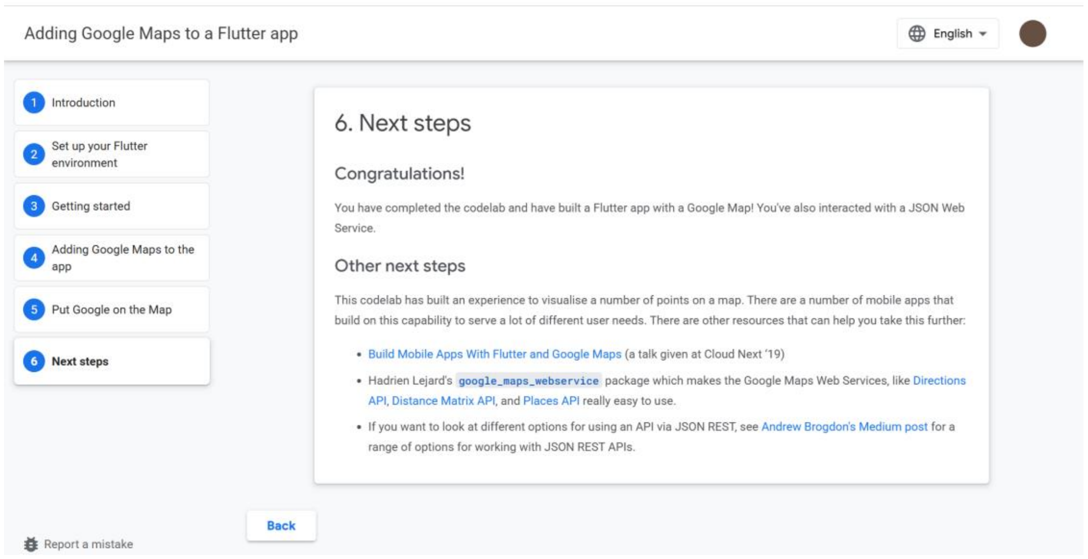
Enable Apps and Web

To ensure multi-platform support, Firebase must be configured for Android, iOS, and Web applications. In the Firebase project console, select Add app and choose the Android icon. Enter the Android package name and download the google-services.json file and place it in the “app/” directory of the Android project. Add the Firebase SDK dependencies in the Android app's “build.gradle” files as per the provided instructions. Last, sync the project with Gradle files to complete Android setup.

Since the main development platform is Android, just add IOS and Web App for reference purpose, there is nothing much needed for configuration.

After completing these steps, the Firebase project is successfully configured to support Android, iOS, and Web platforms, with `UserDetails` and `UserFeedback` collections set up.

5.4.4 Configuration of Google Maps API


Adding Google Maps to a Flutter app
English
1 Introduction
2 Set up your Flutter environment
3 Getting started
4 Adding Google Maps to the app
5 Put Google on the Map
6 Next steps
Report a mistake
Back

6. Next steps

Congratulations!

You have completed the codelab and have built a Flutter app with a Google Map! You've also interacted with a JSON Web Service.

Other next steps

This codelab has built an experience to visualise a number of points on a map. There are a number of mobile apps that build on this capability to serve a lot of different user needs. There are other resources that can help you take this further:

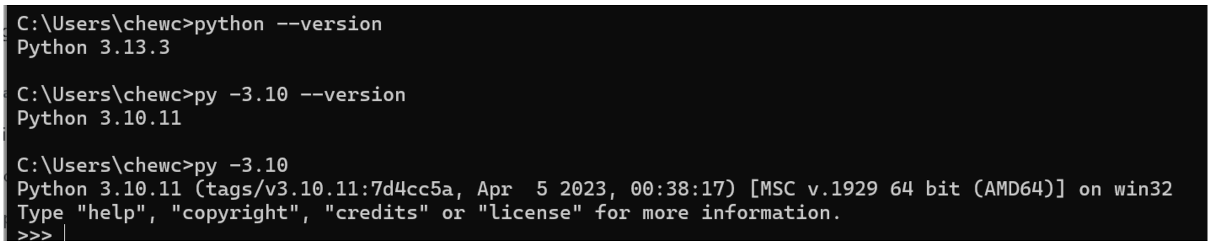
- Build Mobile Apps With Flutter and Google Maps (a talk given at Cloud Next '19)
- Hadrien Lejard's `google_maps_webservice` package which makes the Google Maps Web Services, like Directions API, Distance Matrix API, and Places API really easy to use.
- If you want to look at different options for using an API via JSON REST, see Andrew Brogdon's Medium post for a range of options for working with JSON REST APIs.

Figure 5.44 Adding Google Maps API

The Google Maps API has been configured within the StrokeSense app to enable users to locate hospitals in Malaysia with stroke centers. This integration utilizes the Google Maps JavaScript API to display interactive maps, allowing users to search for nearby medical facilities equipped with specialized stroke care services.

The configuration of the Google Maps API for the StrokeSense app to locate hospitals in Malaysia with stroke centers involves several steps. First, an API key is obtained from the Google Cloud Console by enabling the Google Maps JavaScript API and Places API, ensuring that access is restricted for security. The next step is integrating the API into the app by including the Google Maps JavaScript API script in the app’s HTML file, linking it with the generated API key. The map is then initialized by defining a container in the app’s user interface and configuring the map’s default view to focus on Malaysia, with appropriate zoom levels and center coordinates.

5.4.5 Configuration of Python



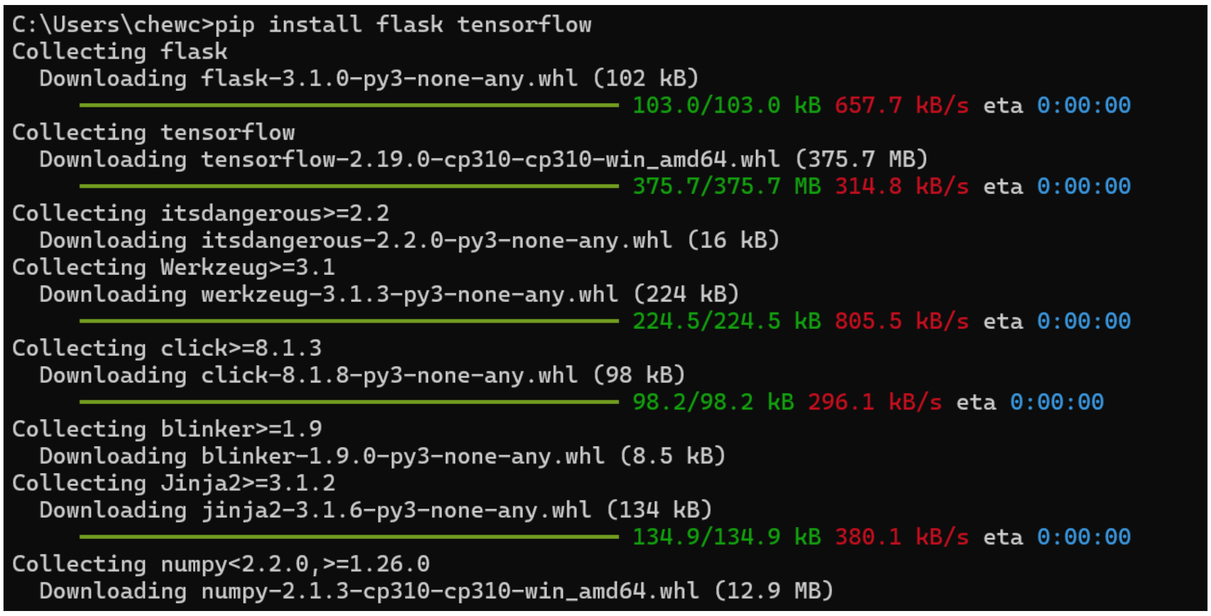
```
C:\Users\chewc>python --version
Python 3.13.3

C:\Users\chewc>py -3.10 --version
Python 3.10.11

C:\Users\chewc>py -3.10
Python 3.10.11 (tags/v3.10.11:7d4cc5a, Apr  5 2023, 00:38:17) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figure 5.45 Configure Python 3.10.11

To download and configure Python 3.10.11 for use with TensorFlow Lite in the StrokeSense app, start by uninstalling any previous Python versions through the "Control Panel" on Windows. Next, visit the official Python website and download Python 3.10.11 by selecting the appropriate installer for your operating system. After downloading, run the installer and ensure to check the box that says "Add Python 3.10 to PATH" before clicking "Install Now" to simplify running Python from the command line.

Once the installation is complete, verify the installation by opening a command prompt or terminal and typing `python --version`, which should return Python 3.10.11. This process ensures that Python 3.10.11 is ready for optimal integration with TensorFlow Lite, enabling smooth machine learning functionality in the StrokeSense app.

5.4.6 Configuration of Flask

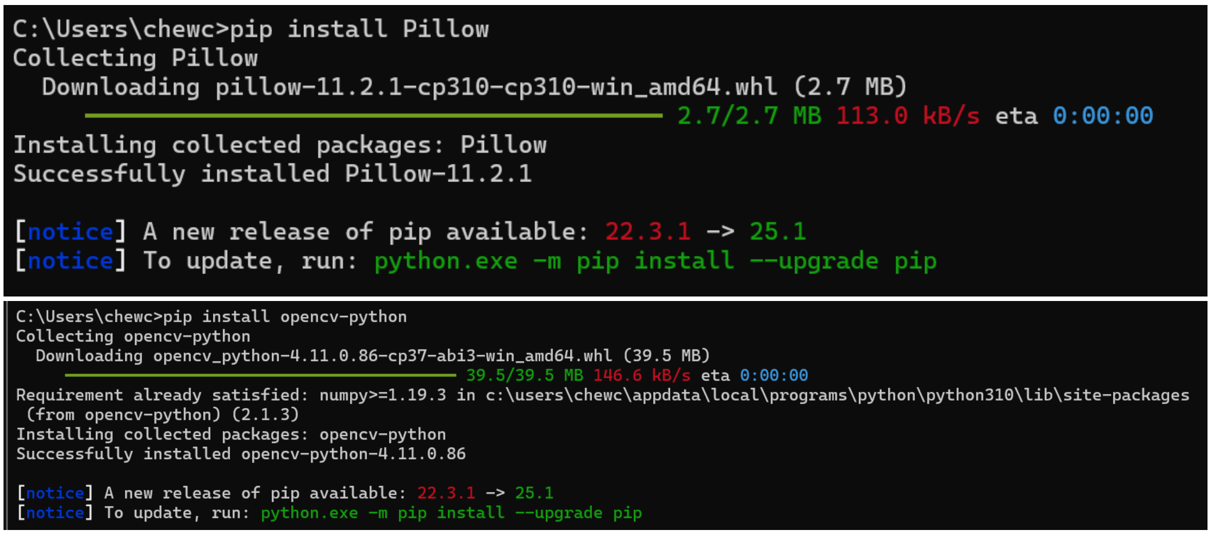


```
C:\Users\chewc>pip install flask tensorflow
Collecting flask
  Downloading flask-3.1.0-py3-none-any.whl (102 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 103.0/103.0 kB 657.7 kB/s eta 0:00:00
Collecting tensorflow
  Downloading tensorflow-2.19.0-cp310-cp310-win_amd64.whl (375.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 375.7/375.7 MB 314.8 kB/s eta 0:00:00
Collecting itsdangerous>=2.2
  Downloading itsdangerous-2.2.0-py3-none-any.whl (16 kB)
Collecting Werkzeug>=3.1
  Downloading werkzeug-3.1.3-py3-none-any.whl (224 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 224.5/224.5 kB 805.5 kB/s eta 0:00:00
Collecting click>=8.1.3
  Downloading click-8.1.8-py3-none-any.whl (98 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 98.2/98.2 kB 296.1 kB/s eta 0:00:00
Collecting blinker>=1.9
  Downloading blinker-1.9.0-py3-none-any.whl (8.5 kB)
Collecting Jinja2>=3.1.2
  Downloading jinja2-3.1.6-py3-none-any.whl (134 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 134.9/134.9 kB 380.1 kB/s eta 0:00:00
Collecting numpy<2.2.0,>=1.26.0
  Downloading numpy-2.1.3-cp310-cp310-win_amd64.whl (12.9 MB)
```

Figure 5.46 Install Flask

Installing Flask

The first step in setting up a Flask application is to install Flask. This can be done using Python's package manager, pip. Run the command pip install flask in terminal or command prompt. This will download and install Flask along with its dependencies, enabling you to create web applications using this lightweight framework.



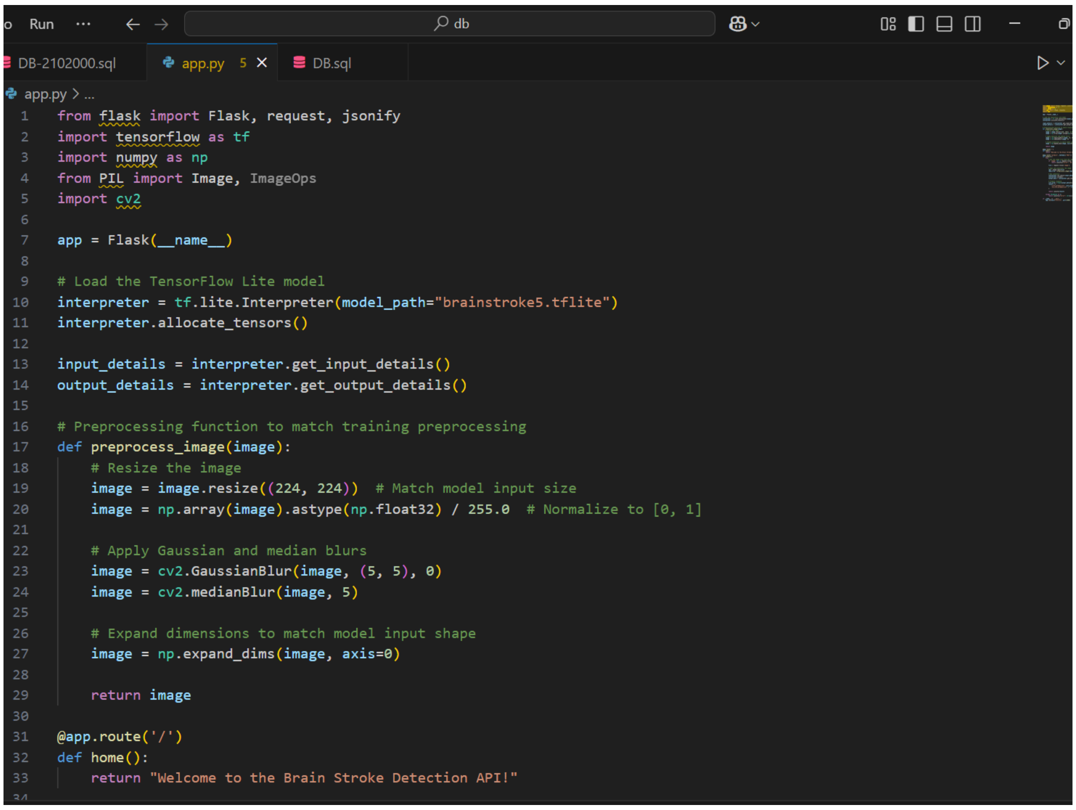
```
C:\Users\chewc>pip install Pillow
Collecting Pillow
  Downloading pillow-11.2.1-cp310-cp310-win_amd64.whl (2.7 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.7/2.7 MB 113.0 kB/s eta 0:00:00
Installing collected packages: Pillow
Successfully installed Pillow-11.2.1

[notice] A new release of pip available: 22.3.1 -> 25.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
C:\Users\chewc>pip install opencv-python
Collecting opencv-python
  Downloading opencv_python-4.11.0.86-cp37-abi3-win_amd64.whl (39.5 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 39.5/39.5 MB 146.6 kB/s eta 0:00:00
Requirement already satisfied: numpy>=1.19.3 in c:\users\chewc\appdata\local\programs\python\python310\lib\site-packages
 (from opencv-python) (2.1.3)
Installing collected packages: opencv-python
Successfully installed opencv-python-4.11.0.86

[notice] A new release of pip available: 22.3.1 -> 25.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

Figure 5.47 Install Flask Required Libraries

Installing Required Libraries

For additional functionality, it is a must to install other libraries such as Pillow for image processing and OpenCV for computer vision tasks. Use pip to install them by running pip install pillow and pip install opencv-python. These libraries will ensure the Flask application has the necessary tools to handle specific tasks like image manipulation or processing.

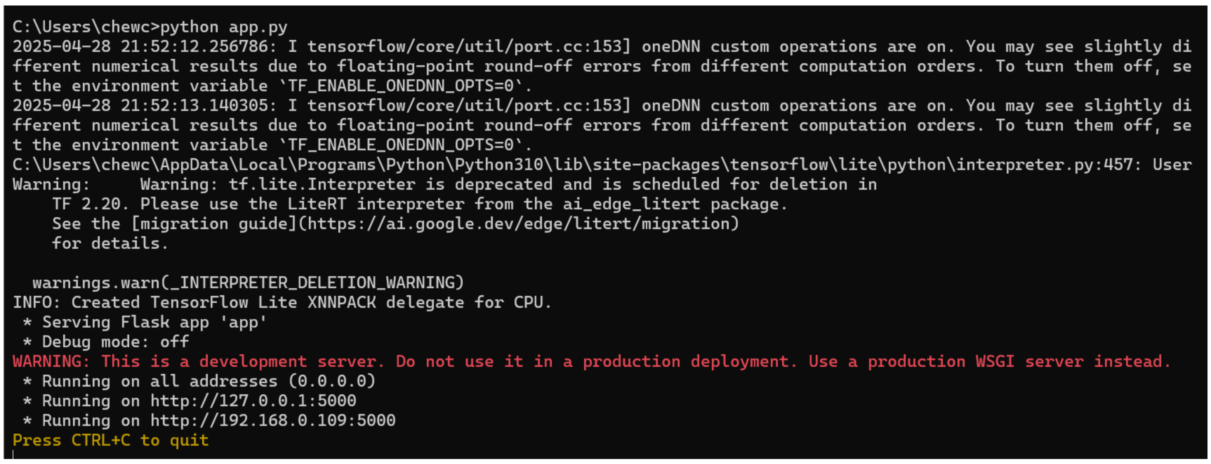


```python
from flask import Flask, request, jsonify
import tensorflow as tf
import numpy as np
from PIL import Image, ImageOps
import cv2

app = Flask(__name__)

# Load the TensorFlow Lite model
interpreter = tf.lite.Interpreter(model_path="brainstroke5.tflite")
interpreter.allocate_tensors()

input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()

# Preprocessing function to match training preprocessing
def preprocess_image(image):
    # Resize the image
    image = image.resize((224, 224))  # Match model input size
    image = np.array(image).astype(np.float32) / 255.0  # Normalize to [0, 1]

    # Apply Gaussian and median blurs
    image = cv2.GaussianBlur(image, (5, 5), 0)
    image = cv2.medianBlur(image, 5)

    # Expand dimensions to match model input shape
    image = np.expand_dims(image, axis=0)

    return image

@app.route('/')
def home():
    return "Welcome to the Brain Stroke Detection API!"
```

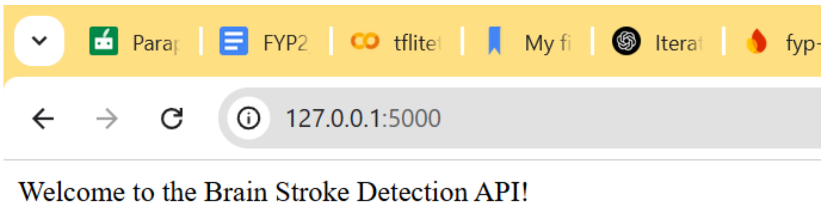
Figure 5.48 Create App.py

Creating a Python Application File

To start building the Flask application, create a Python file named app.py. This file will serve as the entry point for StrokeSense application. Inside the file, it can import Flask, define routes, and set up the application logic.



```
C:\Users\chewc>python app.py
2025-04-28 21:52:12.256786: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly di
fferent numerical results due to floating-point round-off errors from different computation orders. To turn them off, se
t the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-04-28 21:52:13.140305: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly di
fferent numerical results due to floating-point round-off errors from different computation orders. To turn them off, se
t the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
C:\Users\chewc\AppData\Local\Programs\Python\Python310\lib\site-packages\tensorflow\lite\python\interpreter.py:457: User
Warning:     Warning: tf.lite.Interpreter is deprecated and is scheduled for deletion in
    TF 2.20. Please use the LiteRT interpreter from the ai_edge_litert package.
    See the [migration guide](https://ai.google.dev/edge/litert/migration)
    for details.

  warnings.warn(_INTERPRETER_DELETION_WARNING)
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
 * Serving Flask app 'app'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://192.168.0.109:5000
Press CTRL+C to quit
```


127.0.0.1:5000

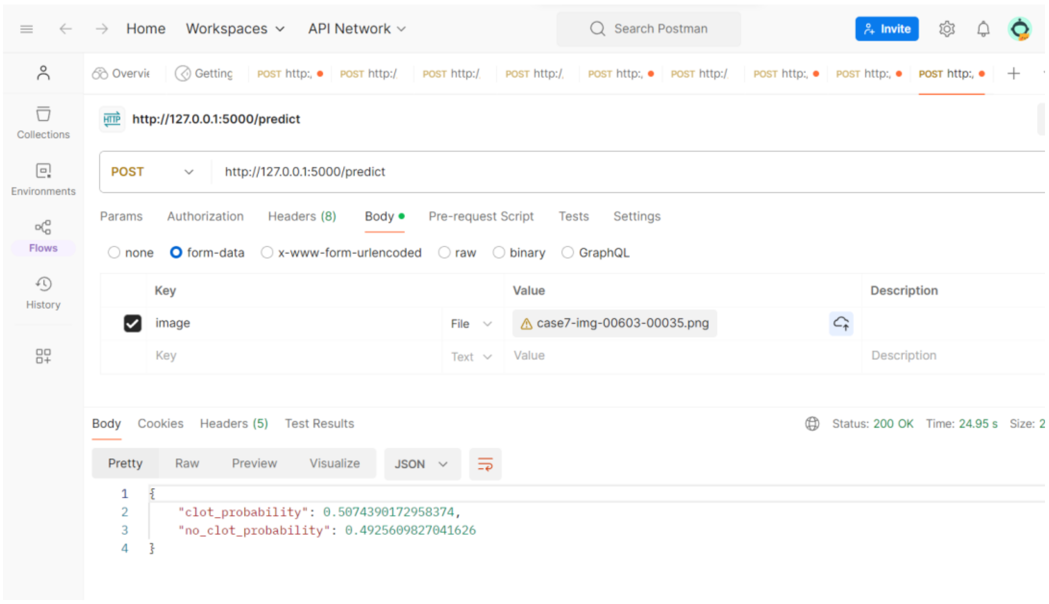
Welcome to the Brain Stroke Detection API!

Figure 5.49 Run App.py

Running the Python Application

Once the app.py file is ready, run it to start the Flask development server. Execute the command python app.py in the terminal. Flask will run the application on a local server, typically accessible at http://127.0.0.1:5000.

5.4.7 Testing with Postman


POST http://127.0.0.1:5000/predict

Body: form-data

Key: image (File) — Value: case7-img-00603-00035.png

Status: 200 OK Time: 24.95 s

```
{
    "clot_probability": 0.5074390172958374,
    "no_clot_probability": 0.4925609827041626
}
```

Figure 5.50 Postman Testing

After the application is running, you can test its functionality using Postman, a popular API testing tool. Open Postman and make a GET or POST request to your Flask application's endpoint (e.g., http://127.0.0.1:5000). If the server responds as expected, it confirms that your Flask setup is working correctly.

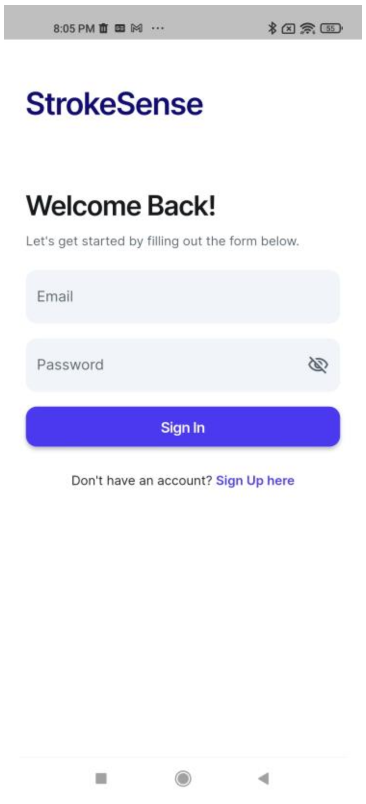
5.5 System Operations

Application Icon and Name

Figure 5.51 StrokeSense Logo

The application icon for StrokeSense features a stylized brain design in a clean, modern blue outline. This visual element symbolizes the app's focus on brain health and clot detection. The choice of a brain icon directly reflects the core function of the app—analyzing medical brain images to detect stroke symptoms using artificial intelligence. The name StrokeSense combines "Stroke" and "Sense" to convey the app's intelligent, responsive approach to clot detection and awareness, making it both meaningful and memorable for users.

Login Page


8:05 PM

StrokeSense

Welcome Back!

Let's get started by filling out the form below.

Email

Password

Sign In

Don't have an account? Sign Up here

Figure 5.52 Login Page

The StrokeSense login page presents a clean and user-friendly interface designed for easy access and smooth user experience. The page prominently displays the app name "StrokeSense" at the top in bold, dark blue font, reinforcing brand identity. Below, a welcoming message “Welcome Back!” greets returning users, followed by brief instructions prompting users to fill in their credentials.

The login form includes two input fields: one for email and another for password, with a toggle icon in the password field for visibility control. A prominent blue Sign In button provides access to the system upon successful login. At the bottom, a secondary prompt offers a link to Sign Up, guiding new users to create an account. The overall layout is minimalist and intuitive, ensuring ease of navigation and accessibility across devices.

Signup Page

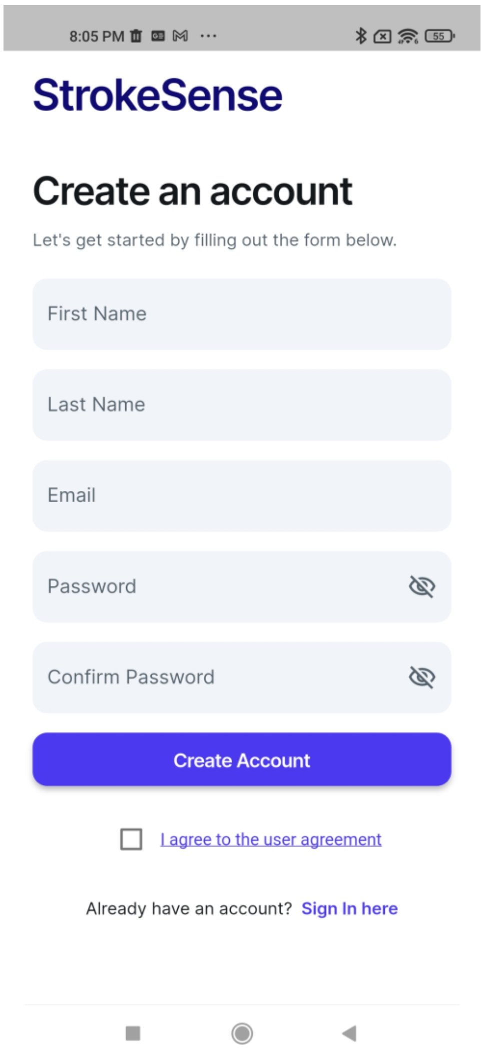

8:05 PM
StrokeSense
Create an account
Let's get started by filling out the form below.
First Name
Last Name
Email
Password
Confirm Password
Create Account
I agree to the user agreement
Already have an account? Sign In here

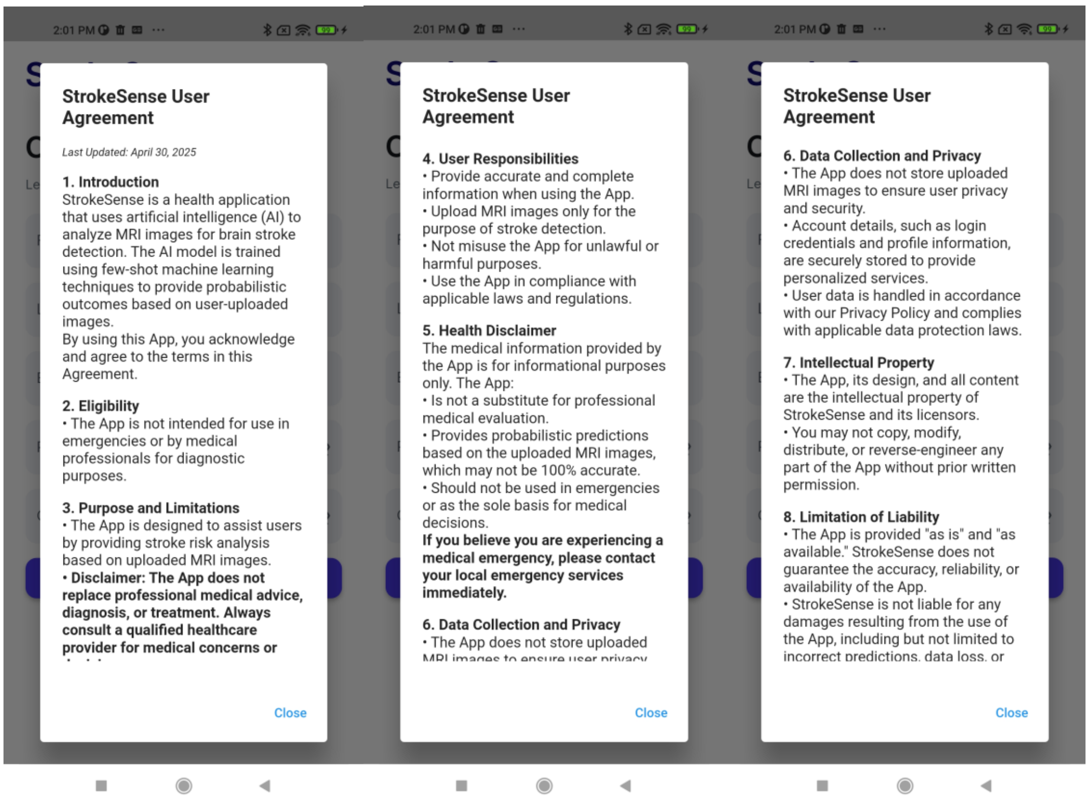
Figure 5.53 Signup Page

The StrokeSense sign-up page offers a structured and welcoming interface for new users to register an account. The page begins with the application name "StrokeSense" displayed at the top in a bold, dark blue font, followed by the title “Create an account” to clearly indicate the purpose of the page. A short instruction encourages users to begin by filling in the required details.

The form includes input fields for First Name, Last Name, Email, Password, and Confirm Password. Each field is clearly labelled and arranged vertically for easy navigation. Password fields include an eye icon, allowing users to toggle the visibility of their input for convenience.

Below the form, users are required to agree to the user agreement by checking a box, with the agreement link highlighted in blue. The Create Account button, styled in a vibrant blue color, allows users to submit the registration form. At the bottom, a prompt guides existing users to the login page via a Sign In here link, enhancing usability and flow between account creation and login.

User Agreement



StrokeSense User Agreement

Last Updated: April 30, 2025

1. Introduction
StrokeSense is a health application that uses artificial intelligence (AI) to analyze MRI images for brain stroke detection. The AI model is trained using few-shot machine learning techniques to provide probabilistic outcomes based on user-uploaded images.
By using this App, you acknowledge and agree to the terms in this Agreement.

2. Eligibility
• The App is not intended for use in emergencies or by medical professionals for diagnostic purposes.

3. Purpose and Limitations
• The App is designed to assist users by providing stroke risk analysis based on uploaded MRI images.
• **Disclaimer: The App does not replace professional medical advice, diagnosis, or treatment. Always consult a qualified healthcare provider for medical concerns or**

Close

StrokeSense User Agreement

4. User Responsibilities
• Provide accurate and complete information when using the App.
• Upload MRI images only for the purpose of stroke detection.
• Not misuse the App for unlawful or harmful purposes.
• Use the App in compliance with applicable laws and regulations.

5. Health Disclaimer
The medical information provided by the App is for informational purposes only. The App:
• Is not a substitute for professional medical evaluation.
• Provides probabilistic predictions based on the uploaded MRI images, which may not be 100% accurate.
• Should not be used in emergencies or as the sole basis for medical decisions.
If you believe you are experiencing a medical emergency, please contact your local emergency services immediately.

6. Data Collection and Privacy
• The App does not store uploaded MRI images to ensure user privacy

Close

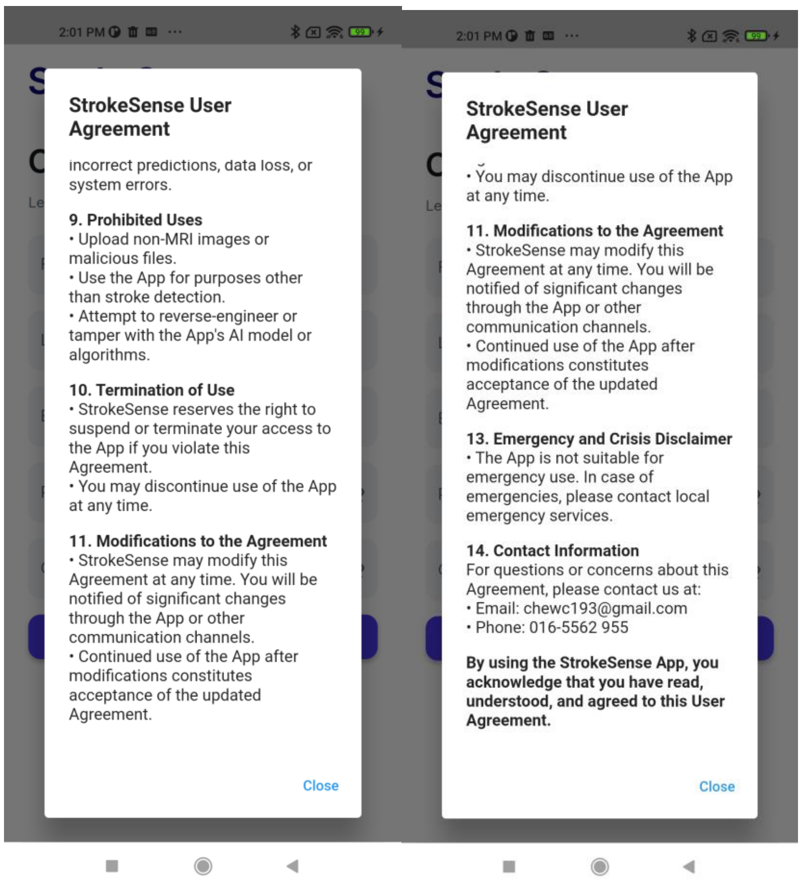
StrokeSense User Agreement

6. Data Collection and Privacy
• The App does not store uploaded MRI images to ensure user privacy and security.
• Account details, such as login credentials and profile information, are securely stored to provide personalized services.
• User data is handled in accordance with our Privacy Policy and complies with applicable data protection laws.

7. Intellectual Property
• The App, its design, and all content are the intellectual property of StrokeSense and its licensors.
• You may not copy, modify, distribute, or reverse-engineer any part of the App without prior written permission.

8. Limitation of Liability
• The App is provided "as is" and "as available." StrokeSense does not guarantee the accuracy, reliability, or availability of the App.
• StrokeSense is not liable for any damages resulting from the use of the App, including but not limited to incorrect predictions, data loss, or

Close



StrokeSense User Agreement

incorrect predictions, data loss, or system errors.

9. Prohibited Uses
• Upload non-MRI images or malicious files.
• Use the App for purposes other than stroke detection.
• Attempt to reverse-engineer or tamper with the App's AI model or algorithms.

10. Termination of Use
• StrokeSense reserves the right to suspend or terminate your access to the App if you violate this Agreement.
• You may discontinue use of the App at any time.

11. Modifications to the Agreement
• StrokeSense may modify this Agreement at any time. You will be notified of significant changes through the App or other communication channels.
• Continued use of the App after modifications constitutes acceptance of the updated Agreement.

Close

StrokeSense User Agreement

• You may discontinue use of the App at any time.

11. Modifications to the Agreement
• StrokeSense may modify this Agreement at any time. You will be notified of significant changes through the App or other communication channels.
• Continued use of the App after modifications constitutes acceptance of the updated Agreement.

13. Emergency and Crisis Disclaimer
• The App is not suitable for emergency use. In case of emergencies, please contact local emergency services.

14. Contact Information
For questions or concerns about this Agreement, please contact us at:
• Email: chewc193@gmail.com
• Phone: 016-5562 955

By using the StrokeSense App, you acknowledge that you have read, understood, and agreed to this User Agreement.

Close

Figure 5.54 User Agreement

The StrokeSense User Agreement outlines the terms and conditions associated with the usage of the StrokeSense app. The agreement begins with an introduction, describing StrokeSense as a health application powered by artificial intelligence (AI) to analyze MRI images for clot detection. It emphasizes that the app uses machine learning techniques to provide probabilistic outcomes and that users must acknowledge and agree to the terms before using the app.

The agreement specifies eligibility by stating that the app is not intended for use in emergencies or as a substitute for medical professionals. It further defines the purpose and limitations of the app, highlighting that it assists users by providing stroke risk analysis based on MRI images. It explicitly states that the app does not replace professional medical advice and urges users to consult qualified healthcare providers for medical concerns.

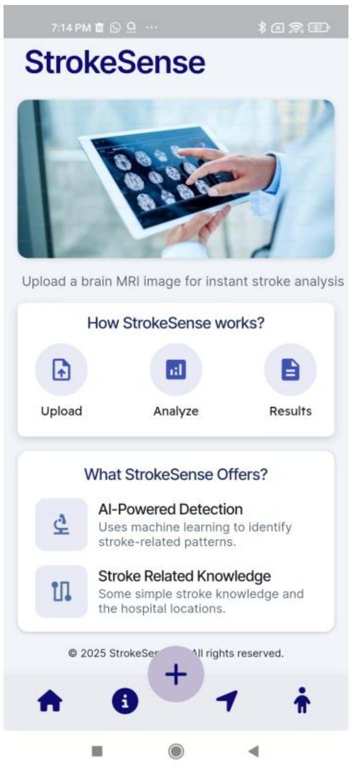
The user responsibilities section requires users to provide accurate and complete information, refrain from misuse, and comply with applicable laws and regulations. A health disclaimer stresses that the app's predictions are for informational purposes only and not a substitute for medical evaluation. It warns that the results may not be 100% accurate and should not be relied on in emergencies.

The agreement also includes clauses on data collection and privacy, intellectual property, and limitation of liability. The intellectual property section prohibits copying, modifying, or redistributing the app's content without prior permission. The limitation of liability clause states that the app is provided “as is” and disclaims any warranties regarding its accuracy, reliability, or availability.

Additionally, the agreement lists prohibited uses, such as uploading malicious images or using the app for purposes other than clot detection. It reserves the right to terminate user access for violations and permits users to discontinue using the app at any time. The agreement also allows StrokeSense to modify its terms and notifies users of significant changes through the app or other communication channels.

Lastly, it includes an emergency and crisis disclaimer, advising users to contact local emergency services in case of an emergency. The agreement concludes with contact information for support and a statement requiring users to confirm that they have read, understood, and agreed to the terms before using the app.

Home Page



StrokeSense

Upload a brain MRI image for instant stroke analysis

How StrokeSense works?

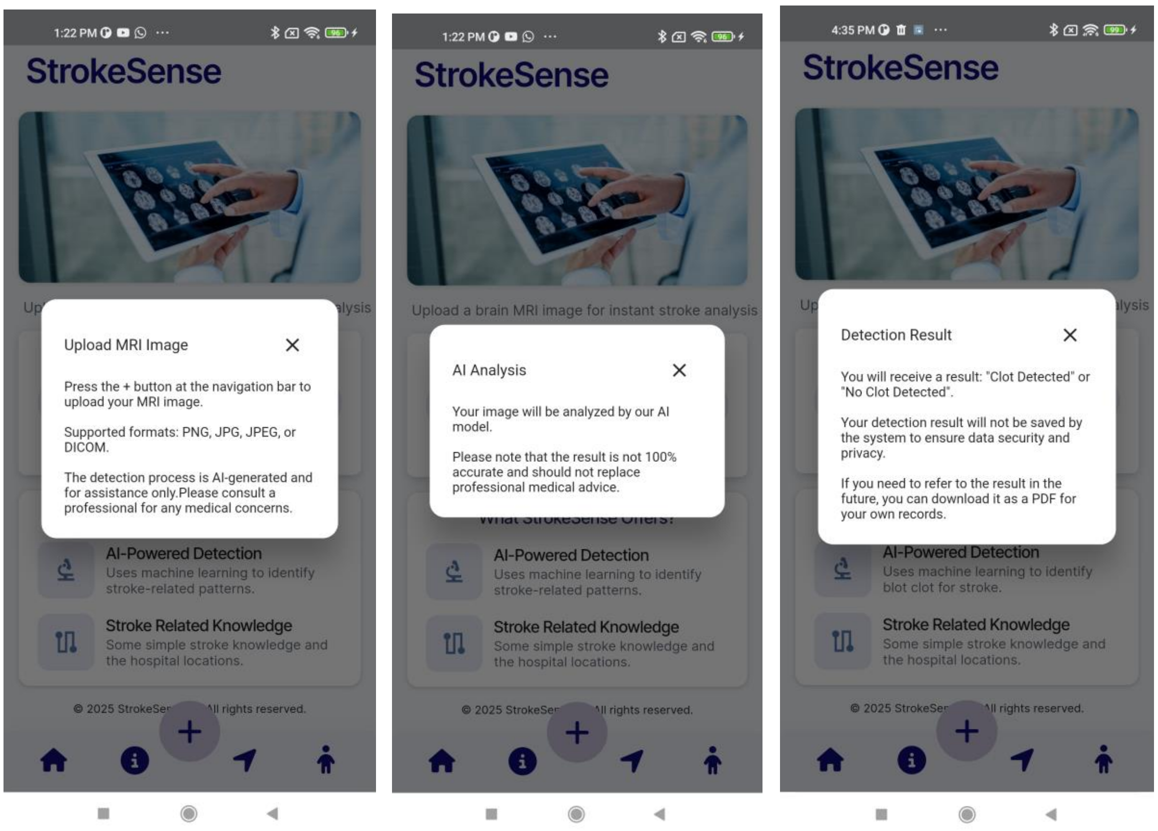
What StrokeSense Offers?

AI-Powered Detection
Uses machine learning to identify stroke-related patterns.

Stroke Related Knowledge
Some simple stroke knowledge and the hospital locations.

© 2025 StrokeSense. All rights reserved.

Figure 5.55 Home Page



Upload MRI Image

Press the + button at the navigation bar to upload your MRI image.

Supported formats: PNG, JPG, JPEG, or DICOM.

The detection process is AI-generated and for assistance only.Please consult a professional for any medical concerns.

AI Analysis

Your image will be analyzed by our AI model.

Please note that the result is not 100% accurate and should not replace professional medical advice.

Detection Result

You will receive a result: "Clot Detected" or "No Clot Detected".

Your detection result will not be saved by the system to ensure data security and privacy.

If you need to refer to the result in the future, you can download it as a PDF for your own records.

AI-Powered Detection
Uses machine learning to identify blot clot for stroke.

Stroke Related Knowledge
Some simple stroke knowledge and the hospital locations.

Figure 5.56 How StrokeSense Works

The StrokeSense home page serves as the central hub for users, providing an intuitive and visually appealing interface designed for ease of navigation. At the top of the page, the application prominently features the StrokeSense logo, establishing its branding. Beneath the logo, a banner image of a healthcare professional analyzing brain scans emphasizes the app's focus on MRI analysis for clot detection. Below the image, a concise tagline invites users to "Upload a brain MRI image for instant stroke analysis," clearly communicating the app's primary purpose.

The "How StrokeSense works?" section is displayed as a card with three interactive icons: Upload, Analyze, and Results. These icons provide users with a streamlined understanding of the app's workflow. When 3 of the icon is clicked, it sequentially triggers three informative pop-up cards explaining how the app works. The first card, "Upload MRI Image" guides the user to upload an MRI image in supported formats like PNG, JPG, JPEG, or DICOM, with a reminder to consult medical professionals for critical decisions. The second card, "AI Analysis" explains that the uploaded image will be analyzed by the AI model, emphasizing that the result is not 100% accurate and is intended for assistance only. The final card, "Detection Result" informs the user that the detection result—either "Stroke Detected" or "No Stroke Detected"—will not be saved by the system to ensure data security and privacy and advises downloading the result as a PDF for future reference. This step-by-step explanation ensures users understand the process and functionality of the app.

Following this section, another card titled "What StrokeSense Offers?" outlines the app's key functionalities. This includes AI-Powered Detection, which highlights the use of machine learning algorithms to identify stroke-related patterns, and Stroke Related Knowledge, which provides users with essential stroke-related information and hospital locations, enhancing the app's educational value.

The page is complemented by a navigation bar at the bottom, consisting of icons for Home, a central "+" button for uploading images, and access to additional features. The footer includes a copyright statement, reinforcing credibility and professionalism. Overall, the home page is thoughtfully structured to provide users with clear guidance and a seamless experience.

Article Page

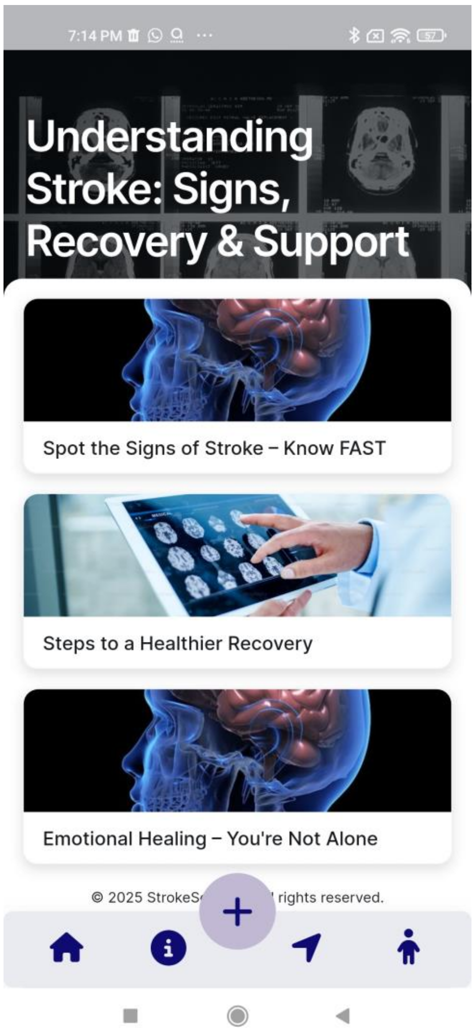

7:14 PM
Understanding Stroke: Signs, Recovery & Support
Spot the Signs of Stroke – Know FAST
Steps to a Healthier Recovery
Emotional Healing – You're Not Alone
© 2025 StrokeS... rights reserved.

Figure 5.57 Article Page

The article page of StrokeSense is designed to provide users with educational content and resources about strokes, focusing on signs, recovery, and support. At the top of the page, a bold header titled "Understanding Stroke: Signs, Recovery & Support" captures the user's attention and sets the theme for the content. The header is complemented by a visually impactful background image of brain scans, reinforcing the medical and informative nature of the page.

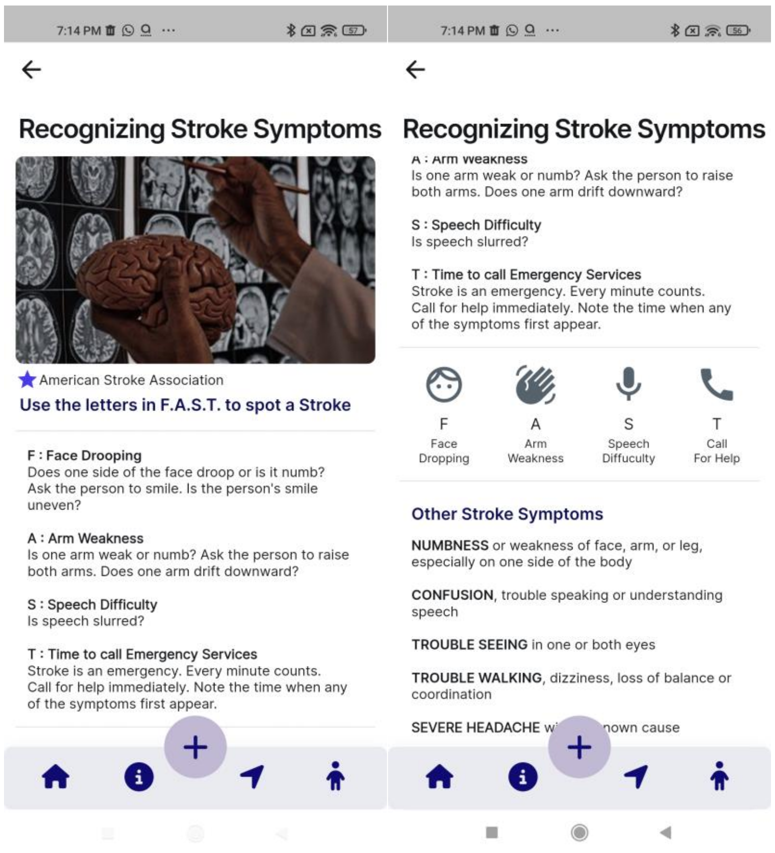
Below the header, the page features a series of interactive article cards, each focusing on a specific topic. The first card, titled "Spot the Signs of Stroke – Know FAST", educates users on recognizing the early warning signs of a stroke using the F.A.S.T. acronym (Face drooping, Arm weakness, Speech difficulty, Time to call emergency services). This card aims to raise awareness about timely stroke identification.

The second card, "Steps to a Healthier Recovery", provides information and guidance on post-stroke recovery. It focuses on actionable steps to improve physical and mental well-being after a stroke, encouraging users to take control of their recovery journey.

The third card, "Emotional Healing – You’re Not Alone", addresses the emotional and psychological challenges often faced by stroke survivors. It emphasizes the importance of emotional healing and offers support resources, reminding users that they are not alone in their recovery.

The page layout is simple and user-friendly, with each article card featuring a relevant image and a descriptive title. The footer also includes a copyright notice to enhance the app's credibility. Overall, the article page effectively combines educational content with an engaging layout to support users in understanding and managing strokes.

Article Content Page



Recognizing Stroke Symptoms

American Stroke Association

Use the letters in F.A.S.T. to spot a Stroke

F : Face Drooping
Does one side of the face droop or is it numb? Ask the person to smile. Is the person's smile uneven?

A : Arm Weakness
Is one arm weak or numb? Ask the person to raise both arms. Does one arm drift downward?

S : Speech Difficulty
Is speech slurred?

T : Time to call Emergency Services
Stroke is an emergency. Every minute counts. Call for help immediately. Note the time when any of the symptoms first appear.

Recognizing Stroke Symptoms

A : Arm Weakness
Is one arm weak or numb? Ask the person to raise both arms. Does one arm drift downward?

S : Speech Difficulty
Is speech slurred?

T : Time to call Emergency Services
Stroke is an emergency. Every minute counts. Call for help immediately. Note the time when any of the symptoms first appear.

Other Stroke Symptoms

NUMBNESS or weakness of face, arm, or leg, especially on one side of the body

CONFUSION, trouble speaking or understanding speech

TROUBLE SEEING in one or both eyes

TROUBLE WALKING, dizziness, loss of balance or coordination

SEVERE HEADACHE w... nown cause

Figure 5.58 Article Content Page

The layout of the article pages in the StrokeSense app is designed to provide users with structured, easy-to-understand, and actionable information about stroke-related topics. Each article page is visually organized with a clean and user-friendly interface. At the top, the article title is prominently displayed, ensuring the reader immediately understands the focus of the content. This is followed by a relevant and engaging image, which visually supports the topic and draws the user's attention.

The content is divided into clearly labelled sections for better readability. The primary section usually begins with key insights or actionable advice, organized in bullet points or step-by-step instructions, as shown in the example of the "Recognizing Stroke Symptoms" article. This specific article uses the F.A.S.T. acronym (Face Drooping, Arm Weakness, Speech Difficulty, Time to call emergency services) to educate users on identifying stroke symptoms quickly and effectively. Supporting sections provide supplementary information, such as additional symptoms or related advice, to deepen the user's understanding.

Icons and visual aids are strategically integrated to enhance comprehension and make the information more memorable. For example, the F.A.S.T. acronym is accompanied by icons that represent each symptom, making it easier for users to associate the visual cues with essential actions.

The navigation bar at the bottom of the page ensures seamless access to other sections of the app, such as the home page, additional resources, or the option to upload MRI images. Overall, the objectives of the article pages are to educate users, raise awareness about stroke symptoms, and provide guidance for timely action, all while maintaining a visually appealing and easily accessible layout.

Upload/Detection Page

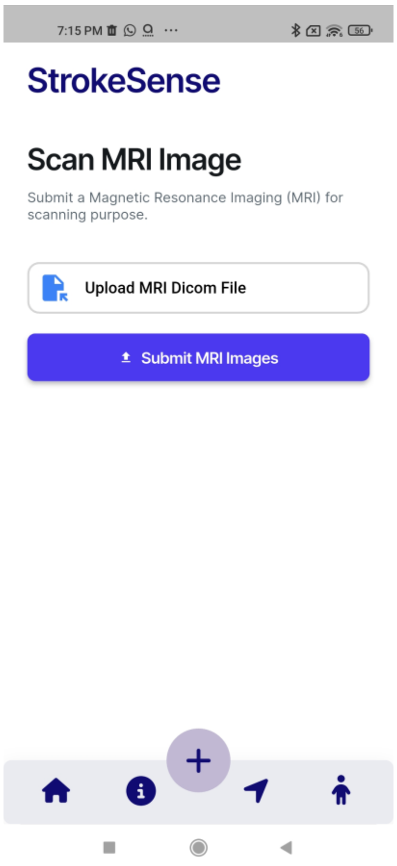


Figure 5.59 Upload/Detection Page

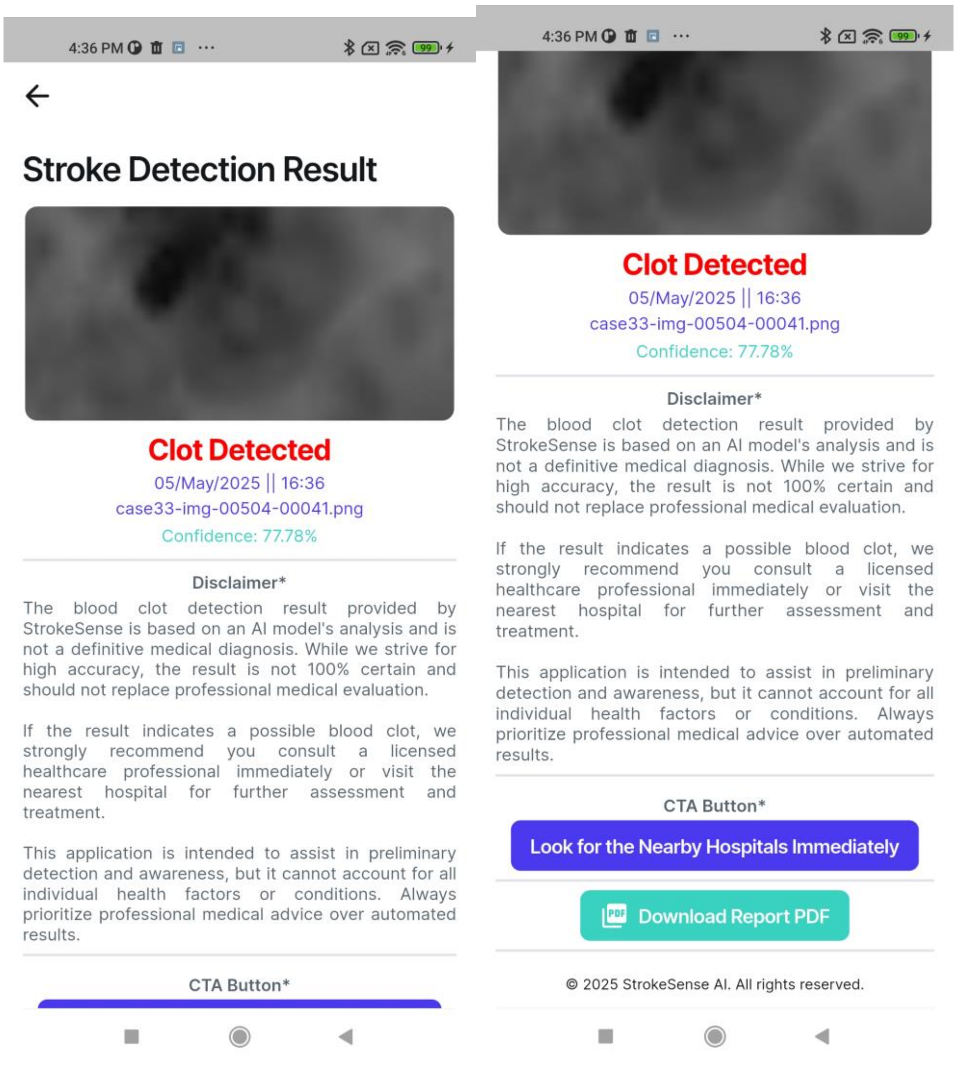
The StrokeSense upload/detection page is designed to facilitate the submission and analysis of MRI images for clot detection. The page features a clean and minimalistic layout with a prominent title, "Scan MRI Image," and a brief description that guides the user to submit a Magnetic Resonance Imaging (MRI) file for scanning purposes. This ensures clarity in its purpose while maintaining ease of use.

Users can upload MRI images in various supported formats, including PNG, JPEG, JPG, and DICOM. The page provides a dedicated button labeled "Upload MRI Dicom File" to allow users to locate and select their MRI file for submission. Once the image is uploaded, it undergoes a backstage conversion process where all file types are automatically converted into the PNG format to ensure compatibility with the backend systems.

A second button, labeled "Submit MRI Images," enables users to initiate the analysis process. Upon submission, the image is passed through the app's API model, which leverages artificial intelligence to assess the MRI image for potential stroke indicators. The model processes the image and determines whether a stroke is detected or not. This result is then communicated back to the user, providing them with instant insights.

The interface is further enhanced by a navigation bar at the bottom, enabling users to switch between the home page, articles, and other app features. Overall, the upload/detection page serves as a streamlined and user-friendly entry point for clot detection, ensuring a seamless experience from image submission to result generation.

Stroke Detected Result Page


4:36 PM

Stroke Detection Result

Clot Detected

05/May/2025 || 16:36

case33-img-00504-00041.png

Confidence: 77.78%

Disclaimer*

The blood clot detection result provided by StrokeSense is based on an AI model's analysis and is not a definitive medical diagnosis. While we strive for high accuracy, the result is not 100% certain and should not replace professional medical evaluation.

If the result indicates a possible blood clot, we strongly recommend you consult a licensed healthcare professional immediately or visit the nearest hospital for further assessment and treatment.

This application is intended to assist in preliminary detection and awareness, but it cannot account for all individual health factors or conditions. Always prioritize professional medical advice over automated results.

CTA Button*

4:36 PM

Clot Detected

05/May/2025 || 16:36

case33-img-00504-00041.png

Confidence: 77.78%

Disclaimer*

The blood clot detection result provided by StrokeSense is based on an AI model's analysis and is not a definitive medical diagnosis. While we strive for high accuracy, the result is not 100% certain and should not replace professional medical evaluation.

If the result indicates a possible blood clot, we strongly recommend you consult a licensed healthcare professional immediately or visit the nearest hospital for further assessment and treatment.

This application is intended to assist in preliminary detection and awareness, but it cannot account for all individual health factors or conditions. Always prioritize professional medical advice over automated results.

CTA Button*

© 2025 StrokeSense AI. All rights reserved.

Figure 5.60 Stroke Detected Result Page

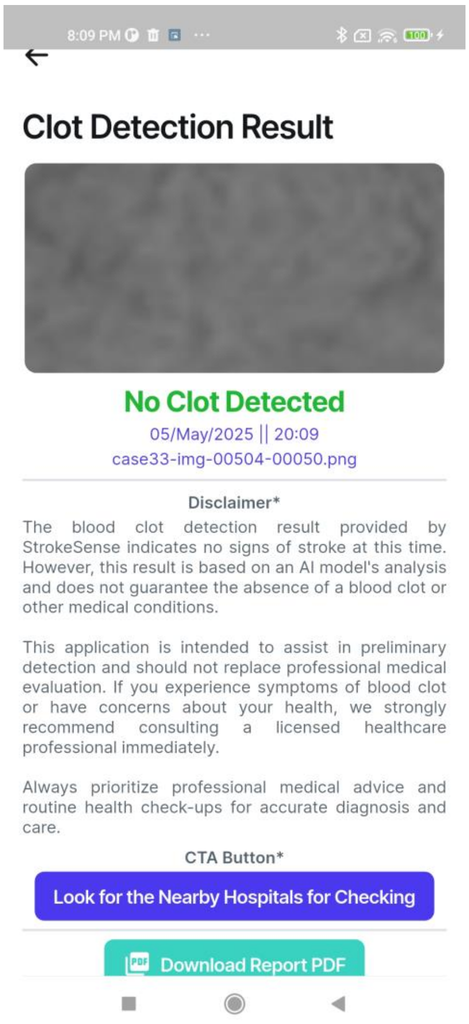
The Clot Detection Result page in StrokeSense provides users with the outcome of their uploaded MRI image analysis, clearly indicating if a stroke has been detected. At the top of the page, the header "Clot Detection Result" ensures immediate recognition of the page's purpose. Beneath the header, a banner image reinforces the context of medical imaging and diagnosis. The result is prominently displayed with the text "Clot Detected" in bold red, drawing attention to the critical outcome. Alongside the result, details such as the date, time, and filename of the uploaded MRI image are provided for reference.

A disclaimer is included to clarify the limitations of the app's AI model. It emphasizes that the result is based on AI analysis and is not a definitive medical diagnosis. Users are informed that while the app strives for high accuracy, the results are not 100% certain and should not replace professional medical evaluation. The disclaimer further advises users to consult a licensed healthcare professional or visit the nearest hospital for further assessment and treatment if a possible stroke is detected.

The page also highlights the app's purpose as a tool for preliminary detection and awareness, noting that it cannot account for all individual health factors or conditions. Users are reminded to prioritize professional medical advice over automated results.

At the bottom of the page, a clear call-to-action (CTA) button labeled "Look for the Nearby Hospitals Immediately" is provided, encouraging users to take prompt action by locating nearby medical facilities. This ensures that users can act quickly in response to the detection result. This button ensures that users can take proactive steps to seek medical attention if needed. Besides, the page also added an action button “Download PDF report” for user to download the generated detection report. This can ease user for refer the detection result since the detection result is not saved into the app to ensure data security. Overall, the page is designed to communicate critical information effectively while promoting responsible and urgent follow-up actions.

No Clot Detected Result Page



The blood clot detection result provided by StrokeSense indicates no signs of stroke at this time. However, this result is based on an AI model's analysis and does not guarantee the absence of a blood clot or other medical conditions.

This application is intended to assist in preliminary detection and should not replace professional medical evaluation. If you experience symptoms of blood clot or have concerns about your health, we strongly recommend consulting a licensed healthcare professional immediately.

Always prioritize professional medical advice and routine health check-ups for accurate diagnosis and care.

Figure 5.61 No Clot Detected Result Page

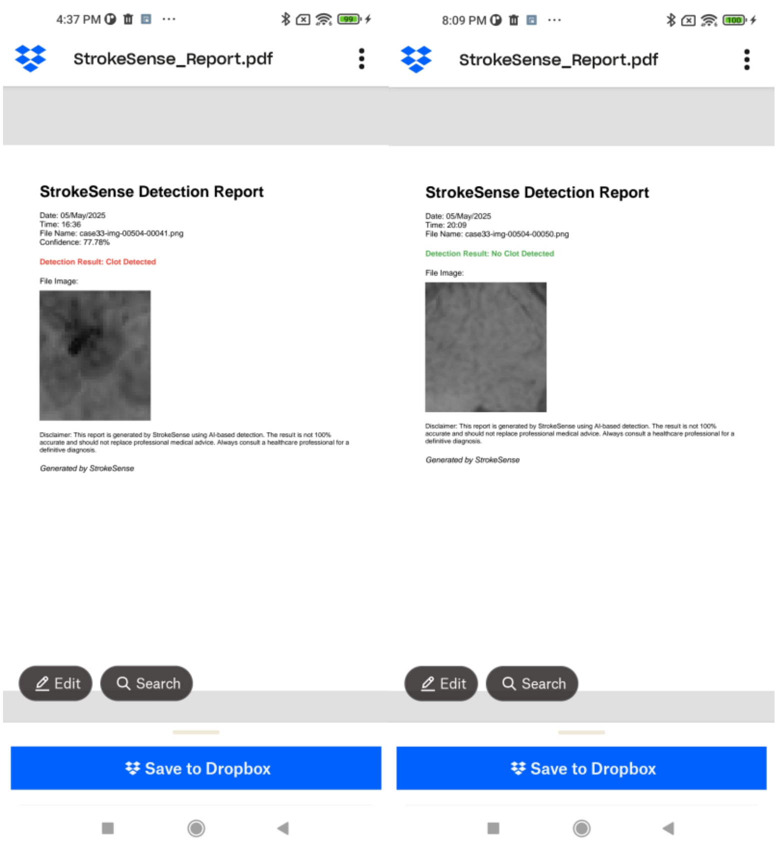
The "No Clot Detected" page in StrokeSense provides users with the results of their MRI image analysis, indicating that no signs of a stroke were found. The page is structured to deliver this reassuring result while still emphasizing caution and promoting responsible follow-up actions. At the top, the header "Clot Detection Result" ensures clarity about the nature of the page. Below the header, a relevant banner image reinforces the medical context of clot detection.

The result is displayed prominently in bold green text with the message "No Clot Detected," providing immediate clarity to the user. Alongside this result, the page also includes details such as the date, time, and file name of the uploaded MRI image for reference and traceability.

A disclaimer follows the result, clearly stating that while no stroke was detected, the result is based on AI analysis and does not guarantee the absence of a stroke or other medical conditions. It emphasizes that the application is intended for preliminary detection and should not replace professional medical evaluation. Users are encouraged to consult a licensed healthcare professional if they experience symptoms of a stroke or have health concerns.

The page also advises users to prioritize professional medical advice and routine health check-ups for accurate diagnosis and care. To encourage further action, a call-to-action (CTA) button labeled "Look for the Nearby Hospitals for Checking" is included at the bottom. This button ensures that users can take proactive steps to seek medical attention if needed. Besides, the page also added an action button “Download PDF report” for user to download the generated detection report. This can ease user for refer the detection result since the detection result is not saved into the app to ensure data security. Overall, the page balances reassurance with caution, promoting both awareness and responsibility.

Detection Result PDF Page



StrokeSense Detection Report

Date: 05/May/2025
Time: 16:36
File Name: case33-img-00504-00041.png
Confidence: 77.78%

Detection Result: Clot Detected

File Image:

Disclaimer: This report is generated by StrokeSense using AI-based detection. The result is not 100% accurate and should not replace professional medical advice. Always consult a healthcare professional for a definitive diagnosis.

Generated by StrokeSense

StrokeSense Detection Report

Date: 05/May/2025
Time: 20:09
File Name: case33-img-00504-00050.png

Detection Result: No Clot Detected

File Image:

Disclaimer: This report is generated by StrokeSense using AI-based detection. The result is not 100% accurate and should not replace professional medical advice. Always consult a healthcare professional for a definitive diagnosis.

Generated by StrokeSense

Figure 5.62 Detected Result PDF

The StrokeSense Detection Report is designed with a clear and user-friendly layout to present the results of the clot detection process effectively. The report is generated in PDF format, making it accessible for sharing and storage. At the top of the report, the header prominently displays the title, "StrokeSense Detection Report," ensuring clarity about the document's purpose. Below the header, essential details such as the date and time of report generation, along with the file name of the analyzed MRI image, are provided for proper documentation and traceability.

The detection results are presented in a bold and color-coded format to enhance visibility. For cases where a stroke is detected, the text "Stroke Detected" is displayed in bold red, conveying urgency. Conversely, for cases where no stroke is detected, the text "No Stroke Detected" is displayed in bold green, indicating a normal result. This immediate visual differentiation ensures that users can quickly interpret the outcome of the analysis.

The report includes the analysed MRI image prominently in the centre, allowing users to visually inspect the image that was used for detection. Accompanying the image is a disclaimer that emphasizes the limitations of the AI-based detection system. This disclaimer ensures transparency and encourages users to seek professional medical consultation for confirmation.

At the bottom of the report, the statement "Generated by StrokeSense" provides attribution to the system. Additional features, such as options to save the report to Dropbox and tools for editing and searching within the document, enhance the usability and functionality of the report. Overall, the UI of the StrokeSense Detection Report balances simplicity and clarity, making it an effective tool for presenting clot detection results to both medical professionals and general users.

Maps Page

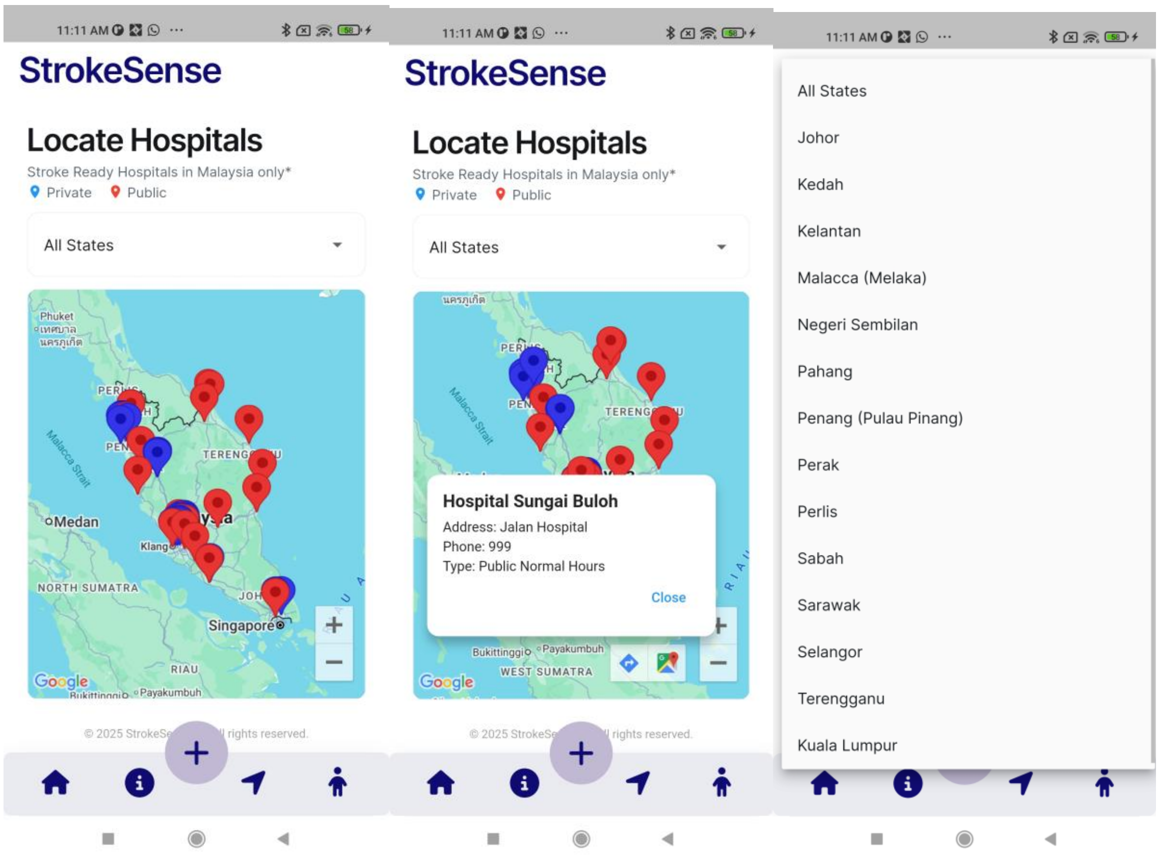


Figure 5.63 Maps Page

The Locate Hospitals page in StrokeSense is designed to help users find hospitals with stroke centers across Malaysia. The page integrates an interactive map and filtering options to provide a user-friendly experience for locating medical facilities. The stroke ready hospital list is provided by **Malaysia Stroke Council** [32].

At the top, the page features a title, "Locate Hospitals," and includes a note clarifying that the locations displayed are specific to Malaysia. Two toggle options, Private and Public, allow users to differentiate between private and public hospitals. These are represented on the map by distinct color-coded icons: blue for private hospitals and red for public hospitals, ensuring visual clarity.

A dropdown menu is provided to narrow down the search to specific states in Malaysia. By default, the dropdown is set to "All States," displaying hospitals from across the country. Users can select a specific state, such as Johor, Selangor, or Kuala Lumpur, to refine their search, making it convenient to locate nearby stroke centers.

The interactive map at the centre of the page displays hospital locations using the color-coded icons. When a user clicks on a location icon, a small card pops up, providing detailed information about the hospital. The card includes Hospital Name, Address, Phone Number, Information of whether it is private or public, along with operating hours.

The map also features standard zooming and panning controls, allowing users to explore different regions easily. At the bottom, the navigation bar ensures seamless access to other app features, maintaining a consistent user experience.

Overall, the Locate Hospitals page is a practical tool for users to quickly identify and access stroke centres, ensuring timely medical attention when needed.

Profile Page

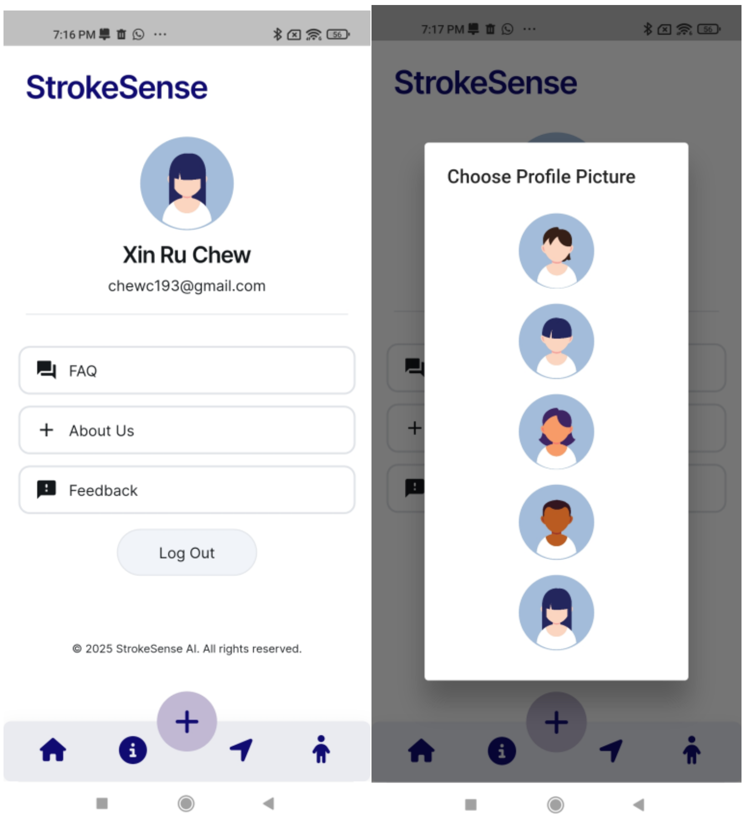


Figure 5.64 Profile Page

The Profile Page in StrokeSense provides users with a personalized space to manage their account information and access various application functionalities. At the top of the page, the user's profile picture is displayed prominently, accompanied by their name and registered email address, offering a quick and clear identification of the logged-in user.

The profile picture is customizable, allowing users to select from a set of predefined avatars. By clicking on the profile picture, a pop-up menu labelled "Choose Profile Picture" appears, presenting a selection of avatar options. This feature adds a layer of personalization, enabling users to tailor their profile to their preferences.

Below the user information, the page includes three functional buttons:

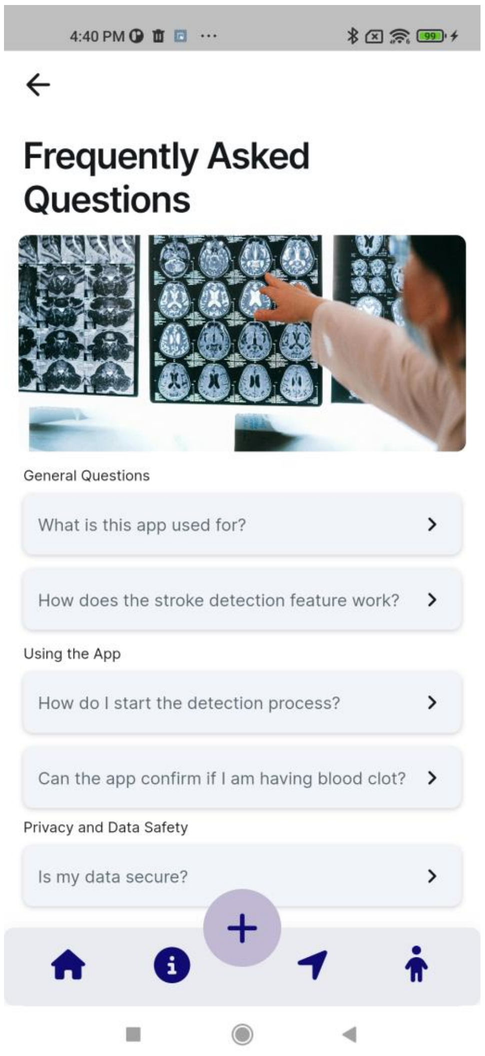
1. **FAQ**: Provides users with answers to frequently asked questions, helping them navigate the app and resolve common queries.
2. **About Us**: Offers information about the StrokeSense app, its purpose, development, and team, fostering transparency and trust.
3. **Feedback**: Gives users an avenue to provide feedback or suggestions, enhancing their engagement and contributing to app improvements.

At the bottom of the page, a Log Out button is available for users to securely exit their account. This ensures privacy and security, especially when the app is accessed on shared devices.

The navigation bar at the bottom of the screen remains consistent with the rest of the app, allowing users to seamlessly switch between different sections, such as the home page and hospital locator. The page is concluded with a footer containing the app's copyright notice.

Overall, the profile page is simple and user-friendly, focusing on personalization, account management, and access to essential resources, ensuring a well-rounded user experience.

Frequently Asked Questions Page



4:40 PM

Frequently Asked Questions

General Questions

What is this app used for?

How does the stroke detection feature work?

Using the App

How do I start the detection process?

Can the app confirm if I am having blood clot?

Privacy and Data Safety

Is my data secure?

Figure 5.65 FAQ Page

The Frequently Asked Questions (FAQ) page in StrokeSense is dedicated to addressing common queries and providing users with clear and concise answers to enhance their understanding of the app. The page is visually structured for ease of navigation, starting with the title "Frequently Asked Questions" prominently displayed at the top, immediately informing users of the page's purpose. Below the title, a banner image reinforces the medical and technical focus of the app.

The FAQ section is organized into three main categories for better clarity and accessibility:

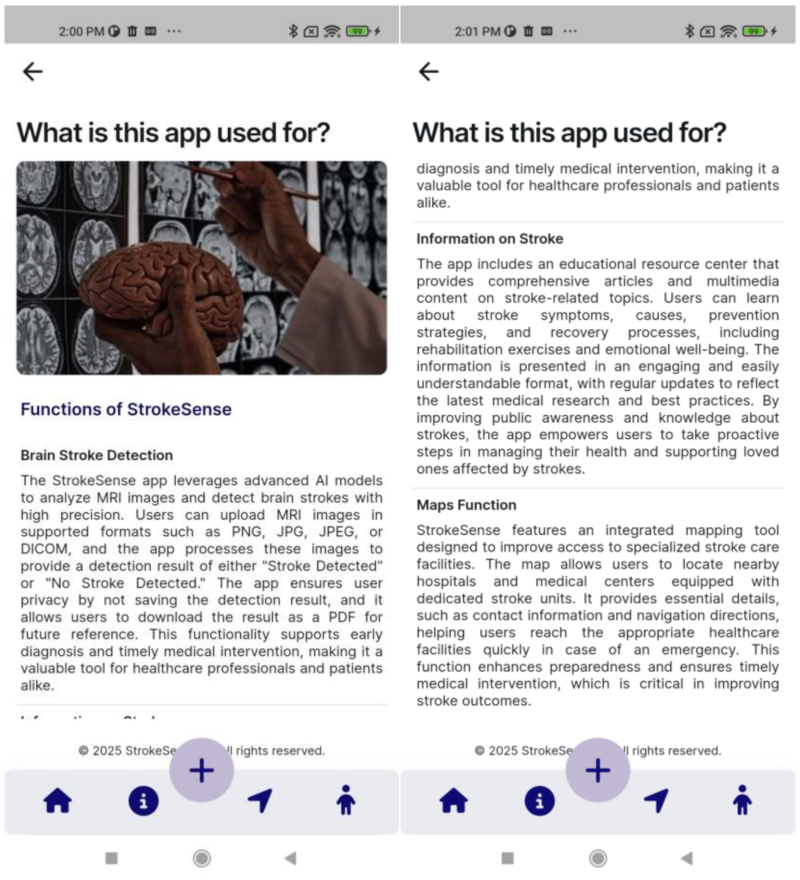
1. **General Questions**: This section explains the app's purpose as a clot detection and awareness tool. It also provides an overview of the AI-based detection process.

2. **Using the App**: This section focused on app functionality, this section guides users like initiating the clot detection feature and clarifies the app's role in providing preliminary detection and its limitations.
3. **Privacy and Data Safety**: This section addresses user concerns about data security, such as assures users of data protection measures and compliance with privacy regulations.

Each question is displayed as an interactive button, allowing users to tap and view detailed answers. This design ensures a clean and clutter-free interface that is easy to navigate. The content focuses on delivering straightforward responses to build user confidence in the app.

The FAQ page is complemented by the navigation bar at the bottom, ensuring users can easily return to other sections of the app. Overall, the page is thoughtfully designed to empower users with the information they need for a seamless and informed experience.

Frequently Asked Questions Single Page



What is this app used for?

Functions of StrokeSense

Brain Stroke Detection

The StrokeSense app leverages advanced AI models to analyze MRI images and detect brain strokes with high precision. Users can upload MRI images in supported formats such as PNG, JPG, JPEG, or DICOM, and the app processes these images to provide a detection result of either "Stroke Detected" or "No Stroke Detected." The app ensures user privacy by not saving the detection result, and it allows users to download the result as a PDF for future reference. This functionality supports early diagnosis and timely medical intervention, making it a valuable tool for healthcare professionals and patients alike.

What is this app used for?

diagnosis and timely medical intervention, making it a valuable tool for healthcare professionals and patients alike.

Information on Stroke

The app includes an educational resource center that provides comprehensive articles and multimedia content on stroke-related topics. Users can learn about stroke symptoms, causes, prevention strategies, and recovery processes, including rehabilitation exercises and emotional well-being. The information is presented in an engaging and easily understandable format, with regular updates to reflect the latest medical research and best practices. By improving public awareness and knowledge about strokes, the app empowers users to take proactive steps in managing their health and supporting loved ones affected by strokes.

Maps Function

StrokeSense features an integrated mapping tool designed to improve access to specialized stroke care facilities. The map allows users to locate nearby hospitals and medical centers equipped with dedicated stroke units. It provides essential details, such as contact information and navigation directions, helping users reach the appropriate healthcare facilities quickly in case of an emergency. This function enhances preparedness and ensures timely medical intervention, which is critical in improving stroke outcomes.

Figure 5.66 FAQ Single Page

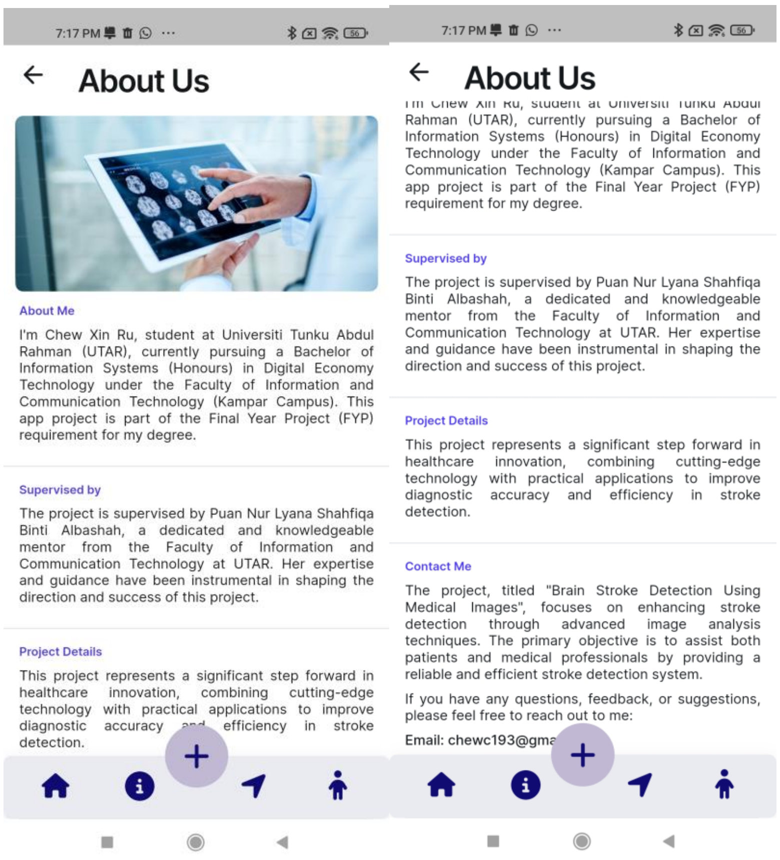
The FAQ detail page in StrokeSense is designed to provide comprehensive yet concise answers to specific user questions, with a clean and structured layout that ensures clarity and ease of navigation. At the top of the page, the question being addressed is prominently displayed as the title, ensuring users immediately know the focus of the content. Below the title, a relevant image visually supports the topic, making the page more engaging and contextual.

The main content section follows, structured with clear headings and brief, informative answers. For example, in this case, the page explains the Functions of StrokeSense and provides a breakdown of its key features, such as Brain Clot Detection, Information on Stroke, and the Maps Function. Each feature is described succinctly, focusing on its purpose and benefits.

The page maintains consistency with the app's overall design by including the persistent navigation bar at the bottom, allowing users to easily switch between other sections of the app, such as the home page or profile. The footer includes a copyright notice, reinforcing the app's credibility.

Overall, the FAQ detail pages aim to provide users with targeted and actionable information, addressing common queries while maintaining a user-friendly and visually cohesive design. These pages are part of the broader FAQ section, which is organized to ensure users can quickly access the information they need.

About Us Page


7:17 PM

About Us

About Me

I'm Chew Xin Ru, student at Universiti Tunku Abdul Rahman (UTAR), currently pursuing a Bachelor of Information Systems (Honours) in Digital Economy Technology under the Faculty of Information and Communication Technology (Kampar Campus). This app project is part of the Final Year Project (FYP) requirement for my degree.

Supervised by

The project is supervised by Puan Nur Lyana Shahfiqa Binti Albashah, a dedicated and knowledgeable mentor from the Faculty of Information and Communication Technology at UTAR. Her expertise and guidance have been instrumental in shaping the direction and success of this project.

Project Details

This project represents a significant step forward in healthcare innovation, combining cutting-edge technology with practical applications to improve diagnostic accuracy and efficiency in stroke detection.

Contact Me

The project, titled "Brain Stroke Detection Using Medical Images", focuses on enhancing stroke detection through advanced image analysis techniques. The primary objective is to assist both patients and medical professionals by providing a reliable and efficient stroke detection system.

If you have any questions, feedback, or suggestions, please feel free to reach out to me:

Email: chewc193@gma

Figure 5.67 About Us Page

The About Us page in StrokeSense provides a detailed overview of the app's background, its purpose, and the individuals involved in its development. The page is structured with multiple sections, each addressing a specific aspect of the project, ensuring clarity and transparency.

About Me: This section introduces the developer, Chew Xin Ru, a student at Universiti Tunku Abdul Rahman (UTAR), pursuing a Bachelor of Information Systems (Honours) in Digital Economy Technology. The project is highlighted as part of the Final Year Project (FYP) requirement, showcasing its academic significance.

Supervised by: This section acknowledges the supervisor, Puan Nur Lyana Shafiqah Binti Albasah, a mentor from the Faculty of Information and Communication Technology at UTAR. Her expertise and guidance are credited with shaping the project’s direction and success.

Project Details: This section outlines the project's objectives and significance. It emphasizes the app's role in healthcare innovation by integrating cutting-edge technology to enhance diagnostic accuracy and efficiency in clot detection. The details highlight the practical applications and the potential impact on healthcare.

Contact Me: The final section provides additional details about the project, titled "Brain Stroke Detection Using Medical Images"* and its mission to assist patients and medical professionals. It invites users to reach out with questions, feedback, or suggestions and includes contact information (email) for communication.

The page is visually consistent with the app's overall design, featuring a navigation bar at the bottom for seamless access to other sections. Each section is clearly labeled, making it easy for users to understand the app's background and purpose. The About Us page effectively builds trust by providing transparency about the app's origins and its development process.

Feedback Page

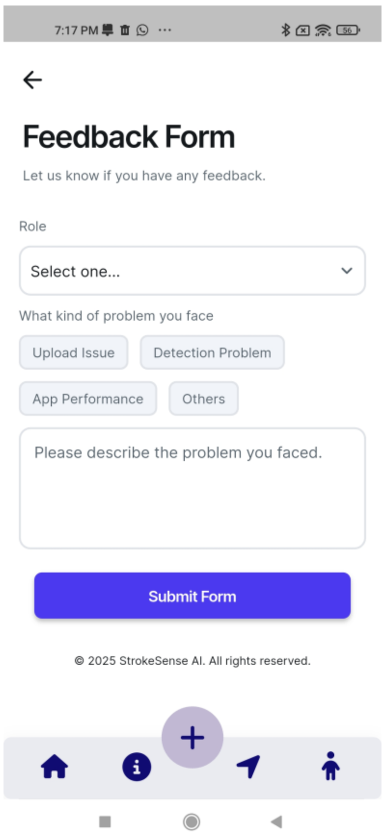

7:17 PM
Feedback Form
Let us know if you have any feedback.
Role
Select one...
What kind of problem you face
Upload Issue
Detection Problem
App Performance
Others
Please describe the problem you faced.
Submit Form
© 2025 StrokeSense AI. All rights reserved.

Figure 5.68 Feedback Page

The Feedback Form page in StrokeSense provides users with a simple and intuitive interface to share their feedback or report issues. The page is structured to ensure users can provide detailed input, helping the developers improve the app's functionality and user experience.

At the top, the page displays the title "Feedback Form" with a brief instruction, "Let us know if you have any feedback," encouraging user participation. Below this, the form is divided into several sections to systematically capture relevant details:

Role Selection: A dropdown menu labeled "Role" allows users to specify their role, such as patient, caregiver, or medical professional. This helps in categorizing the feedback based on the perspective of the user.

Problem Type: A set of predefined buttons enables users to quickly select the type of problem they are facing. The options include Upload Issue for problems related to uploading MRI images, Detection Problem for issues with the clot detection feature, App Performance for feedback on the overall performance of the app, and Others for any other issues not covered by the predefined categories.

Description Field: A text box is provided for users to describe the problem or feedback in detail. This open-ended section allows users to elaborate on their concerns or suggestions.

At the bottom, a prominently displayed "Submit Form" button enables users to send their feedback directly to the developers. The page maintains a clean and user-friendly design, with consistent navigation elements, including a bottom navigation bar for accessing other sections of the app. The footer includes a copyright notice, reinforcing the credibility of the app. Overall, the Feedback Form page is an essential tool for engaging users and gathering actionable insights to enhance the app.

5.6 User Testing and Feedback

After the first deployment of the StrokeSense app, I involved a group of users to test the app and provide valuable feedback to improve its functionality and user experience. During the testing phase, users highlighted a few areas where the app could be enhanced to better meet their needs.

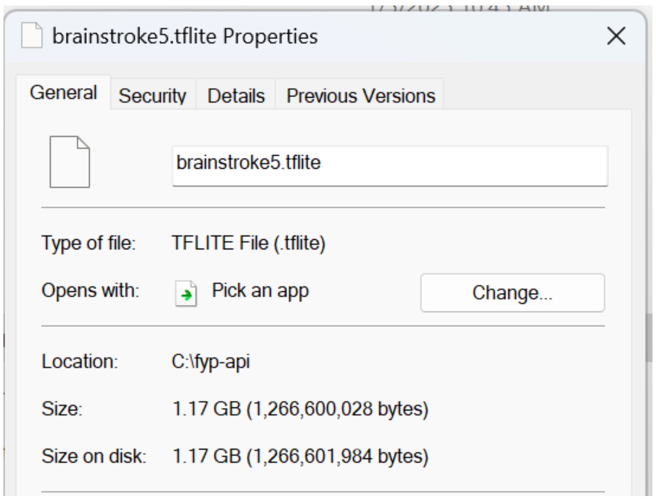
One of the key pieces of feedback was the suggestion to add a **Hospital Map Page**. Users expressed the importance of having a feature that could help them locate nearby hospitals and stroke care facilities. As a result of this feedback, a map functionality was incorporated into the app, allowing users to easily find and navigate to nearby medical facilities equipped to handle stroke emergencies.

Another significant recommendation was to include a **Download PDF** feature for detection results. Users wanted the ability to save their detection results in a secure and portable format for future reference or to share with healthcare professionals. This functionality was promptly added, ensuring that users can download their results directly from the app.

Other general feedback included **improving the app's UI for better navigation** and making the instructions for uploading MRI images clearer. Some users also suggested adding an FAQ section to address common queries and concerns about the app's features and clot detection process.

By incorporating these user-driven improvements, the StrokeSense app has become more user-friendly and effective in addressing the needs of its target audience. The continuous feedback loop ensures that the app evolves to provide better support and value to its users.

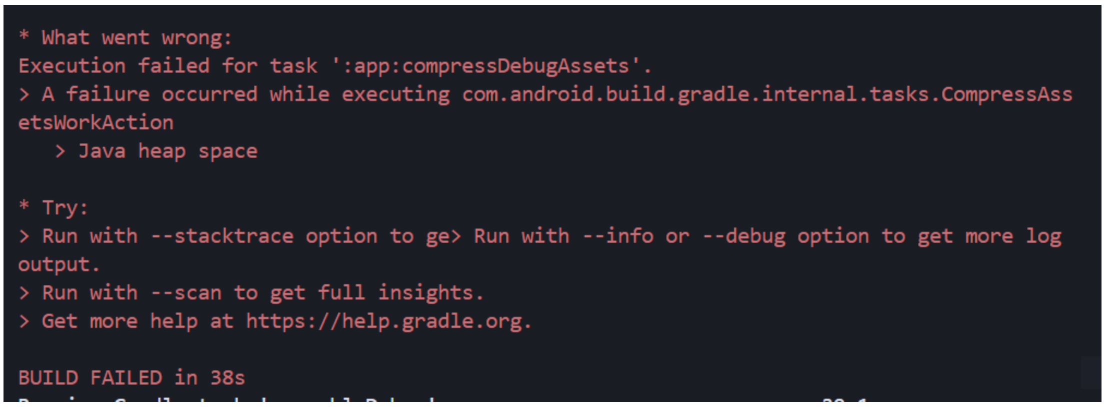
5.7 Implementation Issues and Challenges


brainstroke5.tflite Properties

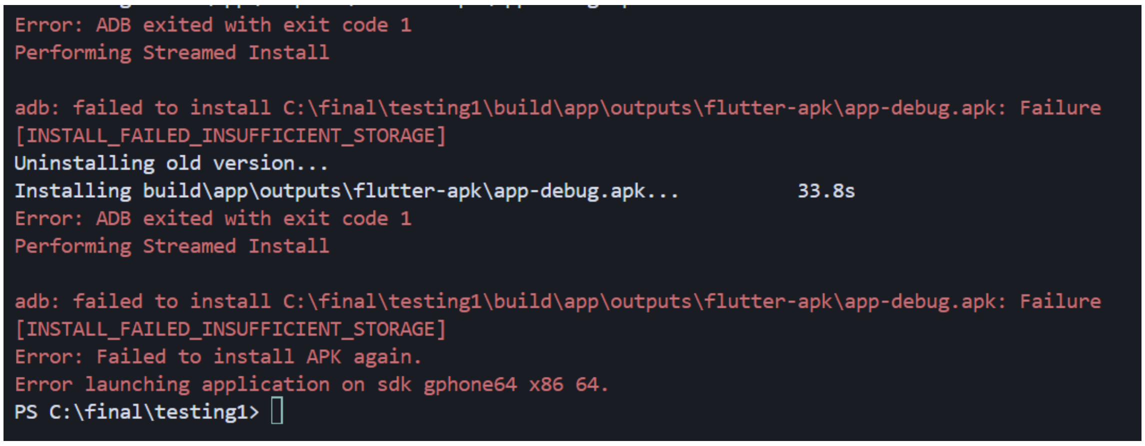
```
brainstroke5.tflite

Type of file:   TFLITE File (.tflite)
Opens with:     Pick an app

Location:       C:\fyp-api
Size:           1.17 GB (1,266,600,028 bytes)
Size on disk:   1.17 GB (1,266,601,984 bytes)
```

Figure 5.69 Brain Clot Detection Model Properties

The first challenge faced during the implementation process was the **excessively large file size of the brain clot detection model**, which stands at 1.17 GB. This made it difficult to manage and integrate the model within the application. The sheer size not only posed storage concerns but also impacted the feasibility of deploying the model for mobile and web applications, where lightweight assets are a priority. This challenge necessitated exploring methods to reduce the model's size without compromising its performance.



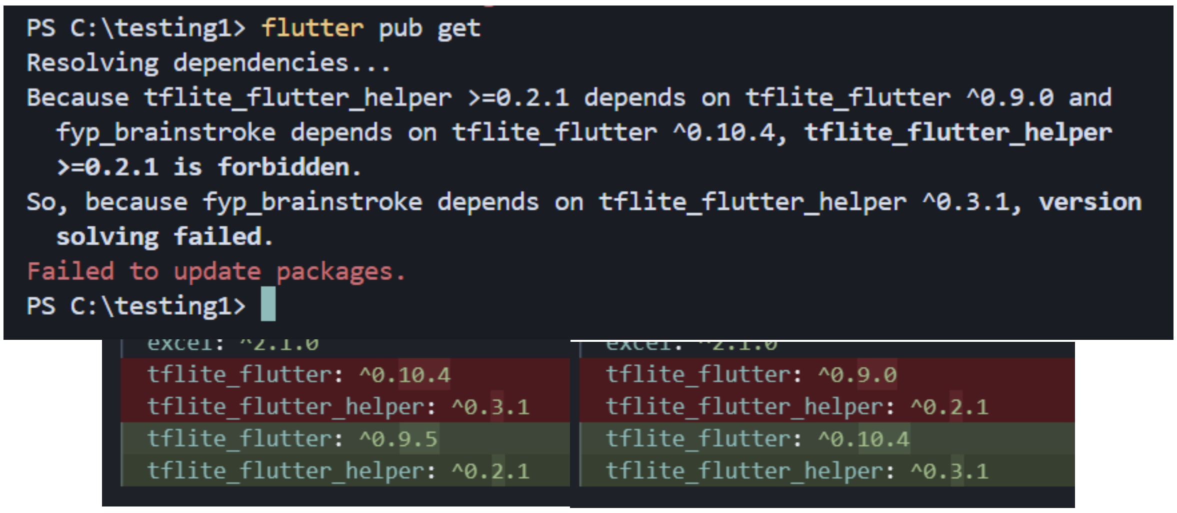
```
* What went wrong:
Execution failed for task ':app:compressDebugAssets'.
> A failure occurred while executing com.android.build.gradle.internal.tasks.CompressAss
etsWorkAction
   > Java heap space

* Try:
> Run with --stacktrace option to ge> Run with --info or --debug option to get more log
output.
> Run with --scan to get full insights.
> Get more help at https://help.gradle.org.

BUILD FAILED in 38s
```



```
Error: ADB exited with exit code 1
Performing Streamed Install

adb: failed to install C:\final\testing1\build\app\outputs\flutter-apk\app-debug.apk: Failure
[INSTALL_FAILED_INSUFFICIENT_STORAGE]
Uninstalling old version...
Installing build\app\outputs\flutter-apk\app-debug.apk...          33.8s
Error: ADB exited with exit code 1
Performing Streamed Install

adb: failed to install C:\final\testing1\build\app\outputs\flutter-apk\app-debug.apk: Failure
[INSTALL_FAILED_INSUFFICIENT_STORAGE]
Error: Failed to install APK again.
Error launching application on sdk gphone64 x86 64.
PS C:\final\testing1>
```

Figure 5.70 Excessive File Size Error

An initial attempt to directly integrate the model into the Flutter project by embedding it within the local file structure proved unsuccessful. Gradle, the build tool used for Flutter projects, encountered errors during the asset compression stage, as the **model surpassed its size-handling limits**. This highlighted the limitations of the current build system in managing large assets and forced a reevaluation of the integration approach.



```
PS C:\testing1> flutter pub get
Resolving dependencies...
Because tflite_flutter_helper >=0.2.1 depends on tflite_flutter ^0.9.0 and
  fyp_brainstroke depends on tflite_flutter ^0.10.4, tflite_flutter_helper
  >=0.2.1 is forbidden.
So, because fyp_brainstroke depends on tflite_flutter_helper ^0.3.1, version
  solving failed.
Failed to update packages.
PS C:\testing1>
```

```
excel: ^2.1.0                       excel: ^2.1.0
tflite_flutter: ^0.10.4             tflite_flutter: ^0.9.0
tflite_flutter_helper: ^0.3.1       tflite_flutter_helper: ^0.2.1
tflite_flutter: ^0.9.5              tflite_flutter: ^0.10.4
tflite_flutter_helper: ^0.2.1       tflite_flutter_helper: ^0.3.1
```

Figure 5.71 Version Incompatible Error

To address compatibility, TensorFlow Lite (TFLite) was adopted, but this introduced new issues in the form of **version conflicts**. Ensuring compatibility between TFLite, the Flutter environment, and other dependencies proved to be a significant challenge. The continuous occurrence of version mismatches made it difficult to achieve a stable configuration, further delaying progress and complicating the implementation process.

Efforts to reduce the model's size focused on **post-training quantization** using TensorFlow Lite with a representative dataset. While this approach successfully minimized the model size to a certain extent, the resulting size was still too large to meet the requirements for smooth integration. This highlighted the limitations of quantization techniques for models of this complexity and scale, necessitating further exploration of alternative optimization strategies.

Since the local implementation hits error, API was used to solve the insufficient storage problem. Running the model using a **Flask API** presented another challenge, as the model initially appeared to function but produced **incorrect outputs**. **Every image detected had the same clot possibility**, indicating that the **model was not working as intended**. This issue emphasized the need for a deeper understanding of the model's requirements and the necessity of debugging the API integration to ensure correct functionality.

Upon further investigation, it was discovered that the model is a **Siamese network**, which operates differently from standard detection models. Unlike typical models that process a single image, the Siamese network **requires a pair of images as input to calculate their similarity**. The Flask API, however, was only passing a single image, which led to erroneous results. To resolve this, a batch of reference images must be prepared and stored in the backend, enabling the uploaded image to be processed correctly by comparing it to the reference images. This adjustment is critical for ensuring the model operates as intended, but it also adds complexity to the implementation process.

5.8 Concluding Remark

In conclusion, the BrainStroke Detection app represents a significant advancement in leveraging AI technology for early clot detection and awareness. The app's development journey encompassed various aspects, including the training process for the clot detection model, which required the careful selection of datasets, preprocessing techniques, and optimization methods to ensure high accuracy and reliability. The project's settings and configurations involved integrating TensorFlow Lite and implementing post-training quantization, while also addressing compatibility issues to create a seamless deployment environment.

The system operations were designed to provide an intuitive and user-friendly experience, incorporating features like clot detection results, hospital locators with comprehensive mapping, and FAQs to enhance user understanding. However, the development process was not without its challenges. Implementation issues and challenges, such as the large model file size, version conflicts, and the unique requirements of the Siamese network, posed significant obstacles that required innovative solutions, including the use of Flask APIs and backend reference images for accurate processing.

Despite these hurdles, the app successfully combines cutting-edge technology with practical applications, aiming to improve diagnostic efficiency and enhance user awareness. The lessons learned from the challenges faced during implementation have laid the groundwork for future improvements and refinements to the system, ensuring its continued effectiveness and reliability in supporting clot detection and healthcare innovation.

Chapter 6

System Evaluation and Discussions

6.1 Functional Testing and Performance Metrics

This section focuses on evaluating the overall performance and functionality of the StrokeSense app. The system testing process includes functional testing to verify that all features, such as login, signup, clot detection, hospital mapping, and user feedback, work as intended. Stress testing is conducted to ensure the app performs reliably under heavy usage, while load testing evaluates its ability to handle multiple simultaneous requests, particularly for the Flask API.

Performance metrics are used to measure the app's accuracy, response time, and efficiency. For instance, the clot detection model is evaluated based on metrics such as similarity to determine its diagnostic accuracy. The app's response time is measured to ensure timely results, while resource utilization is analyzed to optimize performance on mobile devices. These metrics provide insights into the app's robustness and help identify areas for improvement.

6.1.1 Sign Up Testing

The Sign-Up Testing focuses on evaluating the functionality, validations, and user experience of the account registration process. This test ensures that users can create accounts successfully and that all input fields are properly validated to prevent errors or incomplete submissions.

The **objective** of the sign-up testing is to verify that:

1. All input fields (First Name, Last Name, Email, Password, Confirm Password) accept valid data and provide appropriate error messages for invalid inputs.
2. The "Create Account" button is enabled only when all required fields are completed and valid.
3. The checkbox for agreeing to the user agreement must be selected before account creation is allowed.
4. Password and Confirm Password fields match and meet security requirements.
5. Successful account creation redirects users to the appropriate page.

No	Test Case	Inputs	Expected Output	Actual Output	Response Time	Remark
1	Enter valid details	Fill in all fields with valid data (e.g., name, email, password)	The data should be properly recorded in the fields.	The data should be properly recorded in the fields.	1 second	PASS
2	Leave mandatory fields empty	Leave one or more required fields (e.g., First Name, Email) empty	Display error message: "This field is required."	Display error message: "This field is required."	1 second	PASS
3	Enter invalid email format	Enter an invalid email format (e.g., missing "@" or domain	Display error message: "The email address is badly formatted."	Display error message: "The email address is badly formatted."	1 second	PASS
4	Password does not meet criteria	Enter a password that does not meet security requirement (e.g., too short)	Display error message: "Password should be at least 6 characters."	Display error message: "Password should be at least 6 characters."	1 second	PASS
5	Password and Confirm	Enter different values in	Display error message:	Display error message:	1 second	PASS

	Password mismatch	the Password and Confirm Password fields	"Passwords do not match."	"Passwords do not match."		
6	Submit without agreeing to terms	Leave the "I agree to the user agreement" checkbox unchecked and click Create Account	Display error message: "Please agree to the user agreement."	Display error message: "Please agree to the user agreement."	1 second	PASS
7	View password toggle	Click the eye icon in the Password and Confirm Password fields	Toggle between hiding and showing the password input	Toggle between hiding and showing the password input	1 second	PASS
8	Submit sign up form	Click on the "Create Account" button	The sign up form submitted successfully and return to login page	The sign up form submitted successfully and return to login page	1 second (Saved into firebase)	PASS
9	Sign In if already have account	Click on the “Sign In Here” button	Direct to login page from the sign up page	Direct to login page from the sign up page	1 second	PASS

Table 6.1 Sign-Up Testing

The Sign-Up Testing successfully verified all functionalities. The form provides a user-friendly and secure experience for account creation. Future improvements could include implementing password strength indicators and allowing users to view entered passwords by toggling visibility. Additional real-time email validation (e.g., checking for existing accounts) could also enhance the user experience.

6.1.2 Login Testing

The Login Testing focuses on verifying the functionality, validations, and user experience of the login process. This test ensures that registered users can log in securely and efficiently, and that appropriate error messages are displayed for invalid attempts.

The **objective** of the login testing is to ensure that:

1. Users can log in successfully using valid credentials.
2. Input validation is enforced for both the email and password fields.
3. Proper error messages are displayed for invalid email formats, incorrect credentials, or empty fields.
4. The "Sign In" button is enabled only when all required fields are filled.
5. Password visibility toggle works as intended.
6. Users are redirected to the appropriate home page upon successful login.

No	Test Case	Inputs	Expected Output	Actual Output	Response Time	Remark
1	Login with valid credentials	Enter a registered email and correct password	Successfully log in and redirect to the home page	Successfully log in and redirect to the home page	1 second	PASS
2	Login with empty fields	Leave the email and/or password field empty	Display error message: "This field is required."	Display error message: "This field is required."	1 second	PASS

3	Login with incorrect password	Enter a registered email but an incorrect password	Display error message: "The supplied auth credential is incorrect."	Display error message: "The supplied auth credential is incorrect."	1 second	PASS
4	Login with incorrect email	Enter a unregistered email	Display error message: "The supplied auth credential is incorrect."	Display error message: "The supplied auth credential is incorrect."	1 second	PASS
5	Password visibility toggle	Click the eye icon in the Password fields	Toggle between hiding and showing the password input	Toggle between hiding and showing the password input	1 second	PASS
6	Submit with all correct requirement	Fill both fields with valid data and click "Sign In"	Successfully log in and redirect to the home page	Successfully log in and redirect to the home page	1 second	PASS

Table 6.2 Login Testing

The Login Testing successfully validated all key functionalities. The form provides a secure and user-friendly experience for logging in. Future enhancements could include enabling CAPTCHA for additional security, providing real-time feedback for password strength, and integrating biometric authentication options for convenience.

6.1.3 Hospital Mapping Testing

The Hospital Mapping Test evaluates the functionality and usability of the hospital mapping feature within the application. This test ensures that users can differentiate hospitals based on their states, view detailed information for selected hospitals, and access navigation directions seamlessly via Google Maps.

The **objective** of the hospital mapping test is to validate that the dropdown list for states, location icons, and Google Maps integration work as intended. Specifically, the test focuses on ensuring that:

1. Users can select a state from the dropdown list and view hospitals located within that state.
2. Clicking on the location icon provides detailed information about the selected hospital, such as its name, address, and contact details.
3. Clicking on the Google Maps icon opens the Google Maps application with a real-time navigation route to the selected hospital.

No	Test Case	Inputs	Expected Output	Actual Output	Response Time	Remark
1	Navigate to map page	Map icon under navigation bar was clicked	From home page of StrokeSense navigate to map page	From home page of StrokeSense navigate to map page	2 seconds (Fetching API data of labeled hospitals from Google Maps API)	PASS
2	Select a state from the dropdown	Choose a state from the drop down list	Display all the location icons in the selected state	Display all the location icons in the selected state	1 second	PASS
3	View hospital information	Click on the location icon	Display hospital details (name, address, contact information and type) in a card	Display hospital details (name, address, contact information and type) in a card	1 second	PASS
4	Navigate using Google Maps	Click on the Google Maps icon for a hospital	Open Google Maps app with the selected hospital’s icon	Open Google Maps app with the selected hospital’s icon	2 seconds (Transfer from StrokeSense app to Google Maps app)	PASS

			for real-time navigation	for real-time navigation		

Table 6.3 Hospital Mapping Testing

The Hospital Mapping Test successfully verified all functionalities. The feature provides users with a seamless way to identify hospitals in their state, view detailed information, and navigate to the hospital with ease. Future improvements could include enhancing the dropdown list to include search functionality for quicker state selection and adding filters to sort hospitals by specific attributes such as distance or availability of stroke units.

6.1.4 User Feedback Testing

The User Feedback Test focuses on evaluating the functionality and usability of the feedback form within the application. This test ensures that users can effectively provide their feedback or report issues without encountering any errors or disruptions.

The **objective** of the user feedback test is to verify that the feedback form operates as intended, allowing users to:

1. Select their role (e.g., patient, caregiver, or medical professional).
2. Specify the type of issue they are facing (e.g., upload issues, detection problems, app performance, or others).
3. Enter detailed descriptions of the problem in the text field provided.
4. Successfully submit the feedback form for review by the development team.

No	**Test Case**	**Inputs**	**Expected Output**	**Actual Output**	**Response Time**	**Remark**
1	Select User Role	Choose role (Patient, Family of Patient, Medical Staff or Others) from the dropdown list	The dropdown should allow users to select their role, and the selected role should be saved	The dropdown should allow users to select their role, and the selected role should be saved	1 seconds	PASS

2	Select a issue type	Click on the issue type button (Upload Issue, Detection Problem, App Performance)	The selected issue type should be highlighted and properly recorded	The selected issue type should be highlighted and properly recorded	1 second	PASS
3	Enter feedback description	Enter a detailed description of the issue in the text field	The text field should allow users to input text	The text field should allow users to input text	1 second	PASS
4	Submit feedback form	Click on the "Submit Form" button	The feedback form submitted successfully	The feedback form submitted successfully	1 second (Saved into firebase)	PASS
5	Handle missing required fields	Submit the form without selecting a required field (User role, issue type or description)	Display an error message prompting the user to complete all required fields before submission	An error message appears, prompting the user to complete all required fields before submission	1 second	PASS
6	View Submission Confirmatio n Message	Submit the form with all required fields completed	Display a confirmation message: “Feedback submitted successfully!”	Display a confirmation message: “Feedback submitted successfully!”	1 second	PASS

Table 6.4 User Feedback Testing

The User Feedback Form passed all test scenarios successfully. It ensures a seamless and user-friendly experience for reporting issues or providing suggestions.

6.1.5 Detection Testing

The Detection Testing focuses on evaluating the functionality, accuracy, reliability, and responsiveness of the application's detection feature. This test ensures that the detection system performs as expected under different conditions, providing accurate results and proper feedback to the user.

The **objective** of detection testing is to ensure that:

1. The detection feature correctly identifies the target issue or condition (e.g., clot symptoms, anomalies, etc.).
2. The detection system provides accurate and consistent results when tested with various valid inputs.
3. Proper error messages are displayed for unsupported or invalid inputs.
4. The feature responds promptly and provides feedback without unnecessary delays.
5. Users receive actionable recommendations or next steps based on detection results.

No	**Test Case**	**Inputs**	**Expected Output**	**Actual Output**	**Response Time**	**Remark**
1	Detect Valid Input	Provide valid input data (PNG, JPG, JPEG, DICOM)	The system successfully processes the input and provides detection results	The system successfully processes the input and provides detection results	5-15 seconds (Pass to API and perform detection)	PASS
2	Handle invalid input	Provide unsupported or corrupted input data	Display error message: "Invalid input. Please try again with	Display error message: "Invalid input. Please try again with	3-5 second (Read the input format and give response)	PASS

			supported formats."	supported formats."		
3	Display actionable feedback	Provide valid input to the detection system	Display clear and actionable feedback based on the detection results (Disclaimer and CTA)	Display clear and actionable feedback based on the detection results (Disclaimer and CTA)	2 second	PASS
4	Handle high workload	Simulate multiple detection requests continuously	The system processes all requests efficiently without crashing or significant delays	The system processes all requests efficiently without crashing or significant delays	-	PASS

Table 6.5 Detection Testing

The Detection Testing validated that the feature operates reliably and accurately across different scenarios. Future improvements could include enhancing the detection algorithm for more precise analysis, adding support for additional input types, and integrating visual or auditory indicators for improved user experience.

6.1.6 Verification of Detection API Functionality

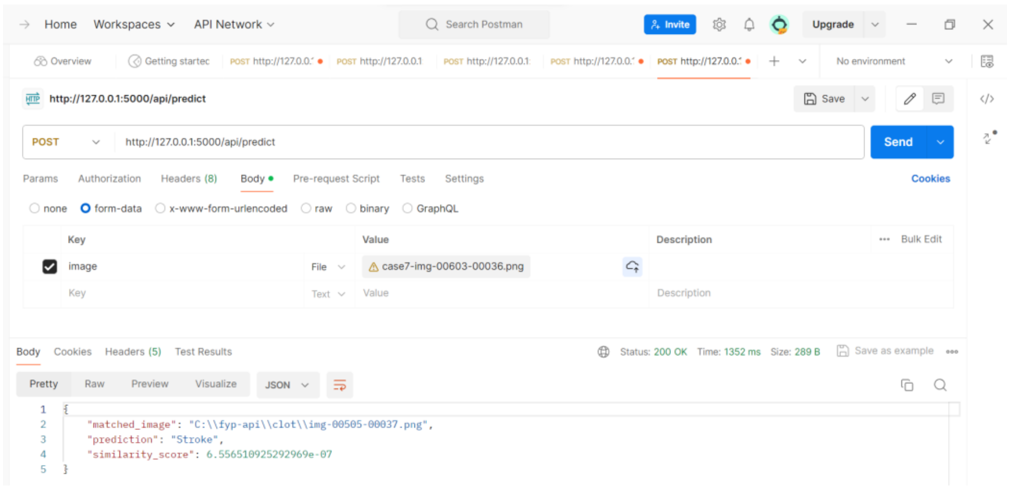
The purpose of this testing is to validate the functionality of the detection model through its API. The goal is to ensure that the detection system performs as expected, processes inputs accurately, and returns reliable results. This test aims to prove that the model works effectively under various scenarios, including valid, invalid, and edge case inputs.

The detection system is a core feature of the application, and its proper functioning is critical to the user experience. By testing the detection, it can be ensured that the model processes valid inputs correctly and delivers accurate results. Also, it verifies error handling for invalid or unsupported inputs. The robustness of the model under edge cases and high usage conditions can be confirmed. It serves as proof that the detection model aligns with the intended requirements and can be relied upon in production.

Test Scenarios:

Valid Input

Submitted a valid base64-encoded image string with the form-data under file and image.



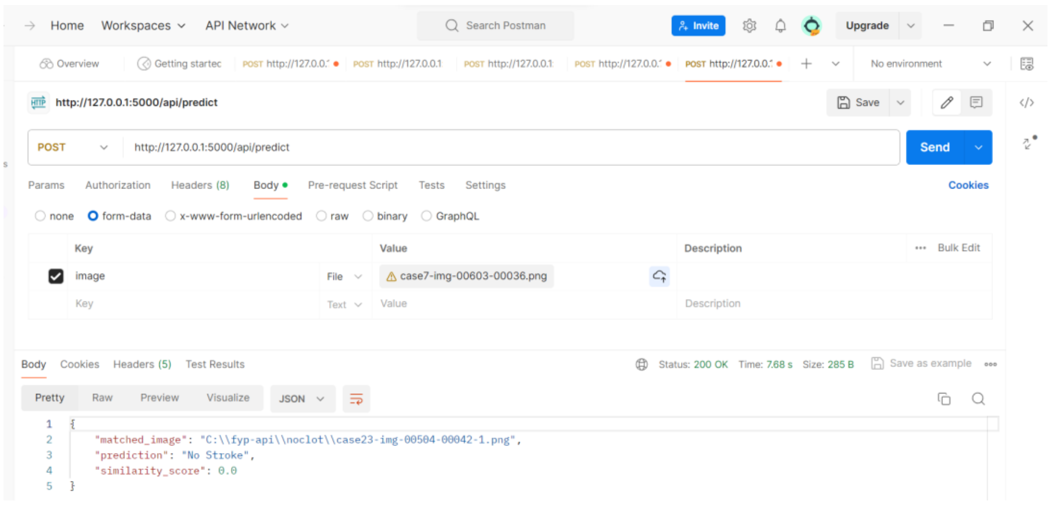
POST http://127.0.0.1:5000/api/predict

Body: form-data — Key: image (File), Value: case7-img-00603-00036.png

Status: 200 OK Time: 1352 ms Size: 289 B

```
{
    "matched_image": "C:\\fyp-api\\clot\\img-00505-00037.png",
    "prediction": "Stroke",
    "similarity_score": 6.556510925292969e-07
}
```

Figure 6.1 Postman Clot Detected



POST http://127.0.0.1:5000/api/predict

Body: form-data — Key: image (File), Value: case7-img-00603-00036.png

Status: 200 OK Time: 7.68 s Size: 285 B

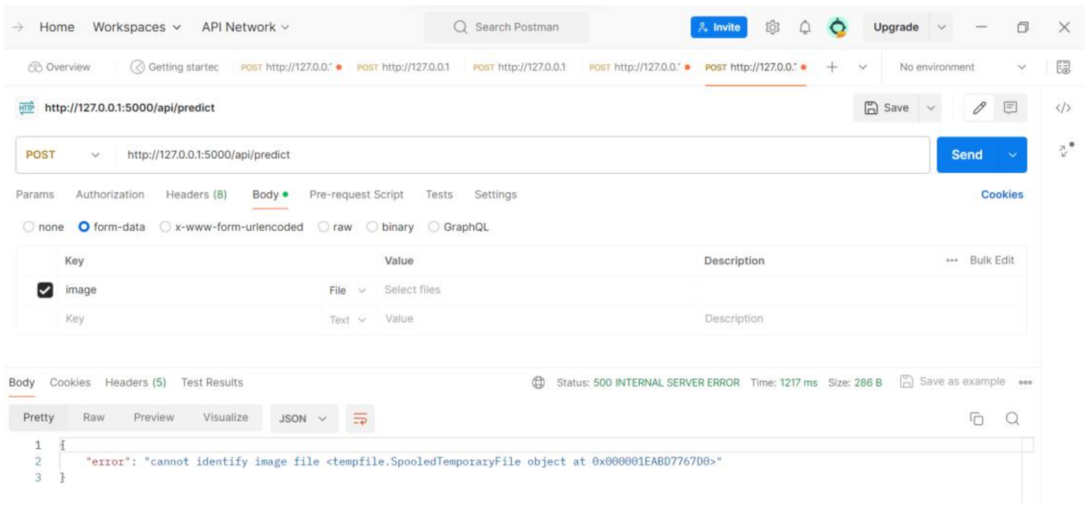
```
{
    "matched_image": "C:\\fyp-api\\noclot\\case23-img-00504-00042-1.png",
    "prediction": "No Stroke",
    "similarity_score": 0.0
}
```

Figure 6.2 Postman No Clot Detected

Both Clot and No-Clot image tested correctly, they shows prediction and the matched image from the training image, and the similarity score with the training image. All requests processed within 10 seconds, server remains stable.

Invalid Input

Submitted a empty JSON body to test error handling.


POST http://127.0.0.1:5000/api/predict

Body: form-data

Key		Value	Description
image	File	Select files	

Status: 500 INTERNAL SERVER ERROR Time: 1217 ms Size: 286 B

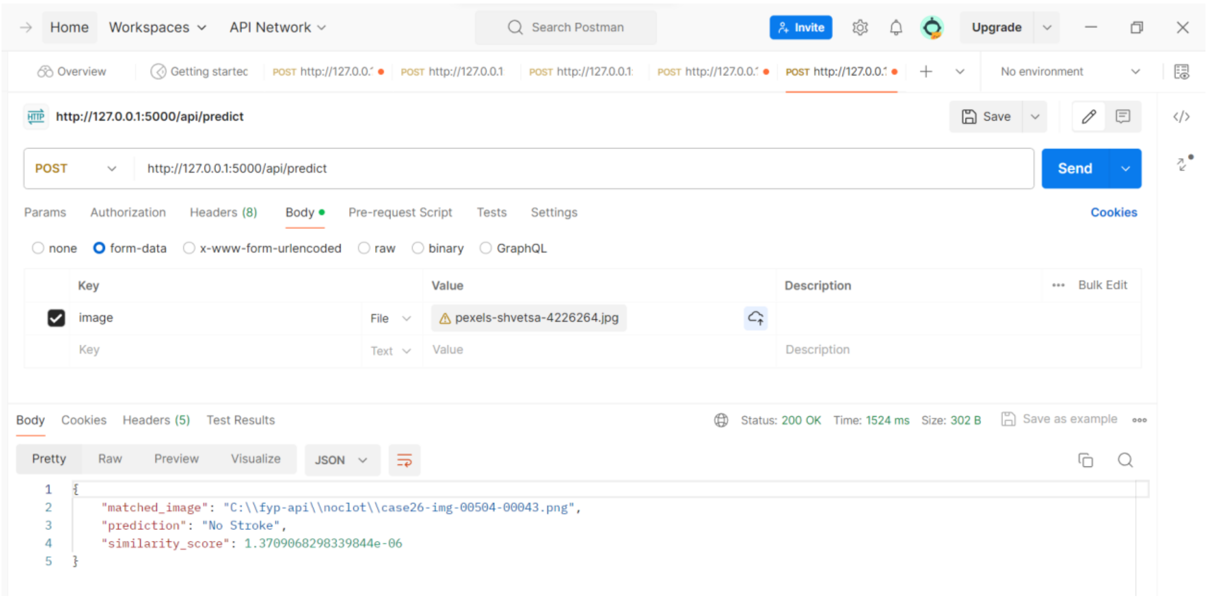
```
{
    "error": "cannot identify image file <tempfile.SpooledTemporaryFile object at 0x000001EABD7767D0>"
}
```

Figure 6.3 Empty Json

With an empty Json tested, it occurs the error that cannot identify the image file. All requests processed within 5 seconds, server remains stable.

Edge Case

Provided borderline inputs (e.g., unclear images) to test robustness.


POST http://127.0.0.1:5000/api/predict

Body: form-data

Key		Value	Description
image	File	pexels-shvetsa-4226264.jpg	

Status: 200 OK Time: 1524 ms Size: 302 B

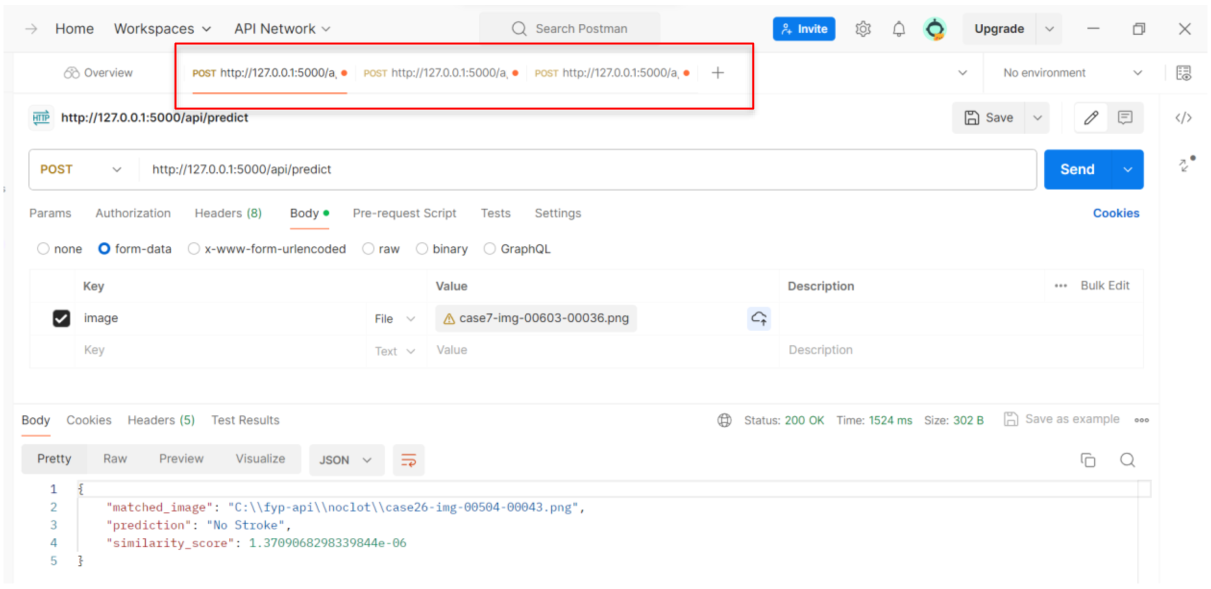
```
{
    "matched_image": "C:\\fyp-api\\noclot\\case26-img-00504-00043.png",
    "prediction": "No Stroke",
    "similarity_score": 1.3709068298339844e-06
}
```

Figure 6.4 Edge Test

With a borderline input, the result is uncertain, but it still can get a similarity score and a result. All requests processed within 5 seconds, server remains stable.

Performance

Simulated 3 detection requests to measure the system's response time and reliability under load.



POST http://127.0.0.1:5000/api/predict

○ none [✓ form-data] ○ x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	Key		Value	Description
✓	image	File	case7-img-00603-00036.png	
	Key	Text	Value	Description

Body Cookies Headers (5) Test Results — Status: 200 OK Time: 1524 ms Size: 302 B

```
{
    "matched_image": "C:\\fyp-api\\noclot\\case26-img-00504-00043.png",
    "prediction": "No Stroke",
    "similarity_score": 1.3709068298339844e-06
}
```

Figure 6.5 Concurrent Requests

Simulated 3 concurrent requests, all requests processed within 3 seconds, server remains stable.

In conclusion, the testing results confirm that the detection model works as intended, accurately identifying valid inputs, handling errors gracefully, and performing reliably under stress. This testing serves as proof that the system is ready for deployment and can meet user expectations effectively. By returning actionable and interpretable results, the detection model can build trust among users, displaying its accuracy and reliability in key scenarios.

6.2 Project Challenges

When undertaking a project, challenges often arise from various domains, including ethics and technical processes. These challenges must be identified and addressed to ensure the success of the project. Below is a detailed exploration of key challenges categorized by their respective areas.

Ethical Challenges

First challenge arise is **data privacy and security**. Managing sensitive user data comes with the responsibility to ensure its privacy and security. Challenges include adhering to regulations like GDPR or HIPAA, preventing unauthorized access, and ensuring secure storage and transmission of data. Failure to address these issues can lead to significant legal and reputational risks. Next, **algorithmic bias and fairness**. If the project involves AI or machine learning, biases in training data or algorithm design can lead to unfair outcomes. Ensuring datasets are diverse and auditing algorithms regularly to mitigate bias is crucial to avoid perpetuating inequalities.

Process Challenges

First, **testing and quality assurance**. Ensuring that the system works as intended under all scenarios requires extensive testing, which can be time-consuming and resource intensive. Next, **time and resource constraints**. Projects often face limitations in time, budget, or workforce, making it difficult to deliver high-quality results on time. Prioritizing core features over secondary ones can help, but this requires careful resource allocation and planning.

Technical Challenges

First, **image preprocessing**. Image preprocessing is vital for projects involving computer vision or detection systems, as raw images often contain noise, varying resolutions, or unwanted artifacts. Challenges include normalizing image sizes, applying filters to remove noise, and ensuring that the preprocessing pipeline does not distort critical image features. Additionally, computational overhead during preprocessing can affect real-time applications, necessitating optimized algorithms. Next, the methodology for training the brain clot detection model. Implementing **few-shot learning models** introduces unique challenges, as these models require learning from limited data samples. Ensuring the model generalizes well across diverse tasks, avoiding overfitting, and finding high-quality training data are significant hurdles.

Moreover, fine-tuning few-shot models while balancing computational efficiency can be resource-intensive.

Besides, **system integration**. Integrating multiple components, such as frontend, backend, and third-party APIs, can lead to compatibility and performance issues. Ensuring seamless communication between these modules is challenging but critical. Next, **scalability and performance**. As the user base grows, the system must handle increased traffic without performance degradation. Designing scalable systems and testing under simulated high loads are necessary to ensure reliability. Also, **innovation vs. feasibility**. Balancing innovative features with practical implementation is often tricky. Cutting-edge technologies may require additional research and resources, but feasibility within the constraints of time and budget must be maintained.

User Feedback

First, **engaging multiple stakeholders**. Collecting feedback from diverse user groups, including patients, caregivers and normal users, adds complexity to the feedback process. Each group has unique needs and priorities, which must be understood and addressed.

Sustainability

Maintenance and updates. Ensuring the system remains functional and relevant over time requires ongoing maintenance, including fixing bugs and updating features to adapt to new technologies. Without proper maintenance, systems can quickly become obsolete.

Conclusion

Project challenges are multifaceted and span ethical, technical, and operational domains. Addressing these challenges requires an initiative-taking approach, effective communication, and continuous improvement. Identifying potential obstacles early and implementing strategies to overcome them ensures that the project meets its objectives while maintaining ethical and professional standards.

6.3 Objectives Evaluation

- **Provide Brain Clot Detection with SWI-MRI Images and Few Shot Models**

This objective focuses on creating a specific model for brain clot detection using SWI-MRI images and Few Shot learning models. The evaluation of this objective involves assessing the system's ability to analyze input data and accurately detect brain clot types using Siamese model. The Few Shot learning model’s ability to generalize from limited training samples is also assessed, focusing on its robustness against unseen MRI images and its resistance to overfitting. Additionally, the effectiveness of the preprocessing pipeline in enhancing critical image features and ensuring consistent input quality is evaluated. The user interface is also a critical component, ensuring it is intuitive and user-friendly for both healthcare professionals and general users. The overall success of this objective is determined by the system’s ability to demonstrate high detection accuracy, practical usability, and reliable performance in real-world clinical settings.

- **Enhance Accessibility and Public Awareness of Stroke Diagnosis and Treatment**

This objective aims to improve public accessibility to stroke treatment and increase awareness about stroke diagnosis and recovery. The evaluation involves verifying the functionality and accuracy of an integrated map displaying all hospitals in Malaysia with dedicated stroke centers, ensuring users can quickly locate specialized medical facilities. The educational resource center is evaluated for its ability to convey critical information effectively, such as stroke symptom recognition using the FAST acronym, understanding rehabilitation exercises, and facilitating emotional recovery through stress management techniques. The content is assessed for clarity, relevance, and its ability to engage users from diverse backgrounds. Furthermore, the system’s ability to provide accurate and up-to-date contact information for support services and management resources is reviewed. User engagement metrics, such as the frequency of accessing educational resources and the time taken to locate stroke centers, are pivotal in evaluating the system’s impact. The success of this objective is determined by the platform’s ability to empower users with knowledge, enhance preparedness, and facilitate timely access to specialized stroke care.

Overall Evaluation

The overall success of the project depends on its ability to seamlessly integrate advanced clot detection technology with accessibility and educational tools. The brain clot detection system should demonstrate high accuracy, usability, and reliability, while the public awareness and accessibility features should empower users with knowledge and enable timely medical intervention. Feedback from stakeholders, including patients, caregivers, healthcare professionals, and the general public, will play a critical role in evaluating the achievement of these objectives. The project's success is defined by its impact on improving stroke diagnosis, treatment accessibility, and public awareness in a meaningful way.

6.4 Concluding Remark

Chapter 6 provides a comprehensive evaluation of the project, encompassing function testing and performance metrics, an in-depth discussion of project challenges, and an analysis of the objectives' achievement.

The function testing and performance metrics demonstrated the robustness of the system in meeting its technical requirements. The testing confirmed the system's ability to accurately process SWI-MRI images, detect brain strokes using Few Shot learning models, and deliver reliable results within acceptable performance thresholds. The evaluation also highlighted areas of improvement, such as optimizing image preprocessing pipelines and ensuring consistent performance under high workloads.

The project challenges were multifaceted, spanning technical, ethical, and process domains. Technical challenges included ensuring the scalability of the system, implementing advanced image preprocessing techniques, and overcoming the complexities of Few-Shot learning with limited datasets. Ethical considerations such as data privacy, algorithmic bias, and inclusivity were addressed to ensure the system aligns with professional standards and user expectations. The process challenges, such as requirements gathering and testing under time constraints, showcased the importance of iterative development and stakeholder engagement in overcoming these hurdles. Collecting feedback from patients, caregivers, normal users, and medical staff underscored the diversity of user needs and the critical role of user-centric design.

The objectives evaluation revealed the project's success in achieving its goals. The brain clot detection system not only delivered accurate and timely results but also provided an intuitive interface that caters to both medical professionals and general users. The integration of a hospital mapping feature and an educational resource center improved accessibility and public awareness of stroke diagnosis and treatment. These features empowered users with essential knowledge and facilitated timely access to specialized care, contributing to the overall impact of the project.

In conclusion, Chapter 6 highlights the project's achievements in delivering a reliable, accessible, and user-friendly blood clot detection system. By addressing challenges effectively and meeting its stated objectives, the project sets a solid foundation for further advancements in clot detection and public health awareness. The insights gained from testing, challenges, and evaluations will guide future iterations and ensure continuous improvement in the system's design and functionality**.**

Chapter 7
Conclusion and Recommendations

7.1 Conclusion

The project **successfully fulfilled its objectives** by providing a reliable blood clot detection system and enhancing accessibility and public awareness of stroke diagnosis and treatment. The integration of advanced Few Shot learning models with SWI-MRI images allowed the detection model to accurately classify various types of brain strokes, demonstrating the potential of modern AI technologies in healthcare. The system achieved significant progress in usability and functionality, making it a practical tool for healthcare professionals, caregivers, and patients alike.

The **methodology**, which followed a **prototyping** approach, played a crucial role in the iterative development of the system. This allowed for continuous improvements based on stakeholder feedback and ensured that the final product met both technical and user requirements. The prototype focused on integrating essential features, such as clot detection, hospital mapping, and educational resources, while maintaining a balance between functionality and ease of use. This iterative process allowed the system to evolve into a user-centric solution.

The **system operation was thoroughly evaluated**, confirming its reliability and efficiency in real-world scenarios. The detection model, supported by robust image preprocessing techniques, demonstrated high performance in terms of accuracy, response time. The app’s user interface (UI) further enhanced the system's usability, offering an intuitive and accessible design that catered to both medical professionals and general users. The UI ensured seamless navigation, enabling users to access clot detection results, hospital maps, and educational resources effortlessly.

However, the project faced several **implementation issues**, including challenges related to data availability for Few Shot learning, resource constraints, and ensuring ethical compliance. These challenges were addressed through careful resource allocation, ethical considerations in data handling, and leveraging advanced preprocessing techniques to compensate for limited data. Despite these hurdles, the system successfully met its objectives.

User feedback was a critical component in shaping the system. Input from patients, caregivers, medical staff, and general users provided valuable insights into improving functionality and usability. Feedback highlighted the importance of features like real-time hospital mapping and educational resources, which significantly enhanced the system's impact. Testing results validated the system's performance and reliability, confirming its readiness for deployment in real-world healthcare environments.

In conclusion, the project demonstrated the successful integration of advanced machine learning models, user-centered design, and practical functionality to deliver a comprehensive clot detection and awareness system. By addressing challenges effectively and incorporating user feedback, the system achieved its goals of improving stroke diagnosis, accessibility, and public awareness. The insights gained from this project provide a sturdy foundation for future enhancements, including expanding the model's capabilities, refining the UI, and scaling the system for broader applications in healthcare.

7.2 Recommendations

Enhance Model Using Grad-CAM for Stroke Localization

To further improve the brain clot detection system, implementing Gradient-weighted Class Activation Mapping (Grad-CAM) is recommended. Grad-CAM can provide visual explanations by highlighting the specific regions in MRI images that contributed to the stroke classification decision. This enhancement would allow the system to not only detect strokes but also localize the affected areas within a complete MRI image, offering valuable insights for healthcare professionals. Such visual feedback could improve diagnostic confidence, aid medical interpretation, and foster trust in the system’s capabilities, making it a more comprehensive tool for stroke analysis.

Improve Educational Resources

The educational resource centre should be expanded and enriched to ensure users gain a deeper understanding of stroke symptoms, treatment, and recovery. Adding multimedia content such as interactive videos, infographics, and animations can make the resources more engaging and easier to comprehend. Additionally, multilingual support should be provided to cater to users from diverse linguistic backgrounds. Regular updates with the latest medical research and best practices will ensure the content remains relevant and valuable. These improvements will empower users with the knowledge to recognize stroke symptoms early, facilitate recovery, and support caregivers in managing post-stroke care effectively.

Collaborate with Healthcare Institutions

Partnering with healthcare institutions, such as hospitals, clinics, and government health agencies, can significantly enhance the system’s adoption and impact. Collaboration can provide access to more diverse and extensive datasets for model training and validation, ensuring the system’s robustness and accuracy across different patient demographics. Moreover, integrating the system into existing healthcare workflows will make it more accessible to medical professionals and patients. Healthcare institutions can also assist in promoting the system through awareness campaigns, increasing public trust and acceptance. Such partnerships would further validate the system's reliability and establish its role as a critical tool in stroke diagnosis and treatment.

REFERENCES

[1]V. L. Feigin et al., “Global and regional burden of stroke during 1990–2010: findings from the Global Burden of Disease Study 2010,” The Lancet, vol. 383, no. 9913, pp. 245–255, Jan. 2014, doi: https://doi.org/10.1016/s0140-6736(13)61953-4.

[2]K. S. Tan and N. Venketasubramanian, “Stroke Burden in Malaysia,” Cerebrovascular Diseases Extra, vol. 12, no. 2, pp. 58–62, Mar. 2022, doi: https://doi.org/10.1159/000524271.

[3]World Health Organization, “Medical doctors (per 10 000 population),” www.who.int, 2022. https://www.who.int/data/gho/data/indicators/indicator-details/GHO/medical-doctors-(per-10-000-population)

[4]Nik Nasihah Nik Ramli, Deviga Genasan, & Norzawani Shafika Rossman. (2024). Assessing the Awareness on Symptoms and Risk Factors of Stroke Amongst Rural Community in Central Region of Malaysia: a Cross-Sectional Survey. The Malaysian Journal of Medical Science, 31(1), 150–160. https://doi.org/10.21315/mjms2024.31.1.13

[5]U. R. Acharya, S. L. Oh, Y. Hagiwara, J. H. Tan, H. Adeli, and D. P. Subha, “Automated EEG-based screening of depression using deep convolutional neural network,” Computer Methods and Programs in Biomedicine, vol. 161, pp. 103–113, Jul. 2018, doi: https://doi.org/10.1016/j.cmpb.2018.04.012.

[6]M. Kaur, S. R. Sakhare, K. Wanjale, and F. Akter, “Early Stroke Prediction Methods for Prevention of Strokes,” Behavioural Neurology, vol. 2022, p. e7725597, Apr. 2022, doi: https://doi.org/10.1155/2022/7725597.

[7]J. S. Kumar and P. Bhuvaneswari, “Analysis of Electroencephalography (EEG) Signals and Its Categorization–A Study,” Procedia Engineering, vol. 38, pp. 2525–2536, 2012, doi: https://doi.org/10.1016/j.proeng.2012.06.298.

[8]H. A. Adhi, S. K. Wijaya, Prawito, C. Prawito, and M. Rezal, “Automatic Detection of Ischemic Stroke Based on Scaling Exponent Electroencephalogram Using Extreme Learning Machine,” Automatic Detection of Ischemic Stroke Based on Scaling Exponent

Electroencephalogram Using Extreme Learning Machine, vol. 10.1088/1742-6596/820/1/012005, Jul. 2017.

[9]G. B. Schmid, H. H. Stassen, and R. M. Dünki, "Intraindividual Specificity and Stability of Human EEG: Comparing a Linear Vs a Nonlinear Approach," Methods of Information in Medicine, vol. 39, no. 01, pp. 78–82, 2000, doi: https://doi.org/10.1055/s-0038-1634249.

[10]R. Karthik, R. Menaka, A. Johnson, and S. Anand, "Neuroimaging and deep learning for brain stroke detection - A review of recent advancements and future prospects," Computer Methods and Programs in Biomedicine, vol. 197, p. 105728, Dec. 2020, doi: https://doi.org/10.1016/j.cmpb.2020.105728.

[11]X. Hu et al., "Brain SegNet: 3D local refinement network for brain lesion segmentation," BMC Medical Imaging, vol. 20, no. 1, Feb. 2020, doi: https://doi.org/10.1186/s12880-020-0409-2.

[12]M. Nguyen, Hyun Jeong Kim, Hong Gee Roh, Y.-S. Cho, and Jin Tae Kwak, "Deep Regression Neural Networks for Collateral Imaging from Dynamic Susceptibility contrast-enhanced Magnetic Resonance Perfusion in Acute Ischemic Stroke," International Journal of Computer Assisted Radiology and Surgery, vol. 15, no. 1, pp. 151–162, Sep. 2019, doi: https://doi.org/10.1007/s11548-019-02060-7.

[13]L. Liu, F.-X. Wu, and J. Wang, "Efficient multi-kernel DCNN with Pixel Dropout for Stroke MRI Segmentation," vol. 350, pp. 117–127, Jul. 2019, doi: https://doi.org/10.1016/j.neucom.2019.03.049.

[14]S. Bhattacharjee, Sujata Ghatak, S. Dutta, B. Chatterjee, and Manash Ranjan Gupta, "A Survey on Comparison Analysis between EEG Signal and MRI for Brain Stroke Detection," Advances in intelligent systems and computing, pp. 377–382, Sep. 2018, doi: https://doi.org/10.1007/978-981-13-1501-5_32.

[15]S. Y. Semenov and D. R. Corfield, “Microwave Tomography for Brain Imaging: Feasibility Assessment for Stroke Detection,” International Journal of Antennas and Propagation, vol. 2008, pp. 1–8, 2008, doi: https://doi.org/10.1155/2008/254830.

[16]N. Irishina and A. Torrente, “Brain Stroke Detection by Microwaves Using Prior Information from Clinical Databases,” Abstract and Applied Analysis, vol. 2013, p. e412638, May 2013, doi: https://doi.org/10.1155/2013/412638.

[17]Aryan Jadon, “Image Classification using Few-Shot Learning - Aryan Jadon - Medium,” Medium, Oct. 26, 2022. https://medium.com/@aryanjadon/image-classification-using-few-shot-learning-286572222b2d (accessed Aug. 28, 2024).

[18]Punyakeerthi BL, “Few-Shot Inference: Learning with Just a Glimpse - Punyakeerthi BL - Medium,” Medium, May 07, 2024. https://medium.com/@punya8147_26846/few-shot-inference-learning-with-just-a-glimpse-786fc95ec640#:~:text=In%20traditional%20supervised%20learning%2C%20the (accessed Aug. 28, 2024).

[19]K. Cao, M. Brbić, and J. Leskovec, “Published as a conference paper at ICLR 2021 CONCEPT LEARNERS FOR FEW-SHOT LEARNING,” Mar. 2021. Accessed: Aug. 28, 2024. [Online]. Available: https://arxiv.org/pdf/2007.07375

[20]“Everything you need to know about Few-Shot Learning,” Paperspace Blog, Jun. 29, 2022. https://blog.paperspace.com/few-shot-learning/

[21]“What Is Few-Shot Learning? | IBM,” www.ibm.com, Apr. 11, 2024. https://www.ibm.com/topics/few-shot-learning#:~:text=IBM-

[22]Brainomix Ltd. (2020, March 9). Brainomix 360 mobile. Retrieved May 7, 2025, from App Store website: https://apps.apple.com/us/app/brainomix-360-mobile/id1443749413

[23]Putu Angga Risky Raharja. (2020, August 28). AF Stroke Risk. Retrieved May 7, 2025, from App Store website: https://apps.apple.com/nz/app/af-stroke-risk/id1508729810?platform=iphoneViz.ai, I. (2018, May 20).

[24]Viz Training AI Stroke Care. Retrieved May 7, 2025, from App Store website: https://apps.apple.com/jp/app/viz-training-ai-stroke-care/id1381220026?l=en-US

[25]Nik Nasihah Nik Ramli, Deviga Genasan, & Norzawani Shafika Rossman. (2024). Assessing the Awareness on Symptoms and Risk Factors of Stroke Amongst Rural Community in Central Region of Malaysia: a Cross-Sectional Survey. The Malaysian Journal of Medical Science, 31(1), 150–160. https://doi.org/10.21315/mjms2024.31.1.13Volchko, J. (2017, June). Prototyping Methodology: Steps on How to Use It Correctly. Retrieved from Lumitex.com website: https://www.lumitex.com/blog/prototyping-methodology

[26]P. Kanani and M. Padole, "Deep Learning to Detect Skin Cancer Using Google Colab ," Deep Learning to Detect Skin Cancer Using Google Colab , vol. 8, no. 6, pp. 2249–8958, Aug. 2019, doi: https://doi.org/10.35940/ijeat.F8587.088619.

[27]R. Ramchandra Uppari, "Comparison between KERAS Library and FAST.AI Library Using Convolution Neural Network(Image Classification) Model.," Esource.dbs.ie, 2020. https://esource.dbs.ie/bitstream/10788/4249/1/msc_uppari_rr_2020.pdf (accessed Aug. 28, 2024).

[28]A. Boatman, "JPG vs. PNG: Which Should I Use? | Blog | TechSmith," Welcome to the TechSmith Blog, Mar. 14, 2017. https://www.techsmith.com/blog/jpg-vs-png/.

[29]P. G. Vaz, J. Cardoso, M. Morgado, D. Amaral, and L. F. R. Ferreira, Image Quality of Compressive single-pixel Imaging Using Different Hadamard Orderings, vol. 28, no. 8, 2020, Accessed: Aug. 28, 2024. [Online]. Available: https://opg.optica.org/oe/fulltext.cfm?uri=oe-28-8-11666&id=429765

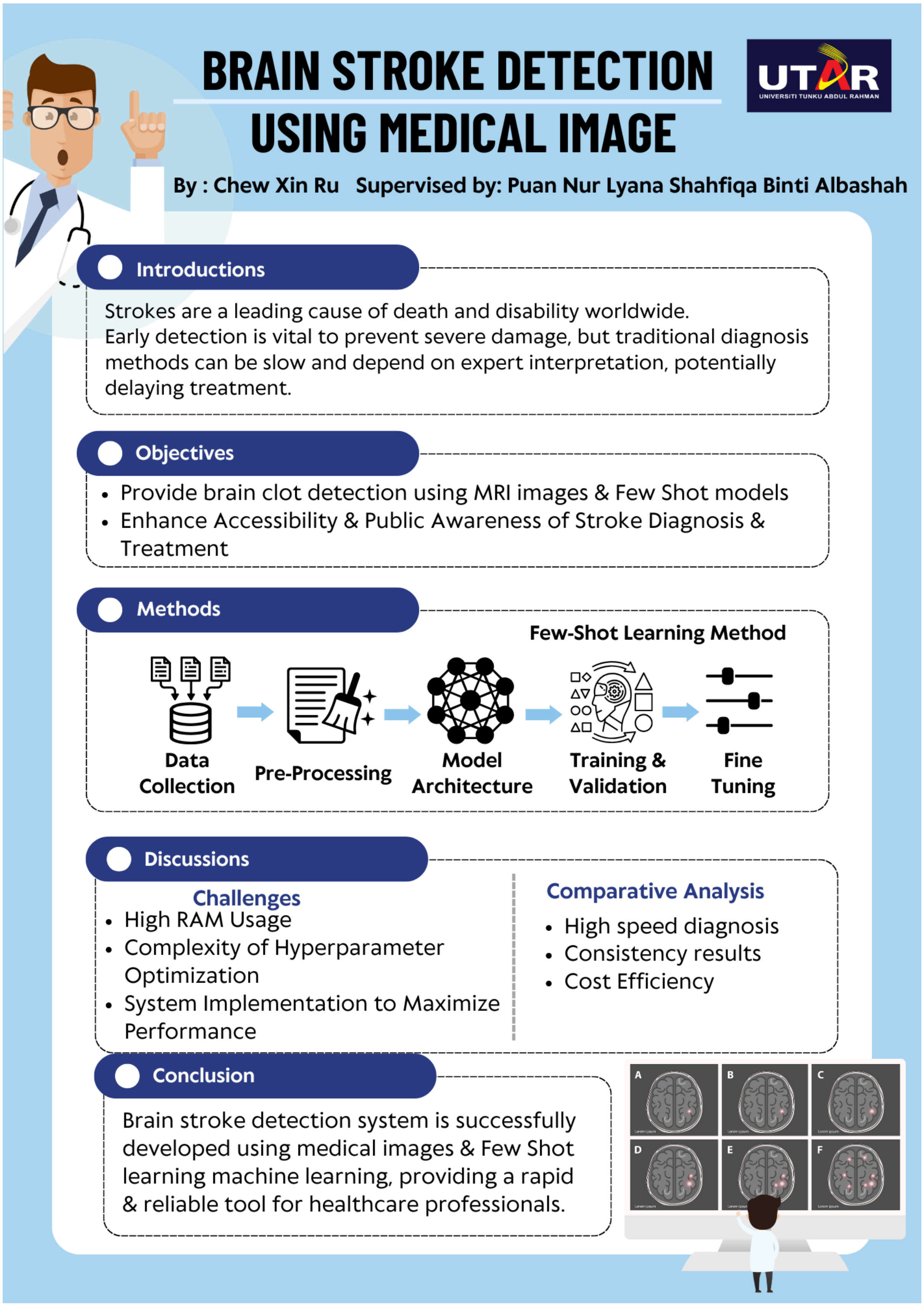
[30]G. Chen, L. Chacón, and T. B. Nguyen, "An Unsupervised machine-learning checkpoint-restart Algorithm Using Gaussian Mixtures for particle-in-cell Simulations," Journal of Computational Physics, vol. 436, p. 110185, Jul. 2021, doi: https://doi.org/10.1016/j.jcp.2021.110185.

[31]D. Danon, M. Arar, D. Cohen-Or, and A. Shamir, “Image Resizing by Reconstruction from Deep Features,” Computational Visual Media, vol. 7, no. 4, Apr. 2021, doi: https://doi.org/10.1007/s41095-021-0216-x.

[32]None Al-Khowarizmi and Halim Maulana, “The Utilization of Gaussian Filter Method on Voice Record Frequency Noise,” Oct. 2020, doi: https://doi.org/10.1109/icoris50180.2020.9320753.

[33]My Stroke Hospitals. (2025). Retrieved May 7, 2025, from Boehringer-ingelheim.com website: https://patient.boehringer-ingelheim.com/my/my-stroke-hospital/locate

POSTER



BRAIN STROKE DETECTION USING MEDICAL IMAGE

UTAR UNIVERSITI TUNKU ABDUL RAHMAN

By : Chew Xin Ru   Supervised by: Puan Nur Lyana Shahfiqa Binti Albashah

Introductions

Strokes are a leading cause of death and disability worldwide.
Early detection is vital to prevent severe damage, but traditional diagnosis methods can be slow and depend on expert interpretation, potentially delaying treatment.

Objectives

- Provide brain clot detection using MRI images & Few Shot models
- Enhance Accessibility & Public Awareness of Stroke Diagnosis & Treatment

Methods

Few-Shot Learning Method

Data Collection → Pre-Processing → Model Architecture → Training & Validation → Fine Tuning

Discussions

Challenges

- High RAM Usage
- Complexity of Hyperparameter Optimization
- System Implementation to Maximize Performance

Comparative Analysis

- High speed diagnosis
- Consistency results
- Cost Efficiency

Conclusion

Brain stroke detection system is successfully developed using medical images & Few Shot learning machine learning, providing a rapid & reliable tool for healthcare professionals.