A MOBILE CLOUD SOLUTION FOR JAPANESE KOI RECOGNITION AND RECOMMENDATION SYSTEM

BY

TAN WEI THONG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONOURS) INFORMATION SYSTEMS ENGINEERING

Faculty of Information and Communication Technology

(Kampar Campus)

JUNE 2024

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: A mobile cloud solution for Japanese Koi recommendation and recognition system

Academic Session: June 2024

I TAN WEI THONG

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

(Author's signature)

Verified by,

(Supervisor's signature)

Address:
793, Jalan Besar,
14200 Sungai Bakap
Pulau Pinang

Ts. Dr. Cheng Wai Khuen

Supervisor's name

Date: 9/9/2024

Date: 9/9/2024

FACULTY/INSTITUTE* OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 9 September 2024

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that ***TAN WEI THONG*** (ID No: ***21ACB05669***) has completed this final year project/ dissertation/ thesis* entitled "**A MOBILE CLOUD SOLUTION FOR JAPANESE KOI RECOGNITION AND RECOMMENDATION SYSTEM**" under the supervision of **Ts Dr Cheng Wai Khuen** (Supervisor) from the Department of **Computer Science** Faculty/Institute* of **INFORMATION AND COMMUNICATION TECHNOLOGY** , and Ts Yong Tien Fui, Albert (Co-Supervisor)* from the Department of **Informations System**, Faculty/Institute* of **INFORMATION AND COMMUNICATION TECHNOLOGY** .

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

TAN WEI THONG

*Delete whichever not applicable

DECLARATION OF ORIGINALITY

I declare that this report entitled "**entitled "A MOBILE CLOUD SOLUTION FOR JAPANESE KOI RECOGNITION AND RECOMMENDATION SYSTEM**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : TAN WEI THONG

Date : 9/9/2024

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Ts Dr Cheng Wai Khuen, and my moderator, Ts Yong Tien Fui for providing me with the opportunity to engage in the field study of Distributed Computing Systems. This opportunity marks my first step towards establishing a career in Cloud Architecture design. Ts Dr Cheng has consistently provided guidance, shared numerous interesting ideas, and helped me refine my project while overcoming numerous obstacles. Ts Yong Tien Fui provide me guidance and give me a lot of ideas for creating the system. Once again, I extend my heartfelt gratitude to my supervisor and moderator.

ABSTRACT

This project introduces a mobile cloud solution for Japanese Koi recognition and recommendation, designed to address significant challenges in the identification and grading of these captivating fish. The current lack of user-friendly technology for recognizing various types of koi fish, coupled with the time-consuming nature of learning to identify them, motivated the development of this innovative system. Additionally, the high market and business value of koi fish, often leading to fraud in purchases, emphasizes the need for a reliable recognition tool in the koi-fish-keeping community. The proposed system leverages Rapid Application Development (RAD) phased development methodology, encompassing planning, analysis, design, and implementation. Notably, it aims to bridge the learning gap for new enthusiasts by providing a convenient and accessible mobile platform for koi fish identification. The project utilizes cutting-edge technology, including cloud computing and Artificial Intelligence, to develop a system capable of accurately identifying koi fish breeds. A key feature of this system is real-time koi detection, allowing users to capture and recognize koi fish instantly through their mobile devices. The motivation behind this project extends beyond academic interests, addressing the practical needs of koi fish enthusiasts. The mobile application, designed for ease of use, allows users to upload images or capture live photos of koi fish for instant recognition. This not only streamlines the learning process but also promotes healthy koi trading by helping users confidently identify and differentiate various grades of Japanese koi. To enhance the learning experience, the system integrates a Chatbot, a large language model, providing users with interactive guidance and a dynamic educational environment. Furthermore, the system contributes to fraud prevention in koi fish trading by ensuring a confident environment for identifying different species and grades. By utilizing a diverse collection of koi fish datasets for training, the system employs cloud computing and AI to store and process information efficiently. The trained model, stored in the cloud, analyzes and predicts koi fish images uploaded or captured in real-time by users, offering detailed information such as type, size, and background. In conclusion, this project's contributions lie in reducing the learning gap for new koi enthusiasts, promoting healthy koi trading practices, and providing a comprehensive, user-friendly platform for recognizing the value of Japanese Koi.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AWS	Amazon Web Service

Chapter 1

Introduction

1.1 Background Information

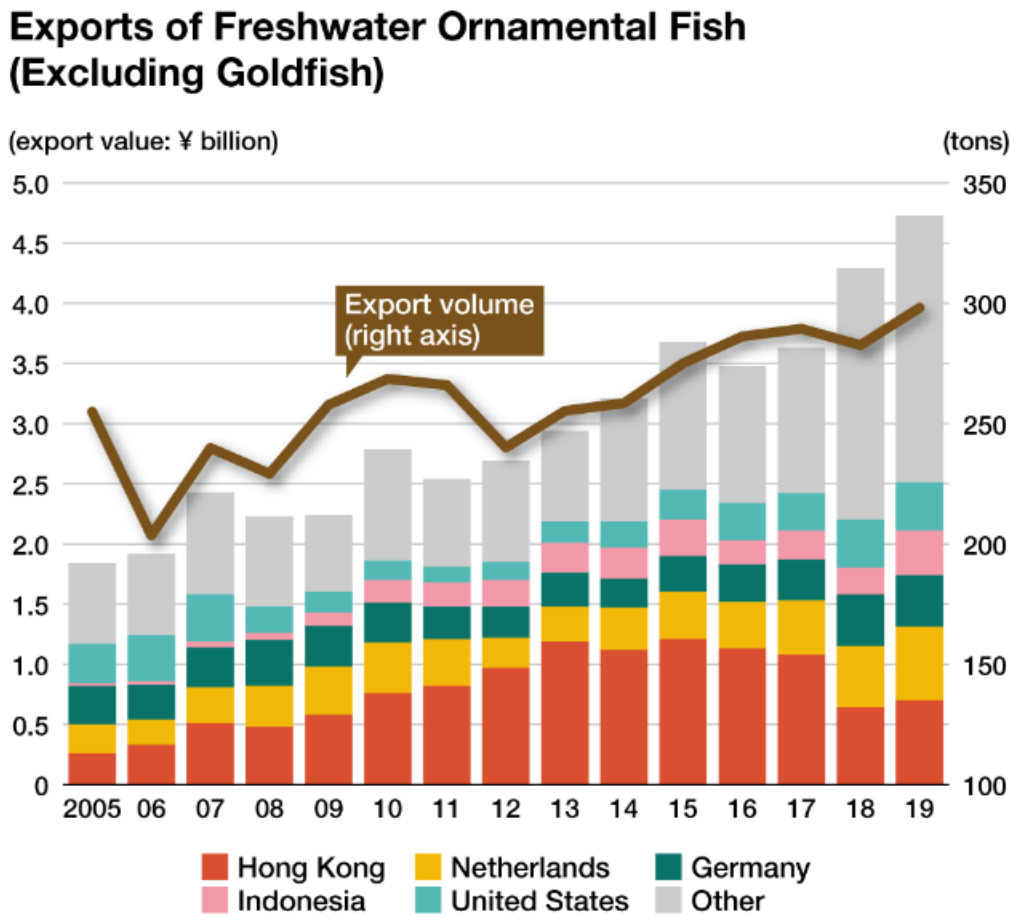


Figure 1.1: The Statistic of Japan's exporting of freshwater ornamental fish (Koi) from 2005 to 2019

Based on the statistics above, the volume of Japan's exports of freshwater ornamental fish, specifically koi, to Hong Kong, the Netherlands, Germany, Indonesia, the United States, and other countries between 2005 and 2019 gradually increased from approximately 200 tons to 330 tons [1]. In terms of price, rarity, beauty, and investment, koi have become popular among people intending to breed fish, especially wealthy individuals. This popularity is attributed to the fact that koi can be seen leisurely swimming in the ponds of homes with spacious gardens, implying that breeding koi is a characteristic of affluent individuals. However, as a beginner, you might purchase an expensive koi from a scammer and later realize that it is a cheaper variety or, worse, not realize the deception at all.

Figure 1.2: different type of koi fishes

Based on the figure above, there are many kinds of koi fish in terms of color, characteristics, and size [2].However, there are over 100 types of koi fish in the world. While it may be difficult to recognize all the different types of koi fish, it is important to learn as much as possible about the various varieties to ensure the proper care and maintenance of these beautiful creatures. Koi fish require specific water conditions, a balanced diet, and regular maintenance to thrive, so it is essential to have a good understanding of their needs. Even though we can search for information about koi on the internet, as mentioned, there are over 100 types of koi fish. Although the internet can be a useful resource for learning about koi fish, it takes time to acquire knowledge, and the process can be overwhelming and confusing. Sorting through the available information and finding reliable sources can be time-consuming. Furthermore, not all information found on the internet is accurate or trustworthy, so it is crucial to use caution and verify information from multiple sources.

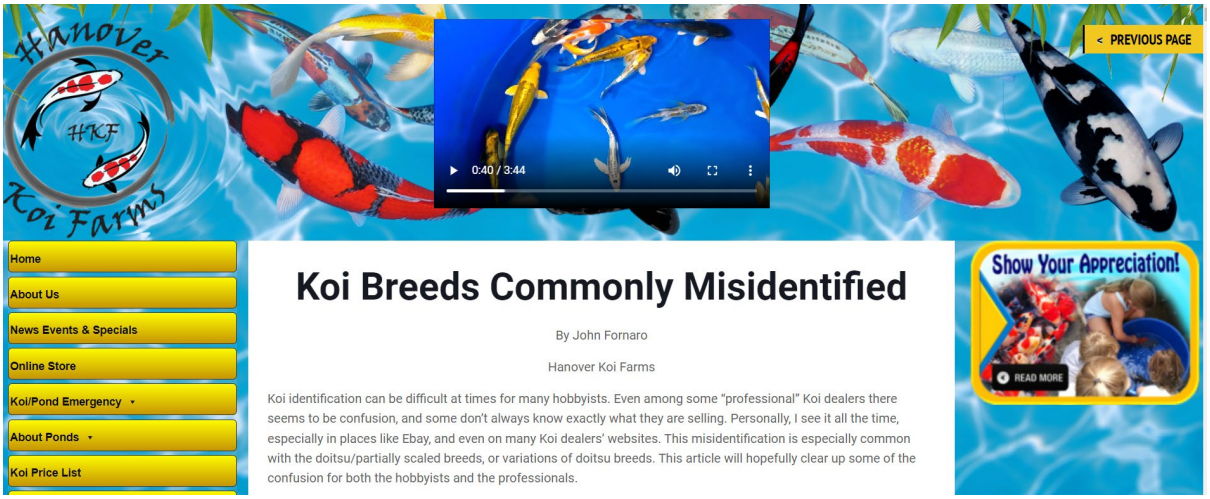


Figure 1.3 Shows an article about "Koi Breeds Commonly Misidentified"

Based on an article posted by John Fornaro for Hanover Koi Farms, it is highlighted that Koi breeding is commonly misidentified by hobbyists [3]. The so-called "professional" Koi sellers may not fully comprehend the various types of Koi, and some of them may not even know the specific types of Koi they are selling. The author notes that many dealer websites provide incorrect information about Koi, either unintentionally or intentionally. In most cases, misidentifications involve the scale variety of Koi or variations in Koi breeds. The article also serves as a guide for beginners, offering basic information about Koi, including the different colors of each part of the fish, such as the body, fins, scales, and head. However, it is emphasized that a Koi hobbyist needs to invest a significant amount of time in reading through the article to gain a better understanding of Koi. Even after reading the article, there is a possibility that hobbyists may still be confused about the types of Koi.

1.2 Problem Statement and Motivation

According to recent research, I have identified certain deficiencies in the existing products in the market, along with some motivations.

- **Technology to recognize type of koi fish is limited.**

  It's surprising that, despite significant technological advancements in recent years, there is a lack of user-friendly technology designed to distinguish between various types of koi fish. While tools for water quality testing and monitoring, as well as measuring and analyzing water parameters, are available, there is a clear gap in readily available and easy-to-use solutions for identifying diverse koi fish breeds. This means that koi fish enthusiasts and professionals often must rely on their own expertise to differentiate between different koi varieties, which can be a challenging and time-consuming task. As technology continues to evolve, it presents an intriguing opportunity for developers and innovators to explore creating accessible and accurate koi fish recognition technology, which could greatly benefit the koi-fish-keeping community.

- **Human need to spend long time in learning to recognize koi fish.**

  Recognizing koi fish can be quite a time-consuming endeavour for humans due to several factors. To begin with, the sheer diversity of koi fish species, numbering well over 100, can overwhelm beginners who may find it daunting to memorize and differentiate between them. Moreover, the challenge doesn't end there. Even those who manage to become adept at recognizing these fish can encounter difficulties as koi have the ability to change their colours and patterns over time. This transformation is influenced by a range of factors, including genetics, environmental conditions, and temperature fluctuations. Consequently, even experienced koi keepers can struggle to accurately identify these beautiful creatures, as their appearance can evolve significantly throughout their lives, further complicating the process of koi fish recognition.

- **High market value and business value of koi fish**

  Koi fish are remarkably prized in both the market and business sectors due to their captivating appearance. Their striking colors and unique patterns make them highly sought after, leading to premium prices, especially for the most visually appealing and larger specimens. Not only for that, but Chinese also hold the belief that

investing in fish will bring about financial prosperity and blessings [4]. This strong demand for koi fish has fostered a thriving industry around them. This industry encompasses various aspects, including breeding, selling, and businesses catering to koi enthusiasts, such as pond construction and the sale of koi-related products like food and health supplies. The allure of koi fish extends beyond their ornamental value; they also hold special significance in Japanese culture, symbolizing qualities like perseverance and good fortune. This cultural connection further elevates their value, making them not just cherished pets but also valuable commodities with a notable presence in both the market and the world of business.

One big issue faced by people who like koi fish is that a long time is required to figure out all the different kinds of koi. With more than 100 types, it can be especially difficult for beginners to remember and distinguish them. The process of learning this can be overwhelming, potentially discouraging newcomers. To address this, research is being planned. An easy-to-use tool is intended to be created, designed to recognize koi fish for users. This tool may employ smart technology, such as analyzing pictures and learning from them, so that anyone, whether a beginner or an expert, can quickly and accurately identify koi fish. The goal is to make the enjoyment of koi fish more accessible to everyone.

Another problem with koi fish is the lack of tools available for recognizing them. Although advanced devices for checking water quality and analyzing it exist, a gap is present in tools for distinguishing koi fish. Current tools are primarily focused on water quality, not fish recognition. To fill this gap, research and development are being planned. A specialized technology for recognizing different koi fish is expected to be created. New technologies, such as computers capable of learning and identifying fish patterns, may be utilized to develop a mobile application. Through this application, koi fish fans will be able to know exactly which type of koi they have, enhancing the overall koi fish experience for everyone.

1.3 Research Objectives

- **To reduce learning gap for new entry coming user in recognizing type of the koi fish.**

The primary objective is to bridge the learning gap for new users interested in koi fish, offering a convenient and mobile-friendly solution for identification and learning. Users can effortlessly upload images of koi fish they're uncertain about to the mobile app, which will then identify the koi fish type. This user-friendly and informative experience aims to empower koi enthusiasts to acquire knowledge and appreciation easily for these beautiful creatures. Additionally, the app will contribute to saving both time and money for users, streamlining the learning process and eliminating the need for costly external resources or experts for koi identification.

- **To promote healthy koi trading by ensuring confident environment when identifying different grade of koi.**

The goal is to empower users by developing a mobile app that to confidently identify and differentiate various grades of Japanese koi. In the world of koi trading, beginners always face the challenge of distinguishing between high-quality koi and potentially fraudulent or misrepresented specimens. This app will serve as a trusted platform for both newcomers and experienced enthusiasts, helping them make decisions by accurately recognizing and grading koi based on their characteristics.

- **To create a comprehensive and user-friendly platform for recognizing the value of Japanese Koi, leveraging the power of Chatbot integration.**

By incorporating Chatbot, a large language model, the app aims to provide a seamless and guided user experience. Users will benefit from interactive and informative guidance, enabling them to learn about Japanese Koi through natural language statements. The integration of Chatbot will facilitate dynamic learning directly within the app, offering a unique educational system tailored to the world of Koi. Furthermore, the app will utilize Gemini API for question-and-answer functionality, enhancing user engagement and promoting a deeper understanding of Japanese Koi through interactive conversations. Overall, the integration with Chatbot aims to create an intuitive and educational platform for Koi enthusiasts.

1.4 Project Scope and Direction

Based on the problem statement stated above, technology with high accuracy to identify the variety of koi fish is limited. In other words, this field of learning has high entry standards, which require beginners to invest a significant amount of time. The difficulty of learning koi-related knowledge might discourage some individuals from pursuing it. Therefore, the scope of the project is to develop a mobile cloud solution application for Japanese Koi recommendation and recognition system. Users can identify and understand the type of Koi fish in the image they provide, along with its characteristics and related information. The mobile application should be able to recognize 11 types of koi fish, such as Asagi, Bekko, Hikarimoyomono, Hikarimuji, Kohaku, Koromo, Showa, Shusui, Taisho Sanke, Tancho, and Utsurimono. Every country has different grading standards for koi fish. In this project, the Malaysia grading standards are used. The main objective of this project is to provide an educational platform for those interested in Japanese koi to learn and identify the type and grade of Japanese koi.

1.5 Contributions

Based on the social perspective, this project will contribute significantly to the koi fish enthusiast community by reducing the learning gap in identifying koi fish species and grades. Additionally, the project's contribution will help individuals avoid falling victim to fraud when purchasing a cheap koi fish at an inflated price. In this case, the project creates and ensures a confident environment for identifying different species and grades of koi by allowing users to upload a koi fish image to the koi fish identifier mobile application.

From a system perspective, a diverse collection of koi fish data sets will be utilized to train the system in identifying koi fish types and grades through cloud computing and Artificial Intelligence. The trained model will be integrated to the app as part of this project. When a user uploads a koi fish image, the trained model will analyze and predict the sent image. Subsequently, the predicted results, including koi fish information such as type, price, and background, will be sent back to the user.

1.6 Report Organization

The details of this project are shown in the following chapters. In Chapter 2, some related backgrounds are reviewed. Then, System Methodology, System Architecture Diagram, User Case Diagram and Activity Diagram of the project are presented in Chapter 3. And then, Chapter 4 show the Block Diagram and Wireframe of the project such as data collection. training model, integrating model, Amazon AWS connection and the user interface of the proposed system. Furthermore, Chapter 5 describes the system implementation including the Data Collection & Annotation, Model Training and Integration, Amazon AWS connection, Firebase Authentication, Chatbot Implementation, Overall System Operation with screenshot and Implementation Issues. Moreover, Model Analyzation and Evaluation, Project Challenges and Objective Evaluation are presented in Chapter 6. Lastly, Chapter 7 talks about Conclusion and Recommendation.

Chapter 2

Literature Review

2.1 System Review

2.1.1 Picture Fish - Fish Identifier

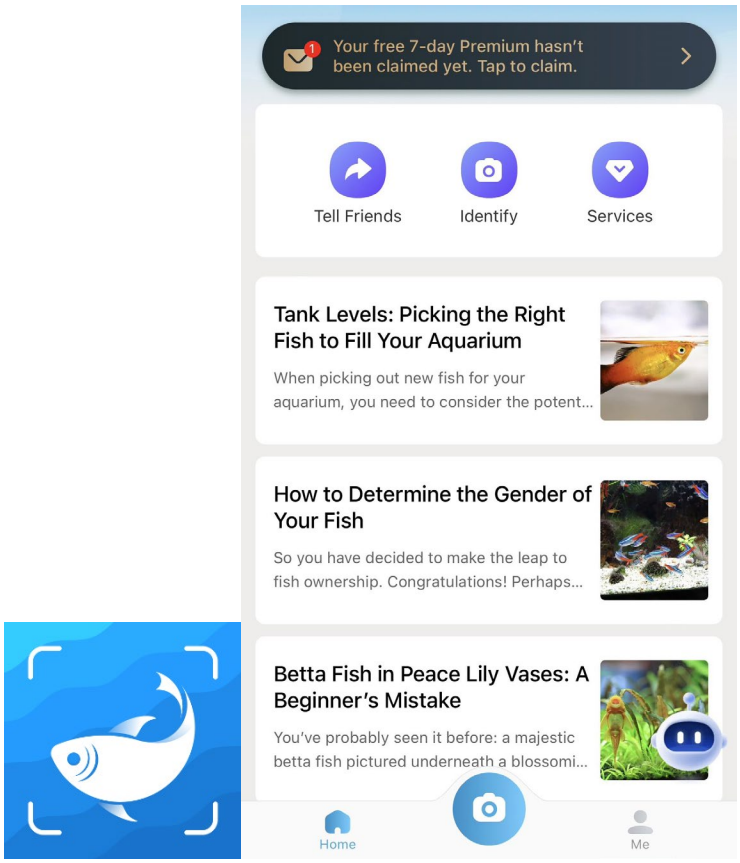


Figure 2.1 Picture Fish - Fish Identifier

Picture Fish - Fish Identifier is a mobile application designed with the primary function of identifying various fish species through uploaded images [5].In addition to its fish recognition feature, the app offers several functionalities to enhance the user experience. Users can access care tips for different fish species, allowing them to provide the best possible care for their aquatic pets. The app also facilitates the creation of a fish collection, allowing users to organize and keep track of the fish they encounter. Furthermore, users could seek advice and answers from fish experts, enhancing their knowledge about fishkeeping. The inclusion of an AI chat box adds an interactive element, allowing users to ask and receive answers to their fish-related questions.

Additionally, the app provides snap tips, offering valuable insights and recommendations related to fish care and maintenance.

Strengths

One of the strengths of Picture Fish lies in its comprehensive features that go beyond fish identification. Users not only benefit from accurate species recognition but also gain access to a wealth of information and support for their fishkeeping endeavours. The app stands out by offering care tips, a fish collection feature, expert advice, an AI chat box, and snap tips, providing a holistic platform for fish enthusiasts.

Weakness

Despite its strengths, Picture Fish - Fish Identifier does have some weaknesses that users should consider. To unlock premium features, users are required to make a purchase, limiting access to certain aspects of the app. While the app offers articles, the availability may be limited, potentially restricting users from accessing a diverse range of content. Another limitation is the fixed picture size requirement, mandating images to be in a square format, which may pose inconvenience for users who prefer or have images in different dimensions. These weaknesses, though present, may be outweighed by the app's strengths for users who prioritize comprehensive fish-related information and support.

2.1.2 Fish Identification – Fish Brain

Figure 2.2 Fish Identification – Fish Brain

Fish Identification – Fish Brain is equipped with powerful AI that facilitates the identification of various fish species, ranging from sharks and whales to tunas, devil fish, bass, butterfly fish, and dolphins [5]. Users can utilize the Fish Identification feature by simply taking or uploading a photo of a fish, allowing the AI to provide detailed information about the species within seconds. Additionally, users have the capability to build a personal fish collection, with scanned fish being stored for future reference.

Strength:

One notable strength of the Fish Identifier lies in its adjustable image recognition, enhancing the accuracy of fish identification based on the images provided. The user interface is designed to be user-friendly, ensuring ease of navigation and a seamless experience for individuals of varying technological expertise. Another advantageous feature is the ability to perform offline scanning, allowing users to identify fish even in environments with limited or no internet connectivity.

Weakness:

Despite its effectiveness in fish identification, the Fish Identifier has some limitations. It exclusively focuses on identifying fish and lacks additional features such as care tips

or comprehensive fish information. This singular functionality may be a drawback for users seeking a more comprehensive tool for managing their aquatic companions.

2.1.3 Fish Identification Fish Snap

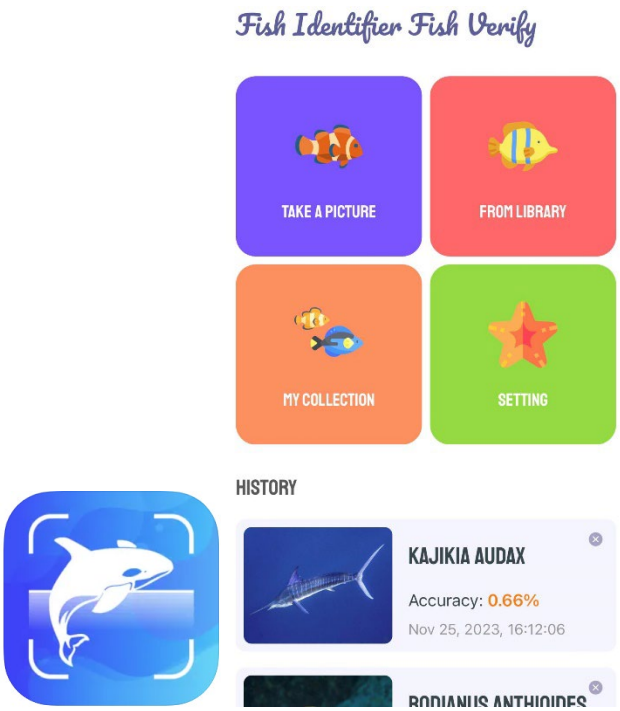


Figure 2.3 Fish Identification Fish Snap

Fish Identification Fish Snap app boasts cutting-edge AI technology that enables users to instantly identify fishes with remarkably accurate metadata [6]. Users can effortlessly capture or upload a snapshot of a fish, and the Fish Identifier app will swiftly provide detailed information about the identified species within seconds. This application is designed to satiate the user's curiosity about the vast array of fish species in our surroundings.

Strengths

One of the notable strengths of the Fish Identifier app lies in its simplicity and efficiency. Users can conveniently take or upload a photo of a fish, and the app's fast and accurate identification system, covering over 3000 species of fishes, promptly delivers comprehensive information. The app features an extensive database that encompasses various details about different types of fishes, and its finely designed, user-friendly interface enhances the overall user experience. Additionally, the app facilitates a quick and accurate identification process, catering to users' interests in exploring the diverse world of fishes.

Weakness

However, the Fish Identifier app does have a notable weakness. Its identification process may lack specificity, as it provides accuracy ratings for different fish images related to a single uploaded fish image. This could potentially lead to confusion or ambiguity, especially when users seek precise identification of a particular fish species. Despite this drawback, the app remains a valuable tool for enthusiasts looking to learn more about the myriad fishes inhabiting our waters.

2.1.4 FishScan – Identify Fish

Figure 2.4 FishScan – Identify Fish

FishScan is an innovative tool designed to turn anyone into a fish expert [7]. This app utilizes powerful AI technology to identify various fish species, ranging from sharks and whales to bass and dolphins. Users can simply take or upload a photo of the fish, and FishScan rapidly provides detailed information about the identified species. The AI's fuzzy search function further assists users in finding similar fish based on the image captured or uploaded. In addition to identification, FishScan offers a comprehensive Fish Encyclopedia featuring information on over 2000 fish species, including scientific and common names, scientific classification, habitat, length, and more. Users can save their identified fish and create a collection, making it convenient for future reference. Moreover, FishScan caters to fish enthusiasts by providing a selection of captivating fish wallpapers, with weekly updates to keep the options fresh. For those seeking knowledge,

the app features a Fish Blog that regularly publishes popular science articles about fish, covering intriguing topics such as the most dangerous shark or the real-life inspiration behind the fish in the cartoon "Finding Nemo."

Strength

FishScan's strengths lie in its user-friendly interface, quick and accurate fish identification, and the wealth of information available in its Fish Encyclopedia. The app caters to both casual fish observers and enthusiasts, offering a versatile platform for learning and exploration. Additionally, the inclusion of interesting fish wallpapers and informative blog articles enhances the overall user experience, making FishScan a comprehensive tool for fish enthusiasts.

Weakness

However, FishScan has a notable weakness—it lacks a fish grade identifier. While it excels in providing information and identification, a feature to assess the quality or grade of the identified fish could be beneficial for users engaged in activities such as fishing or aquarium keeping. Implementing a grading system would enhance the app's utility and appeal to a broader audience with specific needs related to fish quality assessment.

2.1.5 Fish Identifier by Picture

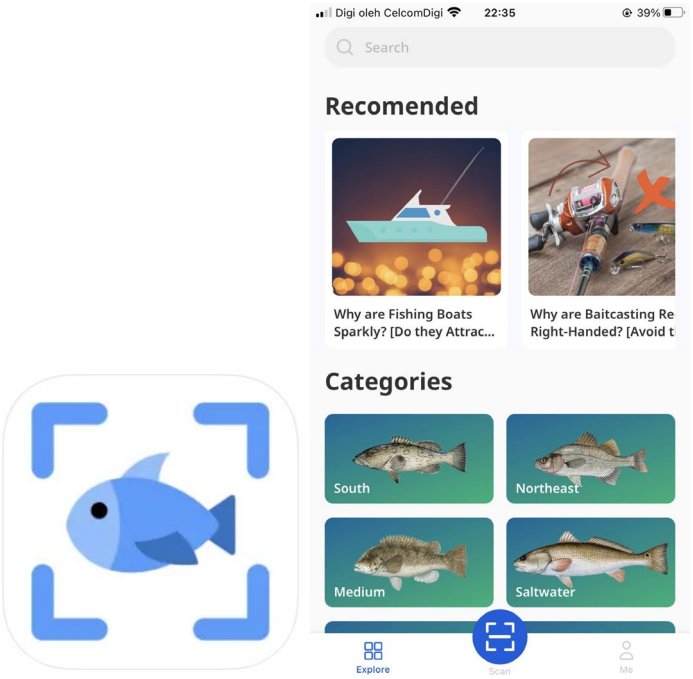


Figure 2.5 Fish Identifier by Picture

Fish Identifier by Picture, a mobile application, serves as a valuable tool for marine life enthusiasts by allowing users to identify fish through uploaded images [8]. With worldwide coverage divided into 31 regions, this app provides a comprehensive exploration of marine life across the globe. This device goes beyond just observing fish, offering a comprehensive collection of all marine species. The Education section is a standout feature, providing valuable bonus material on anatomy and families, enriching the user's knowledge. Boasting an impressive species total of approximately 10,000, it stands out as one of the most comprehensive apps in its category. Additionally, users can curate their personal fish collection, adding a personalized touch to the experience.

Strengths

One of the notable strengths of this marine life exploration device is its ability to provide clear and detailed descriptions of the scanned fish. Users can benefit from additional related articles covering topics such as fishing boats, licenses, and spearfishing, contributing to a well-rounded understanding of the marine environment. The inclusion of such supplementary information is particularly advantageous for beginners, allowing them to quickly expand their knowledge beyond mere observation.

Weakness

However, the application is not without its weaknesses. One notable drawback is the absence of a grading system, limiting the ability to assess or categorize the scanned

marine life. Additionally, the lack of an AI chatbox could be seen as a missed opportunity for user engagement and assistance. The device's interface may also pose challenges, as it is not as user-friendly due to the inclusion of numerous features on a single page. The overwhelming number of features in a consolidated interface may make navigation more complex for users, potentially hindering the overall user experience.

2.2 Summary Table

	Picture Fish - Fish Identifier	Fish Identification – Fish Brain	Fish Identification Fish Snap	FishScan – Identify Fish	Fish Identifier by Picture	Proposed system
Image upload and Photo snapping	Yes	Yes	Yes	Yes	Yes	Yes
AI chat box	Yes	No	No	No	No	Yes
Personal fish collection	Yes	Yes	Yes	Yes	No	Yes
Real-Time Identification	Yes	Yes	Yes	Yes	Yes	Yes
Comprehensive Fish Information	Yes	No	Yes	Yes	Yes	Yes
User-Friendly Interface	Yes	Yes	Yes	Yes	No	Yes
Fish Grade Identifier	No	No	No	No	No	Yes

Table 2.1 Comparison between existing system and proposed system

Table 2.1 shows the comparison between existing systems with the proposed system. The existing system used for comparison Picture Fish - Fish Identifier, Fish Identification – Fish Brain, Fish Identification Fish Snap, FishScan – Identify Fish and Fish Identifier by Picture

2.3 Limitation of Previous Studies

After reviewing several existing products, I have found that there is no system can detect the type of the Japanese Koi in the market. The systems I have reviewed is just only can detect the type of the general fish such as goldfish, whale and shark. Each of them has limitations in terms of identifying the grade of the fish. While these products allow users to scan the fish or upload fish images to identify the fish's identity, they do not facilitate the identification of the fish's grade. Another problem is that both Picture Fish - Fish Identifier and Fish Identifier by Picture do not allow users to adjust the shape of the image; the image is automatically cropped into a square shape, leading to incorrect object detection when parts of the object are not detected in the image.

Furthermore, one of the issues with Fish Identification – Fish Brain, Fish Identification Fish Snap, FishScan – Identify Fish, and Fish Identifier by Picture systems is the absence of an AI chatbot. Users must spend extra time searching for information if they encounter obstacles while using the system to identify the fish. With the integration of an AI chatbot, the app can provide professional answers related to koi fish-related inquiries.

Another problem with the reviewed system Fish Identifier by Picture is the absence of a personal fish collection. Users need to re-upload the image for identification if they accidentally close the system or encounter a human error. Without the collection feature, they cannot look back at the koi fish they identified before. Additionally, Fish Identification – Fish Brain has the issue that the fish information is not comprehensive enough, making it difficult for users to clearly understand the details of the koi fish.

The weakness of the Fish Identifier by Picture system lies in its unfriendly user interface, causing users to struggle to locate or potentially miss important features of the app, leading to uninstallation. The last issue with all the reviewed systems is the absence of grading features. Users are unable to understand the value of the fish without grading features, which may cause them to unintentionally overlook valuable fish.

2.4 Proposed Solutions

In this case, the proposed system aims to develop a mobile solution that allows users to recognize and identify the type and grading of koi fish and receive recommendations. In the proposed system, users only need to register an account to access every feature. This system will enable users to upload a koi fish image or capture a new one, adjust the image size, and, finally, identify and return the koi fish type and grade. The proposed system will also integrate with an AI chatbot that facilitates dynamic learning directly within the app, providing a unique educational system tailored to the world of Koi. Additionally, the proposed system will ensure the development of a user-friendly interface to make it easy for users to navigate and utilize the system.

Chapter 3

System Methodology/Approach OR System Model

3.1 System Requirement

3.1.1 Hardware

The hardware involved in this project includes computer. Specifically, the computer is designated for executing both the front-end and back-end code within an Integrated Development Environment (IDE). Additionally, the computer will be utilized to run emulators for testing the mobile application. This setup ensures efficient development and testing processes for the project.

Table 3.1 Specifications of laptop

Description	Specifications
Model	MSI Bravo 15 B5DD
Processor	AMD Ryzen™ 7 5000 H
Operating System	Windows 11
Graphic	AMD Radeon™ RX 5500M with 4GB GDDR6
Memory	Max 64GB DDR4-3200 2 Slots
Storage	1x M.2 SSD

3.2 Design Specifications

3.2.1 Methodology

The methodology chosen for the development of the mobile cloud solution for Japanese Koi recognition and recommendation system project is the RAD (Rapid Application Development) phased development. This approach comprises four phases: planning, analysis, design, and implementation. Each iteration, starting from the analysis phase and progressing through the implementation phase, will result in a version of the system. Subsequently, the system's version will advance to the next iteration until the final version of the system is achieved. Ultimately, the system will be ready to be used by the end-users.

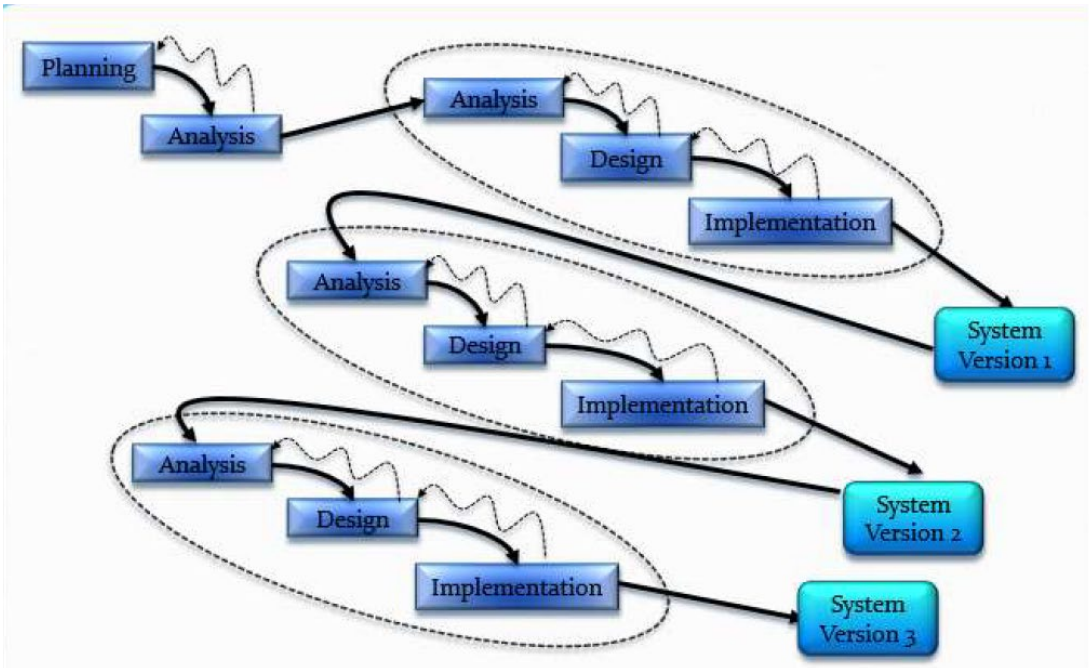
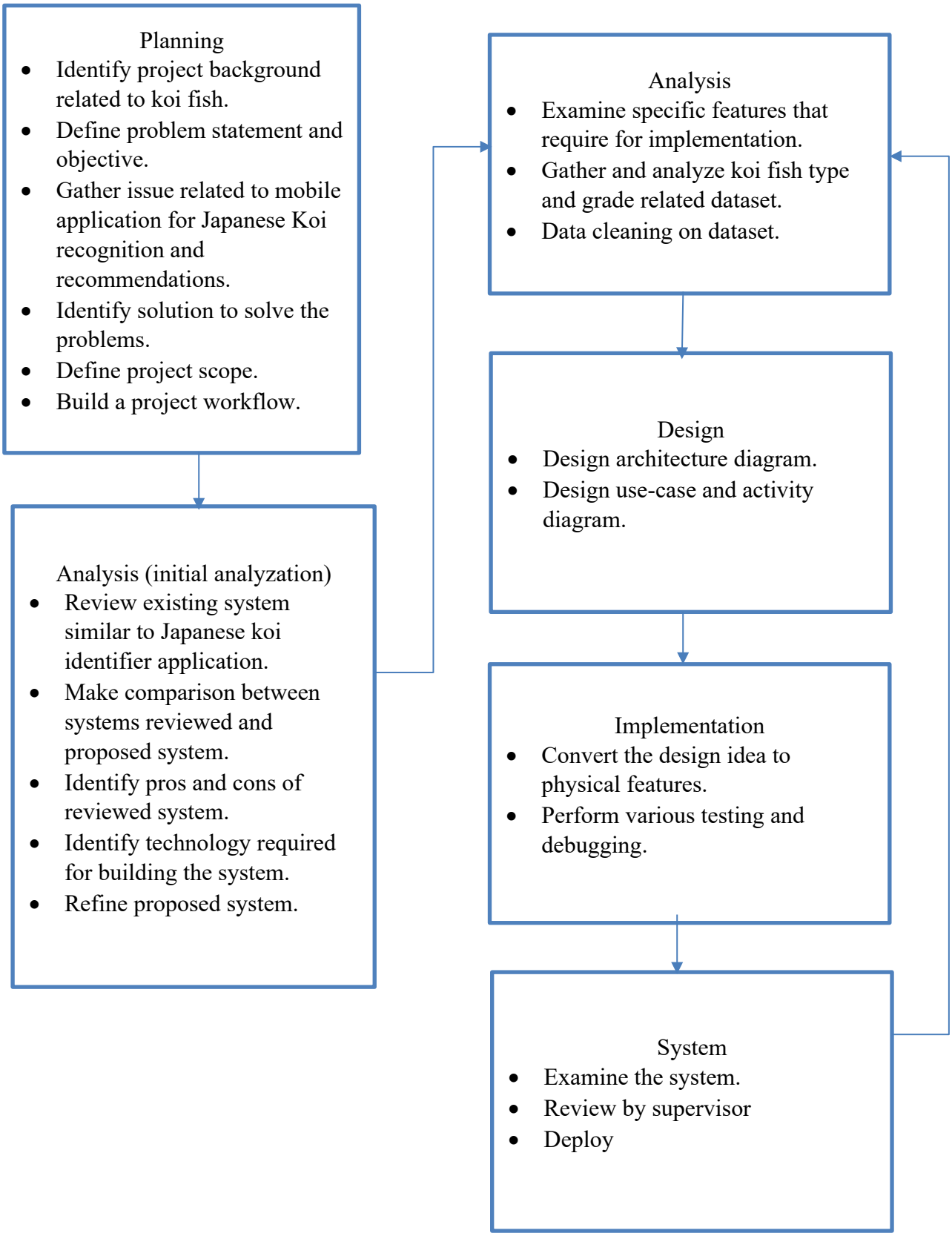


Figure 3.1 Phased Development Methodology

3.2.2 Project Workflow in Phased Development



3.2.2.1 Planning phase

In this first phase of the project, we're laying the groundwork for a mobile app that recognizes and recommends Japanese Koi. We start by looking at the background of the project, understanding the world of koi fish. Then, we clearly state the problem we're trying to solve and the goals we want to achieve. We gather information about the issues users face with a mobile app for recognizing and recommending Japanese Koi. Based on this info, we figure out how to solve these problems. After that, we set the boundaries and goals of the project (project scope). To get a visual idea of how everything will work, we create a project workflow, showing the different steps of the project. This phase is all about getting a solid foundation for building the Japanese Koi recognition and recommendation app.

3.2.2.2 Analysis (initial analyzation) phase

In this phase, our primary task is to investigate existing systems similar to the Japanese koi identifier application. We conduct a detailed comparison between these systems and our proposed one, carefully noting the strengths and weaknesses of each. To enhance our own system, we address any issues identified in the existing systems. Additionally, we gather and analyze a dataset containing information about different types and grades of koi fish, a crucial step in refining our system. Concurrently, we identify the necessary technologies and tools required for building our system. This comprehensive analysis informs the refinement of our proposed system, ensuring that it addresses any shortcomings observed in the existing systems and is well-equipped for effective performance.

3.2.2.3 Analysis phase

In the analysis phase of our development, we carefully inspect and identify the specific features that need to be implemented in the subsequent stages. This ensures a systematic approach, guiding the step-by-step creation of our project with a clear understanding of the essential functionalities.

3.2.2.4 Design phase

In the design phase, we create an architecture diagram outlining the structural components and their interactions within the system. Additionally, we develop use-case and activity diagrams to illustrate the system's functionalities and user interactions, providing a visual guide for the subsequent implementation stage.

3.2.2.5 Implementation phase

In the implementation phase, translate the design concept into tangible physical features, bringing the envisioned ideas into reality. Subsequently, conduct thorough testing and debugging processes to ensure the functionality and reliability of the developed.

3.2.3 Technologies Involved

1. **Dart programming**

Dart are used to build a robust and scalable backend for your application. It facilitate communication between the frontend and backend, enabling connection to AWS services for necessary operations such as storing and retrieving images, managing user data, and invoking the Koi recognition model.

2. **Flutter**

Flutter is a popular cross-platform framework used to develop the frontend of this mobile application.

3. **Gemini**

Gemini is integrated into the project to build a chatbot feature. The users can ask anything that related to koi.

4. **Amazon AWS**

There are 3 services from Amazon AWS that will be implemented in this project. Firstly, Amazon S3 will be used to store images of Japanese Koi, providing a reliable and scalable storage solution for the application. Secondly, Amazon Lambda will serve as a serverless compute service to run the code that response to the events. It would manage the underlying resources automatically. Thirdly, AWS API Gateway will act as a front door to the backend services hosted on AWS, providing an HTTPS endpoint for the frontend to communicate with the cloud. It will route incoming requests to the appropriate Lambda functions and handle the responses.

5. Roboflow

Roboflow is a platform that simplifies building, training, and deploying computer vision models. In this project, Roboflow is used for data collection by uploading and annotating images of Japanese Koi, enhancing the dataset with its augmentation tools.

6. Google Colab Notebook

Google Colab Notebook will be used to train the real-time Japanese Koi detection model and to convert the model file into a format compatible with the mobile app.

7. Firebase

Firebase is a platform developed by Google that provides a variety of tools and services for building and managing mobile and web applications. One of its core features is authentication, which allows you to easily implement secure user sign-up and login processes using email, social media accounts, or other methods. In your project, you used Firebase for authentication to manage user accounts, ensuring that only authorized users can access the app, thereby providing a secure and seamless login experience.

3.3 System Design Diagram

3.3.1 System Architecture Diagram

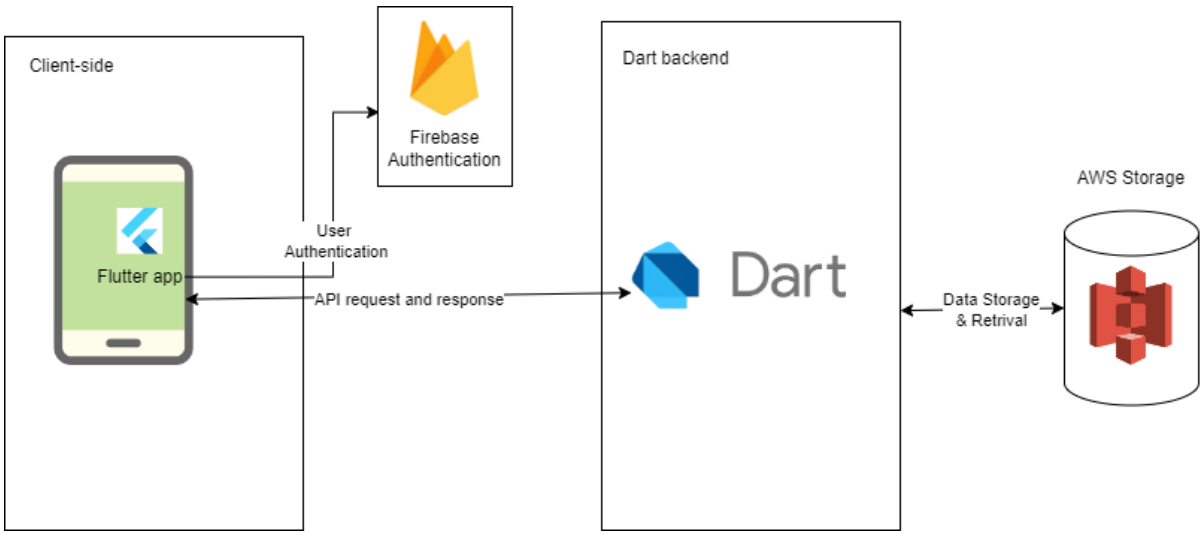


Figure 3.2 System Architecture Diagram

Based on the system architecture above, it includes a Flutter-based frontend running on the user's device, which communicates with several external components. Firebase Authentication, an external service, manages user authentication by verifying login credentials when users access the app. The app interacts with a Dart backend that handles business logic and processes API requests. This backend is responsible for interfacing with AWS Storage, where data such as user files and app-specific information is stored and retrieved. The diagram reflects the flow of data and interactions between these components, ensuring secure authentication, efficient data management, and seamless app functionality.

3.3.2 Use Case Diagram and Description

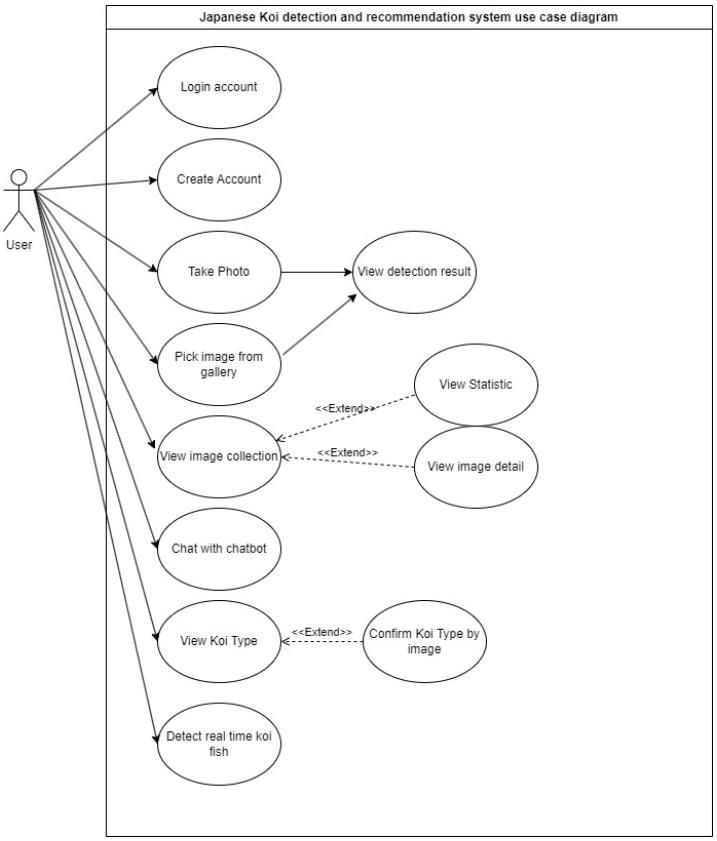


Figure 3.3 Use Case Diagram

Based on the use case diagram for the "Japanese Koi Detection and Recommendation System," here's a description of each use case:

1. **Login Account**: The user logs into the system using their credentials. This step is essential to access personalized features like viewing image collections or detection results.
2. **Create Account**: For new users, this use case allows them to create an account by providing necessary information such as username, email, and password.
3. **Take Photo**: This use case allows the user to take a photo of a koi fish using the application's camera functionality. The captured image is then used for further analysis or detection.
4. **Pick Image from Gallery**: Instead of taking a new photo, the user can select an existing image of a koi fish from their device's gallery. This image can also be analyzed for koi detection.
5. **View Detection Result**: After taking a photo or selecting one from the gallery, the user can view the results of the koi fish detection. The system will display the detected koi type, price range and possibly other details about koi.

6. **View Image Collection**: This use case allows the user to view a collection of previously detected koi fish images. This feature includes sorting options based on detection results.
7. **View Statistic (<<extend>>):** As an extension to view the image collection, the user can view statistics related to the images, such as the number of detections, types of koi detected, and other aggregated data.
8. **View Image Detail (<<extend>>):** Another extension to viewing the image collection is the ability to view detailed information about a specific image. This includes the date of detection, koi type, confidence level of the detection, and other metadata.
9. **Chat with Chatbot:** The user can interact with a chatbot to ask questions or get recommendations related to koi fish. The chatbot provide information on koi types, care tips, or usage instructions for the app.
10. **View List of Koi Type**: This use case provides the user with a list or gallery of different koi types that the system can detect. It includes information such as images, names, and descriptions of each koi type.
11. **Confirm Koi Type by Image (<<extend>>):** This extended use case allows the user to upload an image or take a photo of a Japanese Koi to verify if the koi type in the image matches the selected koi type from the list.
12. **Detect Real-Time Koi Fish**: This advanced use case allows the user to detect koi fish in real time using the device's camera. The system analyses the live feed to identify koi fish as they appear in the frame.

Actor:

1. User: The primary actor interacting with the system. The user can be anyone interested in koi fish detection, from hobbyists to professionals.

3.3.3 Activity Diagram

3.3.3.1 Create Account

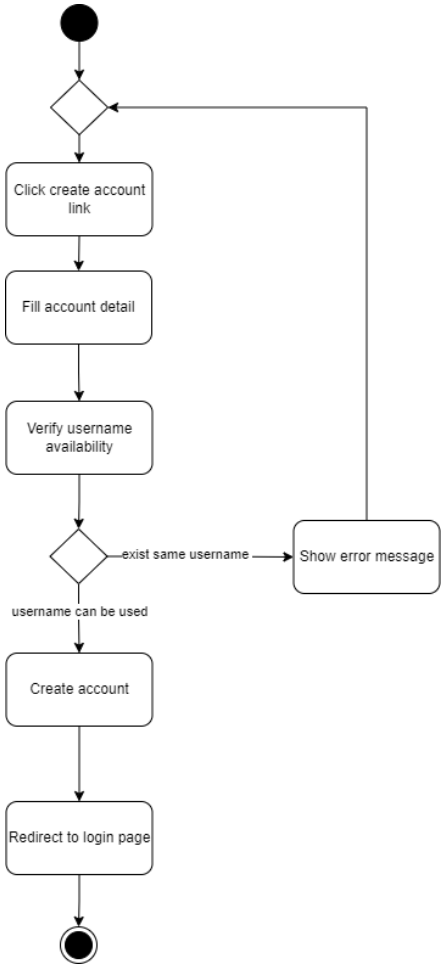


Figure 3.4 Create Account Activity Diagram

When unregistered users wish to access the full features of the application, they must first create a new account. To do this, they click on the "Create Account" link, which navigates them to a registration form. Here, they are prompted to fill in essential information such as their desired username, email address, and password. Once the user completes the form and submits it, the system begins the process of verifying the availability of the chosen username. If the username is already in use by another user, an error message is displayed, prompting the user to choose a different username. However, if the chosen username is available, the account creation process proceeds successfully. The system stores the user's information securely and redirects them to the login page. At this point, the user can log in using their newly created credentials, gaining access to the application's features and content.

3.3.3.2 Log in

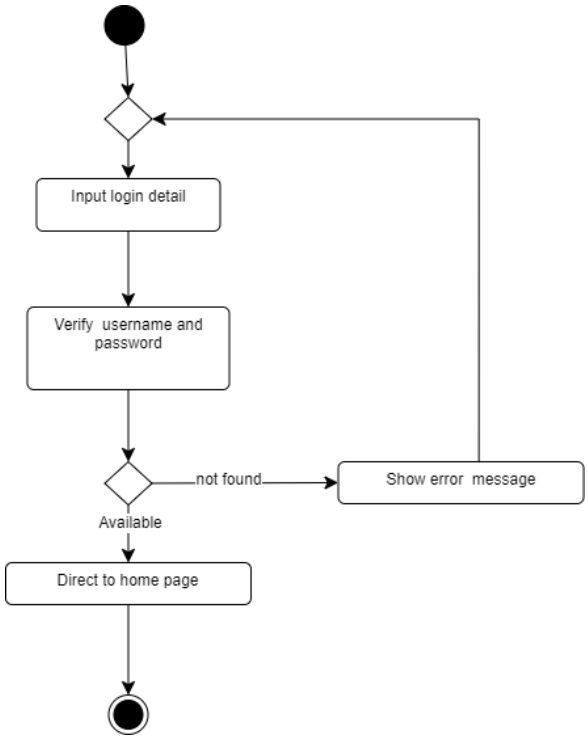


Figure 3.5 Log In Activity Diagram

In the login process, users are required to input their username and password into the designated fields. Once they have entered this information, they can submit the form. At this point, the system begins the validation process, checking the entered username and password against the stored credentials in the database. If the entered username and password are correct, the system grants the user access to the home page, where they can proceed to use the application's features and functionalities. However, if there are any errors detected during the validation process, such as an incorrect username or password, the system displays an error message. This message informs the user of the issue and prompts them to re-enter their credentials.

3.3.3.3 Koi Detection by Taking Photo

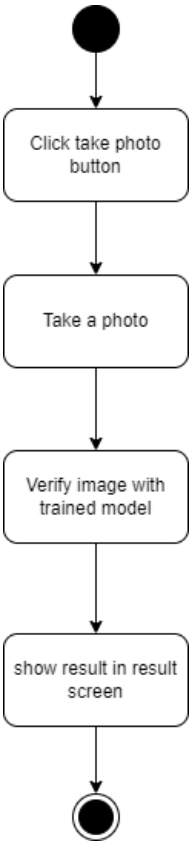


Figure 3.6 Koi Detection by Taking Photo Activity Diagram

When a user wants to identify a koi fish, they click the "Take Photo" button within the app. This action activates the device's camera, allowing the user to capture an image of the koi fish they wish to identify. Once the photo is taken, the app processes the image using its koi fish identification algorithm. This algorithm analyzes the distinct features of the koi fish to determine its type and other relevant information. After the identification process is completed, the app directs the user to a new screen displaying the detection result. Here, the user can view the identified koi fish type, as well as additional details such as the estimated price range and any relevant information about the particular type of koi fish. This feature enables users to quickly and accurately identify koi fish using their mobile device, enhancing their overall experience and knowledge of koi fish varieties.

3.3.3.4 Koi Detection by Picking Image from Gallery

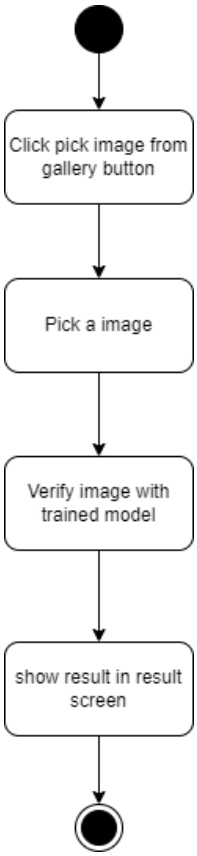


Figure 3.7 Koi Detection by Picking Image from Gallery Activity Diagram

When a user wants to identify a koi image, they have the option to select an image from their device's gallery. To do this, they click on the "Pick Image" button within the app, which opens up the device's gallery interface. The user can then browse through their gallery and select the koi image they wish to identify. Once the user has selected the image, the system verifies the image. After the verification process is complete, the system processes the selected koi image using its identification algorithm. This algorithm analyzes the unique features of the koi fish in the image to determine its type and other relevant details. Once the identification process is finished, the system displays the results on a new screen. Here, the user can view the identified koi fish type, along with any additional information such as the estimated price range and other relevant details. This feature allows users to identify koi images easily and accurately from their gallery, enhancing their overall experience with the app.

3.3.3.5 View image detail and statistic

Figure 3.8 View image detail Activity Diagram

When users wish to view the koi images they have scanned, they must navigate to the screen displaying the collection of koi images within the app. Upon reaching this screen, the system initiates a function to retrieve the images and their corresponding details from the storage. Once the images and details are retrieved, they are displayed on the screen for the user to browse through. Each image is presented in a thumbnail format, allowing users to easily identify them. When a user clicks the statistic button, it directs user to a page that show the statistic of number of koi type collected, the difference categories of price range that user collected or other related statistic. When a user clicks on a specific image, the system directs them to a new screen showing the previous detection result of that particular koi image. This screen provides users with detailed information about the koi fish in the image, such as its type, price range, and any other relevant details. After the user has finished viewing the detection result and closes the screen, the system automatically redirects them back to the screen displaying the collection of koi images.

3.3.3.6 Delete Image

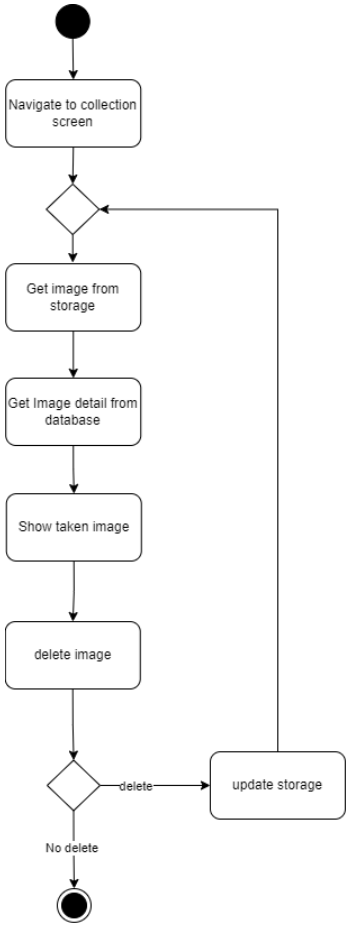


Figure 3.9 Delete Image Activity Diagram

When a user decides to delete a koi image and its detection result, they first need to navigate to the screen displaying the collection of koi fish images within the app. Upon reaching this screen, the system initiates a function to retrieve all the images and their corresponding details from the storage. Once the images and details are successfully retrieved, they are displayed on the screen for the user to browse through. Each image is presented with an option to delete it, typically through a delete button associated with the image. When a user clicks on the delete button for a specific image, a dialog box pops up, asking the user to confirm whether they want to delete the image. If the user selects "Yes," the system updates the storage by removing the image and its corresponding details. The screen is then refreshed to reflect the changes, and the deleted image is no longer displayed in the collection. If the user selects "No" in the dialog box, indicating that they do not want to delete the image, the dialog box simply closes, and no changes are made to the storage or the screen.

3.3.3.7 Chatbot

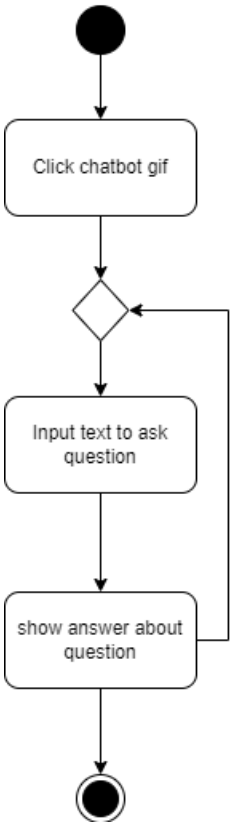


Figure 3.10 Chatbot Activity Diagram

When users have questions related to koi fish, they can easily access support by clicking on the chatbot GIF within the app. This action opens a chat interface where users can input text to ask their questions. The chatbot, equipped with a database of information about koi fish, promptly responds to the user's queries. Users can ask a wide range of questions, such as inquiries about koi fish care, feeding, breeding, and health. The chatbot uses natural language processing to understand the user's questions and provides accurate and relevant answers. This feature enhances the user experience by providing instant access to valuable information about koi fish.

3.3.3.8 Real time koi detection

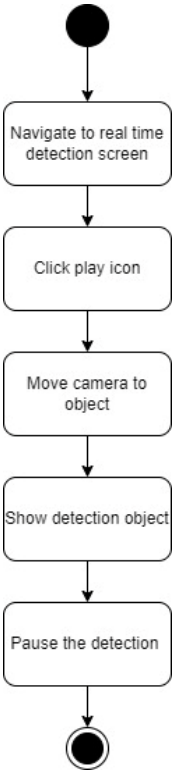


Figure 3.11 Real time koi detection Activity Diagram

The activity diagram outlines the sequence of actions for real-time koi detection. It provides an overview of the user's interaction with the app, from entering the koi detection screen to processing video frames for koi fish detection.

Upon navigating to the koi detection screen, the user sees a play icon to start the detection process. Clicking the play icon activates the device's camera, capturing a live video feed. The app then uses a machine learning model to analyze each frame in real time, searching for features that match a koi fish. When a koi is detected, a bounding box appears around it, showing the detection on the screen and updating with any movement.

The user can control the process by clicking the pause icon to stop the detection. This action halts the camera feed and detection algorithm, allowing the user to review the last detected koi or choose to restart the detection or exit the screen.

3.3.3.9 Koi Learning and Identification

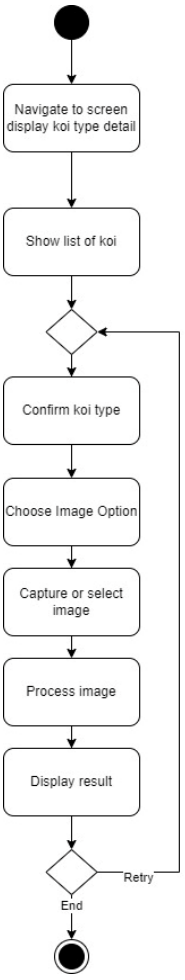


Figure 3.12 Koi Learning and Identification Activity Diagram

The activity diagram illustrates the process of identifying koi fish types in a Flutter app. Users can explore different koi types, each represented as a card that displays an image, name, and brief description. Each card features two icons: a camera icon to take a new photo and an image icon to upload an existing one from the gallery. After selecting an image option, the user provides a photo that the app processes using a pre-trained machine learning model to recognize koi fish types. Based on the analysis, the app offers feedback:

- **Match Found**: A success message confirms that the koi in the image matches the selected type.
- **No Match**: The app informs the user of the mismatch and suggests trying again with a different koi type or image.

3.4 Project Timeline

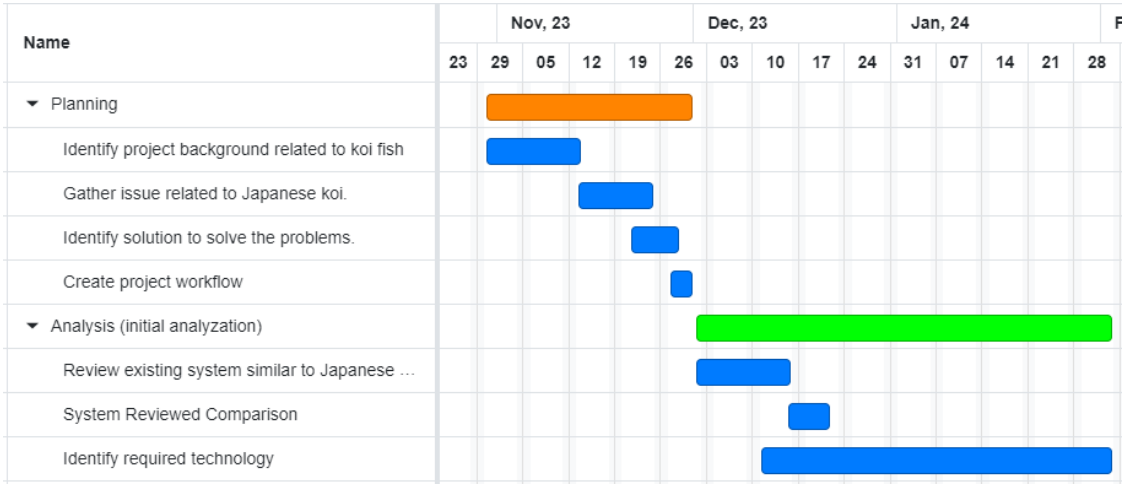


Figure 3.13 Gantt Chart 1

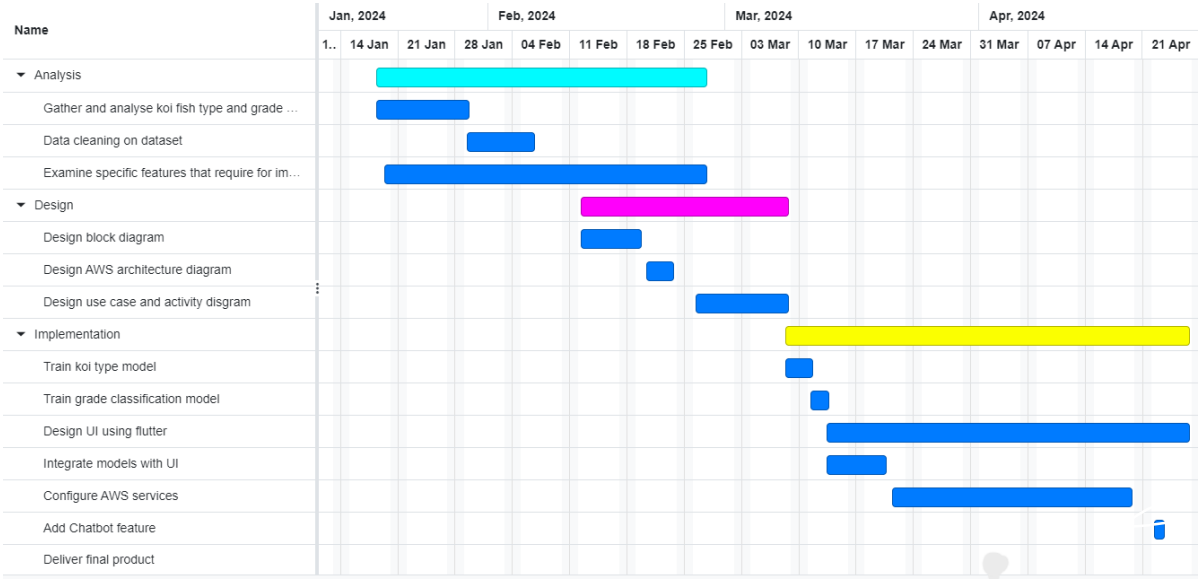


Figure 3.14 Gantt Chart 2

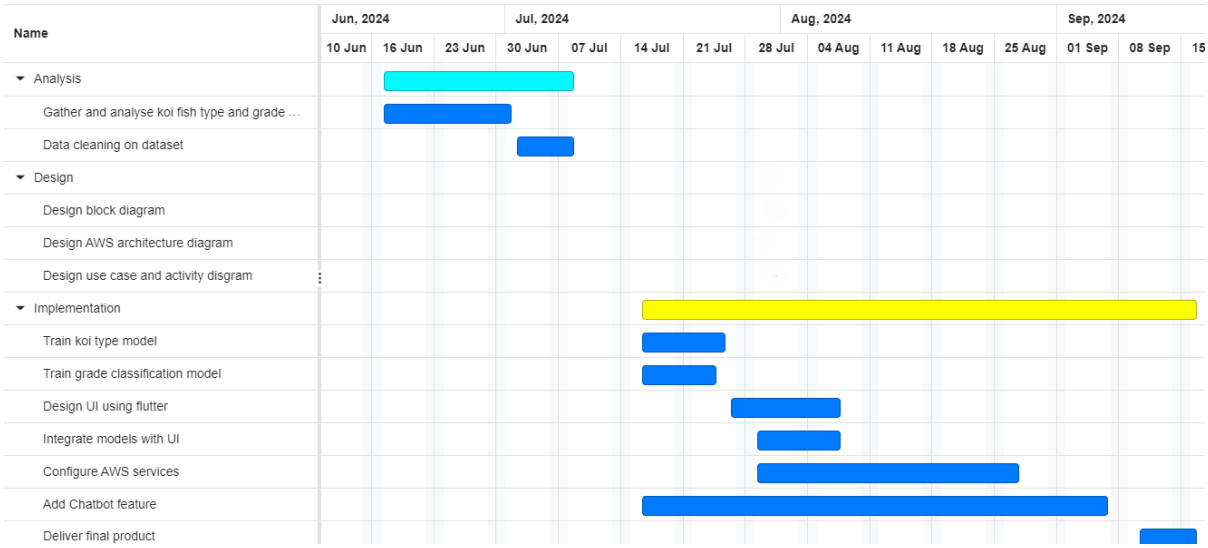


Figure 3.15 Gantt Chart 3

Chapter 4

System Design

4.1 Block Diagram

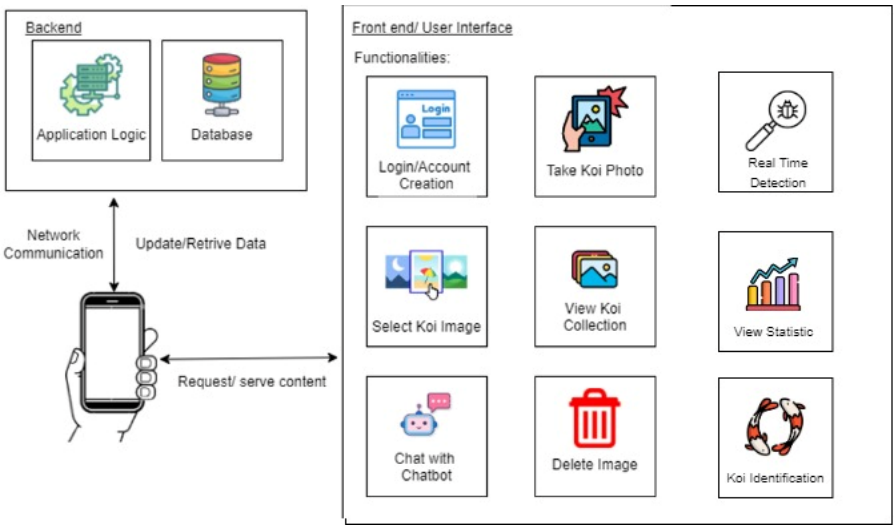


Figure 4.1 Block Diagram

The figure above shows the block diagram of the proposed project. When users access the mobile application, the frontend and backend processes collaborate seamlessly to provide a smooth user experience. On the frontend, users interact with the app's interface, and client-side code processes these inputs locally, managing UI updates and initiating network requests. There are nine functionalities' users could be able to perform, such as logging in, creating an account, detecting a koi image by taking a photo or picking an image from the gallery, viewing a collection of koi images, chatting with a chatbot related to koi, deleting the koi image, detect real time koi, view statistic and identify and learn the koi. For the backend side, it will receive the requests sent by those triggered functionalities by users and then execute application logic, interact with databases, and generate a response. The response is then sent back to the frontend, where client-side code processes the data, updates the UI, and provides immediate feedback to the user. Throughout this process, users can interact with the application seamlessly and utilize all the functionalities within the application.

3.2 Wireframe

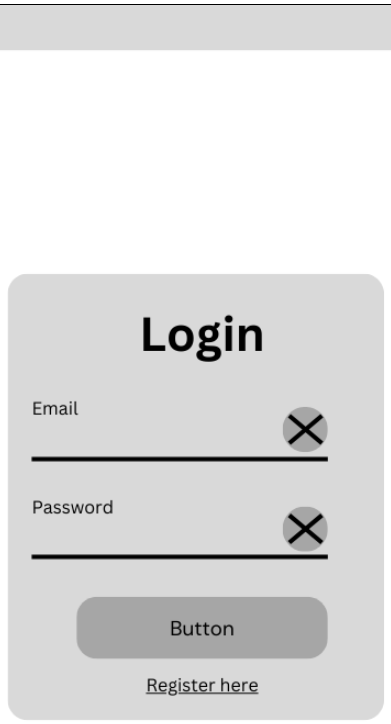


Figure 4.2 Login page

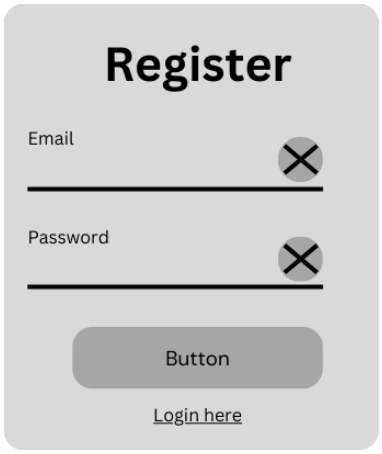


Figure 4.3 Register page

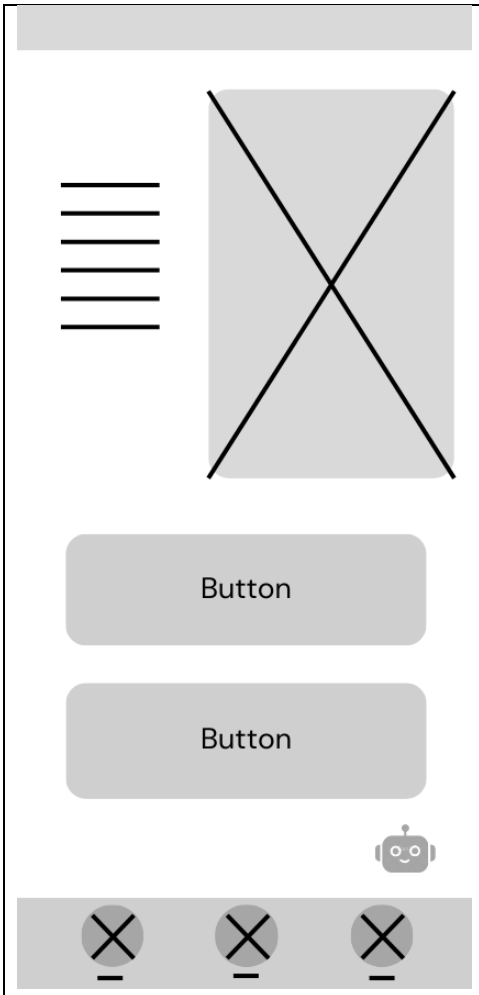


Figure 4.4 Home page

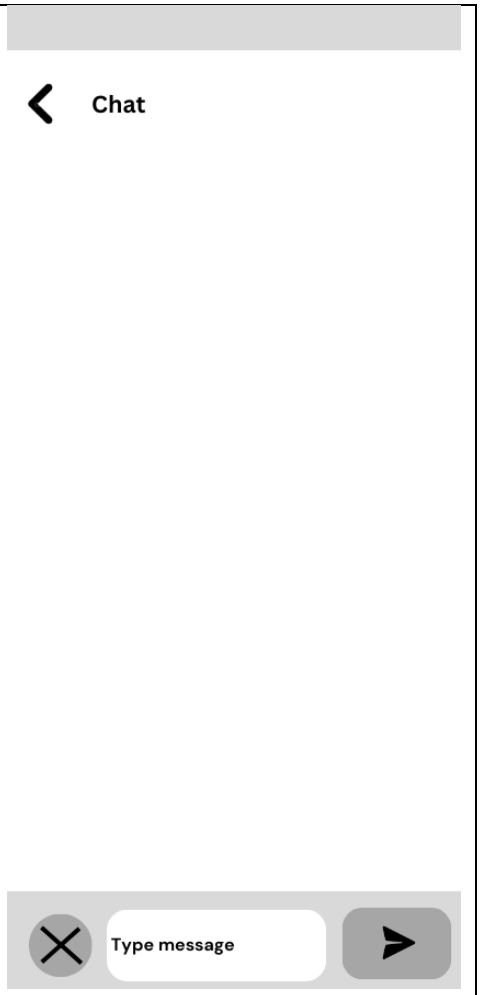


Figure 4.5 Chatbot page

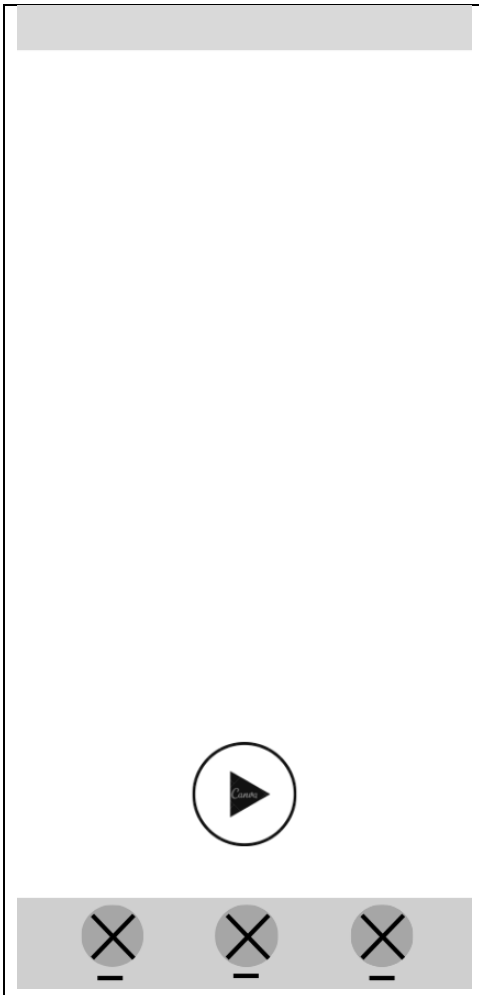


Figure 4.6 Real time detection page

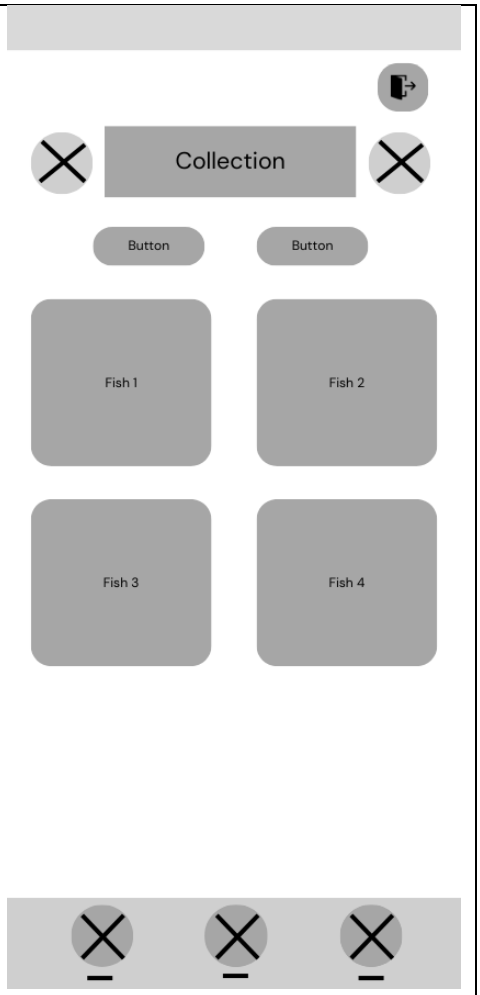


Figure 4.7 Collection page

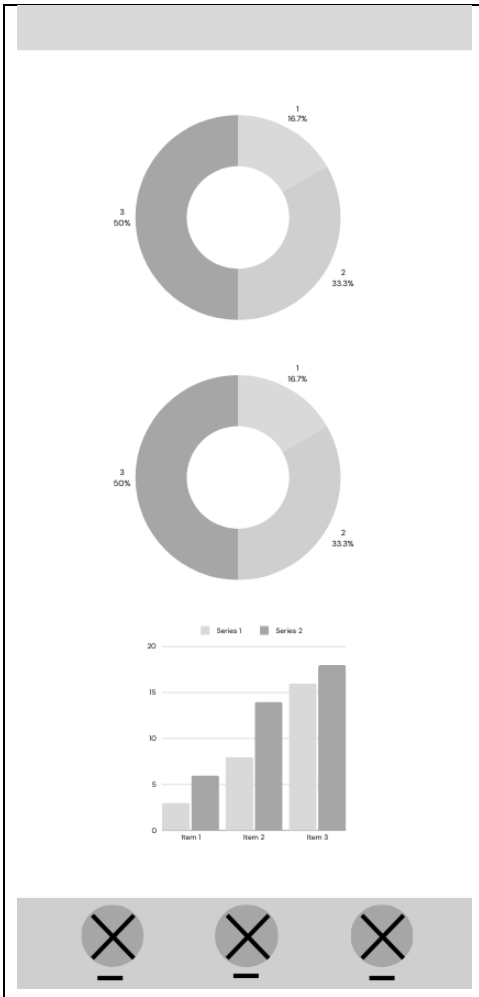


Figure 4.8 Dashboard page

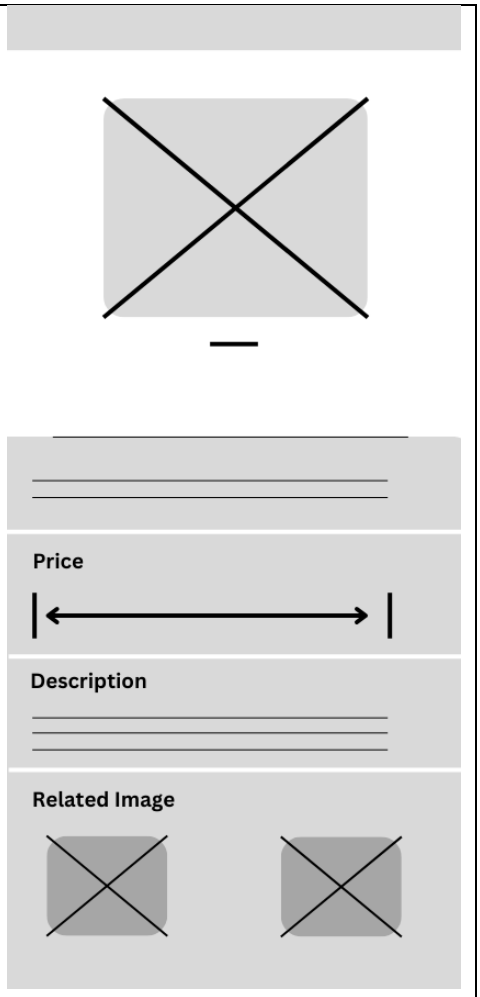


Figure 4.9 Detection result page

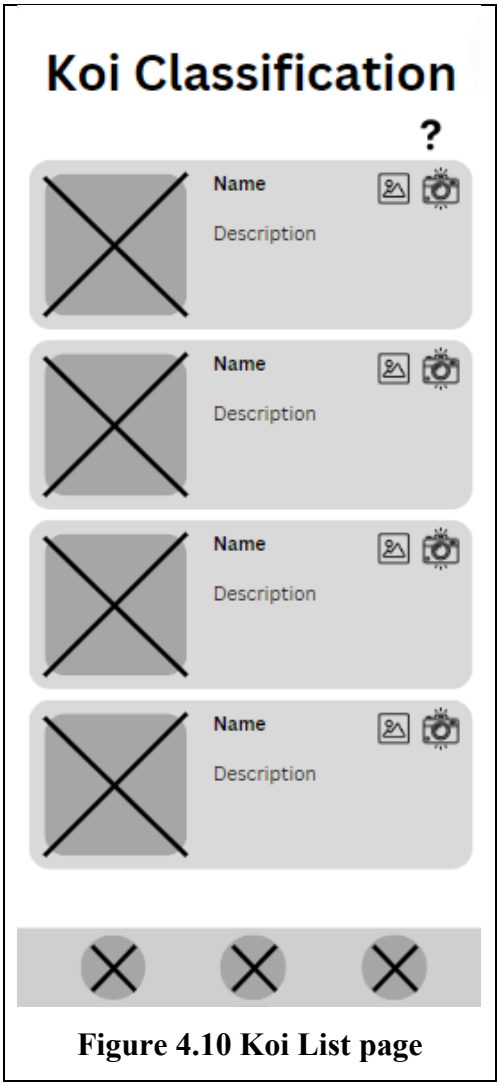


Figure 4.10 Koi List page

Chapter 5

System Implementation

5.1 Data Collection and Annotation

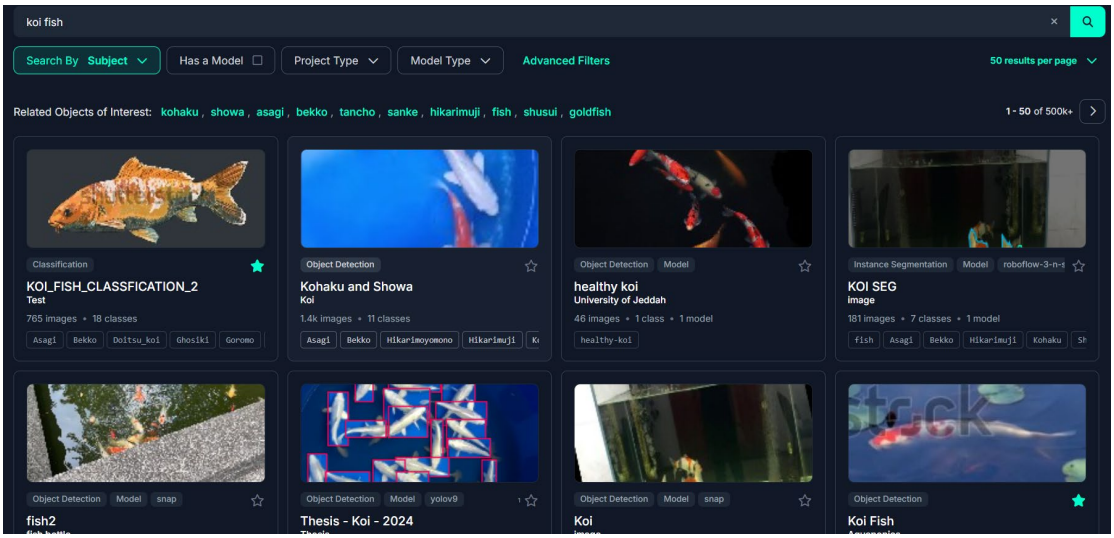


Figure 5.1 Roboflow website for data collection

Before training the model, the koi fish images dataset is downloaded from Roboflow, and additional koi images are gathered from search engines for training purposes. Roboflow [9] is a platform that facilitates the creation, training, and deployment of computer vision models. It offers tools for managing datasets, including image annotation, data augmentation, and version control.

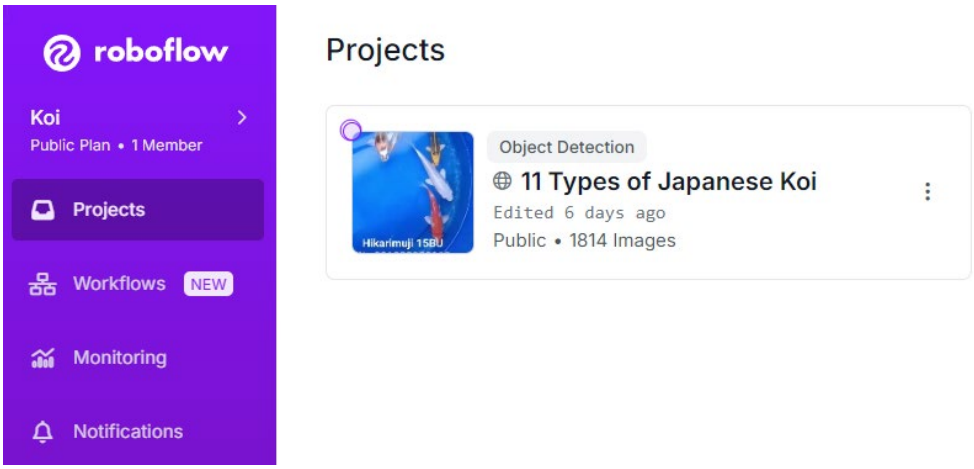


Figure 5.2 Project Creation in Roboflow

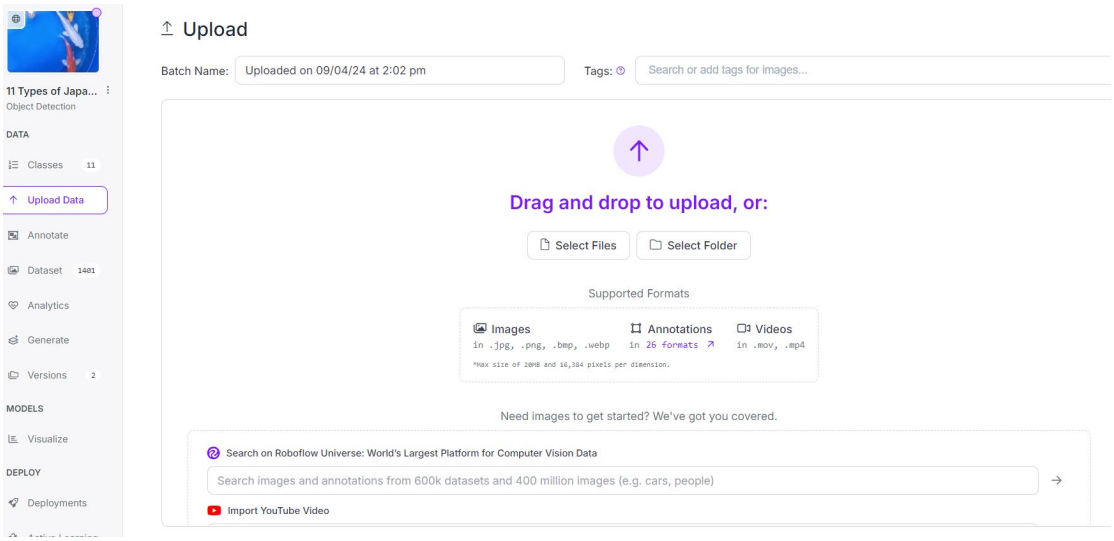


Figure 5.3 Images upload in Roboflow

For the annotation process, the collected images will be annotated using Roboflow. According to Figures 5.2 and 5.3, the initial step involves uploading the collected images to a newly created project in Roboflow before the annotation begins. This preparation ensures that the images are organized and ready for the subsequent annotation tasks.

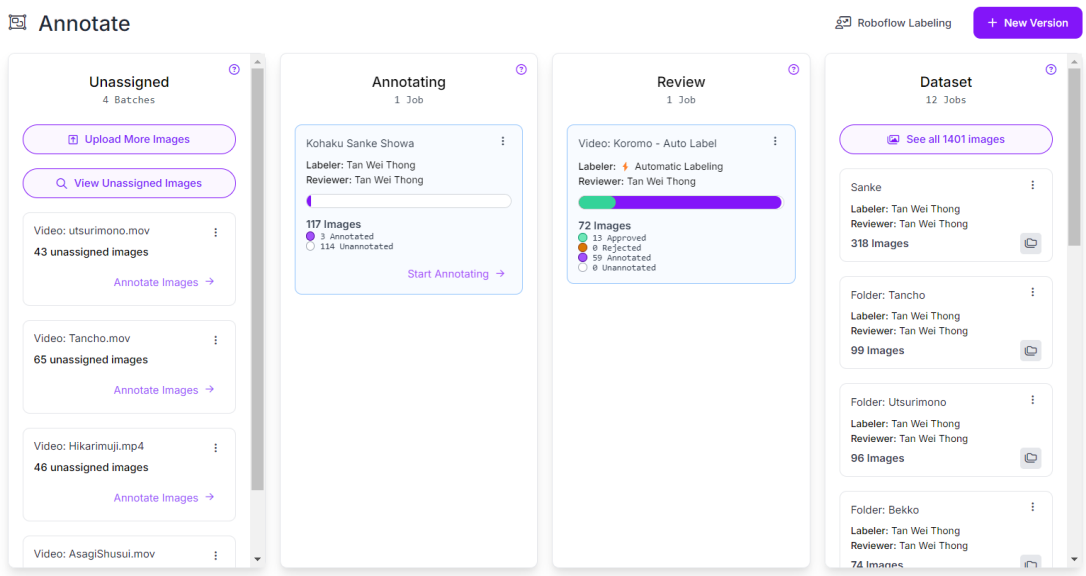


Figure 5.4 Images Assignment and Review in Roboflow

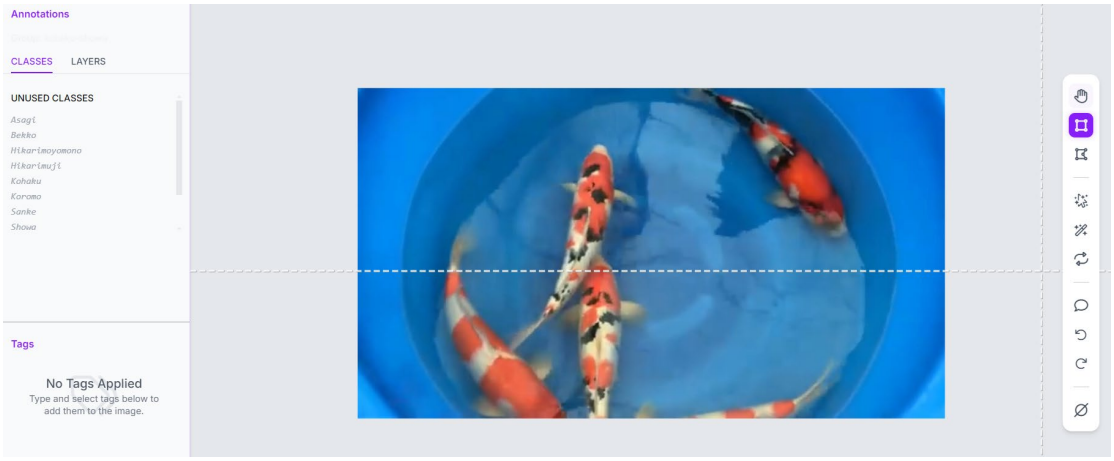


Figure 5.5 Images Annotation in Roboflow

Based on the figure 5.4 and 5.5 above, the image annotation process is divided into four sections: before, during, and after annotation. Initially, after uploading images to the project, they will be placed in the "Unassigned" section. Once images are assigned, they will move to the "Annotating" section, which shows the progress of the annotation. After the annotation is complete, the images will enter the "Review" section for final evaluation. Finally, once the review is finished, the images can be added to the dataset.

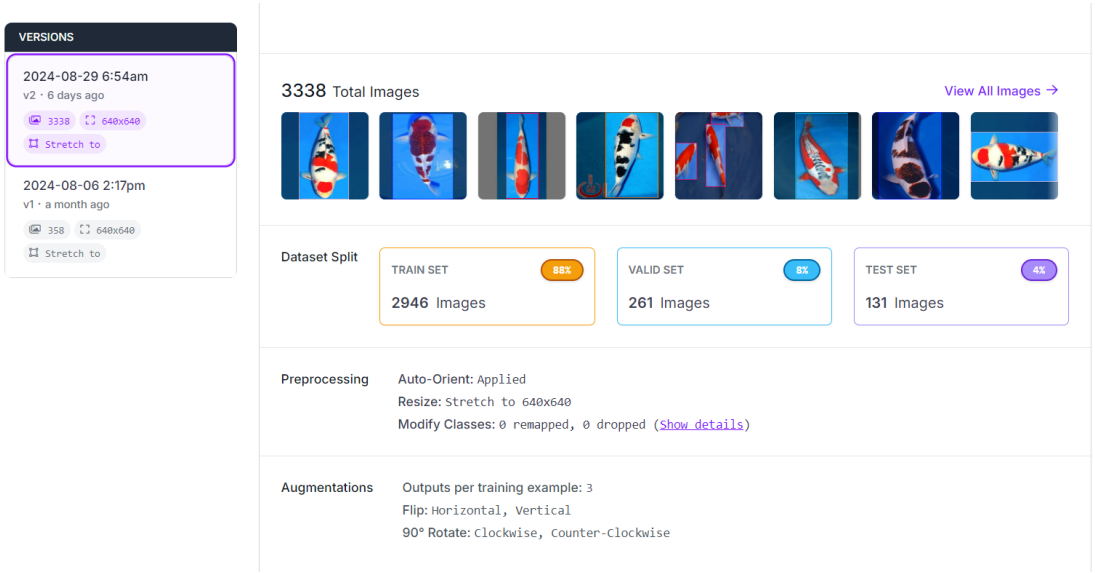


Figure 5.6 Dataset version in Roboflow

Dataset versioning in Roboflow provides a way to manage and track changes to datasets over time. This feature allows for the creation and maintenance of multiple dataset versions, facilitating experimentation with different data modifications, annotations, or augmentations. It enables switching between versions, comparing changes, and ensuring that model training and evaluation utilize the most suitable dataset version.

This approach helps maintain consistency and manage the evolution of data throughout the development process.

5.2 Training and Integration of recognition model

There are two types of model is trained which are Koi price model and Koi type model. The Koi price model will be trained using Teachable Machine as it used for image processing which is a platform that leverages TensorFlow.js while the Koi type model will be trained using Google Colab Notebook using YOLOv8 algorithm as it suitable for both image processing and real-time detection in this project.

5.2.1 Koi Price model

```
!pip install ultralytics==8.0.196
!pip install roboflow
```

Show hidden output

```
[ ] !mkdir datasets
    %cd datasets

    from roboflow import Roboflow
    rf = Roboflow(api_key="AKDfP4DYkkKFsY0mjKC7")
    project = rf.workspace("koi-kccj1").project("koi-price-range")
    version = project.version(1)
    dataset = version.download("yolov8")
```

Figure 5.7: Koi Price dataset preparation

Figure 5.7 show the provided code snippet performs a series of operations to set up and download a dataset from Roboflow using Google Colab Notebook. First, the related library is downloaded. Next, it creates a directory named datasets and changes the working directory to this newly created folder using "!mkdir datasets" and "%cd datasets". Then, it imports the Roboflow class and initializes an instance of it with a specific API key for authentication. The code connects to a workspace and accesses a project titled. Finally, it downloads this dataset in the YOLOv8 format using the version.download("yolov8") method, which prepares the dataset for use in training models with the YOLOv8 object detection framework.

```
[ ] %cd datasets

    !yolo task=detect mode=train model=yolov8s.pt data={dataset.location}/data.yaml epochs=100 imgsz=800 plots=True
```

Show hidden output

```
from ultralytics import YOLO

# Load the YOLOv8 model
model = YOLO("drive/MyDrive/11Koi/koi-price-range.pt")

# Export the model to TFLite format
model.export(format="tflite")  # creates 'yolov8n_float32.tflite'
```

Show hidden output

Figure 5.8: Koi Price Model Training using YOLOv8 and conversion

The code snippet performs several steps to train a YOLOv8 model and convert it to TensorFlow Lite format. First, it changes the working directory to datasets using %cd datasets. It then runs a YOLO command to train a YOLOv8 model using the pre-trained weights yolov8s.pt with the dataset specified in the data.yaml file. The training is set for 50 epochs with images resized to 800 pixels, and plots are enabled to visualize training progress. Following the training, the code imports the YOLO class from the ultralytics library and loads a trained model from a specified path. Finally, the model is exported to TensorFlow Lite format, creating a file named yolov8n_float32.tflite, which is suitable for deployment on mobile or edge devices.

assets	5/9/2024 10:49 AM	File folder	
variables	5/9/2024 10:49 AM	File folder	
best_float16.tflite	5/9/2024 10:49 AM	TFLITE File	21,955 KB
fingerprint.pb	5/9/2024 10:49 AM	PB File	1 KB
metadata.yaml	5/9/2024 10:49 AM	Yaml Source File	1 KB
price.tflite	5/9/2024 10:49 AM	TFLITE File	43,828 KB
saved_model.pb	5/9/2024 10:49 AM	PB File	44,010 KB

Figure 5.9: Downloaded Koi Price model files

When download the model folder from Google Colab Notebook, you'll typically find multiple files in the folder. One of these files contains the labels for each class “metadata.yaml”, which helps you interpret the model's output. The labels provide a human-readable description of what each class represents, making it easier to understand the model's predictions. The other file is the TensorFlow Lite (tflite) model file “price.tflite” itself. This file contains the trained machine learning model, which

has been optimized for deployment on mobile or embedded devices. The tflite model is a compact representation of the original model, designed to be fast and efficient for use in real-time applications.

5.2.2 Koi Type Model

```
[ ]  !pip install ultralytics==8.0.196
     !pip install roboflow
```

Show hidden output

```
!mkdir datasets
%cd datasets

from roboflow import Roboflow
rf = Roboflow(api_key="AKDfP4DYkkKFsY0mjKC7")
project = rf.workspace("koi-kccj1").project("11-types-of-japanese-koi")
version = project.version(2)
dataset = version.download("yolov8")
```

Figure 5.10 Koi Type dataset preparation

As Figure 5.10, similar to Koi Price model, the koi type dataset is needed to be downloaded and stored in a folder.

```
%cd datasets

!yolo task=detect mode=train model=yolov8s.pt data={dataset.location}/data.yaml epochs=50 imgsz=800 plots=True
```

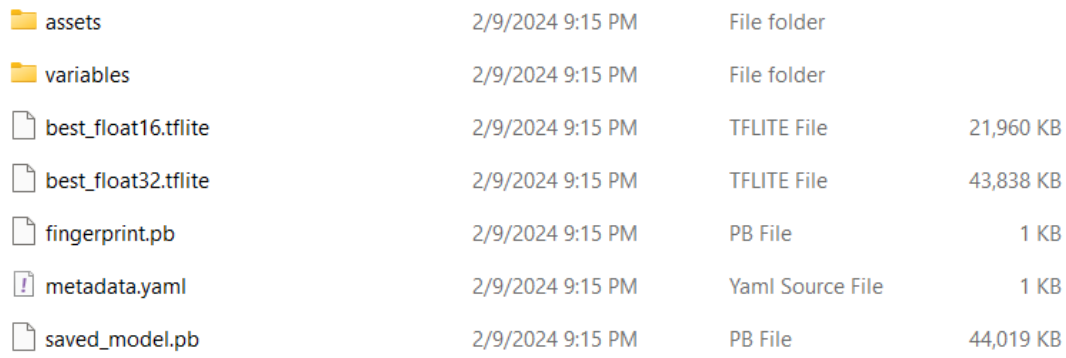
Show hidden output

```
from ultralytics import YOLO

# Load the YOLOv8 model
model = YOLO("drive/MyDrive/11Koi/11koimodel.pt")

# Export the model to TFLite format
model.export(format="tflite")  # creates 'yolov8n_float32.tflite'
```

Show hidden output

Figure 5.11 Koi Type Model Training using YOLOv8 and conversion

According to Figure 5.8, there is a command to train the koi type model similar to train koi price model.

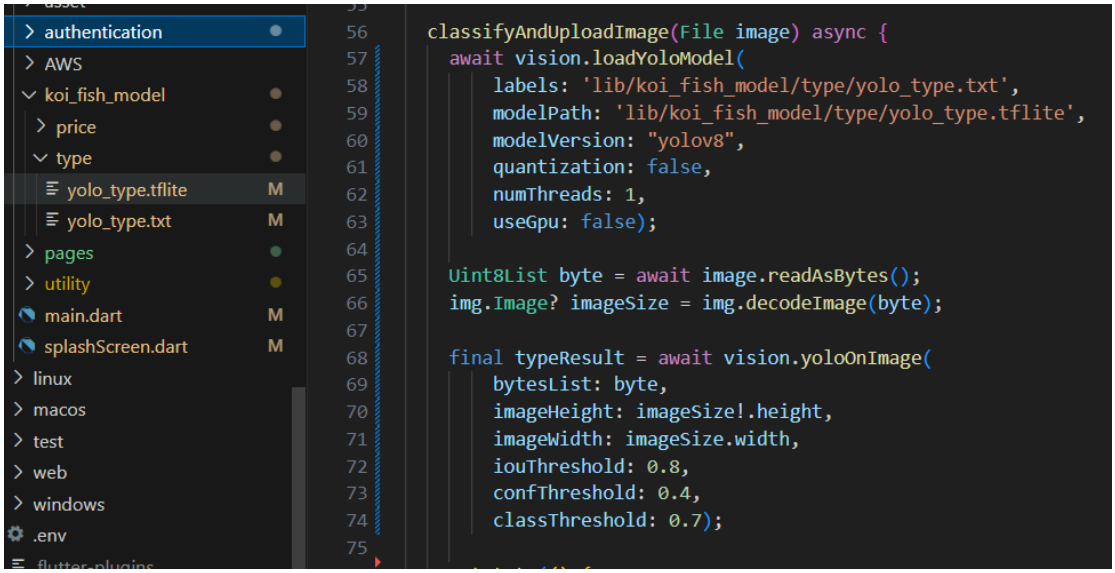


assets	2/9/2024 9:15 PM	File folder	
variables	2/9/2024 9:15 PM	File folder	
best_float16.tflite	2/9/2024 9:15 PM	TFLITE File	21,960 KB
best_float32.tflite	2/9/2024 9:15 PM	TFLITE File	43,838 KB
fingerprint.pb	2/9/2024 9:15 PM	PB File	1 KB
metadata.yaml	2/9/2024 9:15 PM	Yaml Source File	1 KB
saved_model.pb	2/9/2024 9:15 PM	PB File	44,019 KB

Figure 5.12: Downloaded Koi Type model files

"best_float32.tflite" and "metadata.yaml" will be used for this project as shown in the figure above similar to koi price model.

5.2.3 Integration of Models



```
authentication
AWS
koi_fish_model
  price
  type
    yolo_type.tflite M
    yolo_type.txt M
pages
utility
main.dart M
splashScreen.dart M
linux
macos
test
web
windows
.env
```

```
classifyAndUploadImage(File image) async {
  await vision.loadYoloModel(
      labels: 'lib/koi_fish_model/type/yolo_type.txt',
      modelPath: 'lib/koi_fish_model/type/yolo_type.tflite',
      modelVersion: "yolov8",
      quantization: false,
      numThreads: 1,
      useGpu: false);

  Uint8List byte = await image.readAsBytes();
  img.Image? imageSize = img.decodeImage(byte);

  final typeResult = await vision.yoloOnImage(
      bytesList: byte,
      imageHeight: imageSize!.height,
      imageWidth: imageSize.width,
      iouThreshold: 0.8,
      confThreshold: 0.4,
      classThreshold: 0.7);
```

Figure 5.13 Models integration

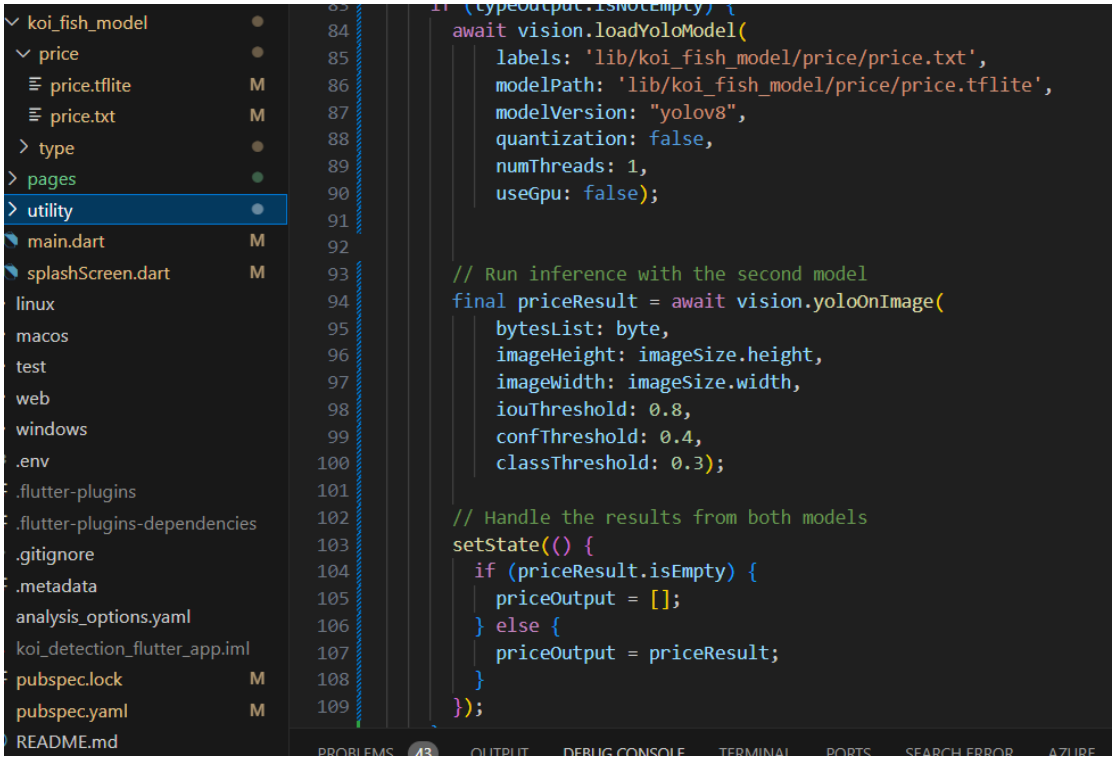


Figure 5.14 Integration of Koi Price model

The provided code snippet in figure 5.10 and 5.11 demonstrates the integration of the Japanese Koi type model and Japanese Koi Price model using the flutter_vision library, as referenced in [10]. This integration allows the models to be loaded and used for koi price classification, koi classification and real time koi detection. The flutter_vision library is crucial for this integration, as it provides the necessary functionality to load and execute TensorFlow Lite models on mobile devices, making it possible to perform detection tasks directly within the Flutter application.

5.3 Amazon Web Services Cloud platform

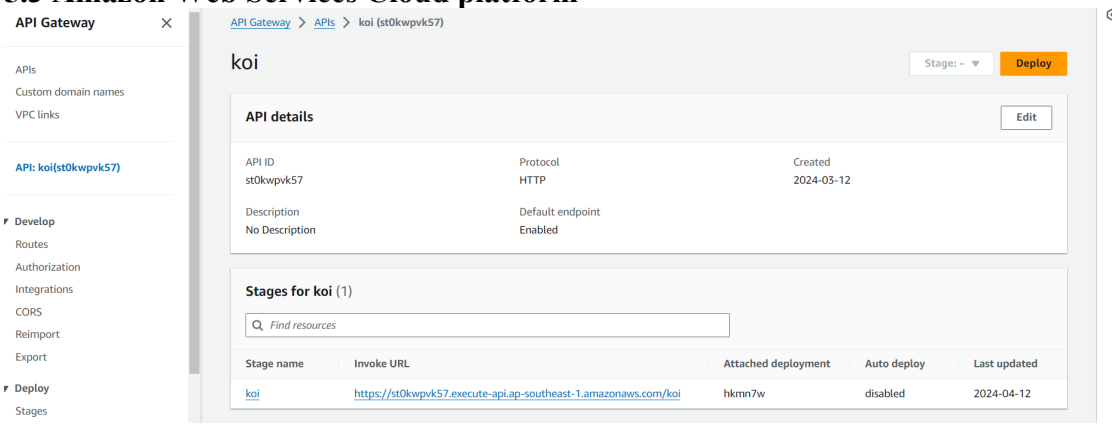


Figure 5.15 AWS API gateway creation

Based on the figure, the "koi" API is developed to manage HTTP requests from the backend within the AWS API Gateway. This API serves as an interface that allows seamless communication between the backend services and the API Gateway, which acts as a gateway for handling incoming requests and routing them to the appropriate backend services. By creating the "koi" API, developers can easily define endpoints and specify how incoming requests should be processed and responded to, providing a flexible and scalable solution for managing HTTP requests in the AWS environment.

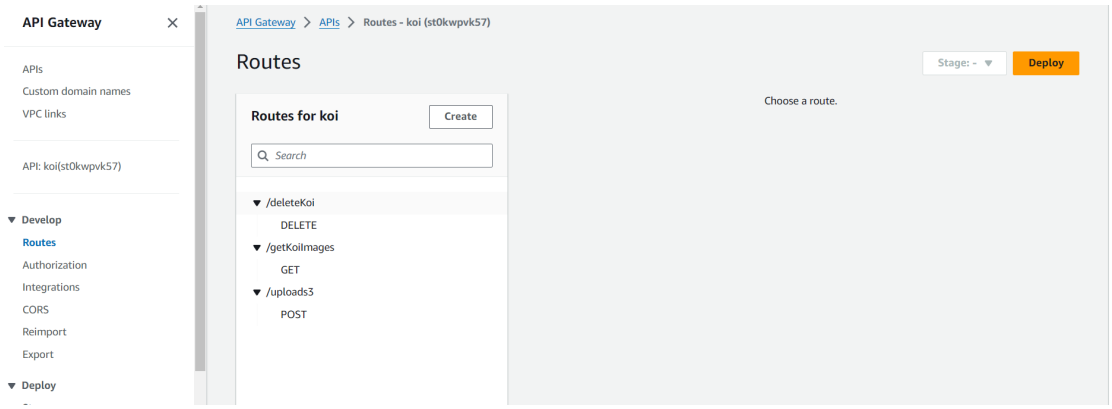


Figure 5.16 Routes create to handle HTTP requests

In the current API Gateway configuration, three routes such as GET, POST, DELETE have been established to handle different types of HTTP requests. By setting up these routes in the API Gateway, a well-structured API is created that can handle a variety of requests and interact with backend services efficiently.

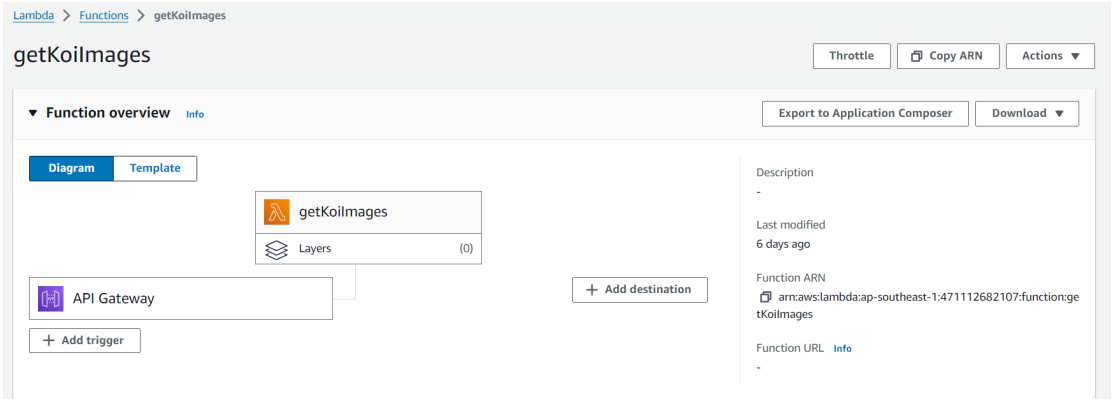


Figure 5.17 Create AWS lambda function to get all user uploaded images and details in S3 Bucket

A lambda function called “getKoiImages” is created and will be triggered by the API Gateway GET request to get the Japanese Koi Image and its result.

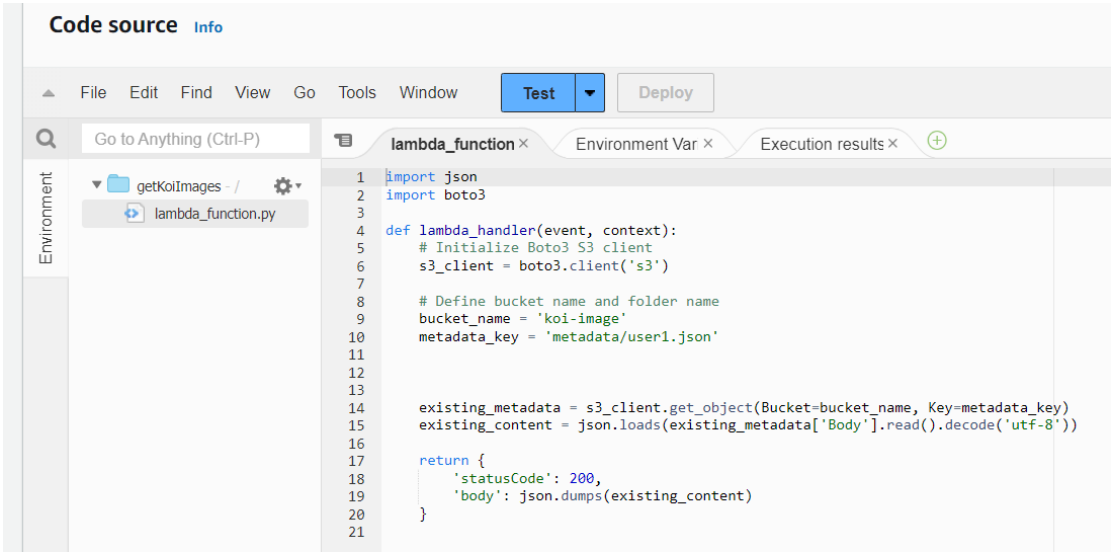


Figure 5.18 Code to retrieve the images and details for current user from S3 Bucket

This lambda function is specifically designed to retrieve Japanese Koi images and their associated results. When the API Gateway receives a GET request, it triggers the "getKoiImages" lambda function, which then retrieves the necessary data from S3 Bucket. The lambda function is responsible for processing the request, retrieving the relevant information, and returning it to the API Gateway. This information includes the Japanese Koi images, as well as metadata associated with them.

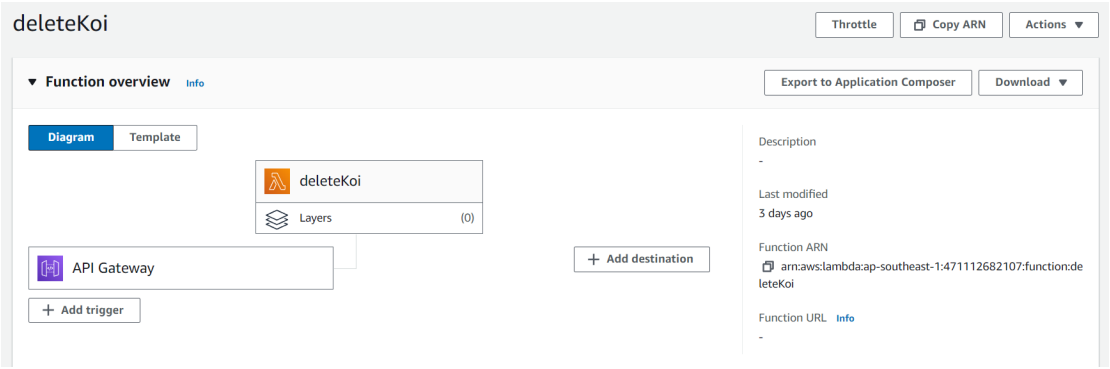


Figure 5.19 Create AWS lambda function to delete selected Japanese Koi image in S3 Bucket

A lambda function called "deleteKoi" is created and will be triggered by the API Gateway DELETE request to delete the Japanese Koi Image and its result in S3 Bucket.

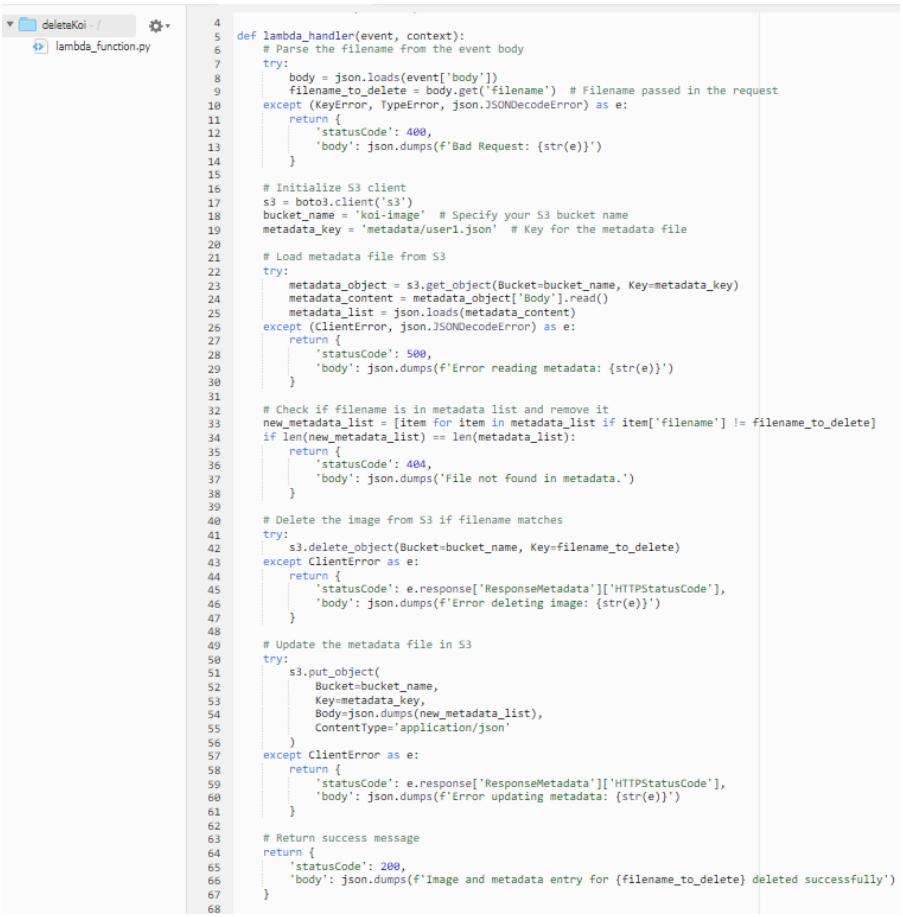


Figure 5.20 Code to delete selected Japanese Koi image from S3 Bucket

This lambda function is specifically designed to delete selected Japanese Koi image and its associated result. When the API Gateway receives a DELETE request, it triggers

the “deleteKoi” lambda function, which then delete the necessary data from S3 Bucket. The lambda function is responsible for processing the request, return message including “fail to delete” or “success to delete” to the API gateway.

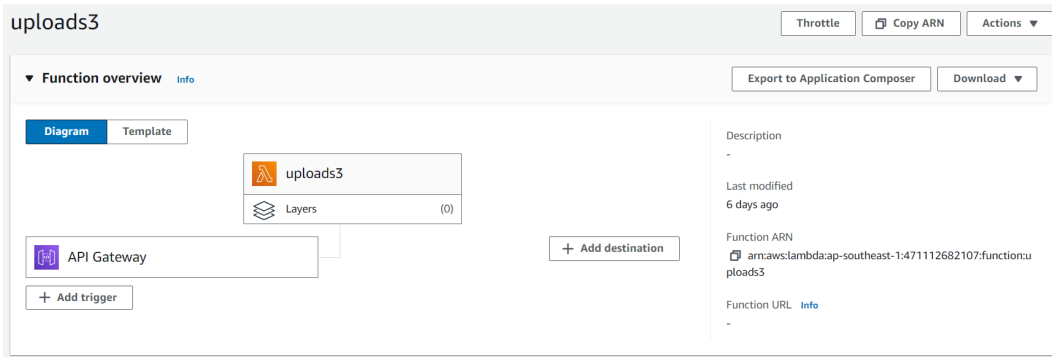


Figure 5.21 Create AWS lambda function to upload Japanese Koi image and its result to S3 Bucket

A lambda function called “uploads3” is created and will be triggered by the API Gateway POST request to upload the Japanese Koi Image and its result in S3 Bucket.

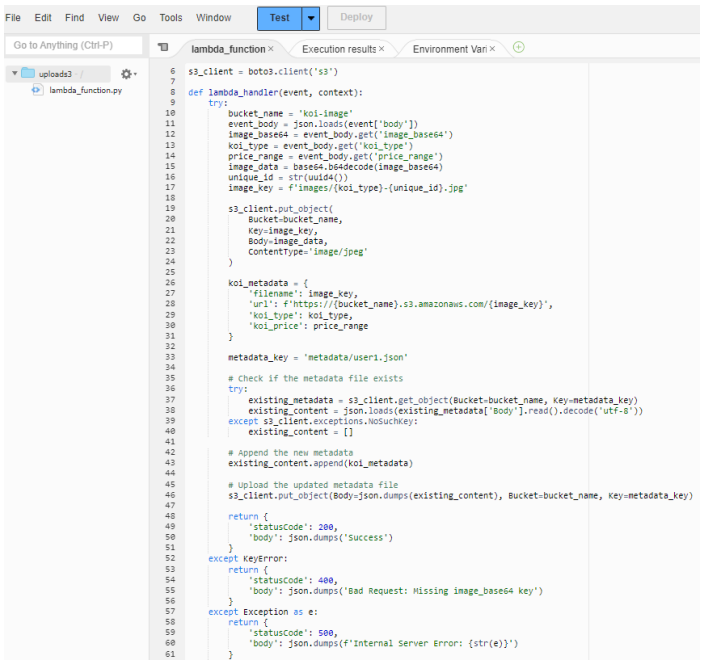


Figure 5.22 Code to upload Japanese Koi image and its detail to S3 Bucket

This lambda function is specifically designed to upload Japanese Koi image and its associated results. When the API Gateway receives a POST request, it triggers the "uploads3" lambda function, which then upload the data to S3 Bucket. The lambda

function is responsible for processing the request, and returning message including “fail to upload” or “success to upload” to the API Gateway.

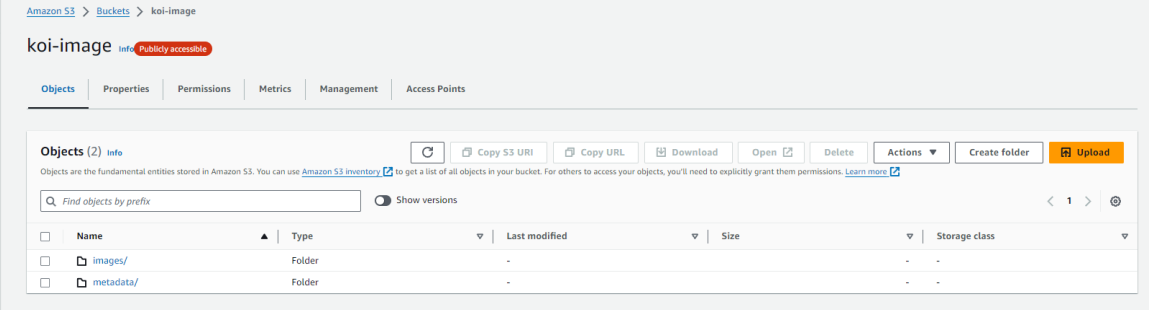


Figure 5.23 Create S3 Bucket

The S3 bucket is created to store the Japanese Koi images and its metadata.

```dart
static Future<void> uploadImage(
    File imagePath, String koiType, String priceRange) async {
  String url =
      'https://st0kwpvk57.execute-api.ap-southeast-1.amazonaws.com/koi/uploads3';

  final bytes = await imagePath.readAsBytes();
  final String base64Image = base64Encode(bytes);

  // Set up POST request arguments

  Map<String, String> headers = {"Content-type": "application/json"};
  String json1 = jsonEncode({
    'image_base64': base64Image,
    'koi_type': koiType,
    'price_range': priceRange
  });

  // Make async POST request
  var response =
      await http.post(Uri.parse(url), headers: headers, body: json1);

  if (response.statusCode == 200) {
    print('Successfully uploaded image.');
  } else if (response.statusCode == 400) {
    print(response.body.toString());
  } else {
    print('Failed to upload image.');
  }
}
```

Figure 5.24 Code to perform POST request to API gateway

This code defines a method “uploadImage” that facilitates the uploading of an image file, along with associated metadata such as koiType and priceRange, to a specified API endpoint. The method first converts the image file to base64 format, which is a common approach for transmitting binary data over HTTP. It then prepares a JSON payload containing the base64-encoded image data and the metadata. The method uses the http package to send a POST request to the specified URL with the JSON payload in the

request body. It checks the response status code to determine if the upload was successful (status code 200) or if there was an error (status code 400).

```dart
static Future<List<dynamic>> getImageUrlsFromLambda() async {
  String lambdaUrl =
      'https://st0kwpvk57.execute-api.ap-southeast-1.amazonaws.com/koi/getKoiImages';

  try {
    // Make GET request to Lambda function
    final response = await http.get(Uri.parse(lambdaUrl));
    if (response.statusCode == 200) {
      // Parse response body
      final List<dynamic> data = json.decode(response.body);

      return data;
    } else {
      // If the server did not return a 200 OK response,
      // throw an exception with the error message received from the server.
      throw Exception(
          'Failed to load image URLs. Status Code: ${response.statusCode}, Body: ${response.body}');
    }
  } catch (e) {
    // Print detailed error message
    print('Error fetching image URLs: $e');
    // Re-throw the exception to propagate it to the caller
    rethrow;
  }
}
```

Figure 5.25 Code to perform GET request to API gateway

This method, getImageUrlsFromLambda, is responsible for fetching a list of image URLs from a Lambda function endpoint. It uses the http package to make a GET request to the specified Lambda URL. Upon receiving a response, the method checks if the status code is 200 (indicating a successful request). If successful, it parses the response body, which is expected to be a JSON array of image URLs, into a list of dynamic objects. It then returns this list to the caller. If the request is unsuccessful (i.e., the status code is not 200), the method throws an exception, providing details such as the status code and the response body. This allows the caller to handle the error appropriately. In case of any other exceptions during the request (e.g., network errors), the method catches the exception, prints a detailed error message, and rethrows the exception to propagate it to the caller for further handling.

```dart
static Future<void> deleteImageFromS3Bucket(String filename) async {
  print('Attempting to delete file: $filename');
  String lambdaUrl = 'https://st0kwpvk57.execute-api.ap-southeast-1.amazonaws.com/koi/deleteKoi';

  try {
    // Prepare headers for JSON content type
    var headers = {'Content-Type': 'application/json'};
    // Encode the filename into JSON for the request body
    var body = jsonEncode({'filename': filename});

    // Make a DELETE request to the Lambda function
    final response = await http.delete(Uri.parse(lambdaUrl), headers: headers, body: body);

    if (response.statusCode == 200) {
      print('Delete successful: ${response.body}');
    } else {
      // If the server did not return a 200 OK response,
      // then throw an exception.
      throw Exception('Failed to delete image. Status Code: ${response.statusCode}, Body: ${response.body}');
    }
  } catch (e) {
    // Log and re-throw the exception to be handled by the caller
    print('Error deleting image: $e');
    rethrow;
  }
}
```

Figure 5.26 Code to perform DELTE request to API gateway

This method, “deleteImageFromS3Bucket”, is designed to handle the deletion of an image file from an S3 bucket. It utilizes the http package to make a DELETE request to a specified Lambda function endpoint, which is responsible for carrying out the deletion process. The method begins by logging a message indicating that it is attempting to delete the specified file. It then prepares the request by setting the appropriate headers for JSON content and encoding the filename into a JSON string for the request body. Upon sending the DELETE request to the Lambda function endpoint, the method awaits the response. If the response status code is 200, it prints a success message indicating that the deletion was successful. However, if the response status code is not 200, the method throws an exception, providing details such as the status code and the response body to indicate that the deletion failed. In the event of an exception during the request (e.g., network errors), the method catches the exception, logs an error message, and rethrows the exception to allow the caller to handle it appropriately.

5.4 Firebase Authentication

Firebase Authentication allows developers to implement various authentication methods in their applications with minimal effort. In this project, Email and Password Authentication is used. Users can create an account using their email address and a password. Firebase handles the authentication process, including secure password storage, sign-in, and account management.

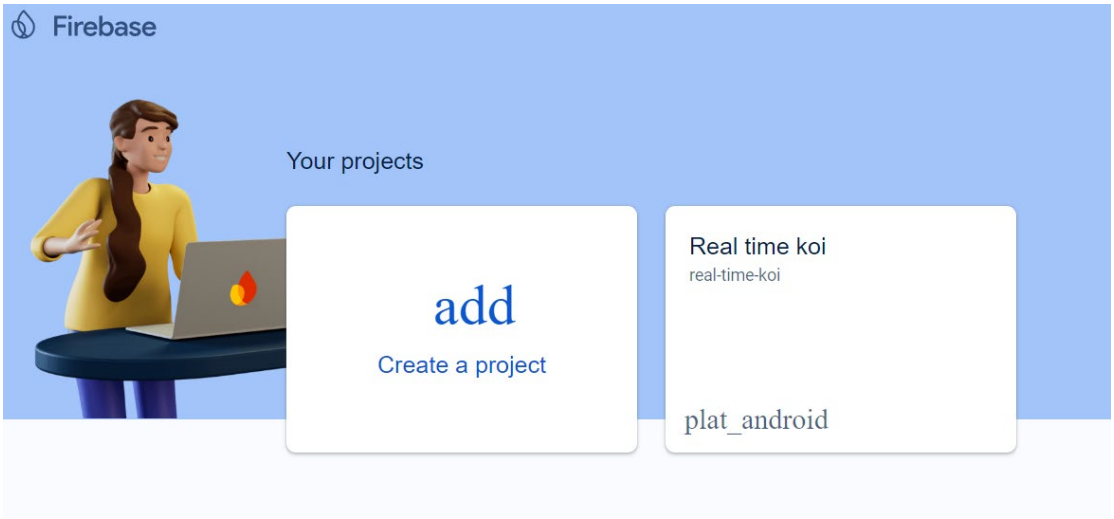


Figure 5.27 Firebase Project Creation

Based on the figure above, it shows Firebase project dashboard. A new project is needed to be created in order to use the Firebase Authentication function.

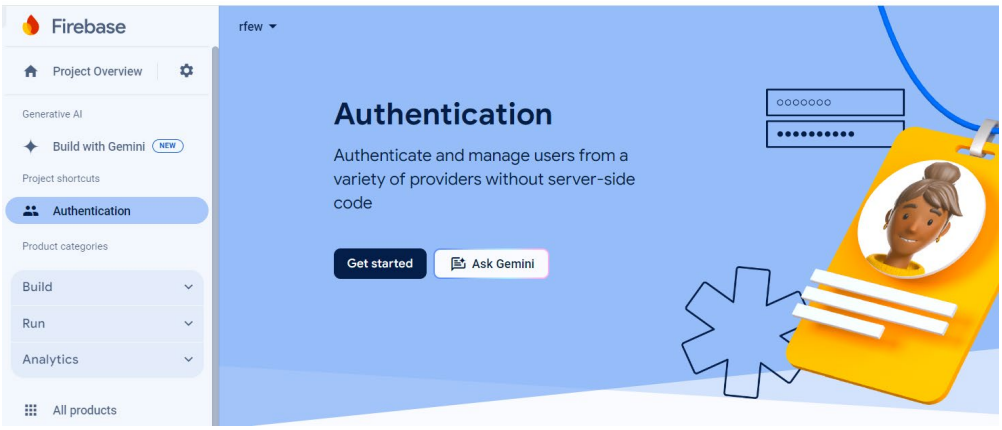


Figure 5.28 Firebase Authentication Setup

There are several functions in Firebase. The authentication is used in this project.

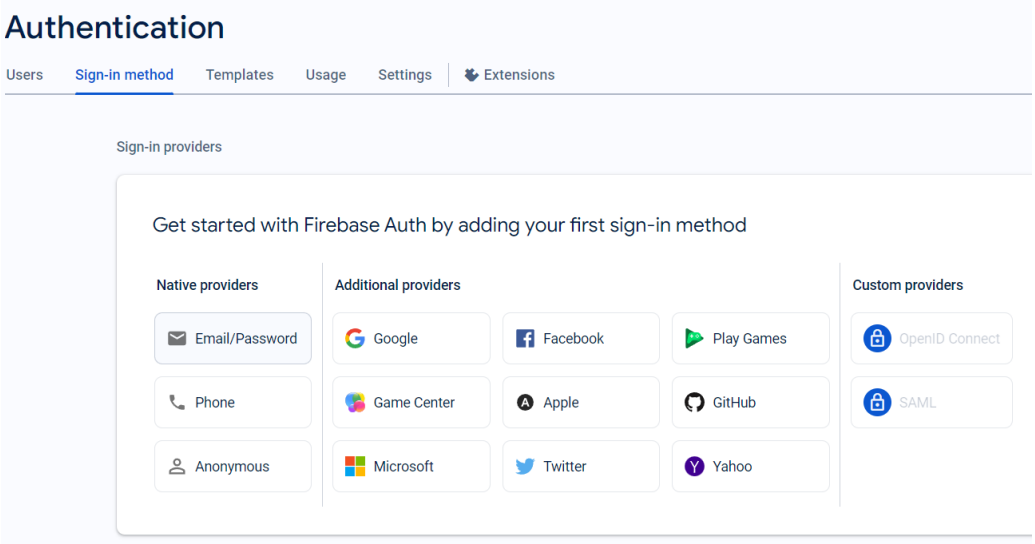


Figure 5.29 Sign-in method selection

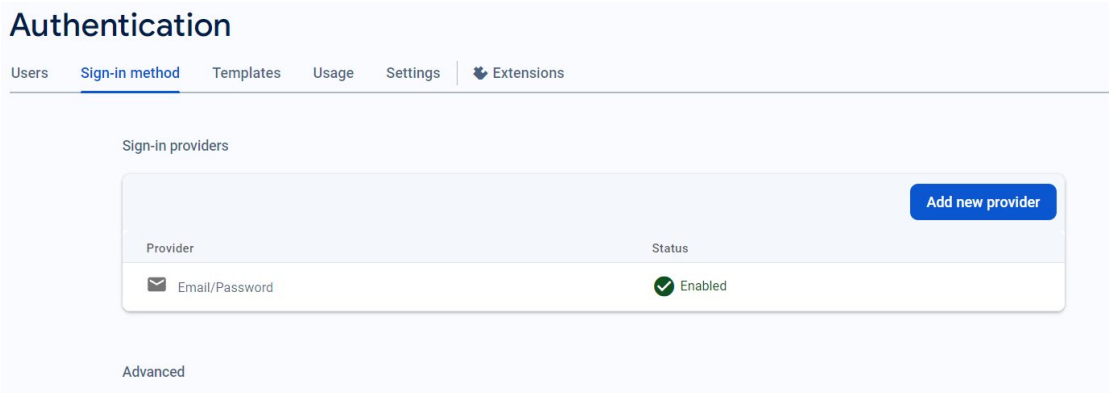


Figure 5.30 Email/Password selection

There are several sign-in method based on figure 5.25. Email/Password is selected in this project according to figure 5.26.

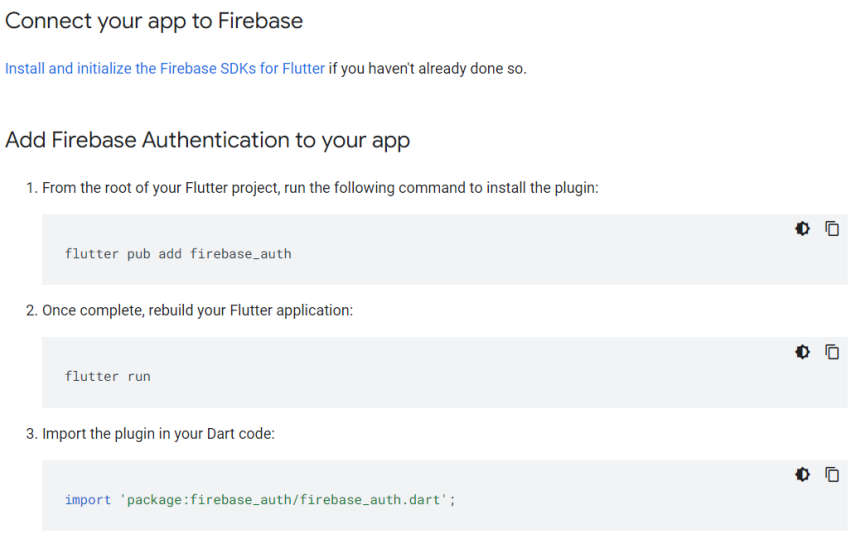


Figure 5.31 Firebase Authentication setup in flutter app

To start using the authentication function, the "firebase_auth" library needs to be added to the Flutter project. After adding it, the project should be rebuilt by running "flutter run". Finally, the library can be imported for use in the project.

```
static const FirebaseOptions ios = FirebaseOptions(
  apiKey: 'AIzaSyBqLWsqFjYAdGyihKTahMRDQMo0N6NVjAs',
  appId: '1:963656261848:ios:d9e01cfe8b675dfcb237ad',
  messagingSenderId: '963656261848',
  projectId: 'flutterfire-ui-codelab',
  storageBucket: 'flutterfire-ui-codelab.appspot.com',
  iosClientId: '963656261848-v7r3vq1v6haupv0l1mdrmsf56ktnua60.apps.googleusercontent.com',
  iosBundleId: 'com.example.complete',
);
```

Figure 5.32 Credential Storing

In the flutter project, the firebase project credentials, and API key are needed to be store in a file to integrate Firebase with Flutter project.

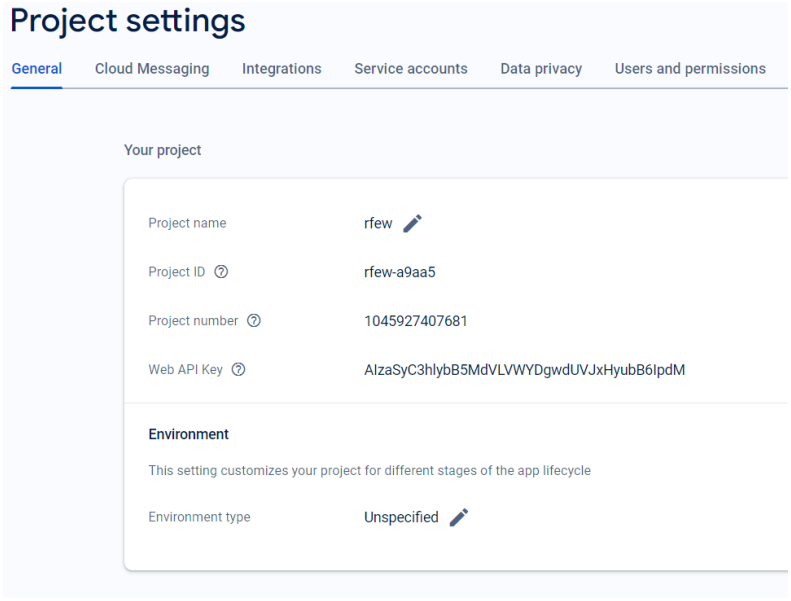


Figure 5.33 Firebase project setting

The credentials and API key can be found in project setting.

```
Future<String?> registration({
  required String email,
  required String password,
}) async {
  try {
    await FirebaseAuth.instance.createUserWithEmailAndPassword(
      email: email,
      password: password,
    );
    return 'Success';
  } on FirebaseAuthException catch (e) {
    if (e.code == 'weak-password') {
      return 'The password provided is too weak.';
    } else if (e.code == 'email-already-in-use') {
      return 'The account already exists for that email.';
    } else {
      return e.message;
    }
  } catch (e) {
    return e.toString();
  }
}
```

Figure 5.34 Create Account function

```dart
Future<String?> login({
  required String email,
  required String password,
}) async {
  try {
    await FirebaseAuth.instance.signInWithEmailAndPassword(
      email: email,
      password: password,
    );
    return 'Success';
  } on FirebaseAuthException catch (e) {
    if (e.code == 'user-not-found') {
      return 'No user found for that email.';
    } else if (e.code == 'wrong-password') {
      return 'Wrong password provided for that user.';
    } else {
      return e.message;
    }
  } catch (e) {
    return e.toString();
  }
}
Future<void> signOut() async{
  await FirebaseAuth.instance.signOut();
```

Figure 5.35 Login and Sign out function

As shown in Figure 5.30 and Figure 5.31, the code snippets provide the functions for account creation, login, and sign-out. With the appropriate credentials and API keys, these functions can be integrated with the login, register, and logout buttons in the user interface.

5.5 Gemini Chatbot

Figure 5.36 Google Gemini AI library

To use Gemini model, “google_generative_ai” library is used in the flutter project.

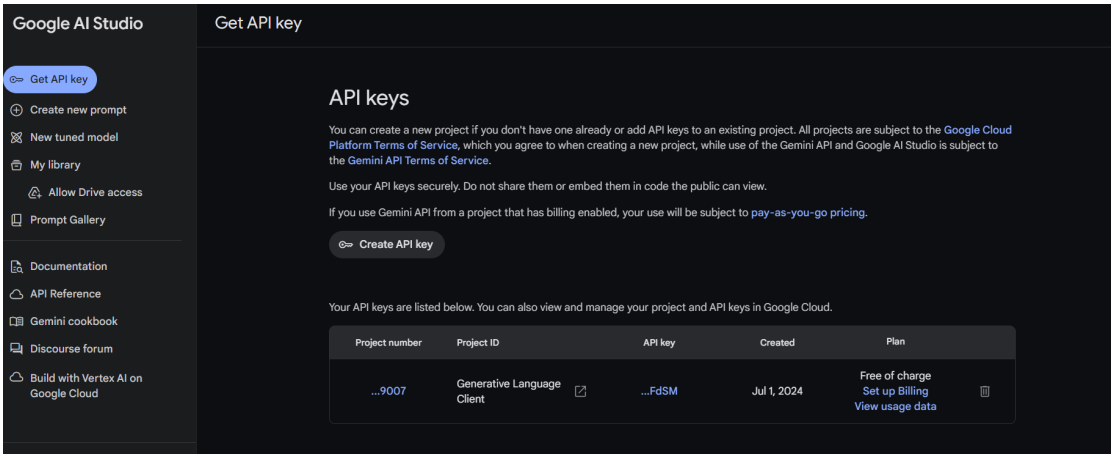


Figure 5.37 Google API key

As shown in the figure, the API key can be obtained from Google AI Studio by creating a new one.

```dart
import 'package:google_generative_ai/google_generative_ai.dart';

const apiKey = ...;

void main() async {
  final model = GenerativeModel(
      model: 'gemini-1.5-flash-latest',
      apiKey: apiKey,
  );

  final prompt = 'Write a story about a magic backpack.';
  final content = [Content.text(prompt)];
  final response = await model.generateContent(content);

  print(response.text);
}
```

Figure 5.38 Library Importing and Model Selection

The code snippet above demonstrates the creation of a chatbot, where the necessary library is imported, and an API key is utilized. The "Gemini-1.5-flash-latest" model is selected for the chatbot. By integrating these essential components with the chatbot's user interface, the chatbot is successfully created.

5.6 System Operation

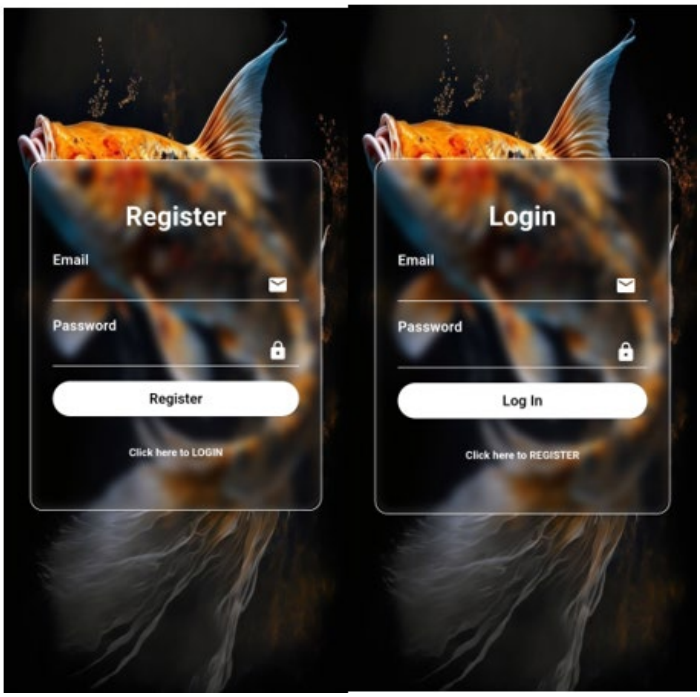


Figure 5.39 Register Screen, and Login Screen

Users can easily create an account and log in using the dedicated registration and login screens. In the registration process, new users are prompted to enter essential details such as a email address, and password. Once the information is submitted, the system verifies its validity, ensuring that the email follows the correct format and unique. Upon successful registration, the user is directed to the login screen.

In the login screen, users who have already registered can enter their credentials, including their email and password, to access the app. The system securely validates the credentials against the stored data, and if correct, it grants the user access to their personalized account, where they can explore all the app's features. If incorrect details are entered, an error message is displayed, prompting the user to re-enter their information or reset their password. This two-step process ensures a secure and seamless entry into the application.

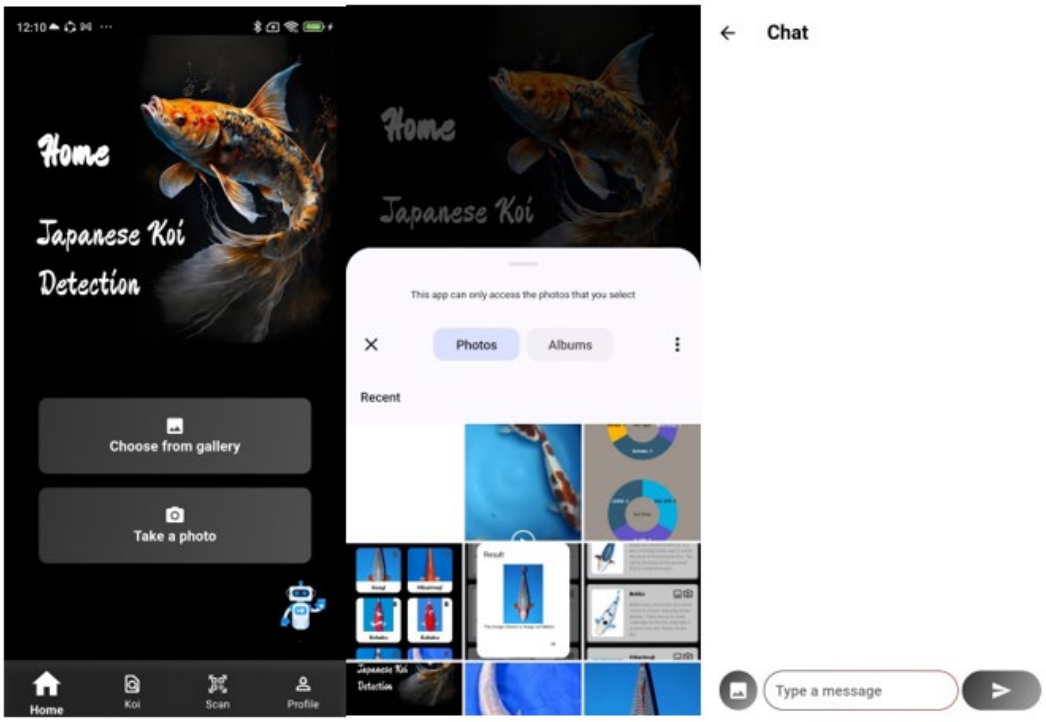


Figure 5.40 Home Screen, Select Image from gallery result Screen and Chatbot Screen

The left side of the figure shows the home screen of the proposed system, featuring two buttons for Japanese Koi detection. Clicking the 'Choose from gallery' button will prompt a bottom sheet to appear, allowing users to select an image from their device's gallery, as depicted in the middle of the figure. Clicking the 'Take a photo' button will direct users to a screen where they can take a photo. Once an image is selected or a photo is taken, it will be sent to the model for classification. The right side of the figure displays chatbot screen which allow users to ask questions about Japanese Koi.

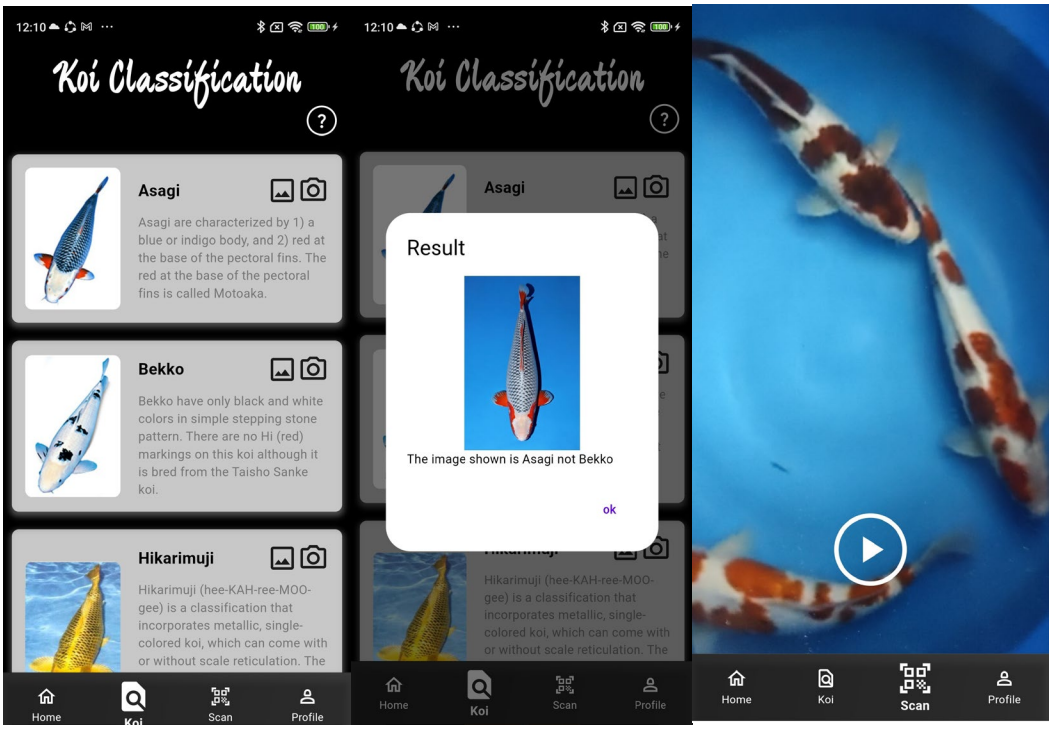


Figure 5.41 Koi List Screen, Result of matching koi screen and Real-time Japanese Koi detection screen.

The left side of the figure show a list of koi type stored in a card. For each card contain the koi type, image and description. Additionally, each card includes two icons: a camera icon to capture a new photo and an image icon to upload one from the gallery. After selecting an image option, the user provides a photo, which the app processes using a pre-trained machine learning model to identify koi fish types. Based on the analysis, the app delivers feedback (as shown in the middle of the figure): if a match is found, a success message confirms the koi matches the selected type; if no match is found, the app notifies the user of the mismatch and suggests trying again with a different image or koi type. The right side of the image is the real-time Japanese Koi detection screen which user can activates the device's camera, capturing a live video feed to detect the real-time Japanese Koi.

Figure 5.42 Collection Screen and Dashboard Screen

The left side of the figure displays the screen where images of Japanese Koi are stored. On the right side, the figure presents statistics of the collected Japanese Koi, including the number of koi types and the distribution across various price ranges.

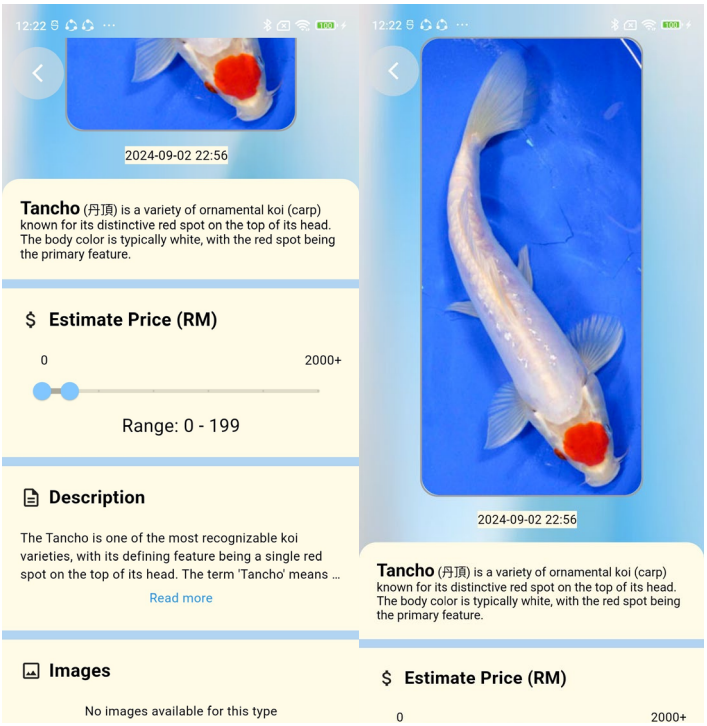


Figure 5.43 Result Screen

The figure above displays the result of the image analysis process. When a user uploads an image for detection or clicks on one of the images in the collection screen (as shown in Figure 5.37), the result will be shown on this screen. The result includes the type of the Japanese Koi, its price range, a description, and related images.

Chapter 6

System Evaluation and Discussion

6.1 Model Evaluation

In this section, two different machine learning models used for koi detection are evaluated: one developed with Teachable Machine and the other using the YOLO (You Only Look Once) algorithm. The evaluation focuses on the effectiveness of each model in two scenarios: real-time detection and static image detection.

6.1.2 Model Overview

Model 1: Teachable Machine Model

During FYP1, Google's Teachable Machine is used as referenced in [11], which leverages TensorFlow.js, to train a model for classifying Japanese Koi types. While the model performed well with static images, it struggled with real-time detection. The primary objective was to utilize its user-friendly interface to develop a model, trained on a labeled dataset of koi images, that could be integrated into our Flutter app for static image classification.

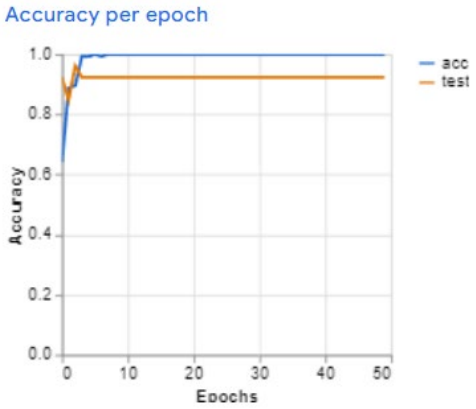


Figure 6.1: Accuracy per epoch

This image shows a plot of **accuracy per epoch** for both the training set (in blue) and the test/validation set (in orange) during the training of a machine learning model.

Key Observations:

1. **Blue Curve (Training Accuracy):**

   The training accuracy increases rapidly within the first few epochs, reaching nearly 100% accuracy. This indicates that the model is quickly learning to

classify the training data correctly. After the initial rapid increase, the training accuracy stays consistently high throughout the remaining epochs.

2. **Orange Curve (Test/Validation Accuracy):**

   The test accuracy also increases significantly during the first few epochs, leveling off at around 90%. After about 10 epochs, the test accuracy stays flat, indicating that no significant improvement in performance on unseen data is happening after this point.

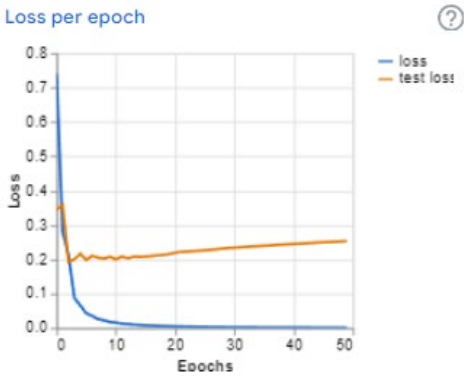


Figure 6.2 Loss per epoch

This figure represents a plot of training loss (in blue) and test (or validation) loss (in orange) over 50 epochs during the training of a machine learning model. Here's what it shows:

1. **Blue Curve (Training Loss):**

The training loss decreases significantly during the first few epochs, indicating that the model is learning from the training data. After about 10 epochs, the curve flattens out, showing that the model has reached a low loss, meaning it is fitting the training data well.

2. **Orange Curve (Test Loss):**

The test loss decreases initially, but around the 5-10 epoch mark, it starts to increase. This is a sign of **overfitting**. After a certain point, the model fits the training data too well and begins to perform worse on unseen test data.

Model 2: YOLO Algorithm Model

During FYP2, A new feature is added to Flutter project for real-time Japanese Koi detection. Model trained using Google's Teachable Machine was not suitable for real-

time object detection. Therefore, the second model utilizes the YOLO algorithm, a state-of-the-art object detection model known for its speed and accuracy in real-time applications. This model was trained on the same dataset, but with the objective of maximizing performance in real-time detection scenarios, as YOLO is particularly well-suited for this purpose.

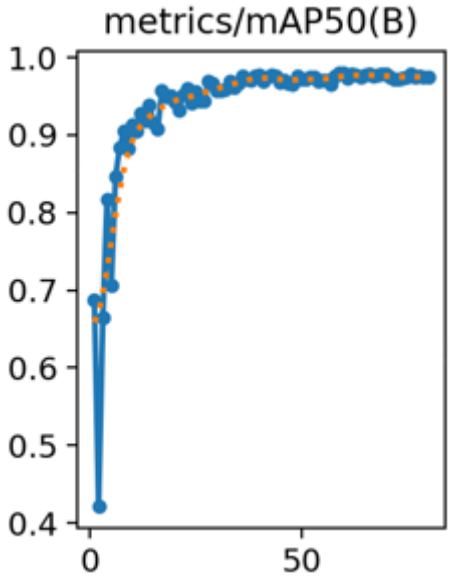


Figure 6.3 Metrics/mAP50(B)

Understanding the Plot:

The provided plot visualizes the mean Average Precision (mAP) metric at a 50% Intersection over Union (IoU) threshold (mAP50) for a given object detection model. This metric is commonly used to evaluate the performance of object detectors, particularly in tasks like image classification and localization.

Key Elements:

- **mAP50(B):** The mean Average Precision for bounding box detection at a 50% IoU threshold.
- **X-axis:** Represents the number of epochs or iterations during the model training process.
- **Y-axis:** Represents the mAP50(B) value.

Interpreting the Curve:

1. **Overall Trend:** The general trend of the curve indicates how the model's performance improves or deteriorates as training progresses. In most cases, you

would expect the curve to rise steadily, indicating that the model is learning to detect objects more accurately.

2. **Convergence:** If the curve plateaus or starts to decrease, it might suggest that the model has converged, meaning that it has reached its optimal performance. This could be due to factors like overfitting or insufficient training data.
3. **Fluctuations:** Small fluctuations in the curve are common and can be attributed to variations in the training data or the model's learning process.

Observations:

- **Rising Trend:** The curve in the provided image shows a clear upward trend, indicating that the model's performance is improving with training.
- **Convergence:** The curve seems to be approaching a plateau, suggesting that the model might be nearing convergence.
- **Fluctuations:** There are minor fluctuations in the curve, which are expected in the training process.

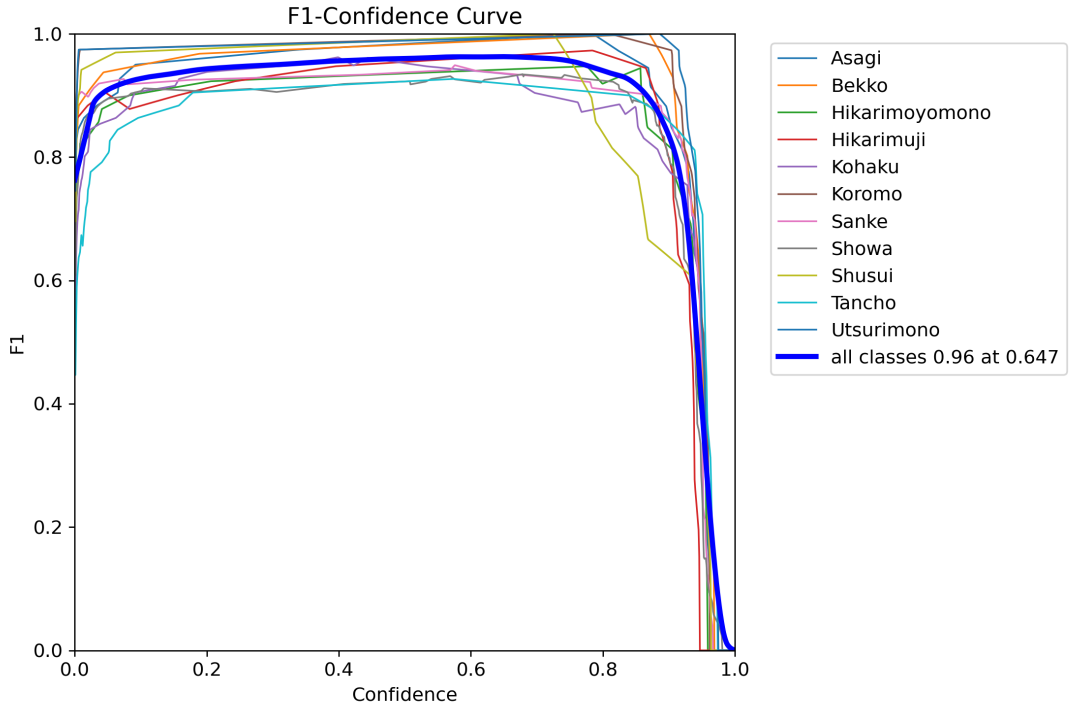


Figure 6.4 F1-Condidence Curve

Understanding the Plot:

The provided F1-Confidence Curve is a visualization tool used in machine learning to assess the performance of a classification model. In this context, the model is likely trained to identify different types of koi fish based on their visual characteristics.

Key Elements:

- **F1-Score:** A metric that combines precision and recall providing a balanced evaluation of a model's performance. A higher F1-score indicates better overall accuracy.
- **Confidence:** The model's estimated probability that a given sample belongs to a particular class.
- **Class Labels:** The different types of koi fish that the model is trying to classify.

Interpreting the Curve:

1. **Overall Performance:** The thick blue line represents the overall performance of the model across all classes. The point where this line reaches its highest F1-score (0.96 in this case) indicates the model's optimal performance.
2. **Class-Specific Performance:** The colored lines represent the performance of the model for each individual koi type. A line that is closer to the top-right corner indicates better performance for that class.
3. **Trade-off between Precision and Recall:** The shape of the curve reveals the trade-off between precision (the proportion of correct positive predictions) and recall (the proportion of positive instances correctly identified). A steeper curve suggests a stronger trade-off, meaning that the model's performance is more sensitive to changes in confidence threshold.

Observations:

- **Strong Overall Performance:** The model achieves a high overall F1-score of 0.96, suggesting that it is generally accurate in classifying koi fish.
- **Class-Specific Variations:** Some classes exhibit slightly different performance levels. For example, "Asagi" and "Bekko" seem to have slightly lower F1-scores compared to other classes.
- **Trade-off Considerations:** The curves for individual classes have varying shapes, indicating different trade-offs between precision and recall. This might be due to the relative frequency of each class in the training data or the inherent difficulty of distinguishing between certain koi types.

6.1.2 Evaluation Criteria

The models were evaluated based on the following criteria:

- **Performance in Real-Time Detection**: The ability of the model to detect koi fish in a live video feed, measured by frame rate (frames per second) and detection latency.
- **Performance in Image Detection**: The ability to accurately classify koi fish from a static image.
- **Usability and Integration**: How easily the models integrate into the Flutter app and their impact on app performance and user experience.

Criteria	**Teachable Machine**	**YOLO Algorithm**
Performance in Image Detection	Perform well in static image	Perform well in static image
Performance in Real-Time Detection	No Response in real-time detection.	Perform well in real-time detection.
Usability and Integration	Easy for static images, difficult for real-time	More complex to integrate, but superior performance

Table 7.1 Comparison between Teachable Machine and YOLO Algorithm

Given these findings, the YOLO model is recommended for use in the Flutter app, especially if real-time detection is a critical feature. While the Teachable Machine model can still be used for simple, static image recognition tasks, the overall user experience and functionality are significantly enhanced by implementing the YOLO algorithm model.

6.2 Project Challenge

Throughout the development of the mobile cloud solution for Japanese Koi recognition, several challenges arose. One of the key obstacles was obtaining a reliable dataset of koi fish images, particularly with labeled information for both types and grades. The koi fish industry lacks a centralized, comprehensive database, which resulted in the need to manually collect and annotate images from various sources. Moreover, ensuring the accuracy and consistency of these labels required significant effort, particularly given the high variance in appearance even within the same species of koi.

Another major challenge was in model integration, specifically achieving real-time detection capabilities. The initial model trained with Google's Teachable Machine was sufficient for static image classification, but it underperformed in real-time detection scenarios. This led to the adoption of a more complex model, YOLO (You Only Look Once), which is optimized for real-time object detection but posed integration and performance hurdles, particularly within the constraints of mobile devices.

Additionally, building a seamless user experience within the app while balancing real-time detection and static image classification posed a significant user interface (UI) challenge. Ensuring that both features worked harmoniously without overwhelming the user, particularly those new to koi fish identification, required extensive iterations and testing. Furthermore, integrating AWS services for cloud storage and processing introduced additional layers of complexity, particularly in maintaining a low-latency connection for real-time operations.

6.3 Objective Evaluation

The project's objectives, as outlined in Chapter 1, aim to bridge the learning gap for new koi enthusiasts and foster a confident environment for identifying different koi species and grades. In terms of evaluation, the project successfully addresses the primary objective of reducing the learning curve by providing a mobile-friendly solution for koi identification. Through the use of cloud-based artificial intelligence models, users can upload images of koi fish for immediate identification, thereby saving time and minimizing reliance on external resources.

Additionally, the app promotes healthy trading by empowering users with the ability to accurately grade koi fish, reducing the likelihood of fraud in koi trading. The integration of machine learning for real-time detection, coupled with an AI chatbot for dynamic learning, ensures a comprehensive and engaging user experience, further aligning with the project's goal of providing a user-friendly platform. While these features demonstrate the project's alignment with its objectives, ongoing improvements in real-time detection accuracy and user interface refinement could further enhance the app's effectiveness.

The third objective, to create a comprehensive and user-friendly platform for recognizing the value of Japanese Koi by leveraging chatbot integration, has been effectively met. The chatbot enhances the user experience by providing interactive guidance and real-time assistance, allowing users to ask questions related to koi identification, care, and trading. This feature facilitates a more dynamic and accessible learning process, helping users quickly resolve queries without needing external resources. The chatbot's natural language processing capabilities enable it to deliver accurate, immediate information, making it a valuable educational tool within the app. While it significantly contributes to the platform's goal of simplifying koi identification

Chapter 7

Conclusion & Recommendation

7.1 Conclusion

The undertaken project addresses significant gaps in the koi fish enthusiast community by developing a sophisticated mobile application designed to simplify the identification of various koi breeds. This initiative aims to transform the koi fish keeping experience, making it more accessible, educational, and enjoyable for both novices and experienced enthusiasts alike.

The research underscored a notable deficiency in current technology with the absence of user-friendly and accurate tools for identifying koi fish types. Despite advancements in aquatic technology focused on water quality and environmental monitoring, there is a lack of efficient solutions for recognizing and differentiating koi fish, necessitating reliance on personal expertise which can be both challenging and time-consuming. This is particularly daunting given the diversity and complexity of koi species, which can change colors and patterns influenced by genetic and environmental factors.

The motivation behind this project stems from the significant ornamental and cultural value of koi fish, which are highly prized in both personal and commercial contexts. The high market and business value of koi fish further highlight the demand for such technology, potentially revolutionizing the commercial aspects of koi breeding and trading.

The proposed mobile application will provide a comprehensive solution by enabling features such as user registration for personalized and secure access, real-time Japanese Koi detection, capabilities for uploading or capturing images directly within the app, and advanced algorithms for analyzing these images to identify and grade koi types. Additionally, the integration of an AI-powered chatbot will offer interactive learning experiences about koi care and breed specifics. The application is designed with a user-friendly interface to ensure easy navigation and utilization, making advanced technology accessible to all users. This project not only fills a technological void but also enhances the educational and commercial experiences associated with koi keeping, enriching community engagement and simplifying the management of these aquatic treasures.

7.2 Recommendation

Based on the successful development and testing of the mobile cloud solution for Japanese Koi recognition and recommendation, several potential improvements and future expansions can be considered to enhance the user experience and extend the application's functionality. First, enhancing the real-time detection capabilities by optimizing models like YOLO for live video feeds could improve the accuracy and usability for users who require instant koi fish identification. Additionally, expanding the dataset to include more koi varieties and higher-resolution images would significantly boost the precision of detection and grading. Incorporating international koi grading standards would also increase the app’s appeal to a broader, global audience.

Further improvements in the user interface and experience (UI/UX) could streamline navigation and offer a more engaging experience, especially by introducing tutorials for new users and improved sorting options for koi image collections. Another major enhancement would be integrating a marketplace where users could list koi for sale directly within the app. This would leverage the identification and grading features to provide verified details, increasing trust and reducing the risk of fraud during transactions. Finally, while the chatbot offers helpful guidance, the app could benefit from additional educational resources like video tutorials or articles on koi care, breeding, and health. These resources could be tailored to users’ detection history and preferences, enriching the learning experience for both new and experienced enthusiasts. By addressing these areas, the app could evolve into a comprehensive platform for koi enthusiasts, offering not just identification, but also trading, education, and an overall richer experience.

REFERENCES

[1] J. Fornaro, "Koi Breeds Commonly Misidentified," [Online]. Available: https://hanoverkoifarms.com/mistaken-identity/.

[2] "Koi Carp Leap from Garden Pond to the World Stage, Enjoying Popularity with Wealthy Asians," 12 November 2020. [Online]. Available: https://www.nippon.com/en/japan-data/h00852/.

[3] "Types of Koi Varieties," [Online]. Available: https://www.kodamakoifarm.com/koi-varieties/.

[4] S. Voon, "Investing in koi," 26 February 2015. [Online]. Available: https://theedgemalaysia.com/article/investing-koi.

[5] Cyborg, "Best Fish Identification Apps for Android and iOS in 2023," 15 August 2023. [Online]. Available: https://naijaknowhow.net/best-fish-identification-apps/.

[6] "Fish Identifier - Fish Verify - Lumos Educational App Store," [Online]. Available: https://www.lumoslearning.com/llwp/resources/applistings.html?iid=1604739565. [Accessed 5 December 2023].

[7] "Make Fish Identification Easy!," [Online]. Available: https://apps.apple.com/ng/app/fishscan-identify-fish/id1577495161. [Accessed 5 December 2023].

[8] "App Detail » Fish Identifier by Picture," ElevenThirteen LLC, [Online]. Available: https://www.148apps.com/app/1626405749/. [Accessed 5 December 2023].

[9] "Roboflow," [Online]. Available: https://universe.roboflow.com/. [Accessed 2 June 2024].

[10] "flutter_vision," [Online]. Available: https://pub.dev/packages/flutter_vision. [Accessed 14 July 2024].

[11] "Teachable Machine," [Online]. Available: https://teachablemachine.withgoogle.com/.

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 3**
Student Name & ID:	
Supervisor: Ts Dr Cheng Wai Khuen	
Project Title: A Mobile Cloud Solution for Japanese Koi Recognition and Recommendation System	

1. WORK DONE

Overall system done such as detection, real time detection, storing image

2. WORK TO BE DONE

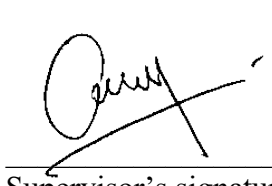
Create statistic for the capture Japanese Koi Image, image filtering and update model

3. PROBLEMS ENCOUNTERED

Time-consuming on annotating Japanese Kois on image, making the model is not up-to date

4. SELF EVALUATION OF THE PROGRESS

Feeling slow to update the model.



Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 10**
Student Name & ID:	
Supervisor: Ts Dr Cheng Wai Khuen	
Project Title: A Mobile Cloud Solution for Japanese Koi Recognition and Recommendation System	

1. WORK DONE

Done creating statistic of the capture image and filtering
Finish updating the model

2. WORK TO BE DONE

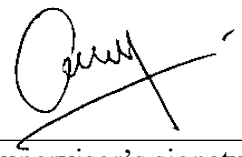
Create another page for learning Japanese Koi making become more learning platform.

3. PROBLEMS ENCOUNTERED

-

4. SELF EVALUATION OF THE PROGRESS

The system is developed with none of the problem so far.



Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 12**
Student Name & ID:	
Supervisor: Ts Dr Cheng Wai Khuen	
Project Title: A Mobile Cloud Solution for Japanese Koi Recognition and Recommendation System	

1. WORK DONE

Create a page enable user to learn the Japanese Koi by capture image.

2. WORK TO BE DONE

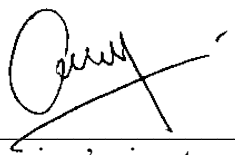
Make the hint and tips more visible to make user better understand the functions of the project.

3. PROBLEMS ENCOUNTERED

-

4. SELF EVALUATION OF THE PROGRESS

Basically, I finish the whole system.



Supervisor's signature

Student's signature

POSTER

Koi Scanner

A mobile cloud solution for Japanese Koi Recognition and Recommendation system

Project Developer: Tan Wei Thong
Project Supervisor: Ts Dr Cheng Wai Khuen

INTORDUCTION

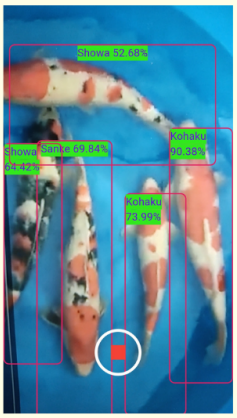
This Japanese Koi Detection system mainly to provide an educational platform for beginner or even koi expert to identify the Japanese Koi type and grade. This would help to learn to understand specific of koi among various variety of Japanese Koi with just an click!!

PROPOSED METHOD

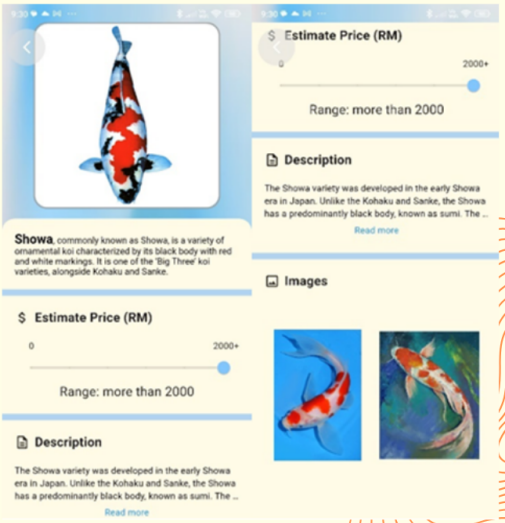
Japanese Koi Type and grade detection

- With just uploading an Japanese Koi fish Image, you can get result directly

REAL TIME DETECTION



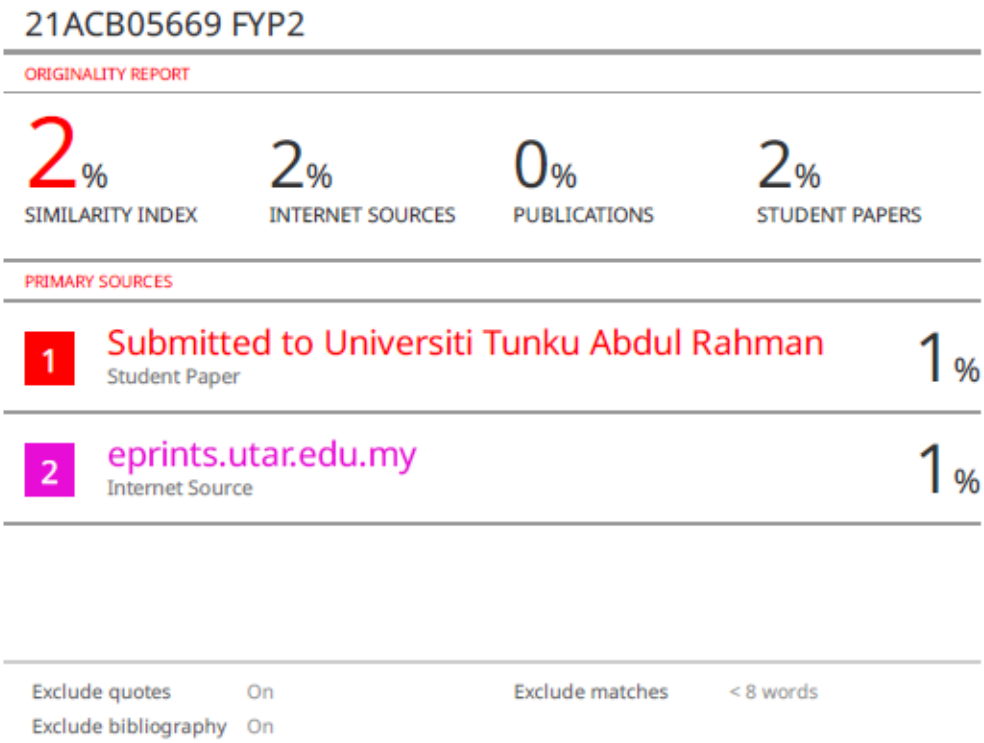
RESULT



CONCLUSION

Japanese Koi Type and Grade Detection feature offers a convenient tool for identifying Japanese Koi. With the integration of a chatbot, it enhances the learning experience, making it a comprehensive platform for enthusiasts of all levels.

PLAGIARISM CHECK RESULT



Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	TAN WEI THONG
ID Number(s)	21ACB05669
Programme / Course	Faculty of Information and Communication / Technology/Bachelor of Information Systems (Honours) Information Systems Engineering
Title of Final Year Project	A Mobile Cloud Solution For Japanese Koi Recognition and Recommendation System

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 2 % **Similarity by source** Internet Sources: 2 % Publications: 0 % Student Papers: 2 %	OK
Number of individual sources listed of more than 3% similarity: 0	

Parameters of originality required and limits approved by UTAR are as Follows:
(i) Overall similarity index is 20% and below, and
(ii) Matching of individual sources listed must be less than 3% each, and
(iii) Matching texts in continuous block must not exceed 8 words
Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Ts. Dr. Cheng Wai Khuen

Date: 9/9/2024

Signature of Co-Supervisor

Name:

Date:

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	21ACB05669
Student Name	Tan Wei Thong
Supervisor Name	Ts. Dr Cheng Wai Kheun

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

(Signature of Student)
Date: 9/9/2024