A CONTEXT-AWARE MOBILE PLATFORM FOR DRUG INTERACTION ALERTS AND PATIENT-CENTERED INSIGHTS

BY

Tan Shin Yi

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

FEBRUARY 2026

COPYRIGHT STATEMENT

This Final Year Project report is submitted in partial fulfillment of the requirements for the degree of Bachelor of Computer Science (Honours) at Universiti Tunku Abdul Rahman (UTAR). This Final Year Project report represents the work of the author, except where due acknowledgment has been made in the text. No part of this Final Year Project report may be reproduced, stored, or transmitted in any form or by any means, whether electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the author or UTAR, in accordance with UTAR's Intellectual Property Policy.

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Ts Dr Chai Meei Tyng, for her guidance and support throughout the development of this safety medication application. She has provided valuable advice, constructive feedback, and continuous encouragement, especially in reviewing the system design and ensuring the accuracy of the project content. This project would not have been possible without her dedication and assistance. A million thanks to you.

ABSTRACT

Medication safety remains a critical healthcare challenge, including risks such as pill misidentification, drug interactions, allergic reactions, scheduling conflicts, and missed-dose errors. This project develops a unified mobile health application integrating six intelligent modules to address these issues.

The system includes pill recognition using ResNet-18 and ResNet-50, drug–drug interaction detection with AI explanations, allergy checking via ingredient matching, real-time drug information retrieval from OpenFDA and DrugBank, pharmacokinetic-aware medication scheduling, and a T-learner-based missed dose decision model trained on 10,000 synthetic patients.

Results show 91.2% accuracy for pill recognition, 0% interaction risk under valid schedules, and a missed-dose model performance of PEHE 7.18, 63.7% accuracy, and 5.2% improvement over rule-based methods.

The novelty lies in integrating six safety modules into a single platform, applying causal learning for missed-dose decisions, and combining pharmacokinetic-aware scheduling with dual CNN-based pill recognition. The system enhances medication safety through an integrated, user-centered framework.

Area of Study: Machine Learning, Mobile Health (mHealth)

Keywords: Pill Image Recognition, Drug-Drug Interaction, Allergy Detection, Smart Drug Scheduling, Missed Dose Decision, T-Learner Model, ResNet, Medication Safety, Clinical Decision Support, Mobile Health Application

TABLE OF CONTENTS

TITLE PAGE **i**

COPYRIGHT STATEMENT **ii**

ACKNOWLEDGEMENTS **iii**

ABSTRACT **iv**

TABLE OF CONTENTS **v-viii**

LIST OF FIGURES **ix-xii**

LIST OF TABLES **xiii**

LIST OF SYMBOLS **xiv**

LIST OF ABBREVIATIONS **xv-xvi**

CHAPTER 1: INTRODUCTION 1-4

1.1 Problem Statement and Motivation 1

1.2 Project Scope 1-2

1.3 Project Objectives 2-3

1.4 Contributions 3-4

1.5 Report Organization 4

CHAPTER 2: LITERATURE REVIEW 5-20

2.1 Review of Existing Mobile Applications 5-9

2.2 Overview of Drug Information Databases 9-10

2.3 Previous Works on Deep Learning 10-14

2.3.1 Pill Image Recognition Using Deep Learning 10-11

2.3.2 Deep Learning for Drug-Drug Interaction Detection 12

2.3.3 Drug Allergy Detection Based on Fuzzy Matching 12-13

2.3.4 Intelligent Medication Scheduling and UMS 14

2.3.5 Knowledge-Based Decision Support Systems 14

2.4 Limitations of Previous Studies 15-16

2.4.1 Limitations of Pill Image Recognition Approaches 15

2.4.2 Limitation of Deep Learning for DDI Detection 15

2.4.3 Limitation of Using Fuzzy Matching in Drug Allergy Detection 15

2.4.4 Limitation of Intelligent Medication Scheduling Systems 16

2.4.5 Limitation of Knowledge-Based DSS 16

2.5 Proposed Solutions 17-20

2.5.1 Proposed Solutions for Pill Image Recognition 17-18

2.5.2 Proposed Solutions for DDI Detection 18-19

2.5.3 Proposed Solution for Drug Allergy Detection 19

2.5.4 Proposed Solution for Intelligent Medication Scheduling Systems 20

2.5.5 Proposed Solution of Knowledge-Based DSS 20

CHAPTER 3: SYSTEM METHODOLOGY 21-26

3.1 System Design Diagram 21-26

3.1.1 System Architecture Diagram 21-22

3.1.2 Use Case Diagram 23-24

3.1.3 Activity Diagram 25-26

CHAPTER 4: SYSTEM DESIGN 27-59

4.1 System Block Diagram 27-28

4.2 Pill Image Recognition Module 28-37

4.2.1 Overall System Architecture of the Pill Image Recognition Module 29-30

4.2.2 CNN-Based Pill Classification System 31-34

4.2.3 ResNet50 Feature Embedding and FAISS Similarity Retrieval System 35-37

4.3 Drug-Drug Interaction Module 37-38

4.4 Drug Information Module 38-41

4.4.1 Drug Information Retrieval and Display 38-40

4.4.2 Active Ingredient Extraction and Allergy Matching 40-41
4.5 Allergy Management Module 42-43
4.6 Smart Scheduling Module 43-54
4.6.1 Drug Schedule Generation and Optimization 44-49
4.6.2 Priority-Based Scheduling Logic 49-50
4.6.3 Risk Calculation and Schedule Adjustment 50-52
4.6.4 Schedule Tracking and Missed Dose Management 53-54
4.7 Missed Dose Decision Module 55-59
4.7.1 Patient Decision Support 55-57
4.7.2 Mapping of User Inputs to Engineered Features 58-59

CHAPTER 5: SYSTEM IMPLEMENTATION 60-101
5.1 Hardware Setup 60
5.2 Software Setup 61
5.3 System Operation 61-99
5.3.1 DrugSafe+ Application Overview 61-62
5.3.2 Pill Image Recognition Module 63-66
5.3.3 Drug-Drug Interaction Module 67-69
5.3.4 Drug Information Module 70-74
5.3.5 Allergy Management Module 75-77
5.3.6 Smart Drug Scheduler Module 78-96
5.3.7 Missed Dose Module 97-99
5.4 Implementation Issues and Challenges 100-101
5.5 Concluding Remark 101

CHAPTER 6: SYSTEM EVALUATION AND DISCUSSION 102-121
6.1 System Testing and Performance Metrics 102-112
6.1.1 CNN-Based Pill Image Classification (Training Phase) 102-107

6.1.2 CNN Embedding-Based Pill Image Representation (Training Phase) 107-111
6.1.3 Performance Evaluation of Missed Dose Prediction Model 112
6.2 Testing Setup and Results 113-118
6.2.1 Testing for CNN Pill Image Classification 113-114
6.2.2 Testing for CNN Embedding-Based Retrieval 115-118
6.2.3 Testing for Missed Dose Decision Module 118
6.3 Project Challenges 118-119
6.4 Objectives Evaluation 119-120
6.5 Concluding Remark 120-121

CHAPTER 7: CONCLUSION AND RECOMMENDATION 122-123
7.1 Conclusion 122
7.2 Recommendation 122-123
REFERENCES 124-126
APPENDIX A-1
POSTER A-1

LIST OF FIGURES

Figure Number	Figure Title	Page
Figure 2-1	Pill Identifier Feature in the Medscape App	6
Figure 2-2	Drug information feature in the Medscape app	6
Figure 2-3	Pill Identifier Feature in the Drugs.com App	7
Figure 2-4	Pill reminder feature in the Pill Reminder app	8
Figure 2-5	Allergy Recorder Feature in the Axilla App	8
Figure 2-6	Pill image recognition framework combining preprocessing and deep learning-based classification	11
Figure 2-7	Workflow of Drug Allergy Detection Using Fuzzy Matching	13
Figure 2-8	Flowchart of the Proposed Pill Image Recognition System Integrating Preprocessing, Data Augmentation, and CNN-Based Classification	18
Figure 2-9	Workflow of Transfer Learning Using Pre-Trained Language Models for DDI Detection	19
Figure 3-1	System Architecture of the Context-Aware Mobile Platform for Drug Interaction Alerts and Patient-Centred Insights	21
Figure 3-2	System Use Case Diagram for the Context-Aware Mobile Platform	23
Figure 3-3	Activity Diagram for Overall System Workflow	25
Figure 4-1	System Block Diagram of the Context-Aware Mobile Platform	27
Figure 4-2	System Block Diagram of the Pill Image Recognition Module	29
Figure 4-3	CNN-Based Pill Classification System Flowchart	31
Figure 4-4	ResNet50 Feature Embedding and FAISS Similarity Retrieval Flowchart	35
Figure 4-5	System Flowchart of the Drug-Drug Interaction Module	37
Figure 4-6	Flowchart of Drug Information Retrieval & Display	39
Figure 4-7	Flowchart of Active Ingredient Extraction and Allergy Matching	40
Figure 4-8	System Flowchart of Allergy Management with Cross-Medication Analysis	42
Figure 4-9	Flowchart of Drug Schedule Generation and Optimization	44
Figure 4-10	Flowchart of Risk Calculation and Schedule Adjustment	50

Figure 4-11	Flowchart of Schedule Tracking and Missed Dose Management	53
Figure 4-12	Flowchart of Missed Dose Decision Module	55
Figure 5-1	Application Logo and Project Name	61
Figure 5-2	Main Interface of the DrugSafe+ Application	62
Figure 5-3	Pill Image Recognition Module Interface	63
Figure 5-4	CNN-Based Classification Results for Fish Oil Sample	64
Figure 5-5	CNN-Based Classification Results for Kremil-S Sample	65
Figure 5-6	CNN Embedding-Based Retrieval Results for Lithium Carbonate Sample	66
Figure 5-7	CNN Embedding-Based Retrieval Results for Creon Sample	66
Figure 5-8	Drug-Drug Interaction Module Interface Design and Medication Selection Function	67
Figure 5-9	Drug-Drug Interaction Report with Risk Classification and Details	68
Figure 5-10	AI-Based Medication Interaction Analysis Interface	69
Figure 5-11	Drug Information Module Interface	70
Figure 5-12	Normal Drug Information View	71
Figure 5-13	Drug Information View with Allergy Alert	72
Figure 5-14	Drug Information View with Allergy Marker and Notification Dot	73
Figure 5-15	Drug Information View with Potential Allergy Warning	74
Figure 5-16	Allergy Management Module Interface	75
Figure 5-17	Allergy Information Card and Editing Interface	76
Figure 5-18	Potential Cross-Reactivity Warning Card	77
Figure 5-19	Smart Drug Scheduler Module Interface	78
Figure 5-20	Frequency and Meal-Time Selection Card in Smart Drug Scheduler Module	79
Figure 5-21	Meal Time Picker for Daily Medication Scheduling	80
Figure 5-22	Desipramine Meal-Time Scheduling Test Case (With Meals)	81
Figure 5-23	Desipramine Meal-Time Scheduling Test Case (After Meals)	82
Figure 5-24	Desipramine Meal-Time Scheduling Test Case (Anytime)	83

Figure 5-25	Desipramine Meal-Time Scheduling Test Case (Before Meals)	84
Figure 5-26	Interaction-Aware Scheduling with Frequency Adjustment	85
Figure 5-27	Multiple Drug Scheduling with Interaction Warning Indicator	86
Figure 5-28	Multi-Drug Scheduling with Interaction Constraints	87
Figure 5-29	Drug Interaction Warnings and Schedule Adjustment Notifications	88
Figure 5-30	Drag-and-Drop Medication Time Adjustment Feature	89
Figure 5-31	Risk Analysis Based on User-Generated Medication Schedule	90
Figure 5-32	24-Hour Risk Analysis Timetable Based on Medication Schedule	91
Figure 5-33	Schedule Control Actions (Update, Reset, and Confirm Functions)	92
Figure 5-34	Medication Tracking Interface with Daily Progress Monitoring	93
Figure 5-35	Main Page with Timetable Overview	94
Figure 5-36	Missed Dose Detection and Automatic Warning with Navigation to Missed Dose Module	95
Figure 5-37	Main Page Daily Timetable Design	96
Figure 5-38	Missed Dose Decision Interface Design with Patient Input Parameters	97
Figure 5-39	Help Function for Kidney Function and Symptom Level Interpretation	98
Figure 5-40	Missed Dose Decision Outcomes Based on User Input	99
Figure 6-1	Dataset Distribution for Pill Image Recognition	102
Figure 6-2	Training and Validation Dataset Split	103
Figure 6-3	ResNet18 Model Architecture for Pill Classification	103
Figure 6-4	Training Performance at Early Epoch	103
Figure 6-5	Training Performance Improvement	104
Figure 6-6	Training Accuracy and Loss Trend	105
Figure 6-7	Best Model Validation Accuracy	105
Figure 6-8	Early Stopping During Training	106
Figure 6-9	Dataset Loading and CSV Structure for CNN Embedding System	108
Figure 6-10	ResNet50 Embedding Model Initialization in PyTorch Environment	108
Figure 6-11	Embedding Extraction Process for All Pill Images	109
Figure 6-12	FAISS Index Construction and Database Creation Output	110

Figure 6-13	Final Embedding Database with Metadata Structure	110
Figure 6-14	Missed Dose Prediction Model Performance Evaluation Results	112
Figure 6-15	Bactidol Pill Classification Result Using CNN Model	113-114
Figure 6-16	Decolgen Pill Classification Result Using CNN Model	114
Figure 6-17	Input Image Upload and Preprocessing Information	116
Figure 6-18	Top-5 Similarity Search Results (FAISS Output)	117
Figure 6-19	Second Test Retrieval Results	118
Figure 6-20	Missed Dose Model Prediction Inference Results	118

LIST OF TABLES

Table Number	Title	Page
Table 2-1	Comparison of Existing Mobile Health Applications	9
Table 4-1	Priority-Based Scheduling Logic for Smart Drug Scheduler	49
Table 4-2	Mapping of User Inputs to Engineered Features for Missed Dose Decision Model	58
Table 5-1	Specifications of Mobile Device	60
Table 5-2	Specifications of Laptop	60
Table 5-3	Software Setup	61
Table 6-1	Performance Summary of CNN Pill Image Recognition Model	107
Table 6-2	Performance Summary of CNN Embedding-Based Pill Image Representation System	110-111

LIST OF SYMBOLS

Symbol	Description
$t_{1/2}$	Drug half-life in hours
μ	Mean vector for ImageNet normalisation
σ	Standard deviation vector for ImageNet normalisation
L	Cross-entropy loss value
y_i	Ground truth label for class i
$\hat{y}_i$	Predicted probability for class i
I_norm	Normalised image tensor
I_tensor	Input image tensor
P(y = i | x)	Probability of class i given input x
z_i	Logit value for class i
e	Raw embedding vector
$\hat{e}$	Normalised embedding vector
sim(q, d)	Similarity score between query q and database image d
C(t)	Normalised drug concentration at time t
k	Elimination rate constant
R_{ij}	Risk score for drug pair (i, j)
R_base	Base risk associated with interaction severity
G_{ij}	Required minimum time gap between drug i and drug j
t_i, t_j	Scheduled administration times of interacting drugs
ITE	Individual Treatment Effect
PEHE	Precision in Estimation of Heterogeneous Effect

LIST OF ABBREVIATIONS

Abbreviation	Description
AI	Artificial Intelligence
API	Application Programming Interface
ATE	Average Treatment Effect
BC5CDR	Bidirectional Encoder Representations from Transformers for Chemical Disease Relations
BioBERT	Biomedical Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Network
CSV	Comma-Separated Values
DDI	Drug-Drug Interaction
eGFR	estimated Glomerular Filtration Rate
FAISS	Facebook AI Similarity Search
FastAPI	Fast Application Programming Interface
FDA	Food and Drug Administration
GLCM	Gray-Level Co-occurrence Matrix
GPT	Generative Pre-trained Transformer
GRU	Gated Recurrent Unit
HOG	Histogram of Oriented Gradients
HTTP	Hypertext Transfer Protocol
HSV	Hue, Saturation, Value
IoT	Internet of Things
ITE	Individual Treatment Effect
JSON	JavaScript Object Notation
kNN	k-Nearest Neighbors
LBP	Local Binary Pattern
LSTM	Long Short-Term Memory
mAP	mean Average Precision
mHealth	Mobile Health
ML	Machine Learning
MongoDB	Humongous Database
NDC	National Drug Code

NER	Named Entity Recognition
NLP	Natural Language Processing
NLM	National Library of Medicine
NumPy	Numerical Python
OCR	Optical Character Recognition
ORB	Oriented FAST and Rotated BRIEF
PEHE	Precision in Estimation of Heterogeneous Effect
RAM	Random Access Memory
ResNet	Residual Network
REST	Representational State Transfer
RGB	Red, Green, Blue
RNN	Recurrent Neural Network
SciBERT	Scientific Bidirectional Encoder Representations from Transformers
SIFT	Scale-Invariant Feature Transform
SSD	Solid State Drive
SURF	Speeded-Up Robust Features
SVM	Support Vector Machine
UMS	Universal Medication Schedule
URL	Uniform Resource Locator

Chapter 1
Introduction

This chapter introduces the motivation and scope of the project. It highlights the limitations of existing medication management systems, particularly in integration and context-awareness. The chapter also outlines the objectives, scope, and contributions of the proposed system.

1.1 Problem Statement and Motivation

Despite the rapid advancement of mobile health applications, existing medication management systems remain fragmented and lack context-awareness, limiting their effectiveness in supporting real-world medication use [1]. Most current solutions provide isolated functionalities such as basic drug reference or simple risk alerts [1][2][3], requiring users to manually interpret and integrate information related to drug identification, safety considerations, and decision-making. This increases the likelihood of user error and reduces system reliability in practical healthcare scenarios.

Furthermore, context-aware technologies in healthcare remain underdeveloped, with limited capability to incorporate user conditions, timing, and risk factors into personalised medication support systems [4]. As a result, users are often required to make medication decisions without sufficient intelligent assistance, particularly in complex multi-medication situations.

Motivated by these limitations, this project aims to develop a context-aware mobile medication management platform that integrates multiple medication-related functions into a unified system. The goal is to move from passive information retrieval towards proactive and personalised decision support by combining medication data with user-specific context. This is especially important for patients with chronic conditions who manage multiple medications daily and require consistent support for safe decision-making. The proposed system aims to improve medication safety, enhance user confidence, and support more effective daily medication management.

1.2 Project Scope

This project delivers a context-aware mobile application prototype for medication management. The final outcome is a fully functional mobile application that integrates multiple

core modules within a unified system. The system includes functionalities for medication identification, drug–drug interaction checking, personalized drug information access, allergy management, medication scheduling, and missed dose decision support. These modules are designed to work together within a single architecture to provide a consistent and user-centered experience.

The scope of this project covers the design and implementation of the mobile application, including the development of the user interface, system logic, and backend processes required to support context-aware functionality. It also includes data processing mechanisms that utilize user inputs and medication-related data to generate personalized responses. However, the project is limited to a prototype-level implementation and does not involve clinical validation or real-world deployment.

1.2 Project Objectives

Primary Objectives

The primary objective of this project is to develop a context-aware mobile medication management platform that integrates multiple medication-related functions into a unified system to enhance medication safety and support informed decision-making.

Sub-Objectives

Objective 1: To enable accurate identification of medications using pill images.

A recognition function is developed to identify medications from captured images by analysing visual features such as shape, colour, and imprint, in order to reduce medication errors caused by variations in pill appearance across manufacturers.

Objective 2: To detect and assess potential drug–drug interactions.

A mechanism is designed to evaluate combinations of medications and identify possible interactions. The system categorises interaction severity and presents warnings to help users avoid unsafe drug combinations.

Objective 3: To provide centralized access to comprehensive drug information.

A feature is implemented to enable retrieval of structured and reliable medication information, including usage, dosage, precautions, and side effects, within a single platform. The system is regularly updated to ensure users receive the most current drug information and provides alerts when medications contain ingredients that match recorded allergies.

Objective 4: To support management and monitoring of medication allergies.

A functionality is introduced to allow users to record known drug allergies within a dedicated allergy management page. The system enables users to view active ingredients, add personal notes, and access potential cross-reactivity information, thereby improving awareness of allergy-related risks and reducing the likelihood of adverse reactions.

Objective 5: To generate optimized medication schedules based on safety constraints.

A scheduling mechanism is developed to organize medication intake by considering interaction risks, appropriate time intervals, and daily routines to ensure safe consumption.

Objective 6: To provide decision support for handling missed medication doses.

A support function is implemented to assist users in deciding whether to take or skip a missed dose based on multiple parameters, including timing conditions and safety considerations.

1.4 Contributions

This project contributes a unified and context-aware mobile medication management platform that integrates multiple medication-related functions into a single system. Unlike conventional applications with static and isolated features, the proposed system incorporates contextual factors such as user condition, timing, and medication relationships to provide more adaptive and relevant decision support. This improves alignment with real-world medication practices and enhances decision reliability, particularly for patients with chronic conditions and older adults managing multiple medications.

From a usability perspective, the system offers a user-centred solution that combines medication information, interaction checking, allergy management, scheduling, and missed-

dose guidance within one platform. This reduces the need for multiple applications and simplifies medication management, improving adherence and reducing medication errors, thereby enhancing patient safety and quality of care.

In addition, this project demonstrates how multiple intelligent modules can be integrated within a single context-aware mobile health architecture. It serves as a reference for future research in combining machine learning with healthcare systems requiring accurate and personalised decision support.

1.5 Report Organization

This report consists of seven chapters: Introduction, Literature Review, System Methodology, System Design, System Implementation, System Evaluation and Discussion, and Conclusion and Recommendations.

Chapter 2 reviews related work in mobile health systems, drug databases, and AI-based approaches for pill recognition, drug interaction detection, allergy management, medication scheduling, and decision support, identifying key research gaps.

Chapter 3 describes the system methodology, including the overall architecture, use case diagram, and system workflow.

Chapter 4 focuses on the design of six core modules: pill image recognition, drug–drug interaction detection, drug information retrieval, allergy management, smart scheduling, and missed dose decision support.

Chapter 5 explains system implementation, including development environment and key challenges such as image recognition limitations and scheduling complexity.

Chapter 6 presents system evaluation, model performance results, and testing analysis.

Chapter 7 concludes the report and provides future improvement recommendations such as dataset expansion, cloud deployment, and model enhancement.

CHAPTER 2

Literature Review

This chapter reviews existing mobile health applications, previous studies on pill recognition, drug-drug interaction detection, and allergy alerts. It highlights the limitations of prior works and shows the need for the proposed system. The chapter also presents the background and technical knowledge required to understand the project.

2.1 Review of Existing Mobile Applications

Existing mobile medication-related applications provide various standalone features such as pill identification, drug information retrieval, reminders, and allergy tracking. However, these functions are often distributed across multiple applications, requiring users to switch between different platforms to obtain complete medication support.

The Medscape application offers several core functionalities. Its pill identifier feature is shown in Figure 2-1, allowing users to identify medication based on pill appearance. In addition, the drug information feature in Medscape, as illustrated in Figure 2-2, provides detailed pharmaceutical data including usage, dosage, and side effects. Similarly, the Drugs.com application also provides a pill identification function as shown in Figure 2-3 and supports drug information access and interaction checking as reflected in Table 2-1.

For medication adherence, the Pill Reminder application focuses mainly on scheduling and reminders, as shown in Figure 2-4, helping users manage their daily medication intake but lacking advanced clinical decision support such as interaction or allergy checking.

In contrast, the Axilla application introduces an allergy recording feature, as shown in Figure 2-5, which allows users to log medication-related allergic reactions for safer medication usage.

Additionally, clinical reference systems such as Micromedex, Epocrates, and Lexicomp are widely used in healthcare settings. These platforms provide highly reliable drug databases, interaction checking, and clinical guidance. However, they are primarily designed for healthcare professionals rather than general users, and their interfaces are often complex and not optimized for everyday patient use.

Overall, as summarized in Table 2-1: Comparison of Existing Mobile Health Applications, each application focuses on specific features rather than providing an integrated solution. This fragmentation forces users to rely on multiple apps to manage different aspects of medication safety, such as identification, scheduling, interaction checking, and allergy tracking. Therefore,

there is a clear need for a unified system that consolidates these functionalities into a single user-friendly platform.

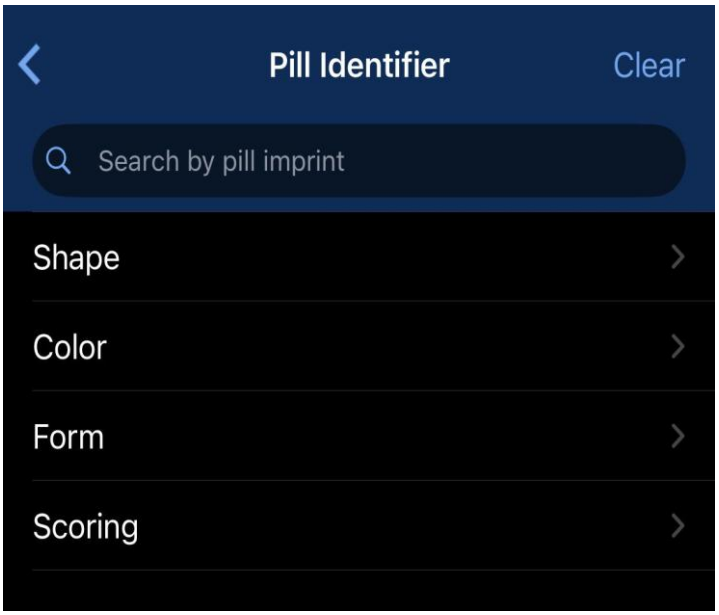


Figure 2-1: Pill Identifier Feature in the Medscape app

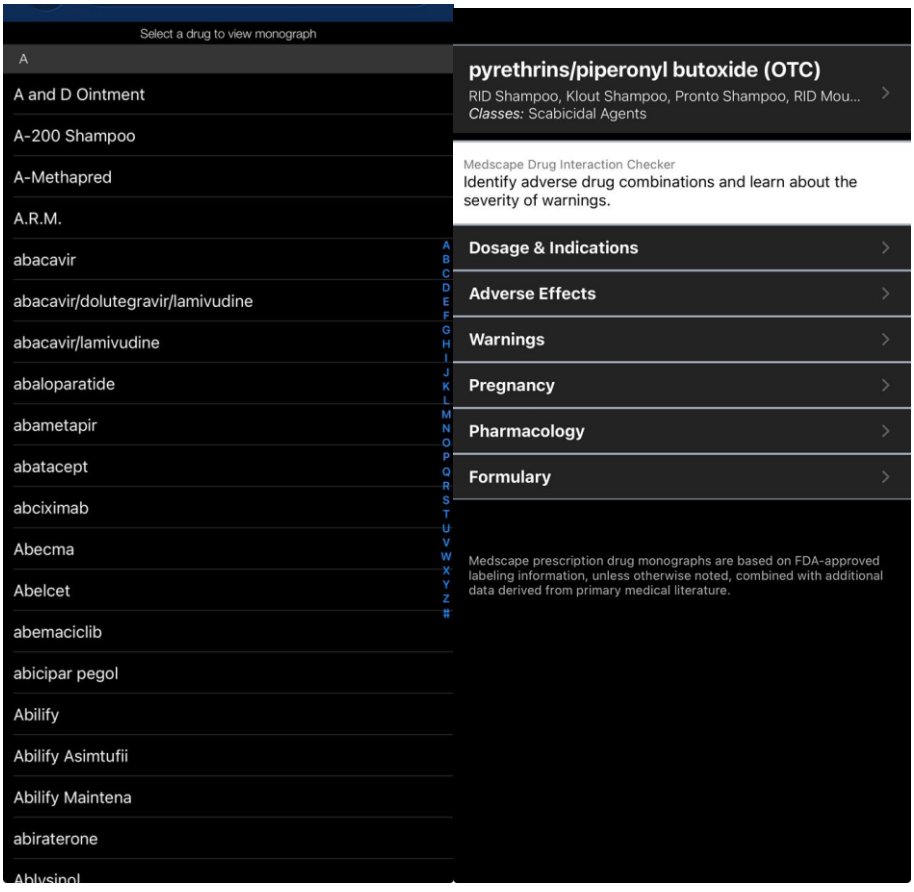


Figure 2-2: Drug information feature in the Medscape app

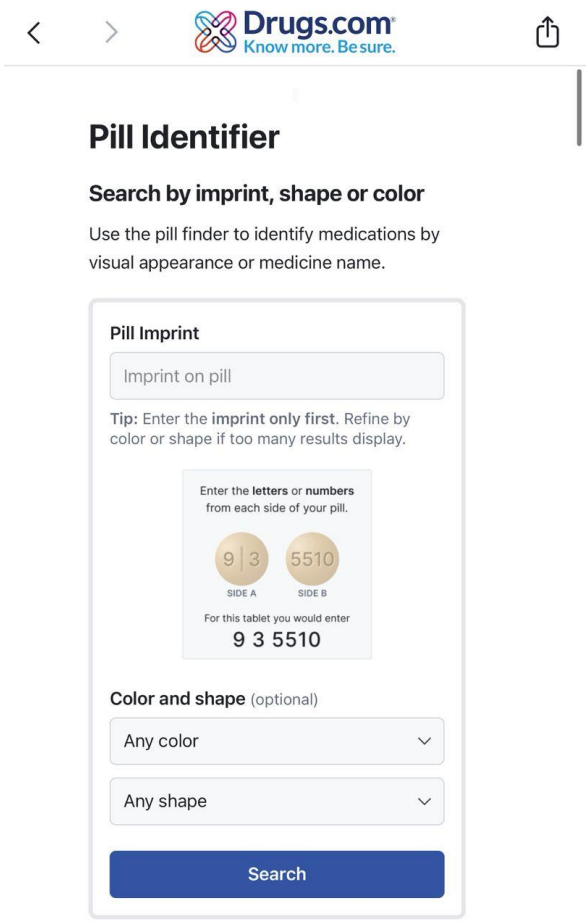


Figure 2-3: Pill Identifier Feature in the Drugs.com app

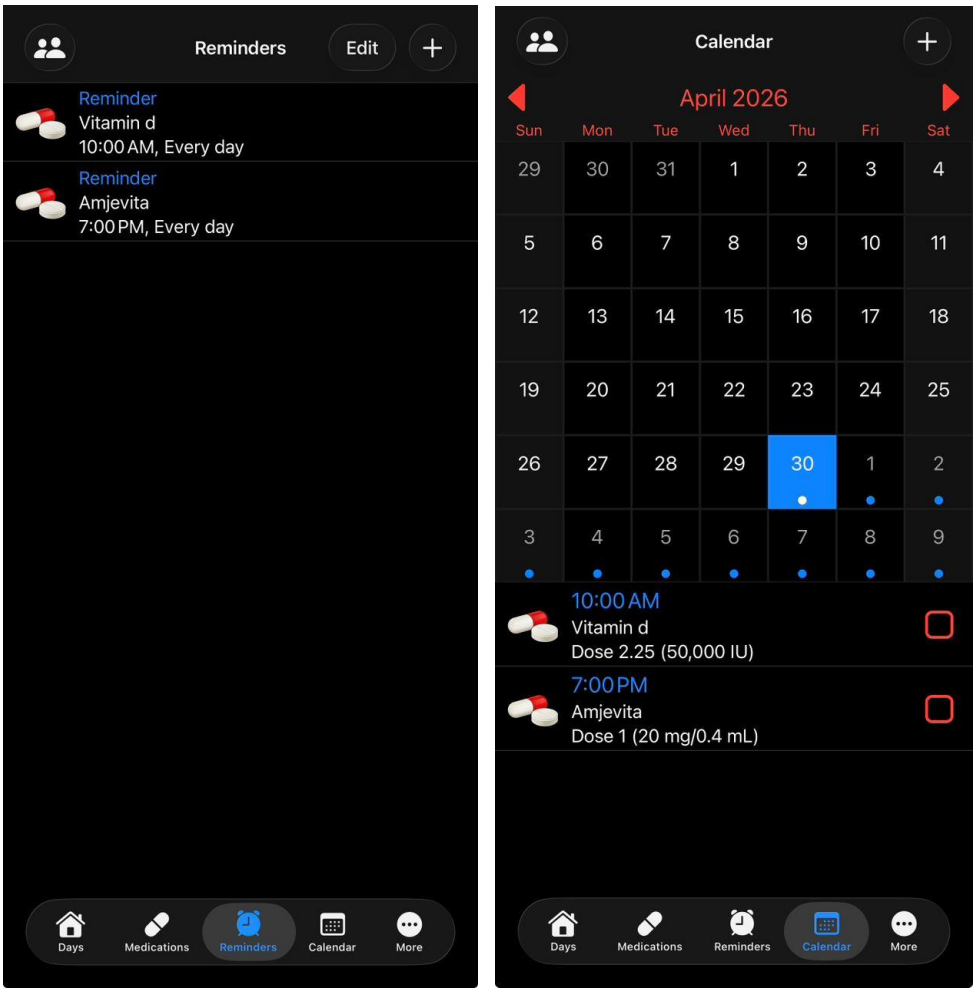


Figure 2-4: Pill reminder feature in the Pill Reminder app

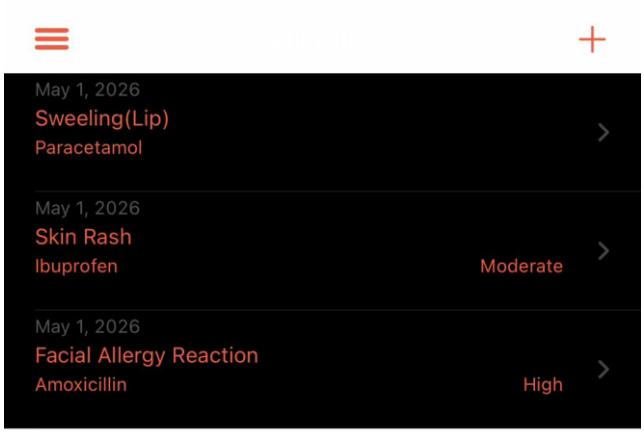


Figure 2-5: Allergy Recorder Feature in the Axilla App

Feature / App	Drugs.com	Medscape	Pill Reminder		Proposed System
Pill Image Recognition		✓			✓
Drug Information Access	✓	✓			✓
Drug-Drug Interaction Alerts	✓	✓			✓
Allergy Alerts					✓
Medication Scheduler and Reminder			✓		
Missed Dose Decision					
Cost	Free with limitations	Free			Prototype under development

Table 2-1: Comparison of Existing Mobile Health Applications [7]

Table 2-1 illustrates the comparison between existing mobile health applications and the proposed system in terms of key medication management features. It highlights those current applications such as Drugs.com, Medscape, and Pill Reminder each provide only partial functionalities, focusing on specific areas like drug information, interaction checking, or medication reminders. In contrast, the proposed system aims to integrate multiple essential features, including pill image recognition, drug interaction alerts, allergy alerts, and medication scheduling—into a single unified platform. This comparison demonstrates the limitation of existing solutions and supports the need for a more comprehensive and integrated medication safety application.

2.2 Overview of Drug Information Databases

Accurate drug information is essential for medication safety, drug interaction detection, and clinical decision support. Common biomedical databases include DrugBank, OpenFDA, and RxNorm, each serving complementary roles.

DrugBank is a comprehensive database providing detailed chemical, pharmacological, and molecular drug information. It supports drug discovery and interaction analysis, including

metabolism and ADMET properties in DrugBank 4.0. However, it focuses mainly on biochemical data and requires external sources for real-time adverse event monitoring [6].

OpenFDA provides structured regulatory data from the U.S. Food and Drug Administration, including drug labels, adverse events, and recalls. It supports pharmacovigilance through RESTful APIs, but is limited to U.S.-based reports, reducing global applicability [7].

RxNorm standardizes drug nomenclature based on active ingredients, dosage forms, and strengths, improving interoperability and reducing naming ambiguity. However, it does not include safety or clinical event data and must be combined with other databases for full functionality [8].

In summary, DrugBank provides pharmacological depth, OpenFDA supports safety monitoring, and RxNorm ensures standardized naming. Together, they form a complementary foundation for medication safety systems.

2.3 Previous Works on Deep Learning

2.3.1 Pill Image Recognition Using Deep Learning

Pill image recognition has been widely studied to improve accurate medication identification. Yaniv et al. introduced the NLM Pill Image Recognition Challenge to evaluate computer vision methods for matching consumer-taken pill images with reference images [9]. The dataset contains 7,000 images of 1,000 pills, and performance was measured using mean Average Precision (mAP). Results showed that deep learning approaches, particularly CNN-based models combined with feature descriptors such as SIFT, achieved the best performance. However, challenges such as lighting variation, background noise, and imprint recognition remain [9].

Further studies have proposed CNN-based models for extracting pill features such as shape, color, and imprint for classification tasks [10]. These models are often combined with classifiers like k-Nearest Neighbors (kNN) or Support Vector Machines (SVM) to improve accuracy. A hybrid CNN + kNN approach achieved over 90% accuracy in real-world settings [10]. Key advantages include automatic feature extraction, improved robustness for subtle visual differences, and scalability for large pill databases.

As shown in Figure 2-6, a typical pill recognition pipeline includes image preprocessing and deep learning classification. Preprocessing may involve filtering, normalization, segmentation, and edge detection for feature enhancement. In the classification stage, models

such as CNN, ResNet-50, and hybrid CNN + kNN or CNN + SVM are commonly used. Experimental results show that CNN + kNN achieves the highest performance (90.8% mAP), outperforming CNN + SVM (86.3%) and ResNet-50 (80%). This demonstrates the effectiveness of hybrid deep learning approaches for pill recognition tasks [10].

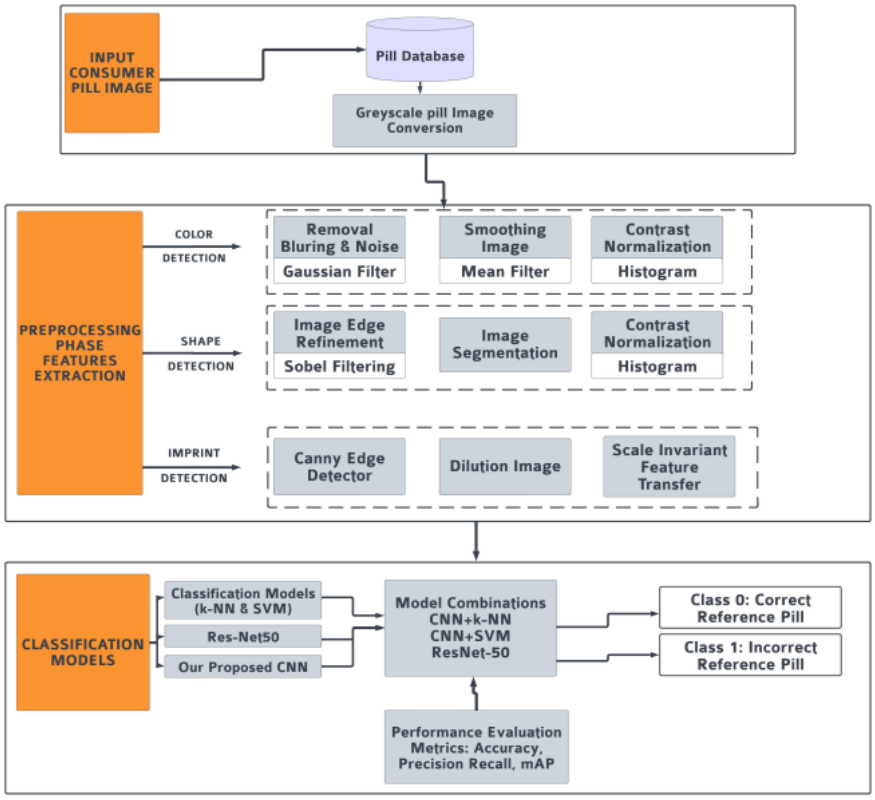


Figure 2-6: Pill image recognition framework combining preprocessing and deep learning-based classification [10]

2.3.2 Deep Learning Approaches for Drug-Drug Interaction Detection

Deep learning methods are widely used for Drug–Drug Interaction (DDI) detection, mainly using CNN, RNN, LSTM, and GRU models to automatically extract interaction information from biomedical text, reducing manual annotation effort [11].

CNN models are effective in capturing local contextual features, such as drug names and nearby keywords, which helps identify whether two drugs are related. They are useful when interaction clues appear within the same sentence or nearby text.

RNN-based models, especially LSTM and GRU, are stronger in modeling sequential and long-distance dependencies in sentences. For example, they can correctly identify interactions even when two drug names are separated by long phrases, as they understand contextual meaning across the sentence.

Overall, deep learning approaches improve DDI detection by reducing manual effort and achieving higher accuracy and recall compared to rule-based methods. Their ability to automatically learn features also improves generalization to unseen data, supporting safer and more scalable clinical decision-making.

2.3.3 Drug Allergy Detection Based on Fuzzy Matching

Drug allergy detection systems often face issues where users enter drug names with spelling errors, abbreviations, or different trade names. Exact matching is not reliable in these cases, so fuzzy matching methods are used to improve accuracy.

Fuzzy matching measures the similarity between user input and standard drug names in the database. Even with minor spelling differences, the system can still identify the correct drug. For example, “Amoxilin” can be matched to “Amoxicillin.”

As shown in Figure 2-7, the process starts when the user inputs or scans a drug name. The system performs fuzzy comparison against the database, and if the similarity score exceeds a threshold, an allergy alert is triggered.

This approach improves fault tolerance and user experience by reducing missed detections caused by input errors. Compared to exact matching, it better reflects real user behavior and is more suitable for mobile-based allergy detection systems [12].

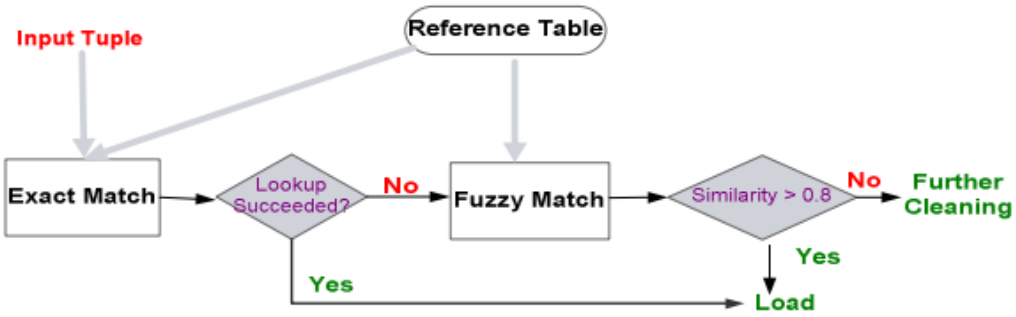


Figure 2-7: Workflow of Drug Allergy Detection Using Fuzzy Matching [12]

2.3.4 Intelligent Medication Scheduling and Universal Medication Schedule (UMS)

Huang et al. [13] proposed a medication reminder system based on a Universal Medication Schedule (UMS) to improve medication adherence and safety. The system integrates drug information, chronopharmacology, and patient-specific factors such as meal and sleep patterns to generate optimized schedules, providing more context-aware planning than traditional reminder apps.

A key contribution is the introduction of Medication Time Constraint with One Drug (MTCOD) and Medication Time Constraint with Multi-Drug (MTCMD). MTCOD defines proper timing for a single drug based on pharmacokinetics and dosage frequency, while MTCMD manages safe time intervals between multiple drugs to reduce interaction risks [13].

The system uses a large drug database combining pharmaceutical and interaction data, enabling automated interaction checking and structured schedule generation aligned with prescriptions and patient routines.

Implemented as a mobile application, it supports real-time reminders, dose confirmation, and delay handling, forming a closed-loop medication management system that improves adherence. However, its reliance on predefined rules and structured data limits adaptability in dynamic clinical environments, requiring further enhancement for real-time intelligence [13].

2.3.5 Knowledge-Based Decision Support Systems

Bindoff et al. [14] proposed a knowledge-based decision support system to assist medication review and detect medication-related problems. The system integrates patient demographics, medical history, symptoms, and medication records to enable context-aware analysis.

Unlike traditional rule-based methods, it uses an incremental learning mechanism that allows experts to update and refine the knowledge base using real clinical cases, improving decision accuracy and consistency over time [14].

The study shows that incorporating clinical context improves the detection of medication-related issues and enhances medication review quality, while maintaining a relatively low false-positive rate. However, the system still relies on expert-defined rules, which limits adaptability in complex or rapidly changing clinical environments [14].

2.4 Limitation of Previous Studies

2.4.1 Limitations of Pill Image Recognition Approaches

Despite strong performance of deep learning-based pill recognition systems, several limitations still exist. Lighting variation, such as changes in brightness, shadows, and reflections, can significantly reduce feature extraction accuracy. Device differences in camera quality and resolution also affect recognition precision. In addition, complex or cluttered backgrounds introduce noise that may lead to misclassification in real-world conditions. These factors collectively reduce the robustness and reliability of pill recognition systems in uncontrolled environments [11][12].

2.4.2 Limitation of Deep Learning for Drug-Drug Interaction Detection

A key limitation of deep learning in drug–drug interaction (DDI) detection is its strong dependency on large amounts of labeled training data. In real-world scenarios, DDI datasets are often sparse and highly imbalanced, with far more negative samples than positive interactions. This can cause models to overfit common patterns while missing rare but clinically important interactions.

In addition, DDI data annotation is costly because it requires biomedical experts to ensure accuracy. These challenges reduce the scalability and reliability of deep learning-based DDI systems in clinical applications and highlight the need for better methods to handle data imbalance and reduce annotation dependence [11].

2.4.3 Limitation of Using Fuzzy Matching in Drug Allergy Detection

Although fuzzy matching is effective in handling spelling errors, abbreviations, and variations in drug names, relying only on this approach has several limitations. It may produce false positives by matching unrelated drugs with similar names, which reduces detection accuracy.

In cases where input is correctly spelled, fuzzy matching is also less efficient and precise compared to exact matching, potentially causing delays in real-time or high-frequency queries and affecting user experience.

Currently, the lack of a hybrid approach combining exact and fuzzy matching limits the overall balance between accuracy and efficiency. These issues highlight the need for a more robust solution to improve the reliability of drug allergy detection systems.

2.4.4 Limitation of Intelligent Medication Scheduling Systems

The study by Huang et al. [15] presents a comprehensive medication reminder system; however, several limitations can be identified.

First, the system relies heavily on structured drug data and predefined rules, which reduces flexibility in handling complex or dynamic clinical scenarios.

Second, its accuracy depends on the completeness and quality of the underlying drug databases, where incomplete data may affect scheduling reliability.

Third, the system requires further validation in real clinical environments, as its performance across diverse populations has not been fully evaluated [15].

2.4.5 Limitation of Knowledge-Based Decision Support Systems

Despite its advantages, the system has several limitations. The development of rules depends heavily on expert knowledge, making it time-consuming and difficult to scale. In addition, rule-based approaches lack flexibility when handling unseen or rapidly changing clinical conditions.

Moreover, the system was evaluated using a limited number of cases and a single expert, which may affect its reliability and generalisability. The absence of data-driven learning techniques further restricts its ability to adapt to complex data patterns [14].

2.5 Proposed Solutions

2.5.1 Proposed Solutions for Pill Image Recognition

To overcome challenges caused by lighting variation and differences in camera quality, a combined approach of image preprocessing, data augmentation, and CNN-based classification is proposed to improve robustness and accuracy in real-world conditions.

Image preprocessing is the first step to standardize input images and improve feature extraction. It includes resizing images to a fixed size for consistent CNN input, noise removal using methods such as Gaussian blur, brightness and contrast normalization to reduce lighting effects, and color space conversion (e.g., RGB to HSV) to enhance color feature detection. These steps reduce the impact of shadows, reflections, and device quality differences, enabling more stable feature extraction [15].

Data augmentation is then applied to increase dataset diversity and improve generalization. It includes geometric transformations (rotation, flipping, scaling, translation), brightness and contrast adjustments, color jittering, and random cropping or occlusion. These techniques help the model learn invariant pill features and reduce overfitting [15].

Finally, CNNs are used to automatically extract deep spatial features such as shape, texture, color, and imprint patterns for classification. When trained with preprocessed and augmented data, CNNs achieve higher stability and accuracy in real-world scenarios [15][16].

Overall, this integrated approach improves recognition performance across different lighting conditions, angles, and devices, ensuring reliable pill identification as shown in Figure 2-8.

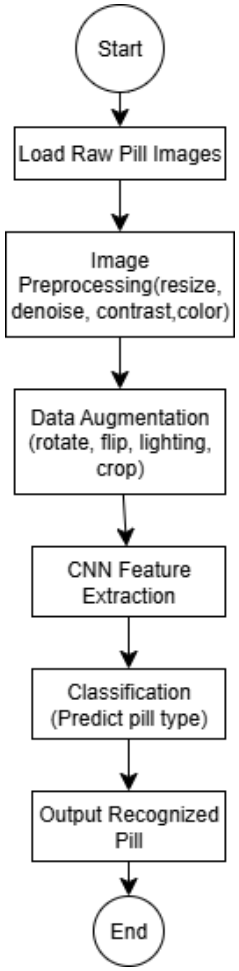


Figure 2-8: Flowchart of the Proposed Pill Image Recognition System Integrating Preprocessing, Data Augmentation, and CNN-Based Classification

2.5.2 Proposed Solutions for DDI Detection

To reduce dependency on large, labeled datasets, the proposed approach uses transfer learning with pre-trained biomedical language models such as BioBERT, SciBERT, and GPT-based models [11]. These models are trained on large biomedical corpora, allowing them to capture domain-specific knowledge and require fewer labeled samples during fine-tuning, thereby reducing annotation cost while maintaining accuracy.

This approach also improves the detection of rare but clinically important drug–drug interactions that are often missed by traditional methods [11]. In addition, transfer learning reduces computational cost, speeds up training, and improves model generalization, making the system more scalable and suitable for real-world clinical use. The overall workflow is illustrated in Figure 2-9.

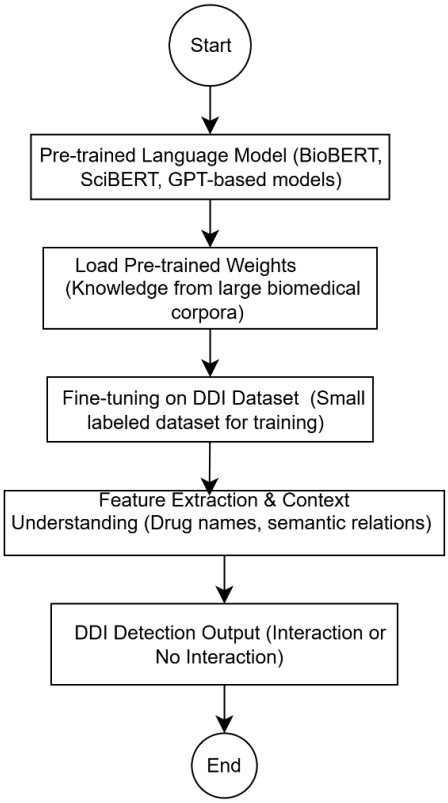


Figure 2-9: Workflow of Transfer Learning Using Pre-Trained Language Models for DDI Detection

2.5.3 Proposed Solution for Drug Allergy Detection

To overcome the limitations of using only fuzzy matching, a hybrid approach combining exact and fuzzy matching is proposed. Exact matching is first applied to quickly and accurately identify correctly spelled drug names. If no match is found, fuzzy matching is used to handle misspellings, abbreviations, and trade name variations. This layered strategy improves efficiency while reducing false positives and maintaining flexibility for real-world inputs.

In addition, a normalized Levenshtein distance is adopted to improve similarity scoring by addressing limitations of traditional edit distance methods, such as bias caused by string length differences [17]. This enhances the accuracy and robustness of similarity matching.

Overall, the hybrid system achieves a better balance between accuracy, efficiency, and usability, making it more suitable for real-world drug allergy detection in mobile healthcare systems.

2.5.4 Proposed Solution for Intelligent Medication Scheduling Systems

To improve adaptability, machine learning and real-time data-driven methods can be introduced to replace rigid rule-based scheduling. By using clinical data, patient behaviour, and historical prescriptions, the system can dynamically adjust medication schedules.

In addition, integration with updated drug databases and electronic health records can improve data accuracy and system interoperability. Large-scale clinical testing and user studies are also needed to validate performance in different real-world scenarios.

2.5.5 Proposed Solution of Knowledge-Based Decision Support Systems

To reduce reliance on manually defined rules, machine learning techniques can be integrated to learn from large-scale clinical data and improve decision accuracy. By incorporating patient-specific factors such as age, renal function, and medical history, the system can provide more personalized recommendations.

A hybrid approach combining rule-based and data-driven methods can further improve system flexibility and robustness. Expanding datasets, involving multiple experts, and integrating real-time clinical data can enhance reliability and support real-world deployment in healthcare environments.

Chapter 3

System Methodology/Approach

3.1 System Design Diagram

3.1.1 System Architecture Diagram

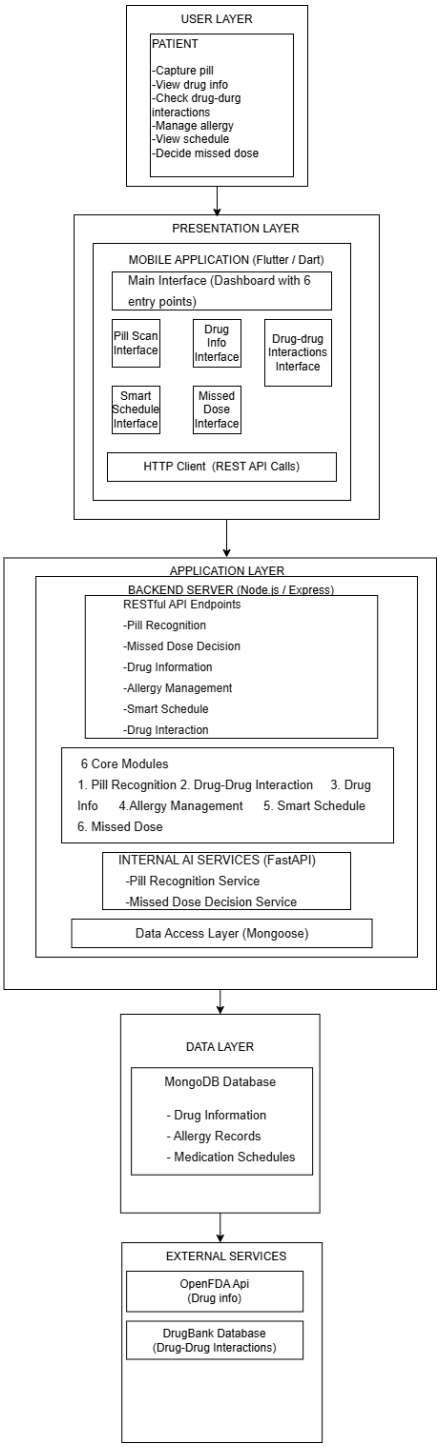


Figure 3-1: System Architecture of the Context-Aware Mobile Platform for Drug Interaction Alerts and Patient-Centered Insights

Figure 3-1 illustrates a layered system architecture consisting of the User Layer, Presentation Layer, Application Layer, Data Layer, and External Services Layer.

At the highest level, the User Layer (Patient) represents end users who interact with the system through a mobile application. Users can perform key operations such as capturing pill images, viewing drug information, checking drug–drug interactions, managing allergy records, viewing medication schedules, and making missed dose decisions. This layer defines the real-world interaction requirements of the system.

The Presentation Layer is implemented using a Flutter-based mobile application, which serves as the system interface. It provides a centralized dashboard with multiple entry points, including pill scanning, drug information access, drug interaction checking, smart scheduling, and missed dose decision support. Communication with the backend is handled through RESTful API calls via an HTTP client.

The Application Layer is implemented using a Node.js (Express) backend, which manages system logic and provides RESTful API endpoints for all core functionalities, including pill recognition, drug information retrieval, drug interaction checking, allergy management, smart scheduling, and missed dose decision support. This layer also integrates six core modules that encapsulate the main system functions. In addition, internal AI services developed using FastAPI handle computationally intensive tasks such as pill recognition and missed dose decision prediction. A data access layer (Mongoose) is used to interface with the database.

The Data Layer consists of a MongoDB database that stores drug information, allergy records, and medication schedules in a structured format to support system operations.

The External Services Layer integrates third-party data sources, including OpenFDA for drug information and DrugBank for drug–drug interaction data, ensuring that the system uses accurate and up-to-date pharmaceutical information.

This layered architecture improves modularity, scalability, and maintainability while enabling efficient integration of AI-driven and context-aware healthcare functionalities.

3.1.2 Use Case Diagram

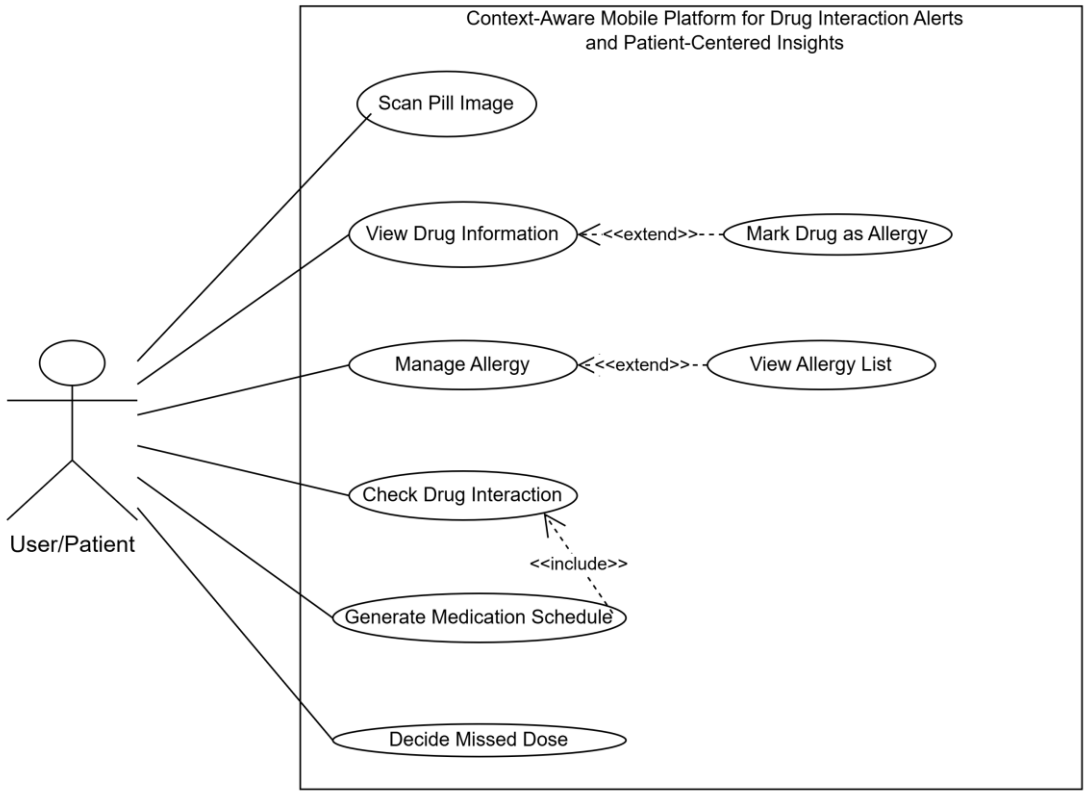


Figure 3-2: System Use Case Diagram for the Context-Aware Mobile Platform

Figure 3-2 presents the use case diagram for the Context-Aware Mobile Platform for Drug Interaction Alerts and Patient-Centered Insights, showing the interaction between the user/patient and the system's main functional modules.

The Scan Pill Image use case allows users to capture medication images using a mobile camera for quick drug identification, improving efficiency and reducing medication errors.

The View Drug Information use case provides detailed and updated drug information, including active ingredients, manufacturer details, storage conditions, and drug–drug interactions. Users can also mark drugs as allergens within this module.

The Manage Allergy use case allows users to view and manage their allergy records, supporting better awareness and safer medication use.

The Check Drug Interaction use case enables users to check potential interactions between medications and provides an independent safety-checking function used across the system.

The Generate Medication Schedule use case creates an optimized medication timetable based on drug half-life, dosage intervals, and interaction data, ensuring a safe and context-aware schedule.

The Missed Dose Decision Support use case provides guidance when a dose is missed based on time delay, patient age, and physiological conditions, and can be accessed independently from the scheduling module.

Overall, the system design is user-centered and context-aware, with modular and loosely coupled functions that operate independently while collectively supporting safe and personalized medication management.

3.1.3 Activity Diagram

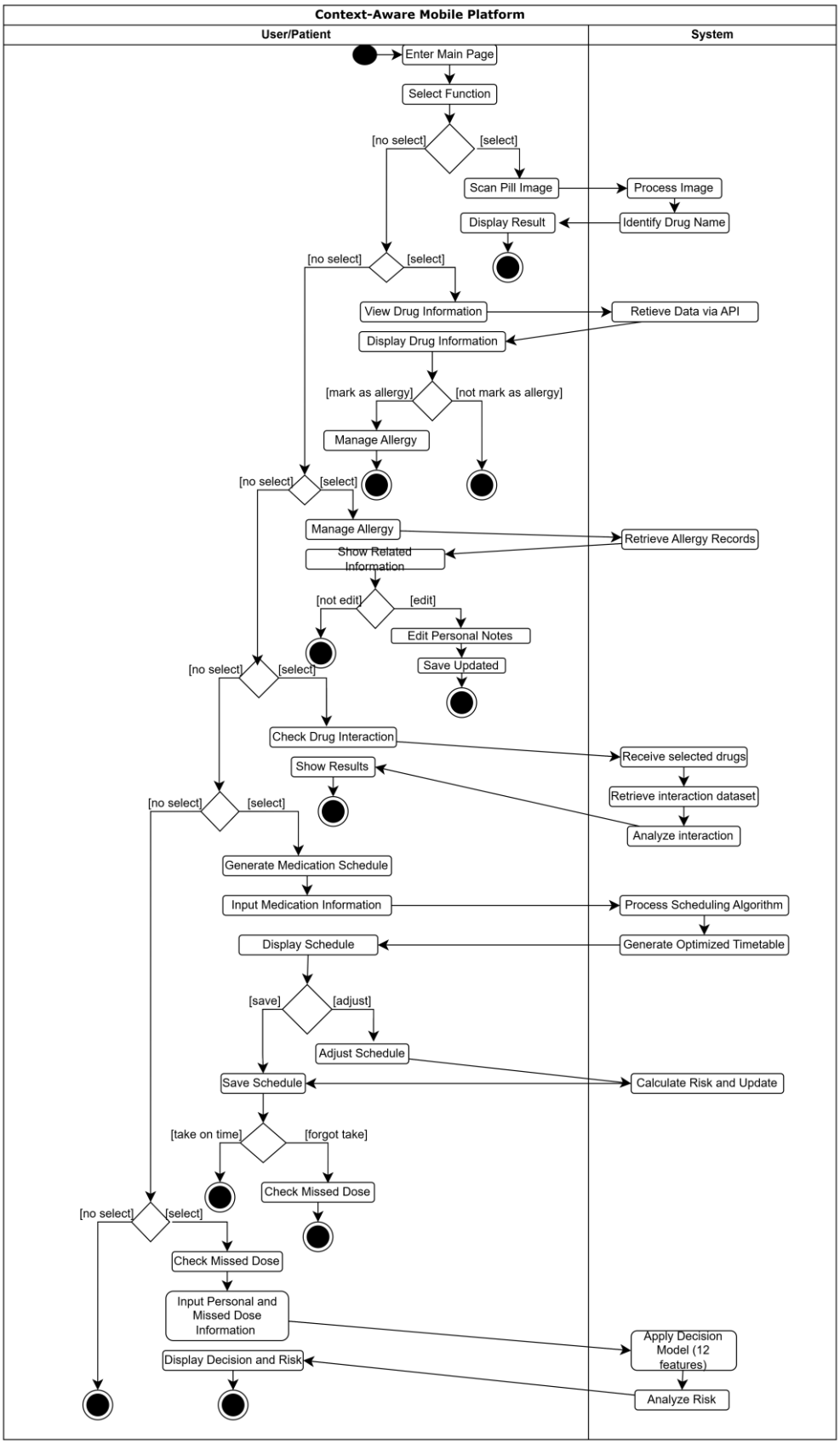


Figure 3-3: Activity Diagram for Overall System Workflow

Figure 3-3 presents the overall system workflow of the Context-Aware Mobile Platform for Drug Interaction Alerts and Patient-Centered Insights. The activity diagram starts when the user accesses the main page, where six main functions are available, and the workflow branches based on user selection.

When Scan Pill Image is selected, the system captures and processes the image, performs pill recognition, identifies the drug name, and displays the result.

For View Drug Information, the system retrieves data from the OpenFDA API and displays detailed drug information, including active ingredients, manufacturer details, storage instructions, and drug–drug interactions. Users can also mark drugs as allergens, which updates allergy records.

The Manage Allergy function allows users to view, edit, and manage their allergy records, including notes and saved updates.

In Check Drug Interaction, the system analyses selected drugs and returns results classified into high, medium, low, or no interaction levels.

The Generate Medication Schedule function collects user inputs and produces an optimized schedule based on drug half-life and dosage intervals. Users can modify the schedule, and changes are saved by the system.

If a dose is missed from the schedule, the workflow can proceed to Decide Missed Dose. This function takes user inputs such as age, renal function, and missed dose details, then generates a recommendation on whether to take or skip the dose, along with a risk level.

Overall, the activity diagram shows a user-driven workflow where all modules are loosely coupled but integrated into a unified medication safety system, allowing flexible navigation while maintaining consistent decision support.

Chapter 4

System Design

4.1 System Block Diagram

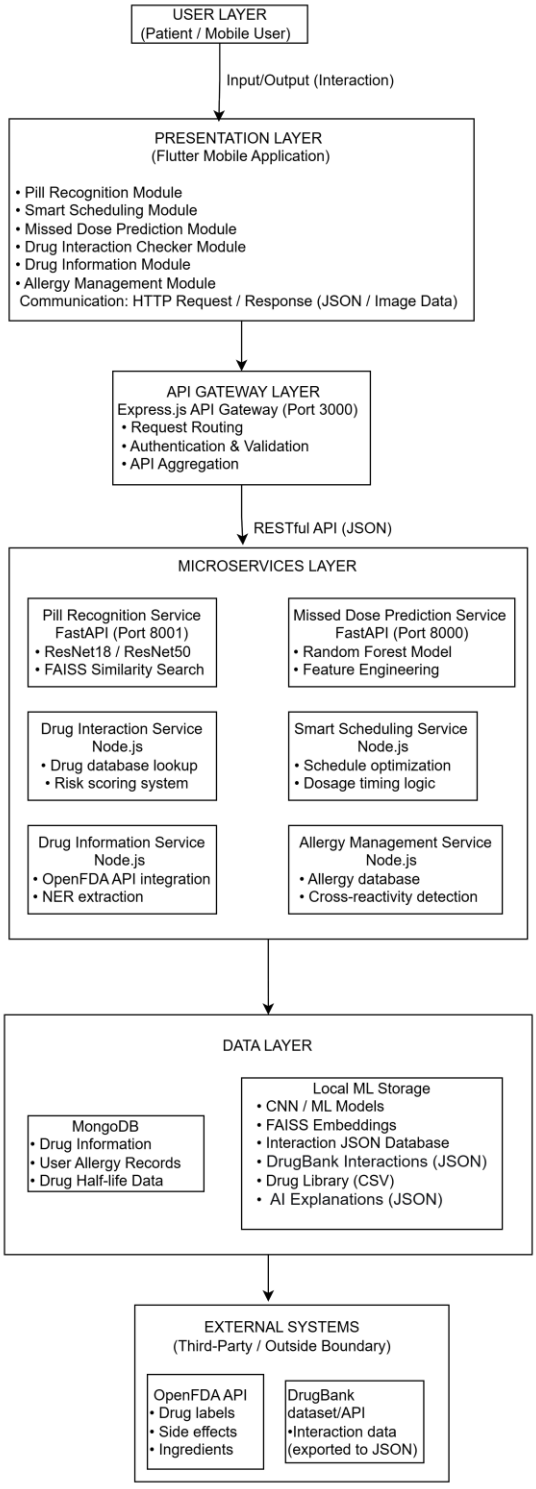


Figure 4-1 System Block Diagram of the Context-Aware Mobile Platform

Figure 4-1 illustrates the overall architecture of the proposed Context-Aware Mobile Platform, which uses a layered and microservices-based design to ensure scalability, modularity, and independent service deployment. The system consists of five layers: User Layer, Presentation Layer, API Gateway Layer, Microservices Layer, Data Layer, and External Systems Layer.

The User Layer represents patients who access the system via a mobile application. The Presentation Layer is developed using Flutter and provides interfaces for pill recognition, smart scheduling, missed dose prediction, drug interaction checking, drug information retrieval, and allergy management. Communication with backend services is handled through RESTful APIs using JSON and image data formats.

The API Gateway Layer, implemented using Express.js on port 3000, acts as a single entry point for all requests, handling routing, validation, authentication, and request forwarding.

The Microservices Layer consists of FastAPI and Node.js services. FastAPI handles AI-intensive tasks such as pill recognition and missed dose prediction, while Node.js manages application logic including interaction checking, scheduling, drug information retrieval, and allergy management. Services communicate via RESTful APIs to maintain loose coupling.

The Data Layer includes MongoDB and local storage. MongoDB stores structured medical data such as drug information, allergy records, and pharmacokinetic data, while local storage stores ML assets such as CNN models, FAISS indexes, interaction datasets, and drug libraries, improving performance and data management efficiency.

The External Systems Layer integrates APIs such as OpenFDA and DrugBank, providing authoritative drug data including labels, ingredients, side effects, and interactions, which improves system accuracy and reliability.

Overall, the architecture forms a scalable and intelligent medication safety platform by combining mobile computing, microservices, machine learning, and external drug data with clear separation of responsibilities across layers.

4.2 Pill Image Recognition Module

The Pill Image Recognition Module enables patients to identify medications using captured pill images, addressing variations in drug appearance such as shape, color, and imprint across different manufacturers. A hybrid AI approach is adopted, combining deep learning classification and image retrieval to improve recognition accuracy and robustness.

4.2.1 Overall System Architecture of the Pill Image Recognition Module

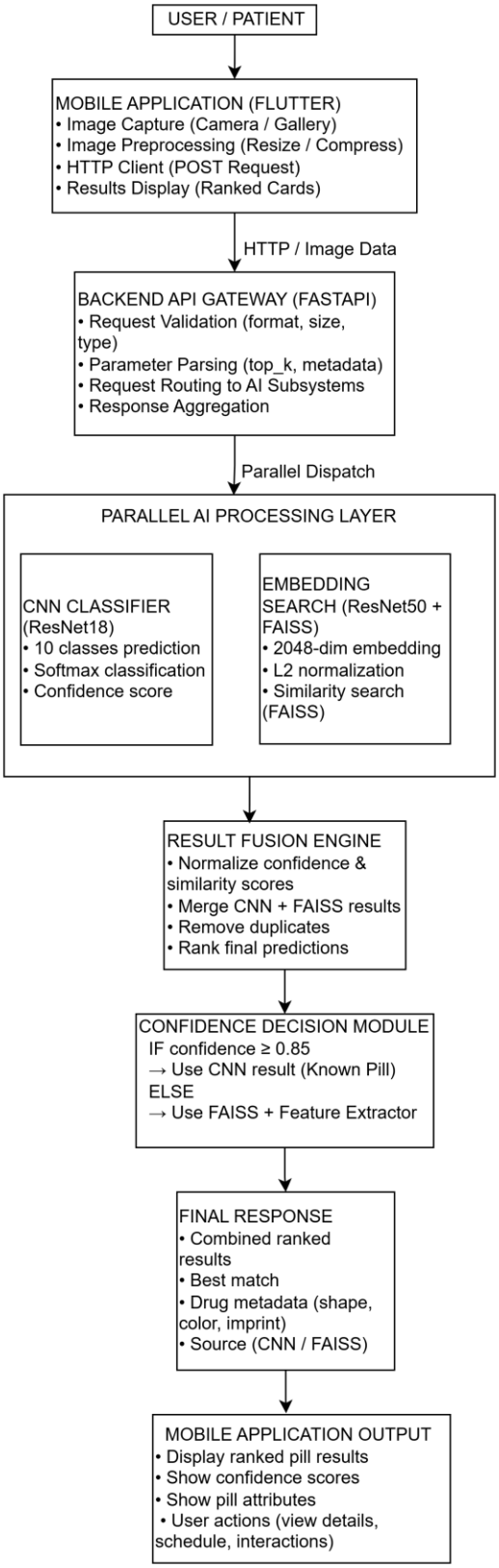


Figure 4-2: System Block Diagram of the Pill Image Recognition Module

Figure 4-2 presents the architecture of the Pill Image Recognition Module, which is designed as a hybrid AI system combining classification and retrieval-based identification.

The process begins when a user captures or uploads a pill image via the Flutter mobile application. The image is preprocessed through resizing and formatting before being transmitted to the backend via a RESTful API.

At the backend, FastAPI acts as the gateway, validating incoming requests and routing data to two parallel AI pipelines, namely a CNN-based classifier (ResNet18) for closed-set pill classification and a feature embedding retrieval system (ResNet50 combined with FAISS) for open-set similarity search.

This dual approach allows the system to efficiently classify known pills while also retrieving visually similar matches for unknown or ambiguous cases.

The outputs from both pipelines are forwarded to a fusion module, where confidence and similarity scores are normalized and ranked. A decision threshold of 0.85 is then applied, where results from the CNN model are returned if confidence is above the threshold, otherwise FAISS retrieval results are used.

The final output is returned to the mobile application, including pill name, confidence score, and key attributes such as shape, color, and imprint.

Overall, the architecture achieves a balance between fast classification and robust retrieval-based identification.

4.2.2 CNN-Based Pill Classification System

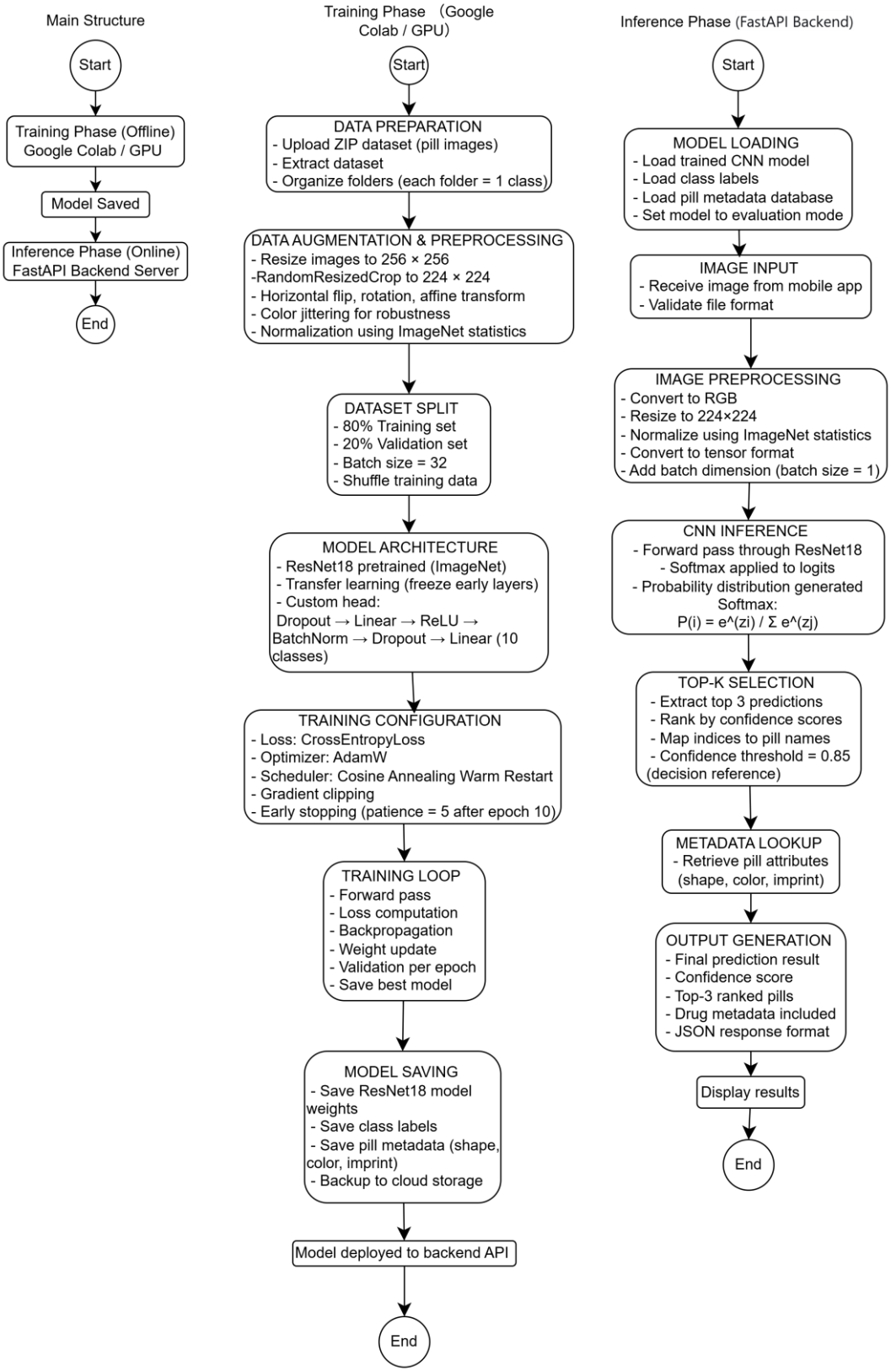


Figure 4-3：CNN-Based Pill Classification System Flowchart

Figure 4-3 illustrates the flowchart of the CNN-based pill classification system, which is one of the two parallel AI processing paths in the Pill Image Recognition Module. The system follows a two-phase architecture consisting of an offline training phase executed on Google Colab with GPU acceleration and an online inference phase deployed on a FastAPI backend server.

The training phase begins with data preparation, where a ZIP file containing pill images organized by class folders is uploaded from the local machine to the Google Colab environment. The system extracts the dataset and verifies its structure to ensure that each folder corresponds to a unique pill class. The dataset consists of ten pill categories, namely Alaxan, Bactidol, Bioflu, Biogesic, DayZinc, Decolgen, Fish Oil, Kremil S, Medicol, and Neozep, with each class containing 1,000 images, resulting in a total of 10,000 images. Sample images are visually inspected to ensure dataset quality and consistency before training begins.

To improve model generalization and reduce overfitting, a data augmentation and preprocessing pipeline is applied. The images are first resized to 256 × 256 pixels, followed by random resized cropping to 224 × 224 pixels with a scaling range between 0.8 and 1.0. Additional augmentations include random horizontal flipping with a probability of 0.5, random rotation within ±15 degrees, color jittering involving variations in brightness, contrast, saturation, and hue, and random affine transformations with up to 10% translation. After augmentation, the images are converted into tensor format with pixel values normalized to the range [0,1], and further normalized using ImageNet statistics with a mean of [0.485, 0.456, 0.406] and a standard deviation of [0.229, 0.224, 0.225] [19].

The dataset is then divided into training and validation sets with an 80:20 ratio. A batch size of 32 is used, and shuffling is enabled during training to improve model robustness. The model architecture is based on ResNet18 pretrained on ImageNet, where early convolutional layers are frozen to preserve general feature extraction capabilities, while deeper layers are fine-tuned for domain-specific learning. The original classification head is replaced with a custom fully connected architecture consisting of dropout layers, linear transformations, ReLU activation, and batch normalization, with the final layer outputting ten classes corresponding to the pill categories.

The training process employs the Cross-Entropy Loss function in conjunction with the AdamW optimizer, configured with a learning rate of 0.0005 and a weight decay of 0.01. The cross-entropy loss measures the discrepancy between the predicted probability distribution and the ground truth label, as defined in Equation (1):

$$L = -\sum_{i=1}^{10} y_i \ log(\hat{y}_i) \quad (1)\ [18]$$

where y_i denotes the ground truth label for class i (with value 1 for the correct class and 0 otherwise), $\hat{y}_i$ represents the predicted probability for class i, and log denotes the natural logarithm. The objective of the training process is to minimize this loss function, where a value of $L = 0$ indicates a perfect prediction.

To further enhance training efficiency, a CosineAnnealingWarmRestarts scheduler is applied with T_0 set to 10 and T_mult set to 2. Gradient clipping with a maximum norm of 1.0 is used to stabilize training, and early stopping with a patience of five epochs is applied after the tenth epoch to prevent overfitting. The model is trained for a maximum of 20 epochs, where each epoch consists of forward propagation, loss computation, backpropagation, parameter updates, and validation evaluation. The model with the highest validation accuracy is saved as the final trained model, along with class labels, input specifications, normalization parameters, and architecture metadata for deployment.

During the inference phase, the backend server loads the trained CNN model, class labels, and pill metadata into memory and sets the model to evaluation mode to disable dropout and batch normalization updates. When the mobile application captures or selects a pill image, the image is transmitted to the backend via an HTTP POST request, where the file format is validated to ensure compatibility.

The input image is then preprocessed by converting it to RGB format to ensure a three-channel input, resizing it to 224 × 224 pixels, converting it into tensor format, and scaling pixel values to the range [0,1]. It is then normalized using ImageNet statistics, and a batch dimension is added to prepare it for model inference.

The normalization process is defined in Equation (2), where the input tensor is standardized using the mean and standard deviation of ImageNet:

$$I_{norm} = \frac{I_{tensor} - \mu}{\sigma} \quad (2)\ [19]$$

where I_{norm} represents the normalized image tensor, I_{tensor} represents the input tensor, μ denotes the mean vector [0.485, 0.456, 0.406], and σ denotes the standard deviation vector [0.229, 0.224, 0.225] [19].

During inference, gradient computation is disabled to improve computational efficiency and reduce memory usage. The preprocessed tensor is then passed through the ResNet18 model

to generate logits, which are subsequently converted into probability distributions using the softmax function as defined in Equation (3):

$$P(y = i|x) = \frac{e^{z_i}}{\sum_{j=1}^{10} e^{z_j}} \quad (3)\ [20]$$

where $P(y = i \mid x)$represents the probability of class i, $z_i$represents the logit value for class i, and $j$denotes the class index.

The confidence score is defined as the highest probability value among all predicted classes, as shown in Equation (4):

$$\text{Confidence} = \max_{i \in \{1,\ldots,10\}} P(y = i \mid x) \quad (4)$$

The system then uses the torch.topk() function to extract the top three predictions with the highest confidence scores. Each predicted class index is mapped to its corresponding pill name using the predefined class label list. A confidence threshold of 0.85 is applied to determine the final decision pathway in the downstream result fusion module, as defined in Equation (5):

$$Output = \begin{cases} CNN\ Result\ (Known\ Pill),\ if\ Confidence \geq 0.85 \\ FAISS\ +\ Feature\ Extractor,\ if\ Confidence\ < 0.85 \end{cases} \quad (5)$$

4.2.3 ResNet50 Feature Embedding and FAISS Similarity Retrieval System

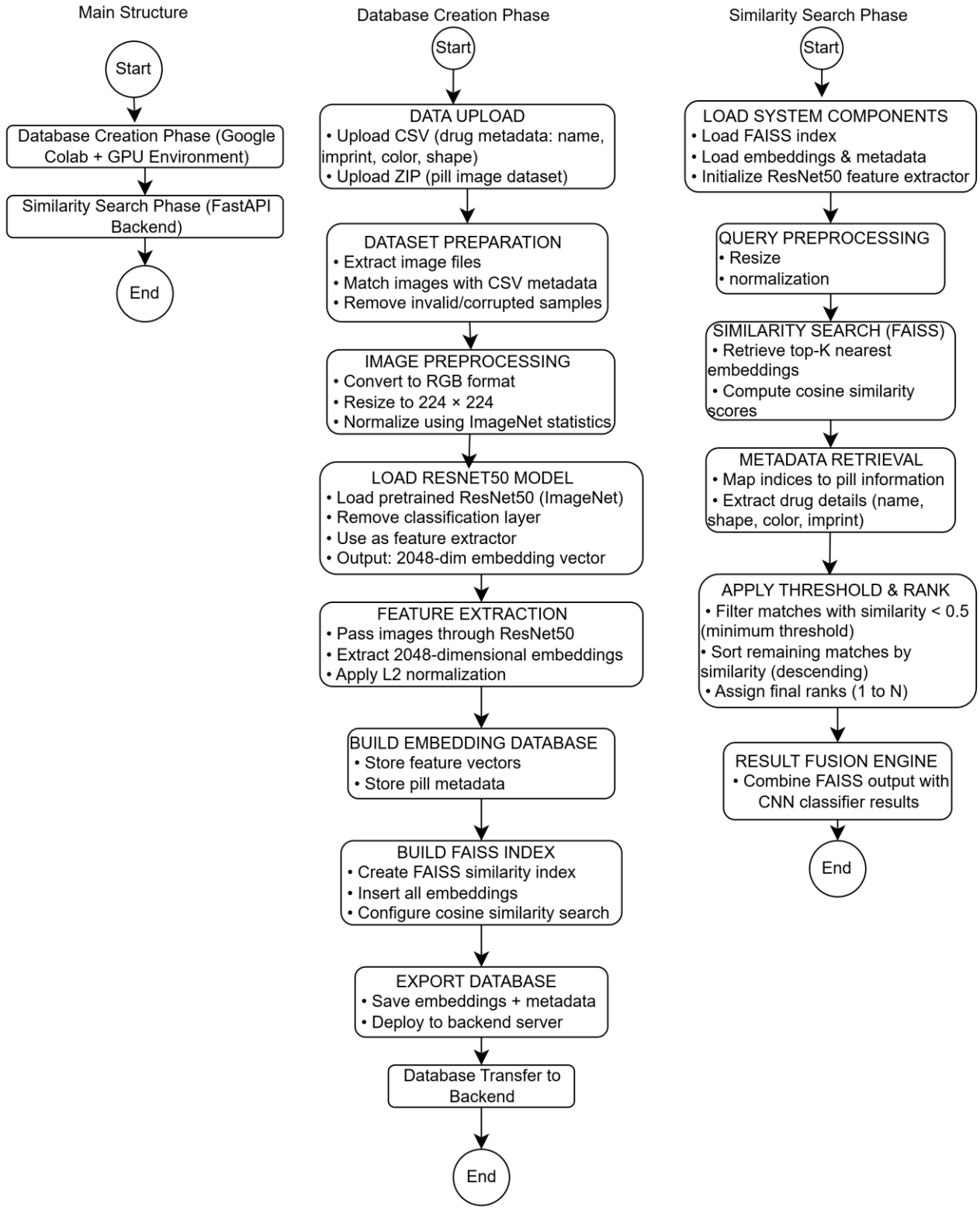


Figure 4-4: ResNet50 Feature Embedding and FAISS Similarity Retrieval Flowchart

Figure 4-4 illustrates the ResNet50 feature embedding and FAISS similarity retrieval system, which is the second parallel path in the pill image recognition module. Unlike the CNN classifier in Figure 4-3 that performs closed-set classification into ten classes, this approach enables open-set retrieval through similarity search against a precomputed embedding database. The system consists of an offline database creation phase (Google Colab with GPU) and an online retrieval phase (FastAPI server).

The database creation phase starts with uploading a CSV file containing pill metadata (drug name, imprint, colour, shape, strength, and NDC code) and a ZIP file of pill images. The dataset is extracted, validated, and mapped to corresponding metadata, with invalid samples removed.

Images are preprocessed by resizing to 224 × 224, converting to RGB, and normalizing using ImageNet statistics (μ = [0.485, 0.456, 0.406], σ = [0.229, 0.224, 0.225]) to ensure consistency with the CNN classifier [19].

A pretrained ResNet50 model is used as a feature extractor by removing its final classification layer, producing a 2048-dimensional embedding vector $e \in \mathbb{R}^{2048}$ for each image.

The raw embedding is then L2-normalized to ensure unit vector representation. This operation is defined in Equation (6):

$$\hat{e} = \frac{e}{||e||2} = \frac{e}{\sqrt{\sum_{k=1}^{2048} e_k^2}} \quad (6)\ [21]$$

where e is the raw embedding vector, ê is the normalized embedding vector, and $e_k$represents the k-th element of the embedding.

All normalized embeddings are stored together with their corresponding metadata. A FAISS index is then constructed using IndexFlatIP, where inner product is used as the similarity metric. Since all vectors are L2-normalized, the inner product is equivalent to cosine similarity.

The similarity between a query embedding and a database embedding is defined in Equation (7):

$$sim(q, d) = \hat{e}_q \cdot \hat{e}_d = \sum_{k=1}^{2048} \hat{e}_{q,k} \cdot \hat{e}_{d,k} \quad (7)\ [22]$$

where sim(q, d) represents the similarity score between query image *q* and database image *d*, ê_q is the normalized query embedding, and ê_d is the normalized database embedding. The similarity score ranges from 0 to 1, where higher values indicate stronger visual similarity.

During the similarity search phase, the query embedding is reshaped into a 1 × 2048 vector and assed to the FAISS index to retrieve the top-K nearest neighbors based on similarity scores.

The retrieved indices are mapped to pill metadata, including drug name, shape, color, imprint, and strength.

A minimum similarity threshold of 0.5 is applied to filter low-confidence matches. The remaining results are ranked in descending order of similarity and returned as a JSON response containing scores and metadata.

Finally, the FAISS retrieval outputs are sent to the result fusion engine, where they are combined with CNN classifier results through score normalization, duplicate removal, and final ranking to produce a unified prediction.

4.3 Drug-Drug Interaction Module

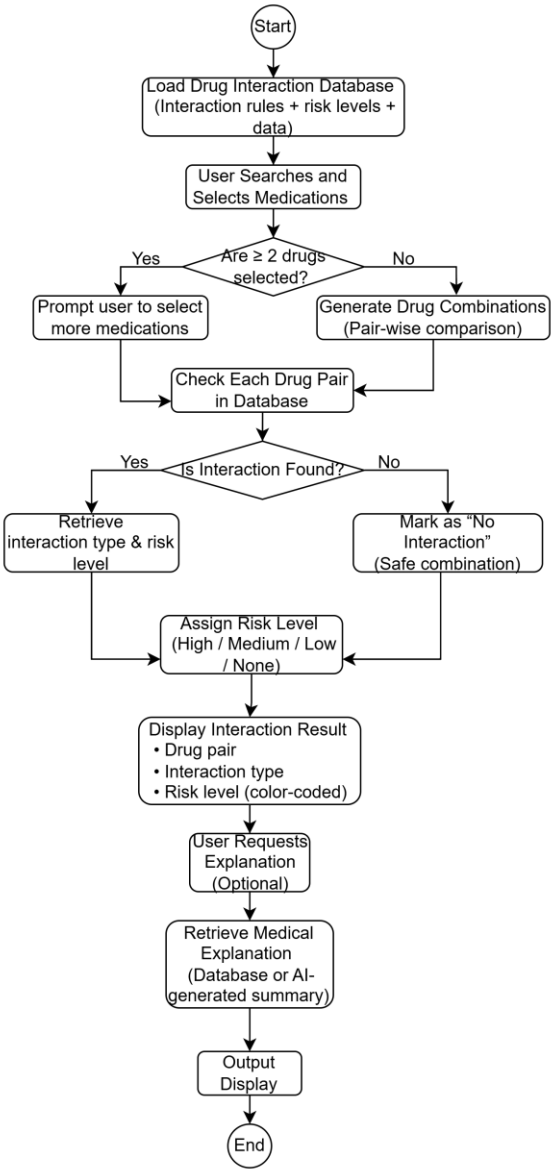


Figure 4-5: System Flowchart of the Drug-Drug Interaction Module

The Drug-Drug Interaction Module helps patients avoid harmful medication combinations that may lead to hospital admissions. Since patients often take multiple drugs from different prescribers, the risk of undetected interactions is high. This module checks drug pairs against an interaction database, classifies risk severity, and provides colour-coded warnings to support safer decision-making.

Figure 4-5 presents the flowchart of the drug–drug interaction checking module, which follows a structured workflow including data preparation, user input processing, interaction analysis, risk classification, and optional AI-based explanation generation.

The process begins with loading a pre-processed drug interaction database, converted from a DDI_data.csv file (DrugBank) into a structured JSON format (e.g., drug_interactions.json). The interaction type information in the dataset covers a total of 222697 documented drug-drug interactions, providing a comprehensive basis for accurate risk assessment and classification. The database stores drug pair relationships, interaction types, and risk levels. Risk classification is determined using keyword-based rules, where medical terms are mapped into High, Medium, or Low severity levels based on clinical significance.

Users then select medications for analysis. A validation step ensures at least two drugs are selected; otherwise, the system prompts the user to add more drugs.

Once valid input is provided, the system generates all unique drug pairs. Each pair is converted into a standardized key format (e.g., DB00006_DB06605) to ensure consistent lookup in the database.

Each drug pair is then checked against the interaction database. If an interaction exists, the system retrieves the interaction type (e.g., serum concentration change, bleeding risk) and its risk level. If no interaction is found, the pair is classified as safe with no known interaction.

Results are displayed in a structured, color-coded format: red (high risk), orange (medium risk), yellow (low risk), and green (no interaction), improving clarity for users.

An optional AI-based explanation feature is provided. The system first retrieves predefined explanations generated using ChatGPT from a database. If no explanation is available, the Gemini API is invoked as a fallback to generate the explanation dynamically. A pattern-based keyword reasoning approach is also applied to present the interaction in simplified terms.

The final output includes drug pair information, interaction type, risk level, and explanation, helping users make informed and safer medication decisions while reducing the risk of harmful drug combinations.

The complete drug–drug interaction dataset used in this module is available for reference at the following link: https://drive.google.com/file/d/1PXhRCYNXlie2eJ1ocbGweQesi1zYPQx5/view?usp=drive_link

4.4 Drug Information Module

The Drug Information Module provides patients with reliable and up-to-date medication information through a centralized platform. Patients often face difficulties obtaining trustworthy drug details due to fragmented sources or outdated leaflets. This module enables users to search for medications and retrieve comprehensive information, including usage, dosage, side effects, and warnings, supporting informed treatment decisions.

4.4.1 Drug Information Retrieval & Display

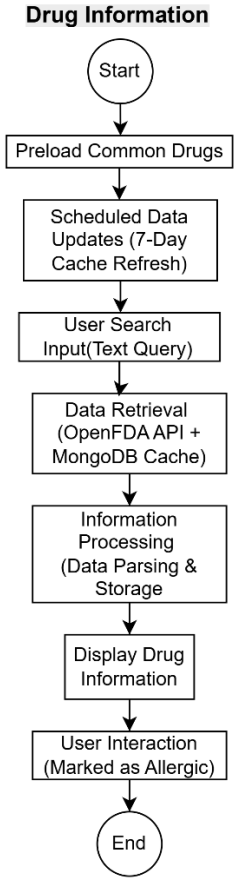


Figure 4-6 Flowchart of Drug Information Retrieval & Display

Based on Figure 4-6, the drug information module follows a clear and sequential workflow. The system first initializes by loading a set of common pharmaceutical compounds to form the initial reference database. Weekly scheduled updates are then performed to ensure that the information remains current. When the user begins a search by entering a text query, the system triggers the data retrieval process. Information is obtained from two sources: the OpenFDA API and the local MongoDB database. The retrieved data is then processed, parsed, and organized into a structured format. Once processed, the information is presented to the user through the medication information interface. The workflow concludes with an interactive feature that allows users to mark specific drugs as allergens, which will be stored for future reference.

4.4.2 Active Ingredient Extraction and Allergy Matching

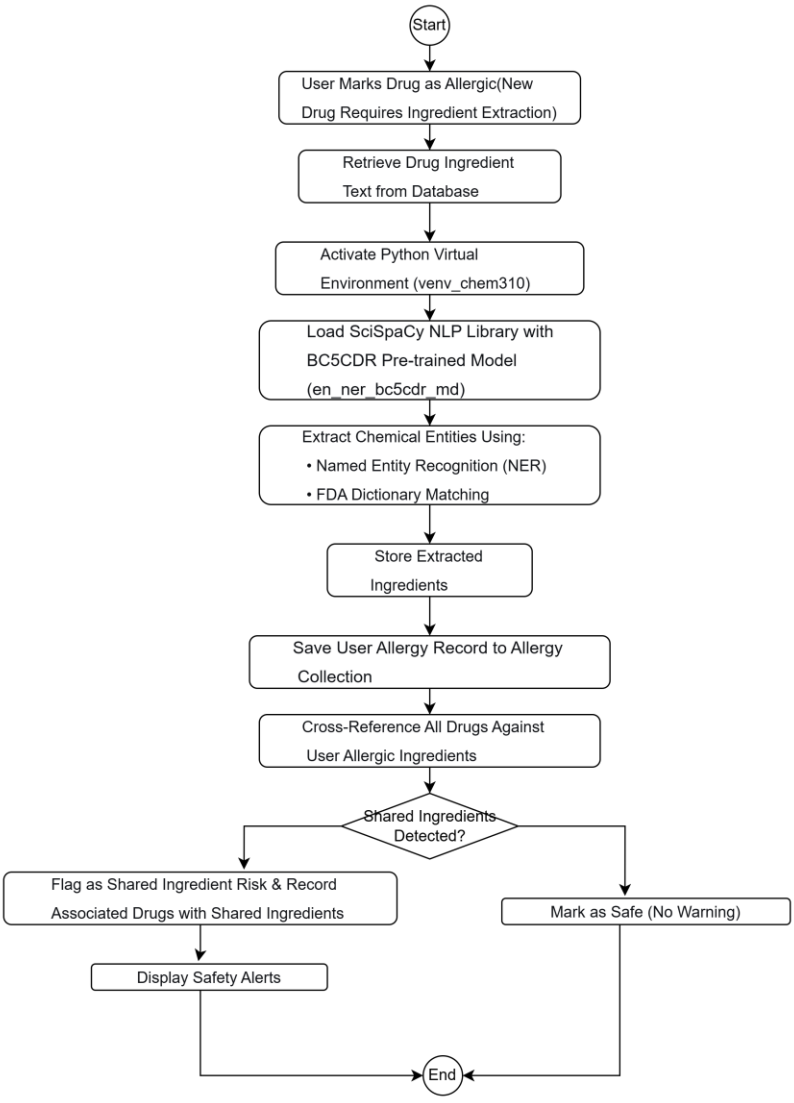


Figure 4-7: Flowchart of Active Ingredient Extraction and Allergy Matching

Figure 4-7 illustrates the workflow of active ingredient extraction and allergy matching within the drug information module. The process starts when a user marks a drug as allergenic or when a new drug requires ingredient extraction.

The system first retrieves ingredient text from the database and activates a Python virtual environment (venv_chem310) to isolate biomedical NLP dependencies. It then loads SciSpaCy with the BC5CDR pre-trained model (en_ner_bc5cdr_md) for chemical and drug entity recognition.

A dual-method extraction approach is used. Named Entity Recognition (NER) identifies chemical entities from the BC5CDR model, while FDA dictionary matching cross-checks ingredients with OpenFDA-based drug name dictionaries. Both outputs are merged to form the final ingredient list.

When a drug is marked as allergic, the system stores the record in the Allergy Collection, including drug name and extracted active ingredients. It then compares all stored drugs against the user's allergy ingredient list.

A decision step checks whether shared ingredients exist. If matches are found, the drug is flagged as an allergy risk and the system records the associated drugs. If no match is found, the drug is marked as safe. The system then generates safety alerts such as color-coded warnings, ingredient-based dialogs, and allergy banners in the interface.

4.5 Allergy Management Module

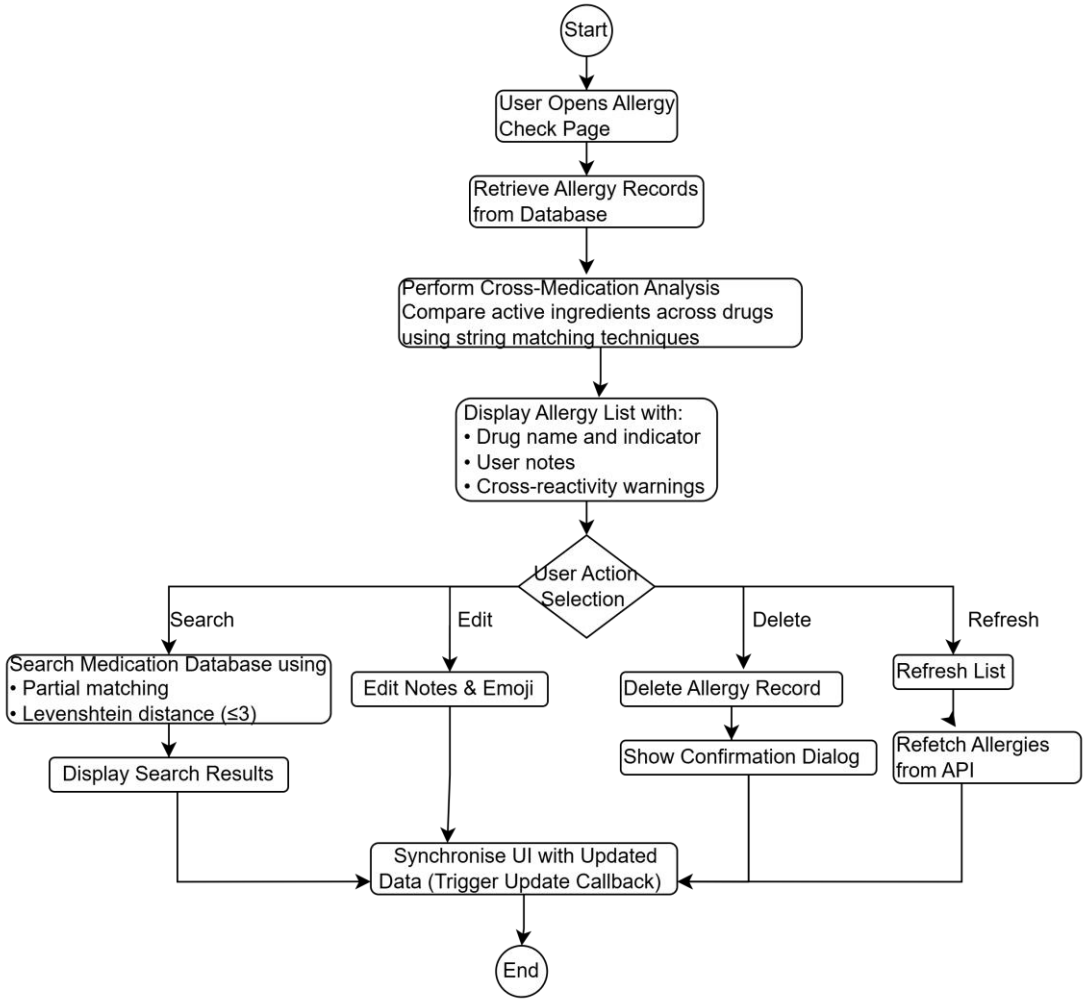


Figure 4-8: System Flowchart of Allergy Management with Cross-Medication Analysis

The Allergy Management Module enables users to record and monitor medication allergies in a personal list. Many patients may forget past adverse drug reactions, increasing the risk of repeated exposure. This module allows users to mark allergic drugs, add notes and emojis, and receive automatic alerts when newly selected medications contain known allergenic ingredients, thereby improving medication safety.

Figure 4-8 illustrates the workflow of the Allergy Management Module, which supports allergy tracking and cross-reactivity detection.

The process begins when the user accesses the Allergy Check Page. Allergy records are retrieved from the database via a GET request to the /api/allergies endpoint. After retrieval, the backend performs cross-medication analysis by comparing the separateIngredients field across all drugs using string matching to identify shared active ingredients that may indicate cross-reactivity.

The system then displays the allergy list, including drug names with emoji indicators, user-defined notes, and cross-reactivity warnings when applicable. Users can perform four main actions within the interface.

First, users can search the medication database using partial text matching combined with the Levenshtein distance algorithm (threshold = 3) to support fuzzy matching and handle spelling variations. Second, users can edit notes and emojis for personalization. Third, users can delete allergy records, with a confirmation step before permanent removal. Fourth, users can refresh the allergy list by re-fetching updated data from the API.

After any modification or refresh, the system synchronizes the interface with updated data and triggers the onAllergyUpdated event handler. This ensures real-time consistency between the Allergy Management Module and the Drug Information Module. The workflow ends once the interface is updated successfully.

4.6 Smart Scheduling Module

The Drug-Drug Interaction Module alerts users about harmful combinations, but patients may still need to take interacting drugs due to medical necessity. The Smart Scheduling Module addresses this by generating optimal daily medication timetables that separate interacting drugs with clinically appropriate time gaps based on drug half-lives and meal times, enabling patients to take all prescribed medications safely.

4.6.1 Drug Schedule Generation and Optimization

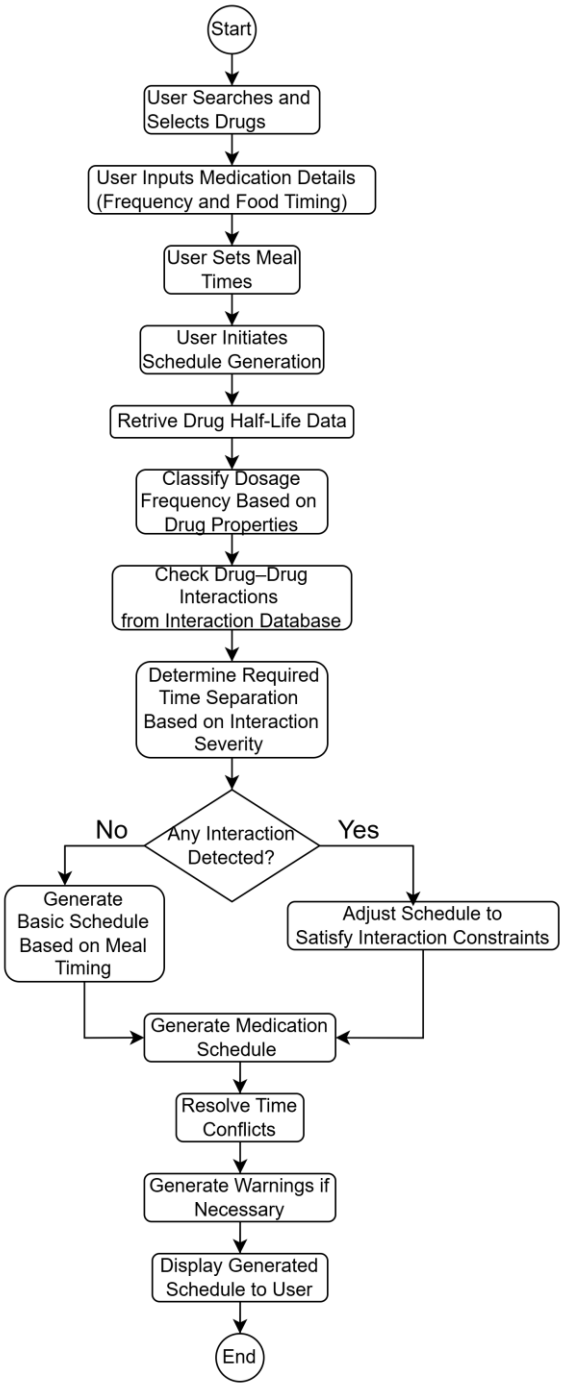


Figure 4-9: Flowchart of Drug Schedule Generation and Optimization

Figure 4-9 illustrates the workflow of the drug schedule generation and optimization module, which generates a daily medication schedule based on user input and drug properties. The process begins when the user selects medications, specifies daily frequency (one to four times per day), and sets food timing preferences (before, after, with, or anytime meals). The

user also inputs breakfast, lunch, and dinner times using a TimePicker widget before schedule generation.

The system retrieves drug half-life data from the drughalflife MongoDB collection using a hierarchical search approach: exact match by drug name, followed by partial match if needed. If no data is available, a default half-life of 12 hours is used as a conservative mid-range value to ensure balanced scheduling.

The half-life value determines the classification and suggested dosing frequency based on predefined mathematical thresholds. Drugs with half-life less than six hours are classified as short, suggesting four times daily dosing because shorter half-life drugs are eliminated from the body more quickly, requiring more frequent administration to maintain therapeutic effect. Drugs with half-life between six and twenty-four hours are classified as medium, suggesting two to three times daily, as these drugs remain in the body for a moderate duration. Drugs with half-life exceeding twenty-four hours are classified as long, suggesting once daily dosing, as these drugs persist in the body for an extended period and require less frequent administration. This classification is expressed in Equation (8):

$$Half - Life\ Classification = \begin{cases} 4\ times\ daily, & if\ t_{1/2} < 6\ hours \\ 2-3\ times\ daily, & if\ 6 \le t_{1/2} \le 24\ hours \\ 1\ time\ daily, & if\ t_{1/2} > 24\ hours \end{cases} \quad (8)$$

where $t_{1/2}$represents the drug half-life in hours. This equation is used to determine the half-life category of each drug and its corresponding suggested daily dosing frequency based on mathematical thresholds.

For every unique pair of selected drugs, the system checks for interactions by querying the drug_interactions.json database, which was originally built for the drug-drug interaction checking module. This database is reused here to detect potential interactions between medications, ensuring that the generated schedule accounts for clinically significant drug pairs. Each interaction record contains an interaction type, risk level (High, Medium, Low, or No), and mechanism description. For each detected interaction, a minimum required time gap between doses is calculated based on the interaction severity. High risk interactions require an eight-hour gap, medium risk requires six hours, low risk requires four hours, and minor risk requires two hours. These gap values are derived from general medication safety principles where higher risk interactions require longer separation periods to minimize concurrent drug exposure. The base gap assignment is shown in Equation (9):

$$Base\ Gap = \begin{cases} 8\ hours, & High\ risk \\ 6\ hours, & Medium\ risk \\ 4\ hours, & Low\ risk \\ 2\ hours, & Minor\ risk \end{cases} \quad (9)$$

If either interacting drug has a half-life exceeding twenty-four hours, the required gap is multiplied by 1.5. This multiplier is applied because drugs with long half-lives remain active in the bloodstream for extended periods, requiring proportionally longer separation to reduce concurrent drug concentration. The 1.5 factor represents a moderate increase that balances safety with scheduling feasibility. The half-life adjustment factor is expressed in Equation (10):

$$Half - Life\ Factor = \begin{cases} 1.5, & if\ \max\left({t_{1/2}}^1, {t_{1/2}}^2\right) > 24\ hours \\ 1.0, & otherwise \end{cases} \quad (10)$$

where ${t_{1/2}}^1$ is half-life of drug 1 and ${t_{1/2}}^2$ is half-life of drug 2. The final required gap is capped at a maximum of twelve hours to ensure the separation requirement remains achievable within a single daily waking period. The complete required gap calculation is shown in Equation (11):

$$Required\ Gap = \min\left(12, Base\ Gap \times Half - Life\ Factor\right) \quad (11)$$

The system then evaluates whether any interactions exist among the selected drugs. When no interactions are detected, a normal schedule is generated based solely on meal times and food timing preferences. When interactions exist, the system branches based on the number of interacting drugs. Prior to any scheduling for interacting drugs, each drug's dosing frequency is first adjusted according to Equation (12), which may reduce a user-selected frequency of four times daily to once daily for drugs with half-life exceeding twenty-four hours, as shown equation (12):

$$Effective\ Frequency = \begin{cases} 1, & if\ t_{1/2} > 24\ hours\ and\ userFreq > 1 \\ suggestedFreq, & if\ suggestedFreq < userFreq \\ userFreq, & otherwise \end{cases} \quad (12)$$

For two interacting drugs, the system calculates whether the required gap can fit within the daily waking hours window from 06:00 to 22:00, which totals sixteen hours. If the required gap is less than or equal to the waking hours, both drugs are scheduled with the necessary time separation, where the times of the second drug are shifted to satisfy the minimum gap requirement. All doses of the second drug are shifted by the same offset, preserving the original

number of daily doses while ensuring the minimum gap is achieved for the earliest dose of the first drug. If shifting a dose would place it outside the waking hours window, the dose time is clamped to the nearest boundary at 06:00 or 22:00, which may reduce the effective separation gap at the edges of the day. After shifting, the system verifies that the adjusted administration times remain within one hour of the user's specified meal times for meal-dependent drugs. If the adjusted times violate these meal timing constraints, the system reverts to scheduling only the first drug and marks the second as problematic. If the required gap exceeds the waking hours, only one drug is scheduled and the other is marked as problematic with a safety warning.

For more than two interacting drugs, the system applies a priority-based scheduling algorithm where drugs are sorted first by the number of interactions (highest first) and then by frequency (lowest first). This ordering prioritizes drugs with more constraints, as they are harder to place later, and drugs with fewer doses, as they are easier to adjust. This strategy increases the likelihood of successfully scheduling all medications within the daily window. The system then iteratively schedules each drug. For each drug, desired times are generated, and conflicts with already-scheduled drugs are identified. For each conflicting drug dose, a forbidden time range is calculated from the other dose time minus the required gap to the other dose time plus the required gap. The valid time range for scheduling is the waking hours window excluding all forbidden ranges. The gap satisfaction condition requires that for every dose pair between interacting drugs, the absolute time difference must be at least the required gap, as expressed in Equation (13):

$$| t_i - t_j | \geq G_{ij}, \quad \forall (i,j) \in D \quad (13)$$

where t_i, t_j are scheduled administration times of interacting drugs, G_{ij} is required minimum time gap and $\mathcal{D}$ is set of interacting drug pairs.

If no continuous valid range exists, the system searches for the closest valid time by checking every thirty-minute interval within waking hours. If still no valid time is found, the system attempts to use standard meal times of 08:00, 12:00, and 19:00 as fallback slots. These standard meal times are only considered as fallback slots if they fall within the 06:00 to 22:00 waking window. When all attempts fail, the drug is marked as problematic and cannot be safely scheduled.

The system then calculates the effective dosing frequency by considering both user preference and the suggested frequency derived from half-life thresholds. If a drug has half-life exceeding twenty-four hours and the user selected a frequency greater than once daily, the

frequency is forced to once daily to prevent drug accumulation. If the suggested frequency is lower than the user-selected frequency, the suggested frequency is used. Otherwise, the user-selected frequency is retained. This logic is formalized in Equation (12):

$$Effective\ Frequency = \begin{cases} 1, \ if\ t_{1/2} > 24\ hours\ and\ userFreq > 1 \\ suggestedFreq, \ if\ suggestedFreq < userFreq \\ userFreq, \ otherwise \end{cases} \quad (12)$$

where $t_{1/2}$ is drug half-life in hours, userFreq is frequency selected by the user (1 to 4 times daily), suggestedFreq is frequency suggested by half-life classification.

Specific administration times are generated based on food timing preference, meal times, and effective frequency. For drugs requiring administration with meals, the time is offset relative to the meal time: before meals subtracts 0.5 hours, after meals adds 0.5 hours, and with meals uses the exact meal time. For drugs with flexible anytime timing, the system assigns standardized time slots within the daily schedule: 10:00 for once daily, 09:00 and 21:00 for twice daily, or 08:00, 14:00, and 20:00 for three times daily. For meal-dependent drugs, doses are distributed across meals according to effective frequency: once daily uses breakfast only, twice daily uses breakfast and dinner, and three times daily uses breakfast, lunch, and dinner. Equation (14) summarizes the meal time offset logic:

$$Scheduled\ Time = \begin{cases} mealTime - 0.5\ hours, \ if\ foodTiming = before \\ mealTime + 0.5\ hours, \ if\ foodTiming = after \\ mealTime, \ if\ foodTiming = with \\ fixedSlot, \ if\ foodTiming = anytime \end{cases} \quad (14)$$

After generating initial times, the system performs conflict resolution to ensure all required gaps are satisfied. When gaps are violated, times are shifted within the waking hours window while maintaining meal time associations where possible. The system builds a schedule entry for each drug containing the drug name, scheduled times, food timing preference, notes including frequency adjustment warnings, and interaction alerts. If any drug cannot be safely scheduled despite all adjustment attempts, a warning entry is created displaying as "Schedule Warning" with detailed notes listing the problematic drugs, required gaps, and general recommendations including consulting a doctor, considering alternative timing, or taking drugs on alternating days.

The final daily schedule is displayed to the user showing the risk summary card with overall risk percentage, hourly risk timeline from 00:00 to 23:00, medication timetable with draggable

time pills for manual adjustment, drop zone containing all 24 hour cells, collapsible warnings section, and drug interactions section. When a user drags a time pill to a new hour cell, the system waits 500 milliseconds before recalculating interaction risks, preventing excessive computational load during rapid drag operations. When no interaction risk is detected (overall risk of 0%), the system returns an empty risk timeline and displays a "No Interaction Risk" message to the user.

It is important to note that this module is designed for informational and scheduling assistance purposes only. The generated schedule and risk calculations are based purely on mathematical formulas, half-life data from the drughalflife collection, and interaction data from the drug_interactions.json database. This module does not provide clinical or medical advice. Users are strongly encouraged to consult qualified healthcare professionals before making any changes to their medication regimen. The workflow then proceeds to risk calculation and daily schedule tracking as detailed in Figure 4-10 and Figure 4-11.

4.6.2 Priority-Based Scheduling Logic

Priority	Factor	Description
1	Drug Interaction Risk Level	Highest priority; determines required separation gap based on severity (High, Medium, Low, Minor) to ensure patient safety.
2	Drug Half-Life	Longer half-life increases required gap, reflecting prolonged drug activity in the body.
3	Waking Hours Constraint	Ensures all medications fit within the user's awake period; otherwise, some drugs may be unscheduled.
4	Required Separation Gap	Enforces minimum time intervals between interacting drugs to prevent overlap.
5	Time Adjustment Capability	Enables dynamic shifting of doses to satisfy scheduling constraints.
6	Meal-Time Constraints	Applies food-related timing rules only when they do not conflict with safety constraints.
7	User-Selected Frequency	Adjusts dosing frequency (1–4 times daily) when constraints cannot be fully satisfied.
8	Half-Life-Based Frequency Reduction	Forces once-daily dosing for drugs with half-life >24 hours, overriding user preference.

Table 4-1: Priority-Based Scheduling Logic for Smart Drug Scheduler

Table 4-1 presents the priority-based decision-making strategy used in the Smart Drug Scheduler to ensure safe and feasible medication planning. The algorithm prioritizes drug–drug interaction risk as the highest concern, followed by pharmacokinetic properties such as drug half-life, which define the required separation intervals between medications.

Subsequently, practical constraints including waking hours and feasible time gaps are evaluated to determine whether all medications can be accommodated within a single day. The system then optimizes scheduling by adjusting timing and frequency while maintaining safety requirements. User preferences, such as meal-time alignment and dosing frequency, are considered only when they do not conflict with higher-priority safety constraints.

If constraints cannot be fully satisfied, the system prevents unsafe scheduling by reducing frequency or omitting specific medications. This ensures that the final schedule remains safe, practical, and user-adaptive, with minimal interaction risk, potentially achieving a risk level of 0%.

4.6.3 Risk Calculation and Schedule Adjustment

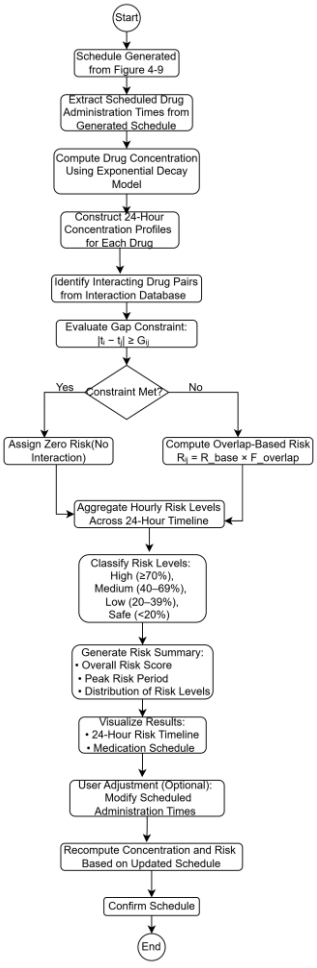


Figure 4-10: Flowchart of Risk Calculation and Schedule Adjustment

Figure 4-10 illustrates the workflow of the risk calculation and schedule adjustment module, which evaluates potential drug interaction risks based on simulated concentration overlap using exponential decay and enables iterative schedule refinement.

The process begins by extracting scheduled administration times from the generated medication schedule. These time points serve as input for concentration modeling. The system computes drug concentration levels using an exponential decay model, where the concentration at time t hours after administration is defined as shown in Equation (15):

$$C(t) = e^{-k.t} \quad (15)\ [23][24]$$

where $C(t)$ represents the normalized drug concentration at time t, assuming a peak concentration of 1.0 at the time of administration, and k represents the elimination rate constant.

The elimination rate constant is derived from the drug half-life $t_{1/2}$ as shown in Equation (16):

$$k = \frac{\ln(2)}{t_{1/2}} \quad (16)\ [23][24]$$

where $t_{1/2}$ represents the drug half-life in hours, obtained from the drughalflife collection. This equation establishes that a longer half-life results in a smaller elimination rate constant, indicating slower drug clearance from the body.

A concentration threshold of 0.1 is applied to determine pharmacological activity, indicating that a drug is considered active when its concentration exceeds 10% of its peak value. For each drug, a 24-hour concentration profile is constructed by selecting the maximum concentration value from all scheduled doses at each time interval.

The system subsequently identifies interacting drug pairs by querying the drug_interactions.json database. For each pair, the gap satisfaction condition is evaluated as shown in Equation (17):

$$| t_i - t_j | \geq G_{ij} \quad (17)$$

where t_i and t_j represent the scheduled administration times of the interacting drugs, and G_{ij} represents the required minimum time separation between the drug pair. If the condition is satisfied, indicating sufficient temporal separation, the corresponding interaction risk is assigned a value of zero.

If the constraint is violated, the system computes an overlap-based risk score as defined in Equation (18):

$$R_{ij} = R_{base} \times \min\left(1.0,\ C_i(h) \times C_j(h) \times 2\right) \quad (18)$$

where R_{ij} represents the risk score contributed by the drug pair, R_{base} represents the base risk associated with the interaction severity, and $C_i(h)$, $C_j(h)$ represent the normalized concentrations of the respective drugs at time interval h.

The base risk values are predefined according to interaction severity, where high-risk interactions are assigned a value of 80, medium-risk interactions 50, low-risk interactions 25, and minor-risk interactions 10.

The risk scores are aggregated across all interacting drug pairs to produce a 24-hour risk timeline. Each time interval is then classified into discrete risk levels, where values greater than or equal to 70 are categorized as high risk, values between 40 and 69 as medium risk, values between 20 and 39 as low risk, and values below 20 as safe.

A comprehensive risk summary is subsequently generated, including the overall risk score, peak risk periods, and the distribution of risk levels throughout the day. These results are presented through a 24-hour risk timeline and a structured medication schedule visualization.

The module further supports interactive schedule adjustment, allowing users to modify administration times by dragging time pills to drop zone cells. Each drag operation triggers a recalculation of concentrations and risk levels with a 500-millisecond delay to optimize performance. Users may also reset to the original schedule or update risk calculations on demand. Once confirmed, the workflow proceeds to schedule tracking as described in Figure 4-11.

4.6.4 Schedule Tracking and Missed Dose Management

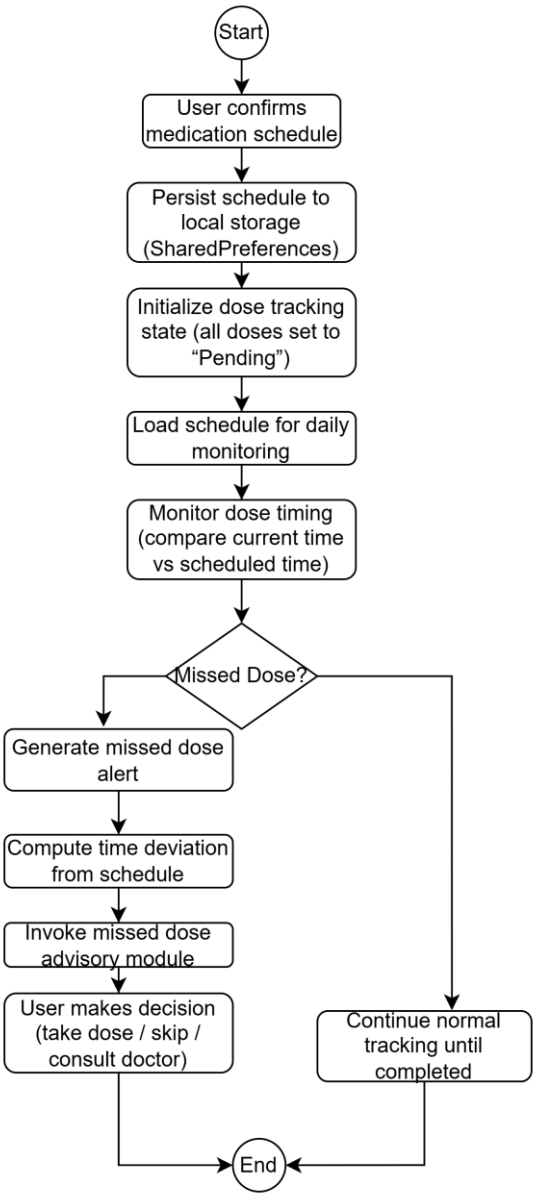


Figure 4-11: Flowchart of Schedule Tracking and Missed Dose Management

Figure 4-11 illustrates the workflow of the schedule tracking and missed dose management module, which supports medication adherence monitoring and missed dose guidance.

The process begins when the user confirms the schedule, which is stored locally using SharedPreferences with a date-based key format schedule_YYYY-MM-DD for daily retrieval without backend requests. Dose tracking is then initialized, with all doses set to a pending state and uniquely identified as taken_{drugId}_{time} with status flags set to false.

The system continuously compares current time with scheduled doses. When a dose time is reached, user confirmation is required via a checkbox interface, and successful intake updates the tracking status and adherence progress.

If a dose is not confirmed after the scheduled time, it is flagged as missed and an alert is generated. Delay is calculated using the time difference between scheduled and current time.

The missed dose decision module is then activated, providing recommendations based on delay duration and drug half-life, including options to take, skip, or consult a doctor.

Schedule deletion is supported with confirmation to prevent accidental data loss. All data is stored locally to ensure privacy and offline access.

The workflow ends when all doses are completed, resolved, or the session is manually terminated.

4.7 Missed Dose Decision Module

The Missed Dose Decision Module helps patients determine whether to take or skip a medication when they forget a scheduled dose. Patients often feel uncertain after missing a dose, unsure if taking it late is safe or if skipping is the better option. This module provides personalized guidance to help users make informed decisions and avoid potential harm from incorrect actions.

4.7.1 Patient Decision Support

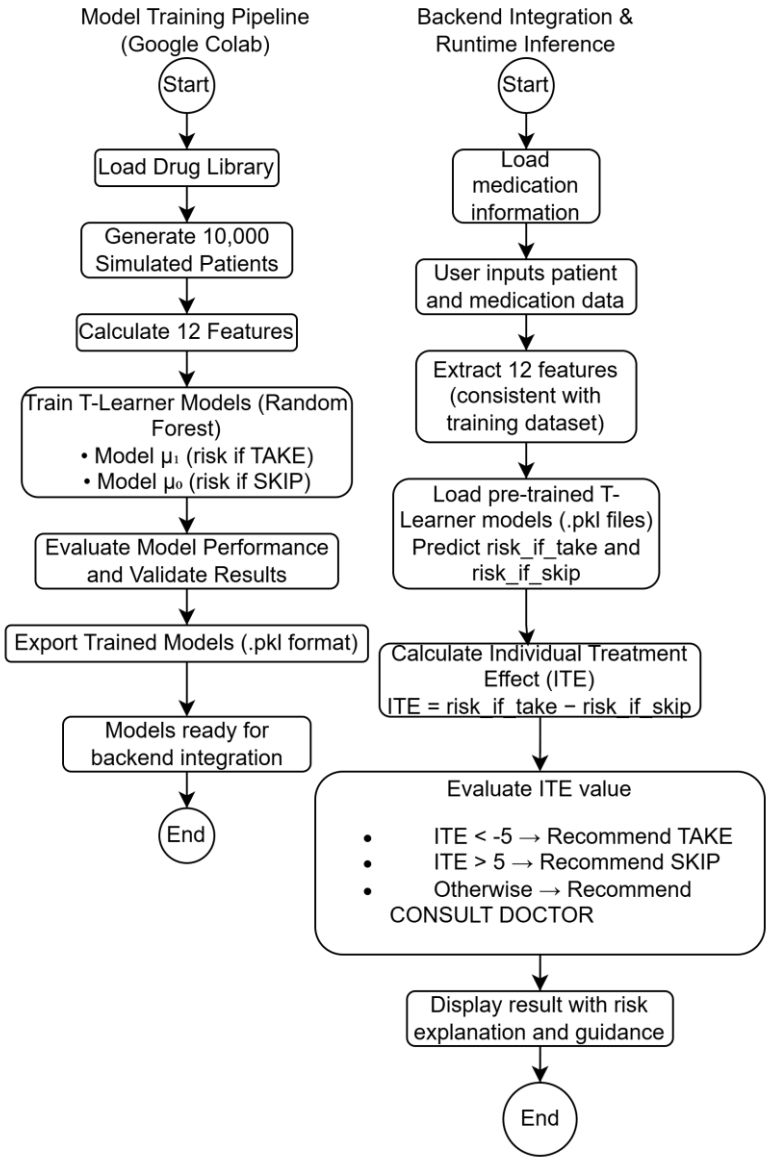


Figure 4-12: Flowchart of Missed Dose Decision Module

Figure 4-12 illustrates the workflow of the missed dose decision module, which employs a T-learner machine learning architecture to provide personalized recommendations when a patient misses a scheduled medication dose. The module consists of two main phases: offline model training and runtime decision support.

The training process begins with the creation of a drug library containing 76 medications, each with defined half-life values and dosing intervals derived from pharmacological data. The system generates 10,000 simulated patient cases, randomly sampling parameters including missed hours (0.5 to 48 hours), age (20 to 90 years), renal function level (1 to 5), symptom severity (1 to 5), and prior adherence rate (0.5 to 1.0). These ranges are selected to represent a

broad spectrum of real-world clinical scenarios, ensuring the trained model generalizes across diverse patient populations.

For each simulated case, the system calculates 12 features as defined in Equation (19). These features are designed to capture the key pharmacokinetic and patient-specific factors that influence the risk-benefit trade-off of taking a missed medication dose:

$$
\begin{aligned}
1.\ & missed\ ratio = \frac{t_{late}}{T_{interval}} \\
2.\ & remaining\ ratio = 0.5^{t_{late}/t_{1/2}} \\
3.\ & accumulation\ factor = \frac{T_{interval}}{t_{1/2}} \\
4.\ & \log missed\ hours = In(t_{late} + 1) \\
5.\ & age\ norm = \frac{age - 50}{20} \\
6.\ & renal\ score = renal\ function \\
7.\ & renal\ x\ age = renal\ score\ \times age\ norm \\
8.\ & symptom\ score = symptom\ severity \\
9.\ & prior\ adherence = prior\ adherence \\
10.\ & time \sin = \sin\left(\frac{2\pi\ \cdot 12}{24}\right) \\
11.\ & time \cos = \cos\left(\frac{2\pi\ \cdot 12}{24}\right) \\
12.\ & half\ life = t_{1/2}
\end{aligned}
\qquad (19)
$$

where t_{late} represents the hours elapsed past the scheduled dose, $T_{interval}$ is the drug's prescribed dosing interval, and $t_{1/2}$ is the drug half-life in hours.

The missed ratio indicates how much of the dosing interval has been missed. The remaining ratio estimates drug still present in the body using exponential decay. The accumulation factor shows the drug's tendency to build up with regular dosing. The log transformation normalizes the wide range of missed hours. Age is normalized to centre the distribution. Renal function and symptom severity are ordinal scores from one to five. The interaction term renal x age captures combined age-kidney effects. Prior adherence reflects the patient's medication-taking consistency. Time sin and time cos encode circadian rhythm fixed at noon. Half-life is the fundamental elimination parameter.

The system then trains two independent Random Forest regressors, each configured with 100 estimators, forming the T-learner architecture. Model μ_1 is trained exclusively on patients who took the missed dose to predict the risk if the dose is taken, while model μ_0 is trained exclusively on patients who skipped the dose to predict the risk if the dose is skipped. Following training, the models are exported as model_treat.pkl and model_control.pkl files.

Once trained, the models are integrated into the backend, where a FastAPI server loads the pickle files at startup. When a user submits a request, the backend receives the input parameters: drug name, hours late, age, renal function level, symptom severity, and prior adherence rate. The system retrieves the corresponding drug's half-life and dosing interval from the drug library and calculates the same 12 features used during training.

The pre-trained models generate two risk predictions: risk if take from model μ_1 and risk if skip from model μ_0. The Individual Treatment Effect (ITE) is calculated as shown in Equation (20):

$$\text{ITE} = \text{risk if take} - \text{risk if skip} \quad (20)$$

A negative ITE indicates that taking the dose reduces risk compared to skipping, while a positive ITE indicates that taking the dose increases risk. The final recommendation is determined by applying clinical decision thresholds: if the ITE is less than -5, the system recommends "TAKE"; if the ITE exceeds 5, the system recommends "SKIP"; otherwise, the system recommends "ASK DOCTOR". The backend returns a JSON response containing the decision, predicted risk values, ITE score, confidence level, and an explanatory message.

4.7.2 Mapping of User Inputs to Engineered Features

User Input	Engineered Feature(s)	Description
Drug Name	Half-life [24]	Retrieved from drug database; represents how long the drug stays active in the body
Hours Late	Missed Ratio, Log Missed Hours	Measures how delayed the dose is and transforms delay into normalized values
Hours Late + Half-life	Remaining Drug Ratio [24]	Estimates how much drug effect remains in the body
Drug Half-life + Dosing Interval	Accumulation Factor [24]	Measures drug accumulation risk over time
Age	Normalized Age	Converts age into a standardized value for model input
Kidney Function	Renal Score [25]	Represents kidney performance level (1–5 scale based on eGFR) [25]
Kidney Function + Age	Renal × Age	Captures combined risk effect of age and kidney condition
Symptom Severity	Symptom Score	Measures current patient condition severity (1–5 scale)
Prior Adherence	Prior Adherence	Represents how consistently the user takes medication on time
Time of Day	Fixed Cyclic Feature (sin) [26]	Encodes daily time cycle pattern
Time of Day	Fixed Cyclic Feature (cos) [26]	Encodes daily time cycle pattern
Drug Properties	Half-life [24]	Pharmacokinetic property from database used in calculations

Table 4-2: Mapping of User Inputs to Engineered Features for Missed Dose Decision Model

Table 4-2 presents the feature engineering process for the missed dose decision model, where raw user inputs are transformed into structured features before model inference. Instead of using raw data directly, each input is converted into medically meaningful and computationally suitable representations.

For instance, hours late are transformed into features such as missed ratio and log-transformed delay to better capture non-linear risk escalation over time. Drug name is mapped

to pharmacokinetic properties, including half-life, which is used to estimate remaining drug concentration and accumulation risk.

Patient-related variables such as age, kidney function, and symptom severity are normalized to reflect overall physiological condition, while adherence history is incorporated to represent medication-taking behaviour and compliance level.

In total, 12 engineered features are generated and used by the machine learning model to predict two outcomes: the risk of taking the missed dose and the risk of skipping it. The final decision (Take, Skip, or Ask Doctor) is determined by comparing both risks using an Individual Treatment Effect framework.

Chapter 5

System Implementation

5.1 Hardware Setup

Description	Specifications
Model	Huawei nova 7 SE / CDY-NX9B
Processor	HiSilicon Kirin 820 5G (Octa-core: 1×2.36 GHz + 3×2.22 GHz + 4×1.84 GHz)
Operating System	Android 10 with EMUI 10.1
Graphic	Mali-G57 MP6 (6-core)
Memory	8 GB RAM
Storage	128 GB internal storage

Table 5-1 Specifications of Mobile Device

Description	Specifications
Model	Vivobook_ASUSLaptop X1502VA_A1502VA
Processor	13th Gen Intel® Core™ i7-13700H, 2.4 GHz, 14 Cores, 20 Threads
Operating System	Windows 11 Home Single Language
Graphic	Intel® Iris® Xe Graphics
Memory	16 GB DDR4/DDR5
Storage	256 GB SSD

Table 5-2 Specifications of laptop

5.2 Software Setup

Component	Configuration / Setup
Flutter Environment	Flutter 3.22.2 (Stable) installed with Android Studio integration
Backend Server (Node.js)	Express.js server running locally on development environment
Machine Learning API	Python FastAPI service
Model Training Environment	Google Colab used for model training and experimentation
Database Setup	MongoDB (Local) with MongoDB Compass
Frontend–Backend Communication	HTTP REST API (JSON format)
Mobile Testing Environment	Physical Android device
Local Storage Configuration	SharedPreferences (Flutter)

Table 5-3 Software Setup

5.3 System Operation

5.3.1 DrugSafe+ Application Overview

Figure 5-1 Application Logo and Project Name

Figure 5-1 shows the application logo featuring a green drug icon and the project name branding, designed to match the theme and purpose of the medication safety system.

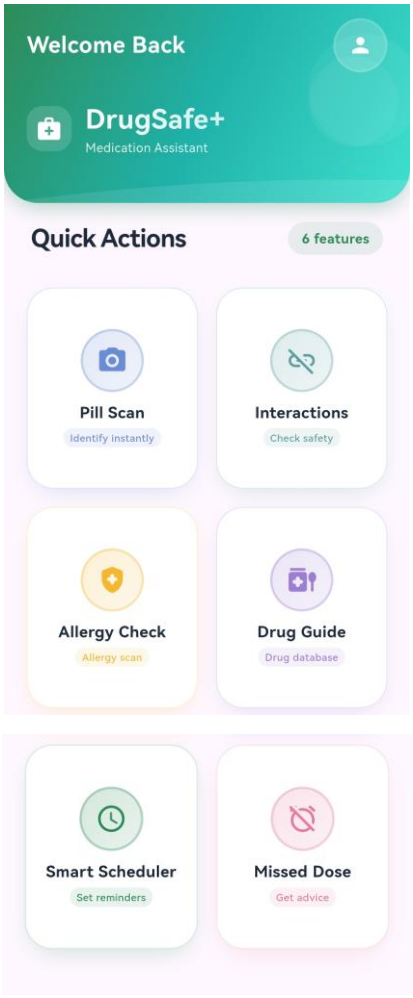


Figure 5-2: Main Interface of the DrugSafe+ Application

Figure 5-2 illustrates the main interface of the DrugSafe+ application. The interface is designed with a clean and user-friendly layout, enabling users to conveniently navigate across the six core modules.

The application title, “DrugSafe+,” is prominently displayed to provide clear system identification and enhance usability.

5.3.2 Pill Image Recognition Module

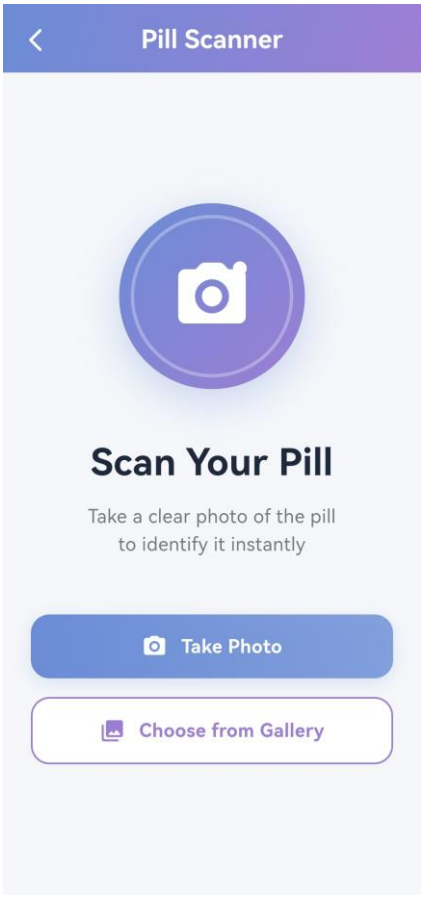


Figure 5-3: Pill Image Recognition Module Interface

Figure 5-3 presents the interface of the Pill Image Recognition module, which enables users to identify medications by scanning pill images. The interface adopts a purple-themed design to enhance visual consistency and user experience. Users are provided with two options for image input: capturing a real-time photo using the device camera or selecting an existing image from the gallery, thereby offering flexibility and convenience in the pill identification process.

Case 1: CNN-Based Classification Testing

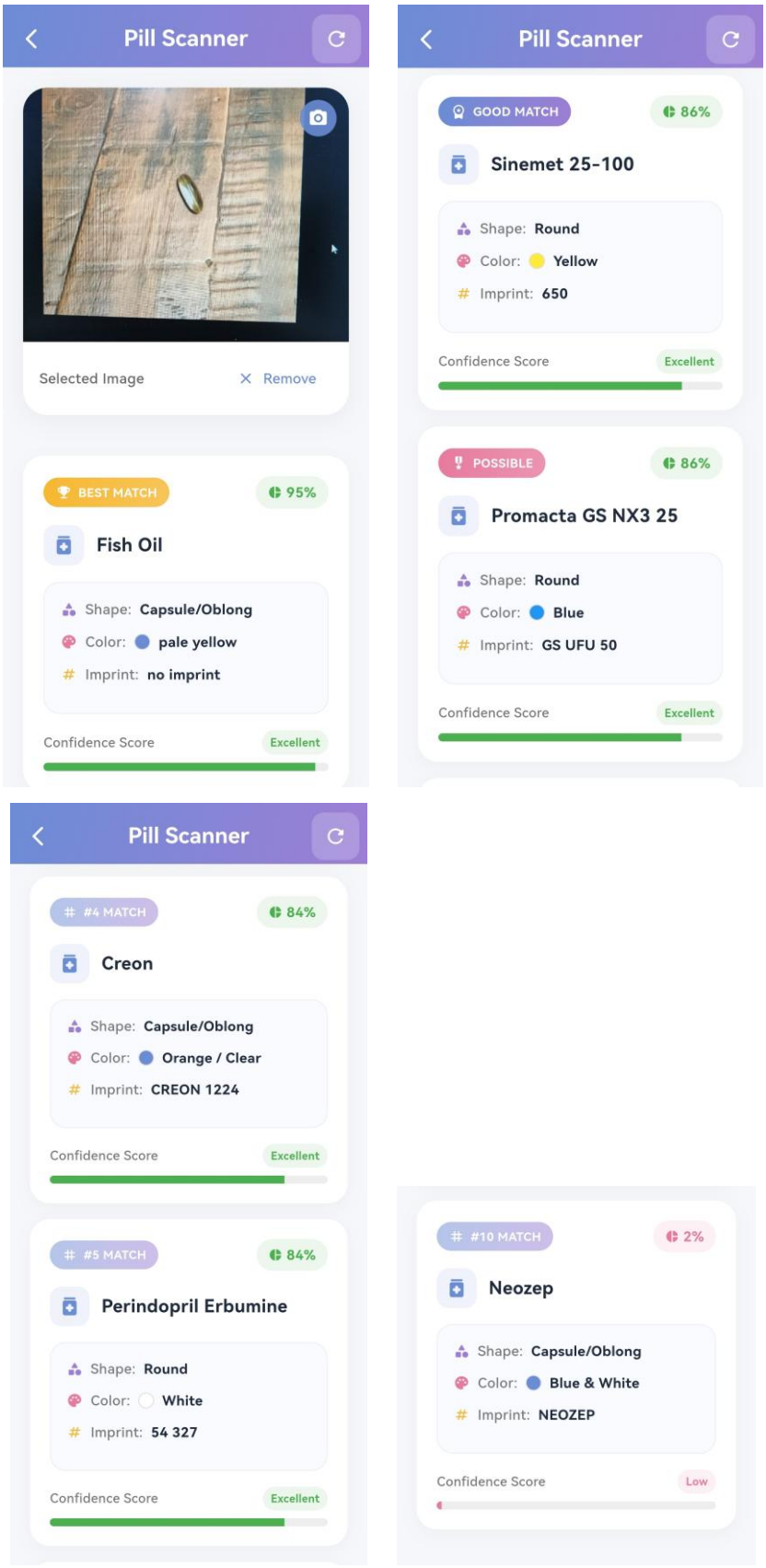


Figure 5-4: CNN-Based Classification Results for Fish Oil Sample

Figure 5-4 illustrates the classification results generated by the CNN-based model when a fish oil sample is tested. The system outputs the top 10 predicted classes, with “fish oil” correctly identified as the top prediction (Top-1) with an accuracy of 95%. This result demonstrates the effectiveness of the model in accurately recognizing the tested medication sample.

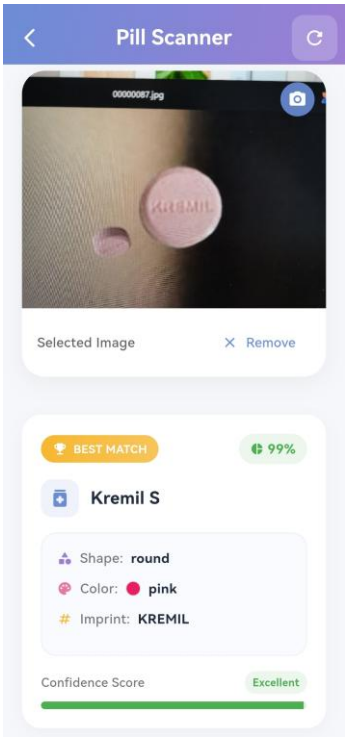


Figure 5-5: CNN-Based Classification Results for Kremil-S Sample

Figure 5-5 presents the classification results produced by the CNN-based model for the second test using a Kremil-S sample. The system displays the top 10 predicted classes, with “Kremil-S” correctly identified as the top prediction (Top-1) with an accuracy of 99%. This result further demonstrates the robustness and high accuracy of the model in medication recognition across different samples.

Case 2: CNN Embedding-Based Training Evaluation

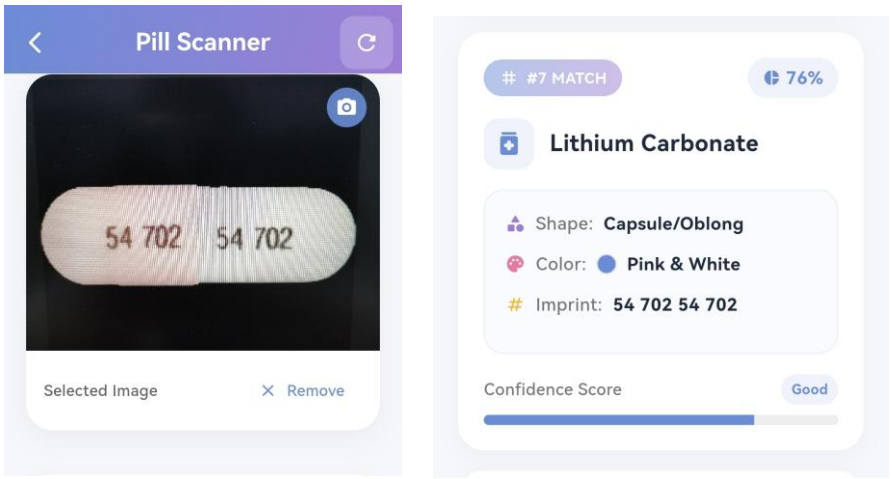


Figure 5-6: CNN Embedding-Based Retrieval Results for Lithium Carbonate Sample

Figure 5-6 presents the results obtained using the CNN embedding-based approach for a Lithium Carbonate sample. The system retrieves and displays the top 10 most similar classes based on feature embeddings, where "Lithium Carbonate" appears at the seventh position (Top-7) with a similarity score of 76%. Although it is not ranked as the top result, the correct class is still included within the top 10 predictions, indicating that the model is capable of capturing relevant feature representations for medication identification.

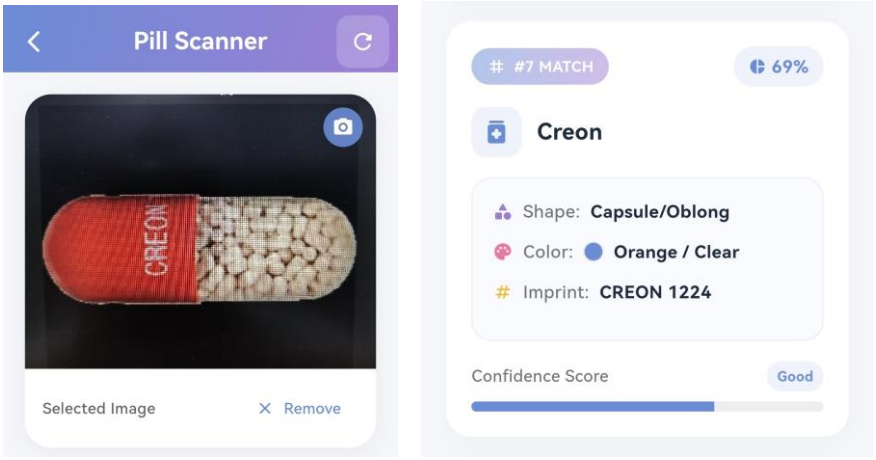


Figure 5-7: CNN Embedding-Based Retrieval Results for Creon Sample

Figure 5-7 presents the results of the second test case using a Creon sample under the CNN embedding-based approach. The system retrieves and displays the top 10 most similar classes based on feature embeddings, where "Creon" is ranked at the seventh position (Top-7) with a similarity score of 69%.

5.3.3 Drug-Drug Interaction Module

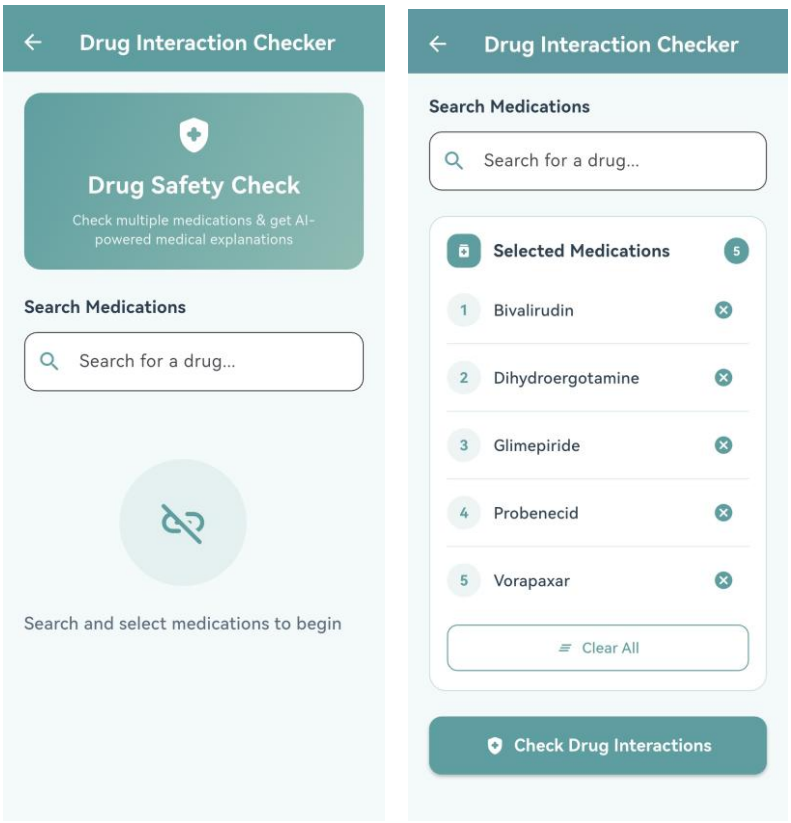


Figure 5-8: Drug-Drug Interaction Module Interface Design and Medication Selection Function

Figure 5-8 illustrates the interface design of the Drug-Drug Interaction module, which allows users to search and select medications for interaction checking. The interface adopts a ocean blue and teal gradient theme to enhance visual clarity and maintain consistency with the overall application design. A dedicated medication selection feature is provided, requiring users to select at least two drugs before performing the interaction analysis. This ensures meaningful comparison between medications and supports an intuitive and user-friendly workflow.

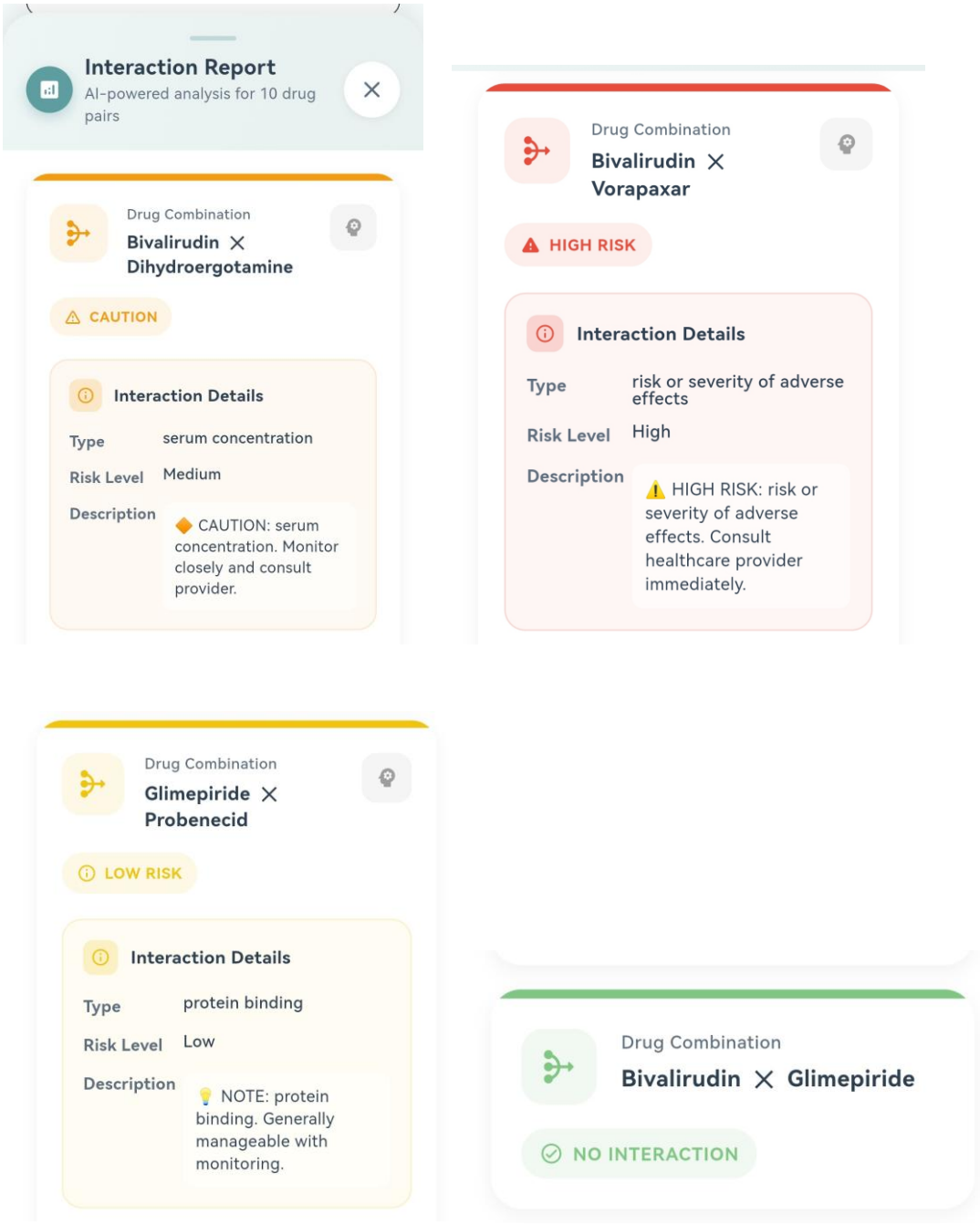


Figure 5-9: Drug-Drug Interaction Report with Risk Classification and Details

Figure 5-9 presents the drug-drug interaction report generated by the system, displaying the interaction results based on different risk levels, including high, medium, low, and no risk. Each risk category is represented using distinct color indicators to enhance visual clarity and improve user understanding. The report also provides interaction information, including the interaction type, risk level, and a brief description, enabling users to quickly interpret potential medication safety issues.

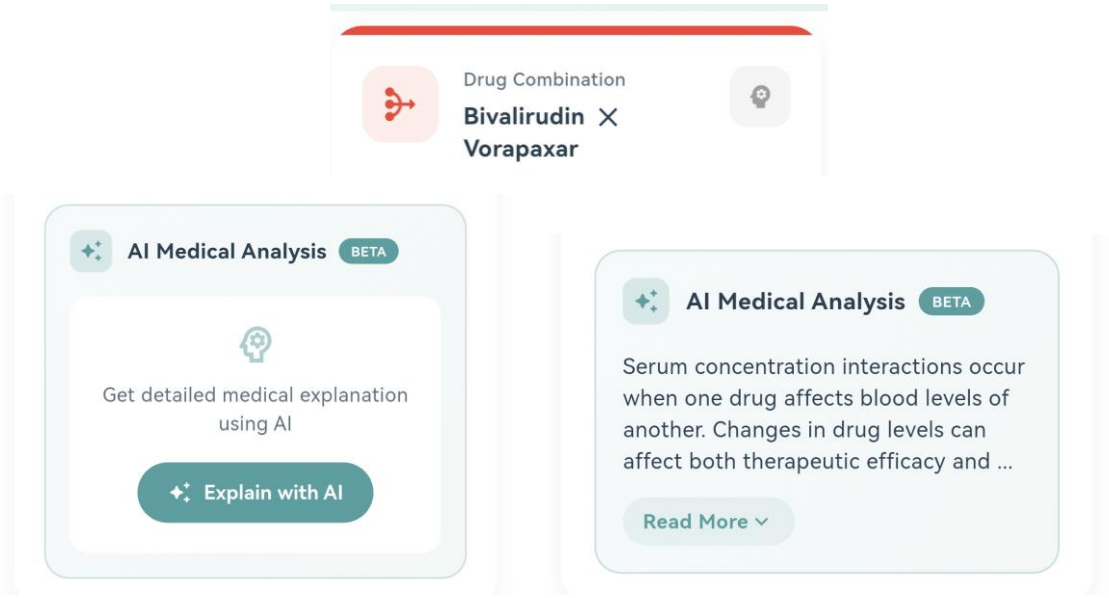


Figure 5-10: AI-Based Medication Interaction Analysis Interface

Figure 5-10 illustrates the AI-based medication interaction analysis interface, which provides an enhanced explanation of detected drug interactions based on interaction type. The interface allows users to access additional information when needed for better understanding. By selecting the “Psychology” icon or the “Explain with AI” function, users can trigger an AI-generated explanation of the interaction results. In addition, a “Read More” option is available to expand the content, providing more detailed insights into the medication interaction and supporting informed decision-making.

5.3.4 Drug Information Module

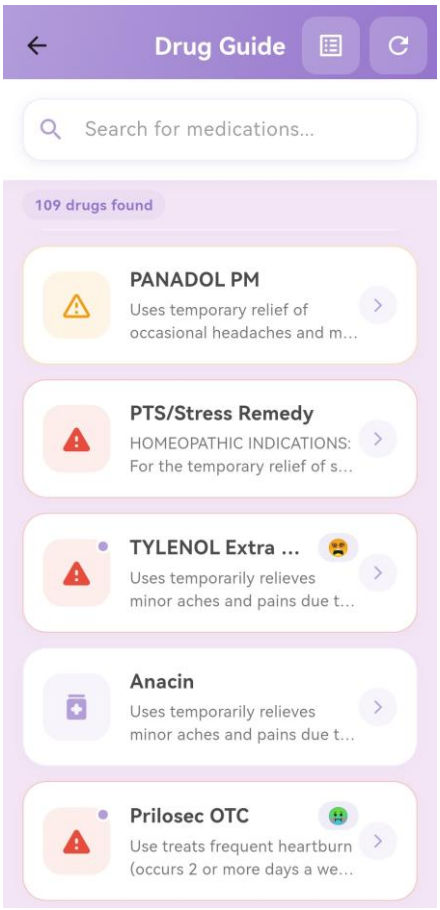


Figure 5-11: Drug Information Module Interface

Figure 5-11 illustrates the Drug Information module interface with a purple-themed layout. The left-side indicators represent medication status: purple indicates a normal drug with no allergy association, yellow highlights drugs sharing active ingredients with recorded allergies, and red marks drugs explicitly identified as allergens. A notification dot indicates entries with user-added allergy notes.

A notebook icon at the top-right provides quick access to the Allergy Management module for reviewing or managing allergy records. This visual system improves clarity and enables users to quickly assess medication safety status.

Case 1: Normal Drug Information Display (Purple Indicator)

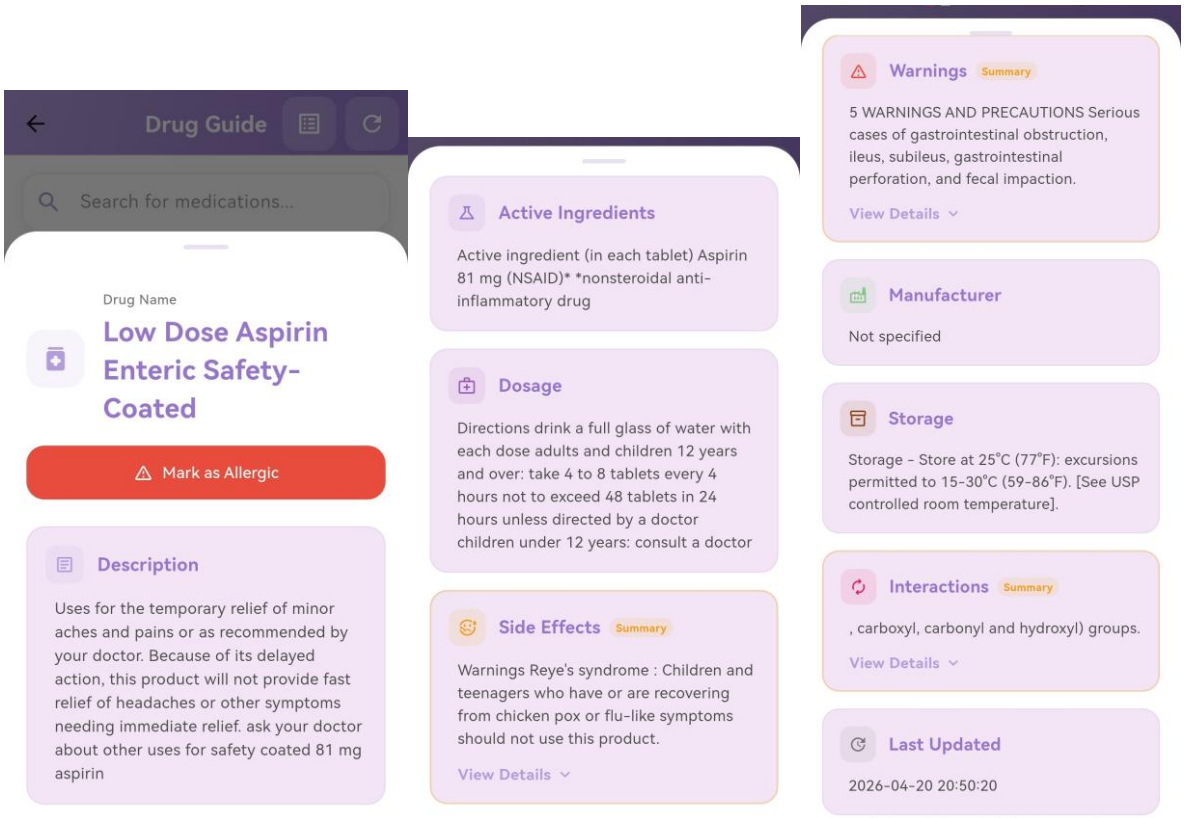


Figure 5-12: Normal Drug Information View

Figure 5-12 illustrates the normal case of the Drug Information module, indicated by a purple status icon. In this state, users can safely view general drug information without allergy-related alerts. The system displays key details including drug description, active ingredients, dosage, side effects, warnings, manufacturer information, storage instructions, interactions, and last updated timestamp.

To improve readability, a concise summary is shown by default, while full details can be accessed via the “View Details” option. Drug information is retrieved from the OpenFDA database and refreshed every 7 days to ensure accuracy and reliability.

Case 2: Allergy Alert Status (Red Indicator)

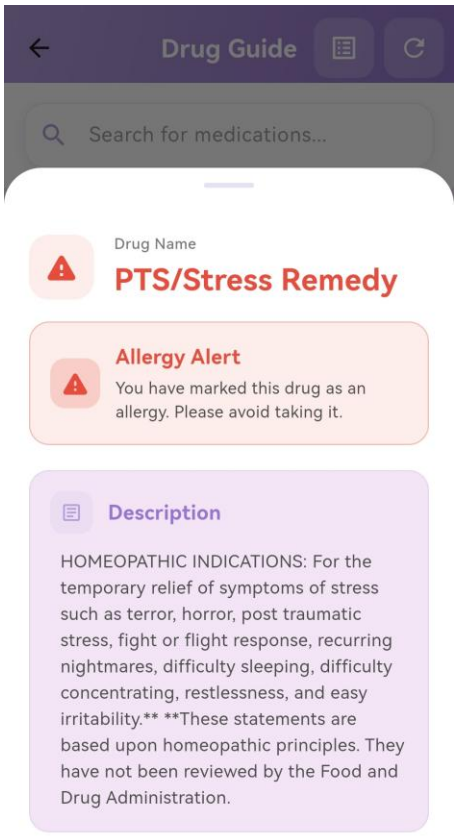


Figure 5-13: Drug Information View with Allergy Alert

Figure 5-13 illustrates the allergy alert state of the Drug Information module, indicated by a red status icon. In this state, the drug has been previously marked as an allergen by the user. The interface highlights the information in red and displays an allergy warning message to alert the user that the medication should be avoided.

This design enables quick identification of unsafe drugs and reduces the risk of accidental exposure to allergens.

Case 3: Allergy Status with Notification Indicator (Red Icon with Dot)

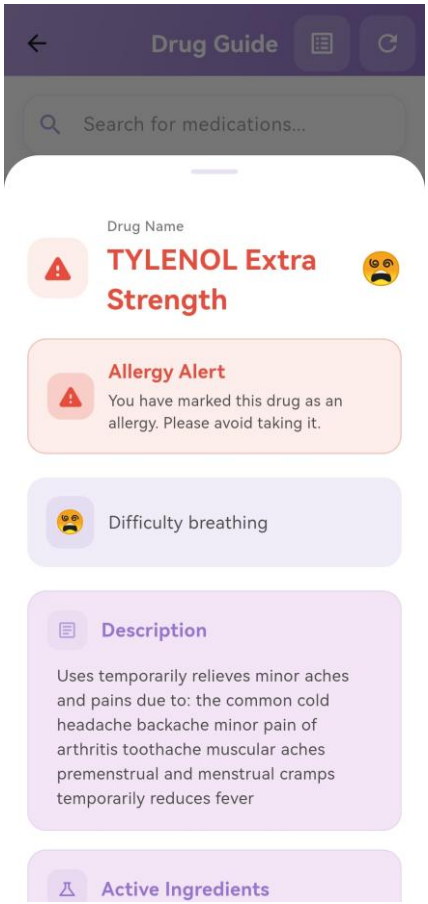


Figure 5-14: Drug Information View with Allergy Marker and Notification Dot

Figure 5-14 illustrates the drug information interface when a medication is both marked as an allergy and contains a user-added personal note from the Allergy Management module. A red warning icon indicates that the drug is recorded as an allergen, while a notification dot shows the existence of additional personal notes.

This combination improves visual clarity by allowing users to quickly identify high-risk medications and access relevant personal allergy information, thereby supporting safer medication management.

Case 4: Potential Allergy Warning (Yellow Indicator)

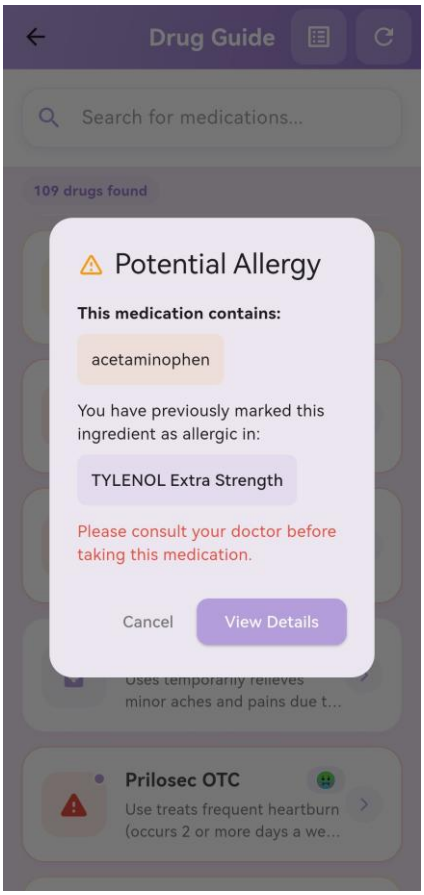


Figure 5-15: Drug Information View with Potential Allergy Warning

Figure 5-15 illustrates the drug information interface with a displayed warning box when a potential allergy risk is detected, indicated by a yellow warning icon. In this case, the system identifies that the medication contains active ingredients that are the same as those found in drugs previously marked as allergies by the user. A warning message is displayed to alert the user of potential cross-reactivity risk and to advise caution before use. The interface also highlights the relevant active ingredients and recommends that the user consult a healthcare professional for further guidance.

5.3.5 Allergy Management Module

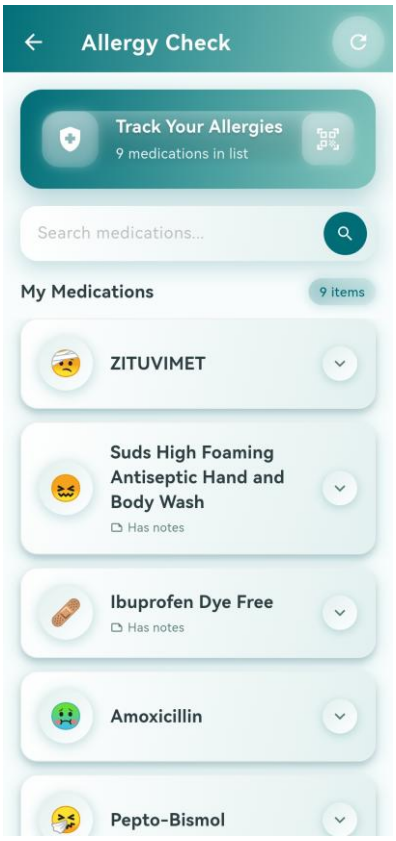


Figure 5-16: Allergy Management Module Interface

Figure 5-16 presents the interface of the Allergy Management module, designed with a teal and mint theme to enhance readability and user comfort. The interface lists user-recorded drug allergies, each with a user-reported feeling indicator and emoji representation.

An expandable dropdown feature allows users to view detailed allergy information. The interface adopts an accordion-style design, where only one record can be expanded at a time while previously opened items automatically collapse. This ensures a clean and organized display of allergy records.

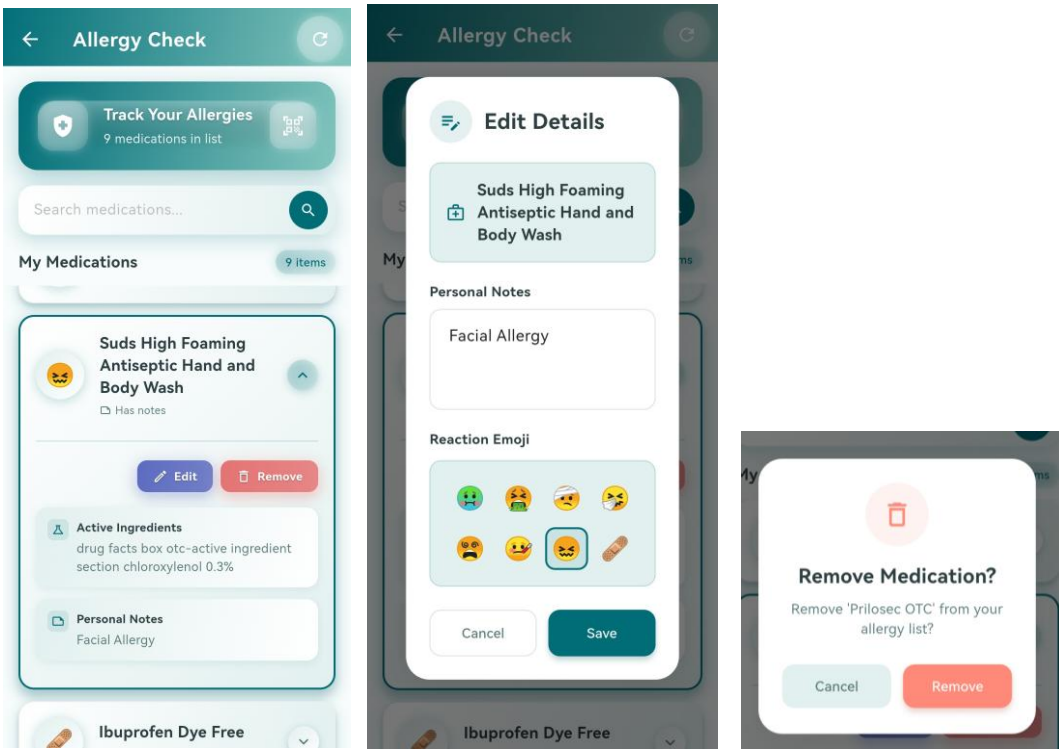


Figure 5-17: Allergy Information Card and Editing Interface

Figure 5-17 illustrates the allergy information card and editing interface within the Allergy Management module. The interface displays key details of each recorded allergy, including active ingredients. Users are provided with the option to add personal notes and select a reaction emoji to describe their individual response to the allergy. In addition, edit and remove functions are available, allowing users to update or delete allergy records to maintain accurate and personalized allergy information.

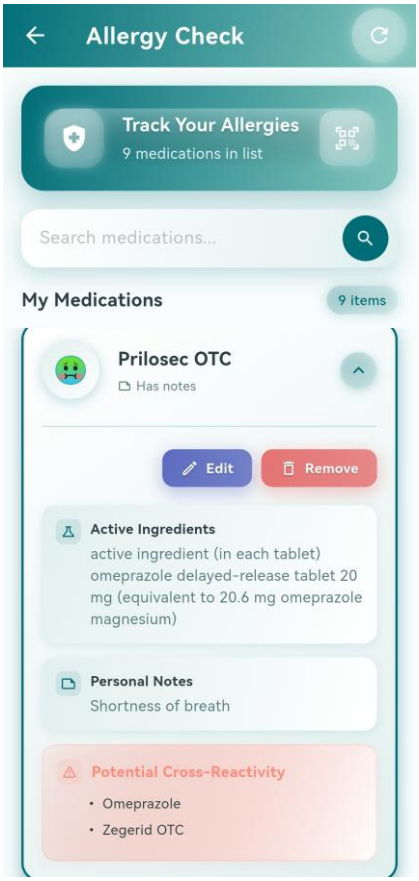


Figure 5-18: Potential Cross-Reactivity Warning Card

Figure 5-18 presents the potential cross-reactivity warning card within the Allergy Management module. This feature alerts users when other medications contain active ingredients that are the same as or similar to those in the currently selected drug, which has been marked as an allergy. The system highlights potential risks to support safer medication use and to prevent accidental exposure to allergenic substances. This warning mechanism helps users identify related medications that should be used with caution due to possible cross-reactivity.

5.3.6 Smart Drug Scheduler Module

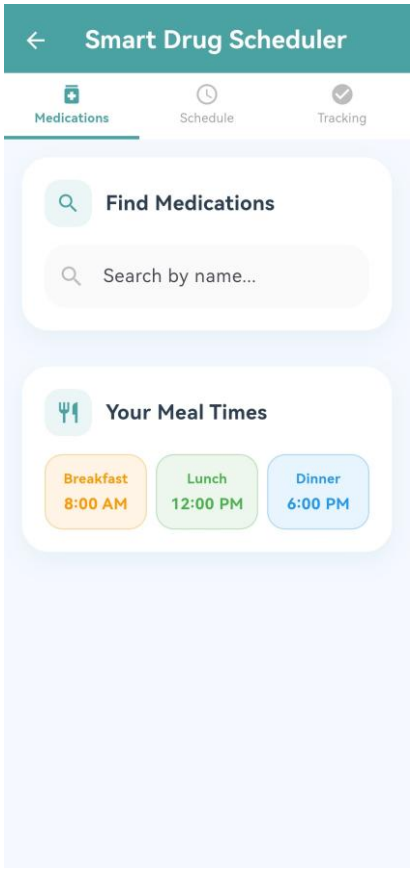


Figure 5-19: Smart Drug Scheduler Module Interface

Figure 5-19 shows the Smart Drug Scheduler interface for daily medication planning. Users input drug name, dosing frequency, and meal-related timing (before, after, with meals, or anytime), and can also set personalized meal times based on their routine. The system supports daily scheduling only, without weekly or monthly planning.

The interface includes three main tabs: Medication for entering drug details, Schedule for displaying the generated timetable, and Tracking for monitoring intake, marking doses as taken, and receiving reminders.

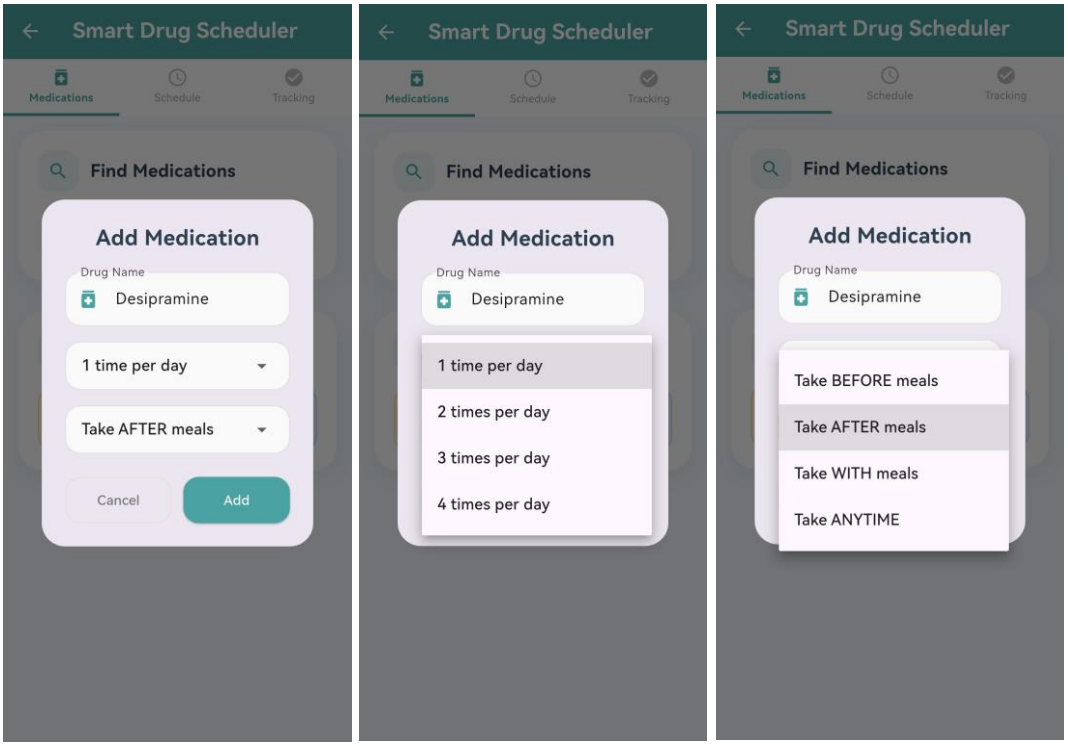


Figure 5-20: Frequency and Meal-Time Selection Card in Smart Drug Scheduler Module

Figure 5-20 illustrates the selection card within the Smart Drug Scheduler module, where users configure medication intake settings. The card allows users to select the drug consumption frequency and define the timing relative to meals, including before meals, after meals, with meals, or anytime. This interface provides a structured and user-friendly method for setting daily medication schedules according to individual dietary routines and prescription requirements.

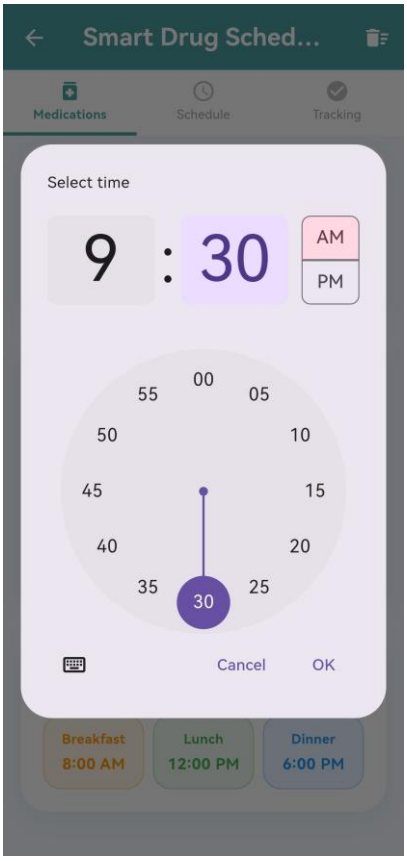


Figure 5-21: Meal Time Picker for Daily Medication Scheduling

Figure 5-21 illustrates the time picker interface within the Smart Drug Scheduler module for configuring meal times, including breakfast, lunch, and dinner. This feature allows users to set specific time values for each meal according to their daily routine.

Test Case 1: Single Drug Normal Scheduling Verification (Desipramine: 1x Daily, Meal Times: 9:30 AM, 2:00 PM, 7:00 PM)

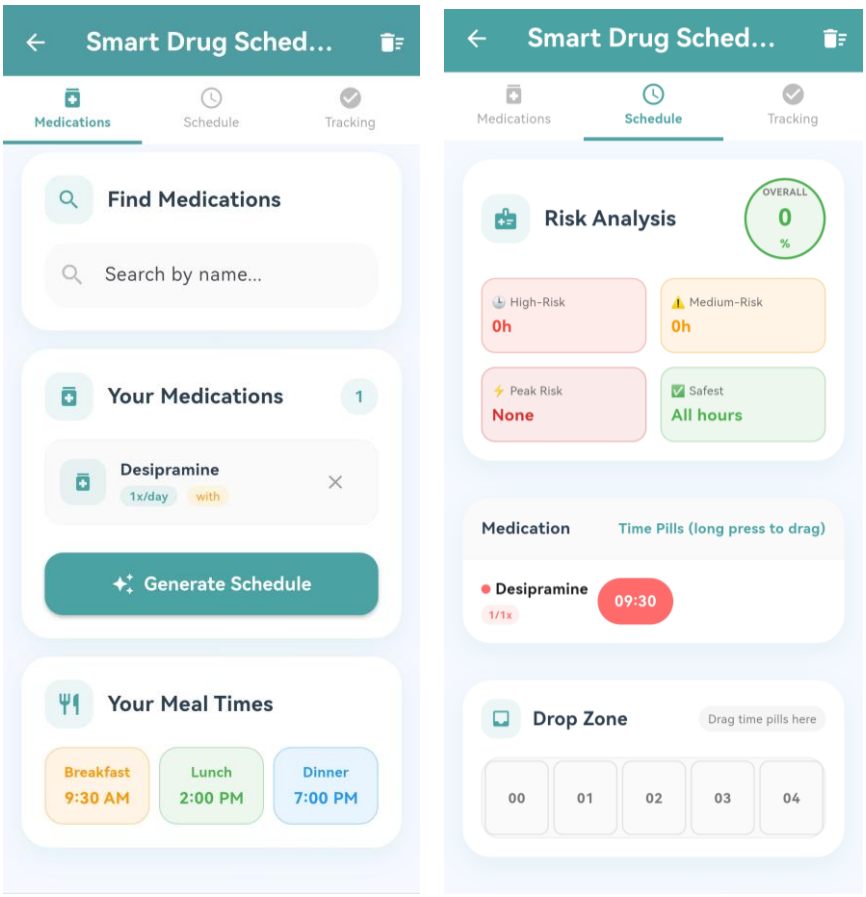


Figure 5-22: Desipramine Meal-Time Scheduling Test Case (With Meals)

Figure 5-22 illustrates the scheduling result for Desipramine under the “with meals” condition. The medication intake is aligned exactly with the selected meal time.

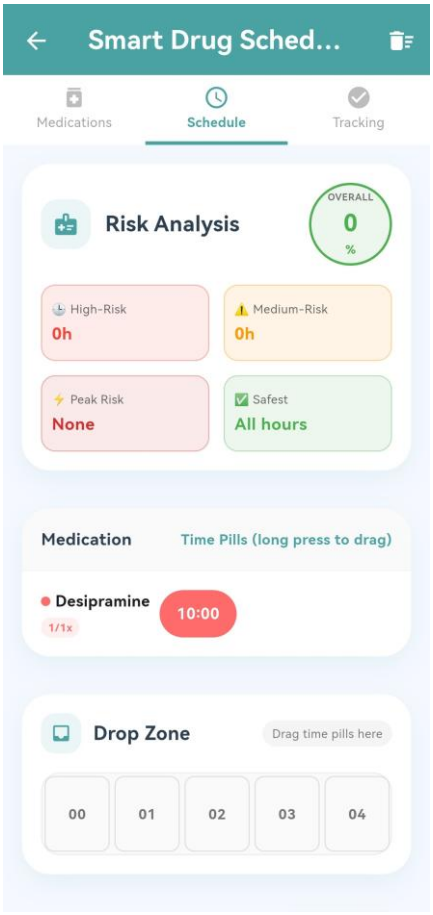


Figure 5-23: Desipramine Meal-Time Scheduling Test Case (After Meals)

Figure 5-23 illustrates the scheduling result for Desipramine under the "after meals" condition. The system schedules the medication 30 minutes after the selected meal time, ensuring compliance with the after-meal intake rule.

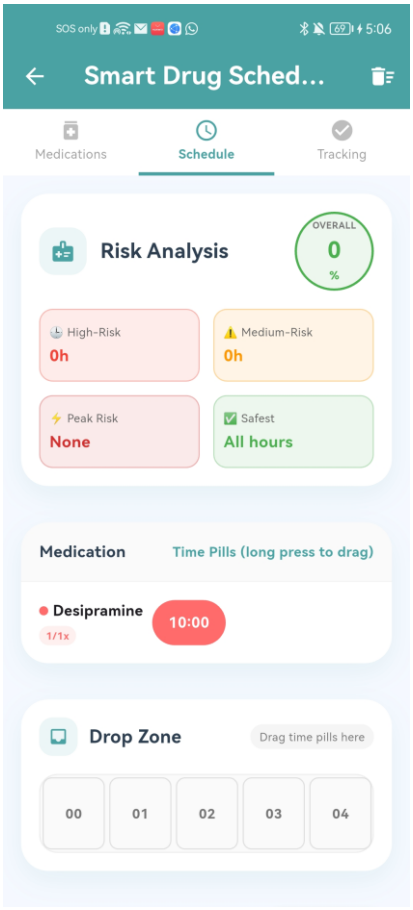


Figure 5-24: Desipramine Meal-Time Scheduling Test Case (Anytime)

Figure 5-24 presents the scheduling result for Desipramine under the “anytime” condition. The system assigns an appropriate time slot independent of meal times while maintaining consistency with the daily schedule.

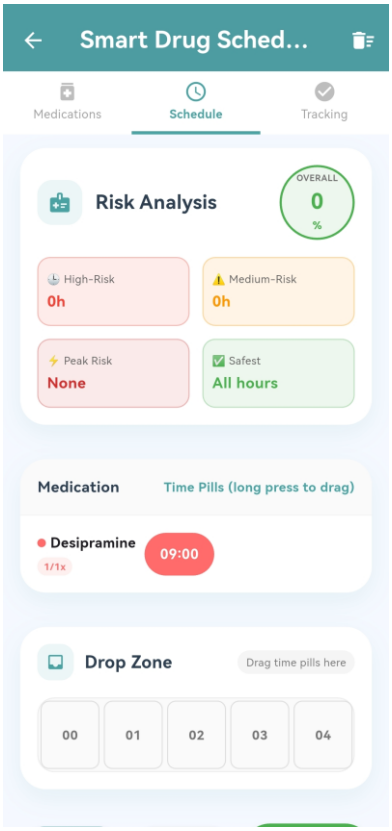


Figure 5-25: Desipramine Meal-Time Scheduling Test Case (Before Meals)

Figure 5-25 illustrates the scheduling result for Desipramine under the "before meals" condition. Based on the predefined meal times, the system schedules the medication 30 minutes before the selected meal time, demonstrating correct alignment with the before-meal rule.

Test Case 2: Two-Drug Interaction Scheduling Verification (Amoxicillin: 1x Daily, After Meals; Nicardipine: 3x Daily, After Meals; Meal Times: 9:30 AM, 2:00 PM, 8:00 PM)

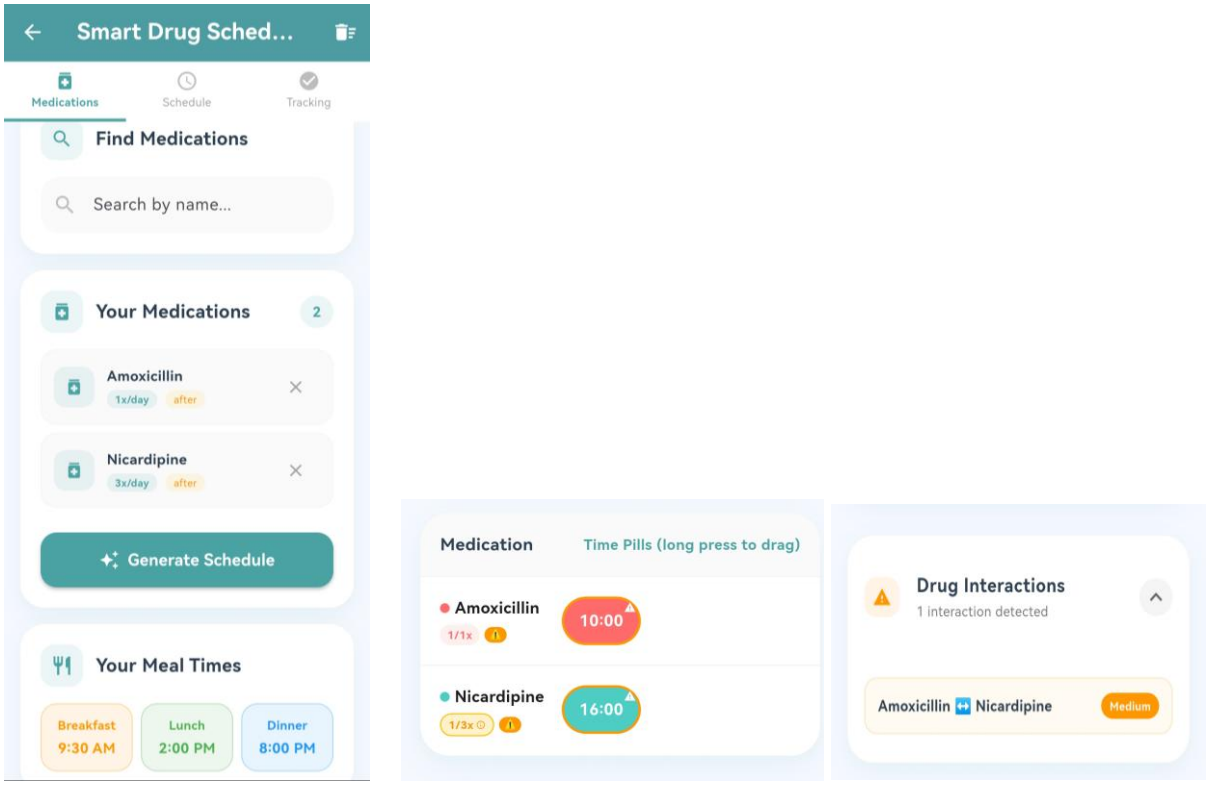


Figure 5-26: Interaction-Aware Scheduling with Frequency Adjustment

Figure 5-26 shows the scheduling output for Amoxicillin (1× daily, after meals) and Nicardipine (3× daily, after meals) with a detected drug–drug interaction. The system prioritizes safety by adjusting the schedule, including reducing Nicardipine to a safer suggested frequency based on interaction constraints.

A warning message is displayed to notify the user of the interaction. An expandable section is also provided to view or hide detailed interaction information, improving interface clarity. This demonstrates the system's ability to dynamically modify scheduling to ensure a safer medication plan.

Test Case 3: Multiple Drug Scheduling Verification (Apalutamide: 4x Daily, Anytime; Sildenafil: 3x Daily, With Meals; Aspirin: 1x Daily, Anytime; Meal Times: 8:00 AM, 12:00 PM, 6:00 PM)

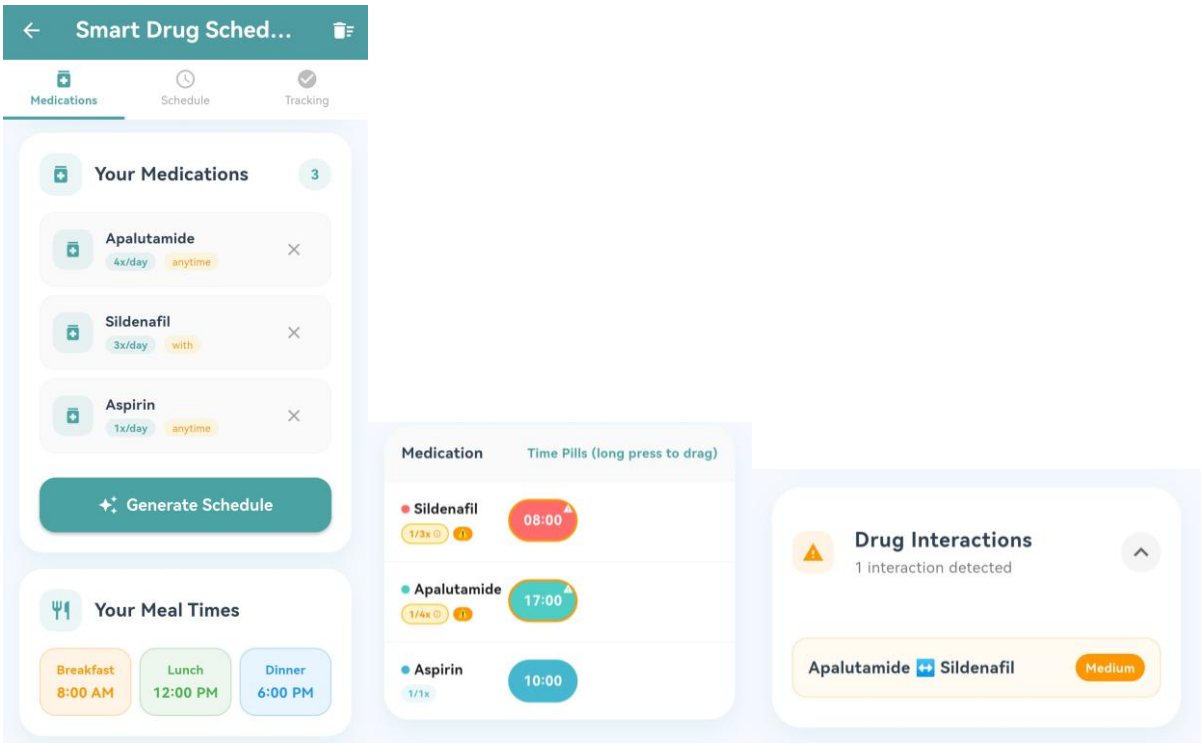


Figure 5-27: Multiple Drug Scheduling with Interaction Warning Indicator

Figure 5-27 illustrates the generated timetable for multiple drugs, where interaction risks are present. A warning icon is displayed below each affected drug in the timetable to indicate the presence of drug-drug interactions. The interaction warning section provides the interacting drug pairs along with their corresponding risk levels, allowing users to clearly identify which medications interact with each other. This visual indication enhances user awareness of potential risks and supports safer and more informed medication scheduling decisions.

Test Case 4: Multiple Drug Scheduling with Constraint Violations and Interaction Handling (Asenapine: 4x Daily, After Meals; Atorvastatin: 4x Daily, After Meals; Ziprasidone: 1x Daily, Anytime; Amoxicillin: 1x Daily, Anytime; Simvastatin: 1x Daily, After Meals; Meal Times: 10:00 AM, 1:00 PM, 8:00 PM)

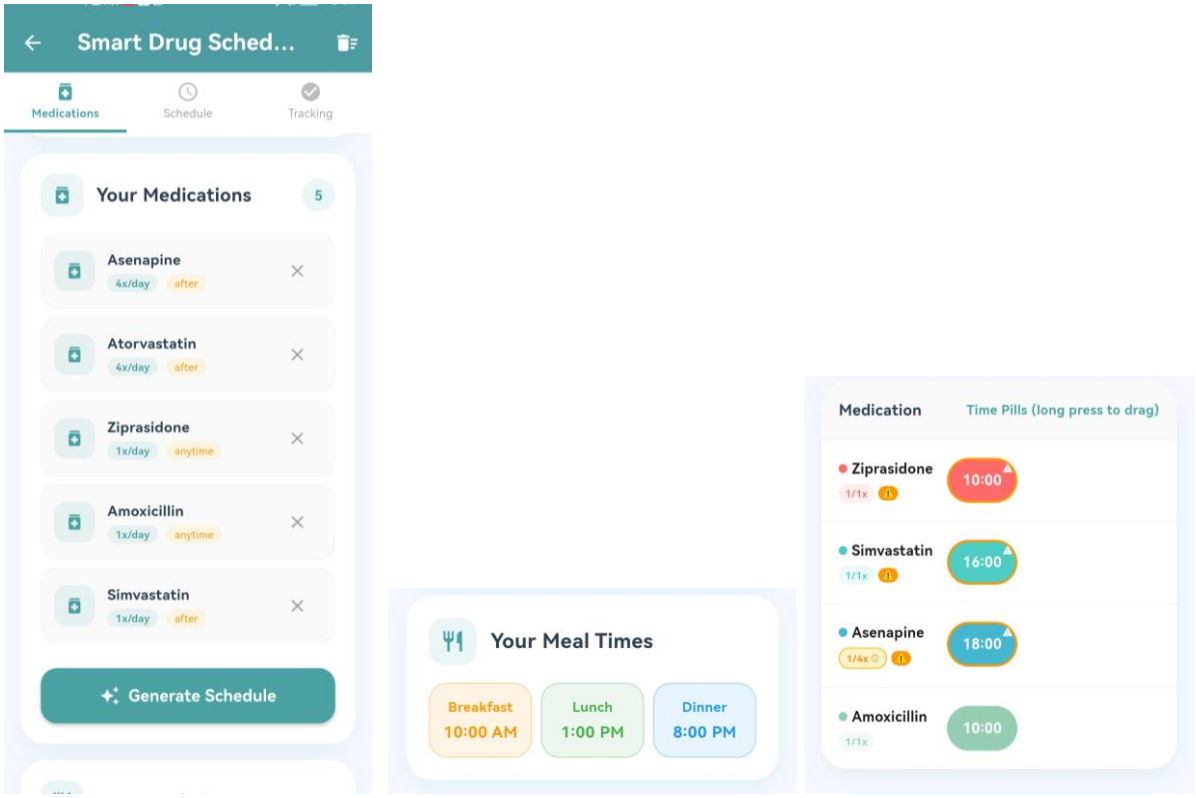


Figure 5-28: Multi-Drug Scheduling with Interaction Constraints

Figure 5-28 illustrates the generated timetable for multiple drugs based on predefined meal times. The schedule is aligned with meal-based timing rules; however, due to multiple drug-drug interactions and insufficient available time gaps within a single day, not all selected medications can be fully scheduled. In such cases, the system applies optimization by adjusting dosing frequency to suggested levels where possible, while certain drugs are excluded from scheduling to ensure safety constraints are maintained. This process ensures that the final generated schedule maintains a risk level of 0%, prioritizing patient safety while preserving feasible medication timing.

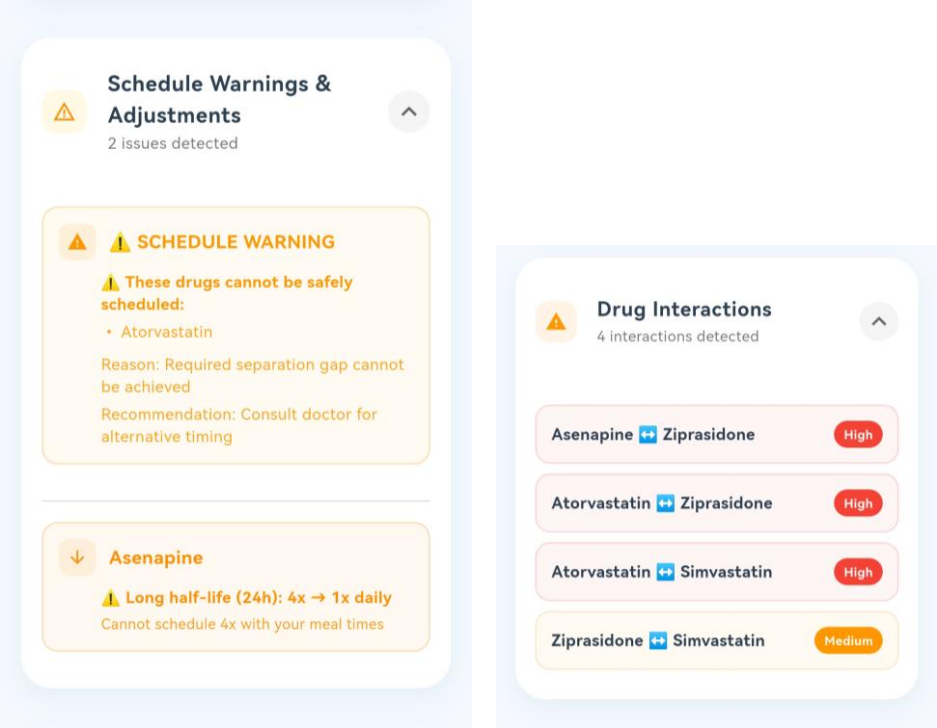


Figure 5-29: Drug Interaction Warnings and Schedule Adjustment Notifications

Figure 5-29 illustrates the scheduling warning messages indicate cases where certain drugs cannot be safely scheduled, providing the reasons and recommended actions for the user. Some warnings also show adjustments in medication frequency due to long half-life properties of specific drugs. In addition, the drug interaction section displays detected interactions between medications along with their corresponding risk levels, allowing users to clearly understand the safety implications of the generated schedule.

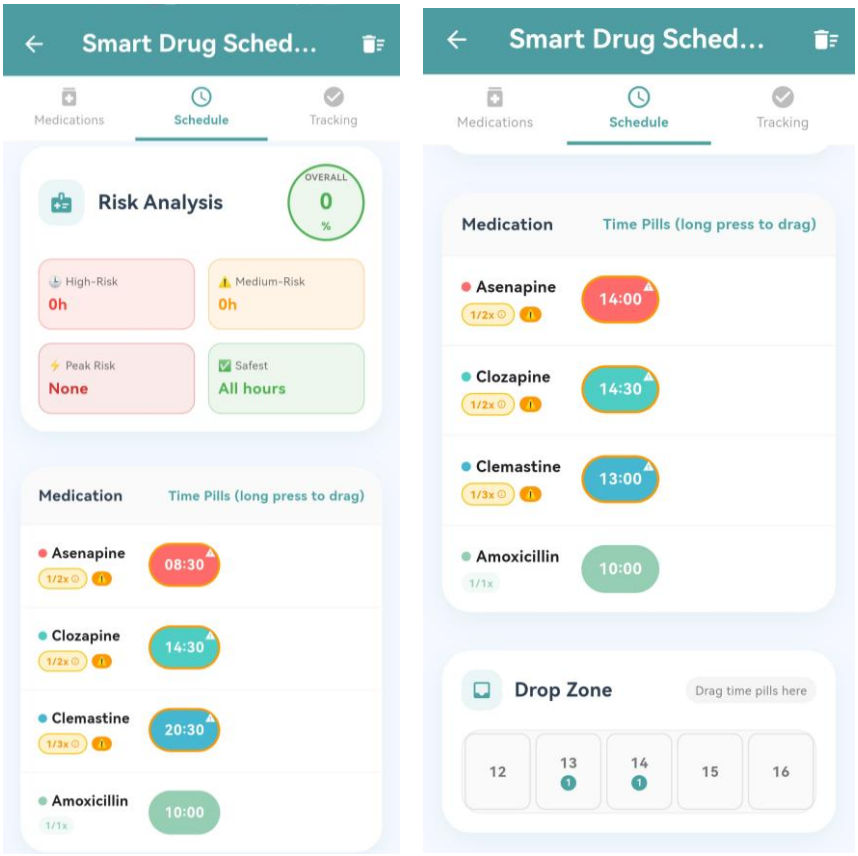


Figure 5-30: Drag-and-Drop Medication Time Adjustment Feature

Figure 5-30 illustrates the drag-and-drop feature for adjusting medication schedules to user-preferred time slots. If users are not satisfied with the automatically generated schedule, they can manually drag and drop the medication time to a desired position within the timetable. In this example, the intake time of Asenapine is adjusted from 8:30 AM to 2:00 PM, while Clemastine is adjusted from 7:30 PM to 1:00 PM. This feature allows users to customize their medication schedule while maintaining system consistency and updating the overall timetable accordingly.

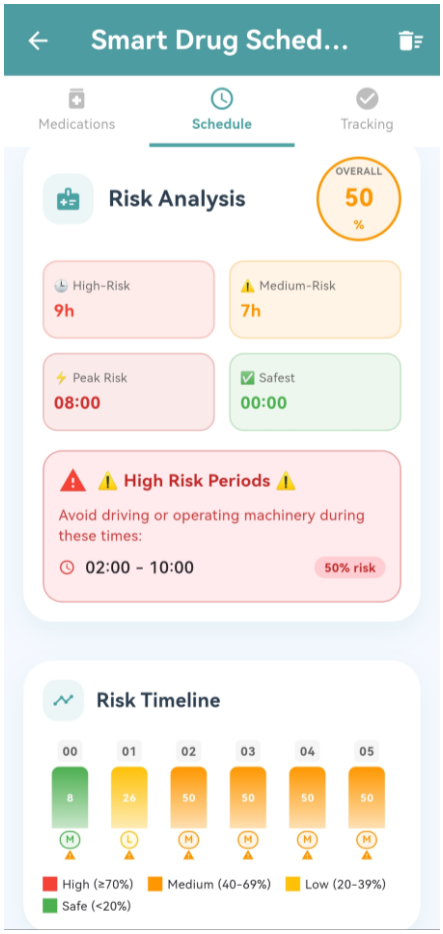


Figure 5-31: Risk Analysis Based on User-Generated Medication Schedule

Figure 5-31 illustrates the risk analysis results generated based on the user-defined medication schedule. The system provides an overall risk percentage to indicate the safety level of the schedule. In addition, detailed breakdowns are displayed, including high-risk hours, medium-risk hours, peak risk time, and safe time periods. High-risk periods are clearly highlighted to help users understand when potential drug interaction risks are most significant. A safety warning is also included, advising users to avoid driving or operating machinery during high-risk periods to prevent possible adverse effects.

Figure 5-32: 24-Hour Risk Analysis Timetable Based on Medication Schedule

Figure 5-32 illustrates the risk timetable generated for a full 24-hour period based on the user's medication schedule. The system performs hourly risk evaluation and visualizes risk levels across the day. Risk classification is divided into four categories: high risk (≥70%), medium risk (40–69%), low risk (20–39%), and safe risk (<20%). Each time slot is mapped according to the calculated interaction and scheduling conditions, allowing users to clearly observe fluctuations in risk intensity throughout the day. This visualization helps users identify safer time periods and avoid high-risk intervals when taking multiple medications.

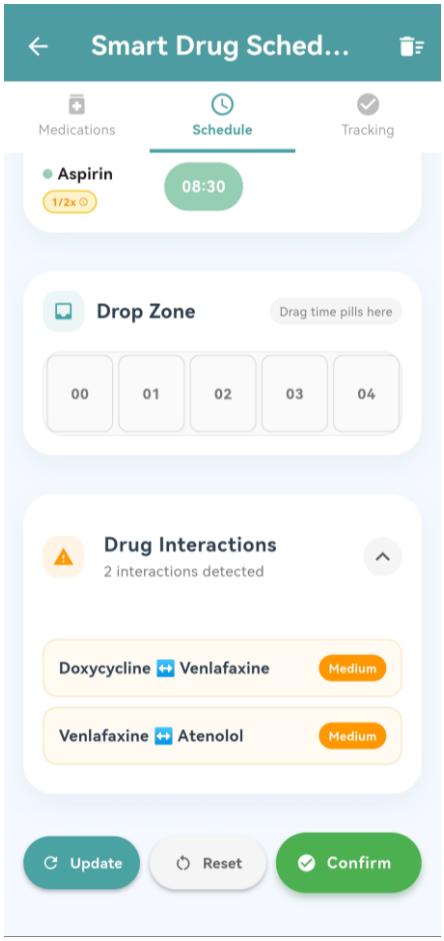


Figure 5-33: Schedule Control Actions (Update, Reset, and Confirm Functions)

Figure 5-33 illustrates the bottom control section of the scheduling interface, which includes three functional buttons: Update, Reset, and Confirm. The Update button allows users to refresh and regenerate the schedule based on the latest input or changes. The Reset button clears the current schedule and restores the default or initial state. The Confirm button finalizes the generated timetable and navigates the user to the Tracking tab, where the confirmed schedule is displayed for medication monitoring and intake tracking.

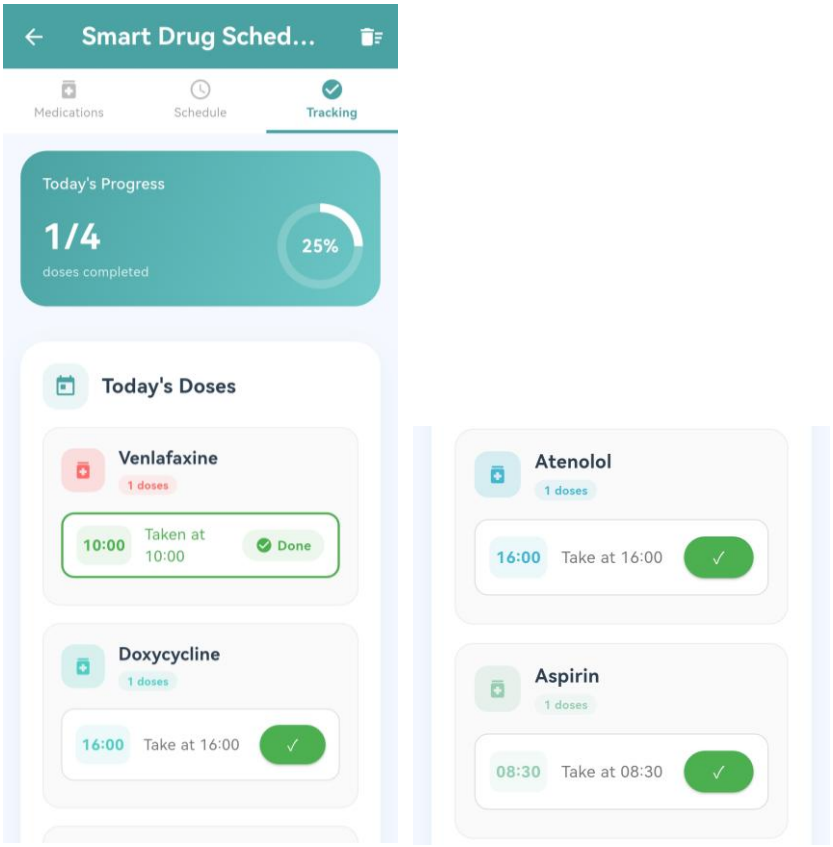


Figure 5-34: Medication Tracking Interface with Daily Progress Monitoring

Figure 5-34 illustrates the design of the Medication Tracking interface, which allows users to monitor their daily medication intake progress. The interface displays a list of scheduled medications for the current day, along with a progress tracking system. Users can mark each dose as completed by ticking the corresponding checkbox when the medication is taken. The progress indicator updates automatically based on completed doses, providing a clear overview of adherence to the prescribed schedule and supporting effective medication compliance.

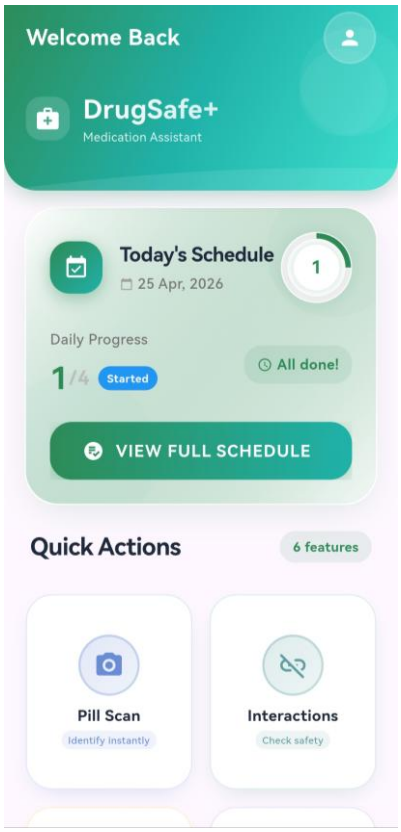


Figure 5-35: Main Page with Timetable Overview

Figure 5-35 illustrates the main page interface displaying the daily timetable overview of scheduled medications. The timetable provides a simplified view of the user's medication plan, allowing quick reference to all scheduled doses within the day. This design enables users to easily monitor their medication schedule directly from the main page.

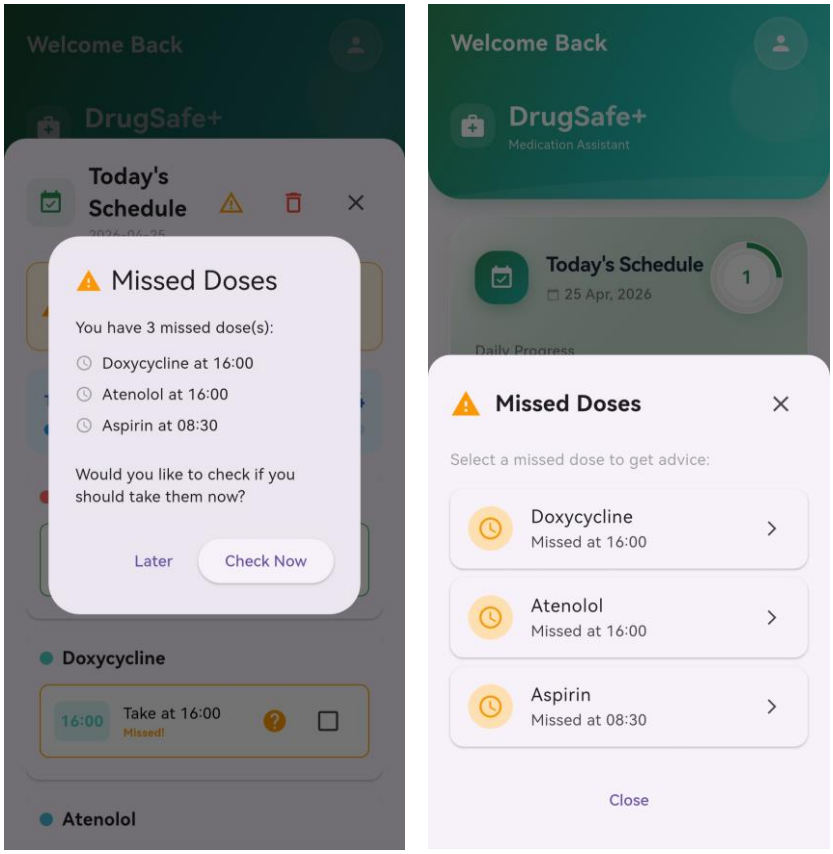


Figure 5-36: Missed Dose Detection and Automatic Warning with Navigation to Missed Dose Module

Figure 5-36 illustrates the system behavior when a scheduled medication dose is missed. When the user accesses the schedule view and the current time exceeds the planned medication time, the system automatically triggers a missed dose warning. A warning message is displayed to alert the user, indicating that the dose has not been taken on time. The interface also provides an option to navigate directly to the Missed Dose module, allowing the user to decide whether to take the missed medication immediately or review further actions. After selecting the "Check Now" option, the system prompts the user to select the specific medication that was missed. Once selected, the Missed Dose module is automatically populated with relevant information, including the drug name and the missed time, to assist the user in making an appropriate decision. This feature helps ensure timely intervention and improves medication adherence.

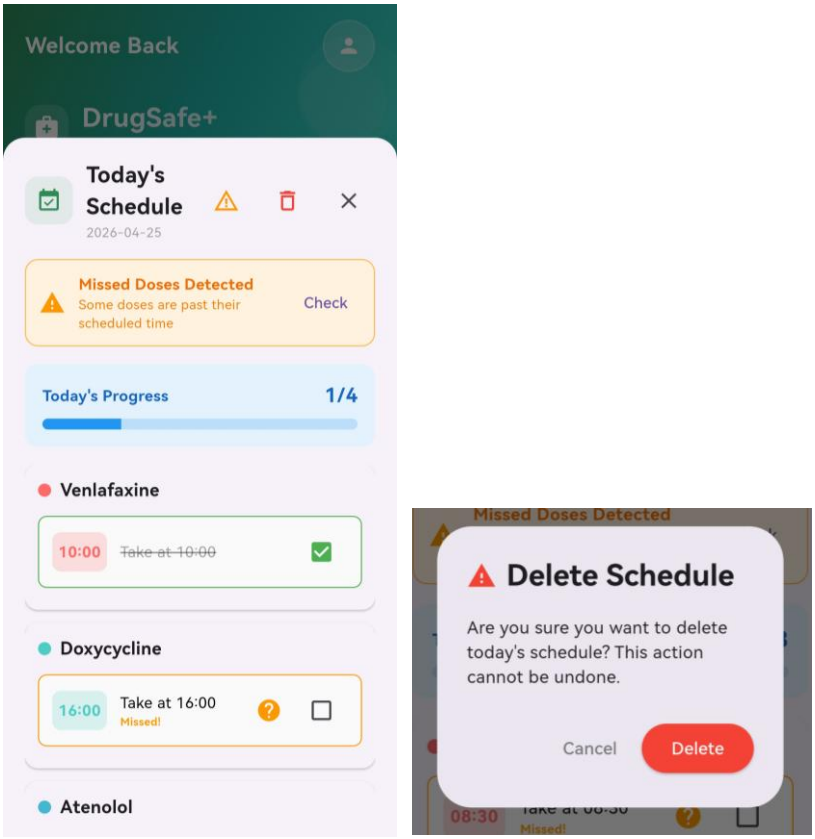


Figure 5-37: Main Page Daily Timetable Design

Figure 5-37 illustrates the main page interface displaying the daily medication timetable. The timetable presents all scheduled medications for a single day in a structured layout for easy viewing. A delete function is provided, allowing users to remove selected medication entries from the daily schedule when necessary. This design enables users to manage and update their medication plan efficiently while maintaining a clear overview of their daily schedule. In addition to automatic missed-dose detection and warning alerts, users can manually trigger the missed dose check by selecting the “Check” option within the yellow warning container on the timetable, allowing the system to verify and confirm any overdue or unconsumed medication doses.

5.3.7 Missed Dose Module

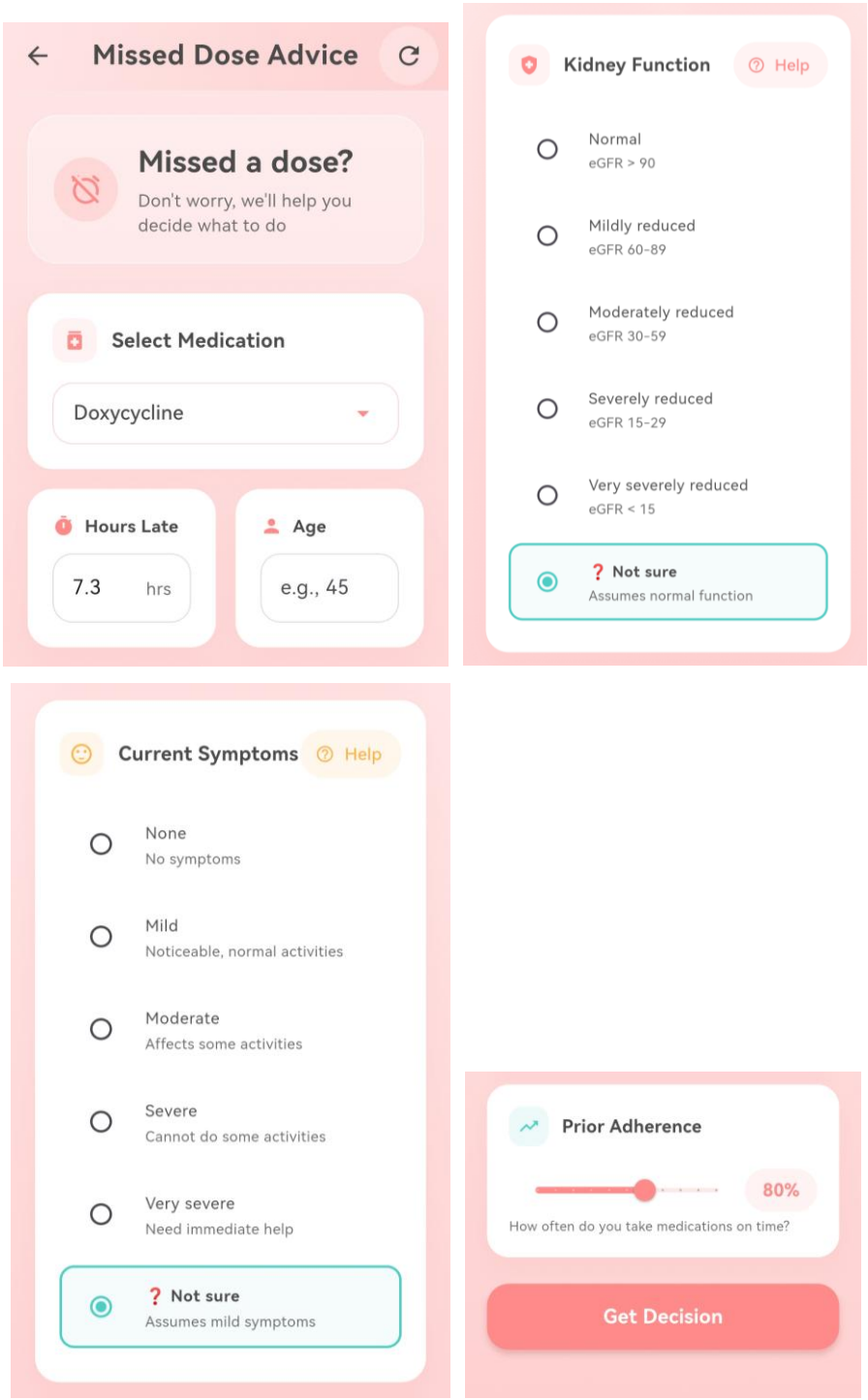


Figure 5-38: Missed Dose Decision Interface Design with Patient Input Parameters

Figure 5-38 illustrates the interface design of the Missed Dose Decision module, presented with a pink-themed layout. Users can access this interface from the timetable when a missed dose is detected, where the system automatically populates key information such as the drug name and hours late, calculated as the difference between the current time and the scheduled

time. Alternatively, users may enter missed dose information manually, in which case all required fields must be completed by the user.

The interface collects essential information to generate appropriate advice, including drug name, hours late, user age, and kidney function status based on eGFR values. Kidney function categories include normal (eGFR > 90), mildly reduced (eGFR 60–89), moderately reduced (eGFR 30–59), severely reduced (eGFR 15–29), and very severely reduced (eGFR < 15). If the user is unsure, the system assumes normal kidney function by default.

In addition, users are required to indicate current symptoms, categorized as none, mild, moderate, severe, or very severe, with an option for "not sure," which defaults to mild symptoms. The interface also includes a prior adherence input, allowing users to indicate how often they take medications on time using a percentage-based slider. This comprehensive data collection supports personalized and context-aware missed dose recommendations.

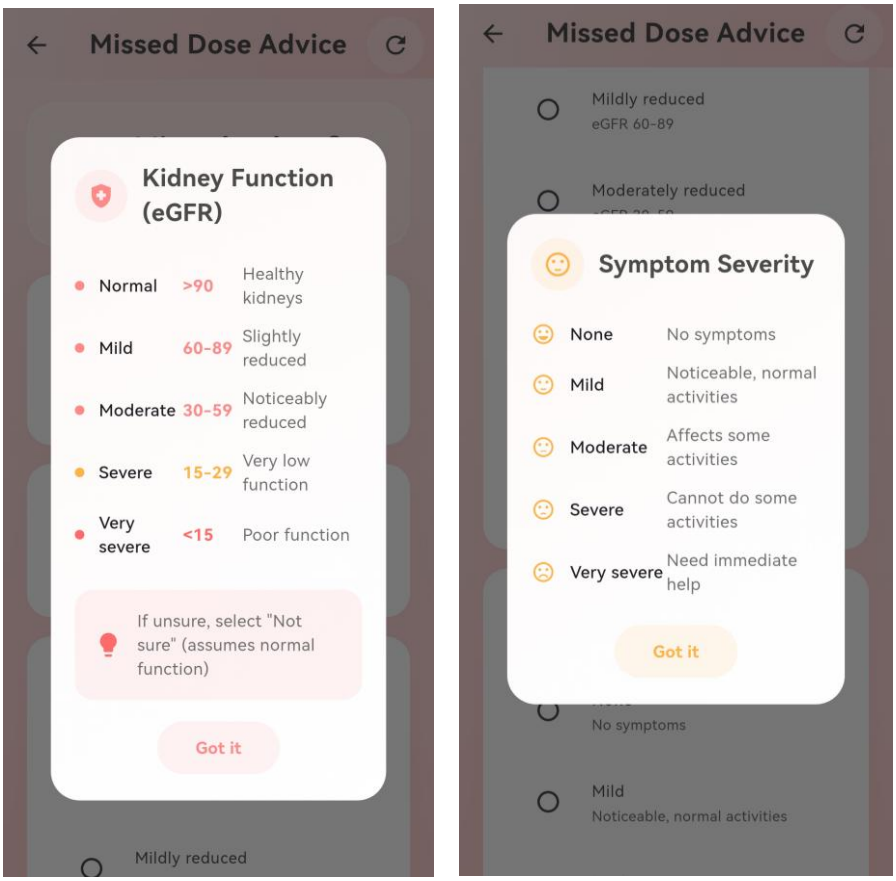


Figure 5-39: Help Function for Kidney Function and Symptom Level Interpretation

Figure 5-39 illustrates the help feature provided within the Missed Dose Decision interface. If users are unsure about the meaning of kidney function categories or symptom severity levels, they can select the help option available in the corresponding input sections. Upon activation, the system displays explanatory information for each category and level, enabling users to better understand the definitions and select the most appropriate option. This feature enhances usability and supports more accurate user input for personalized medication advice.

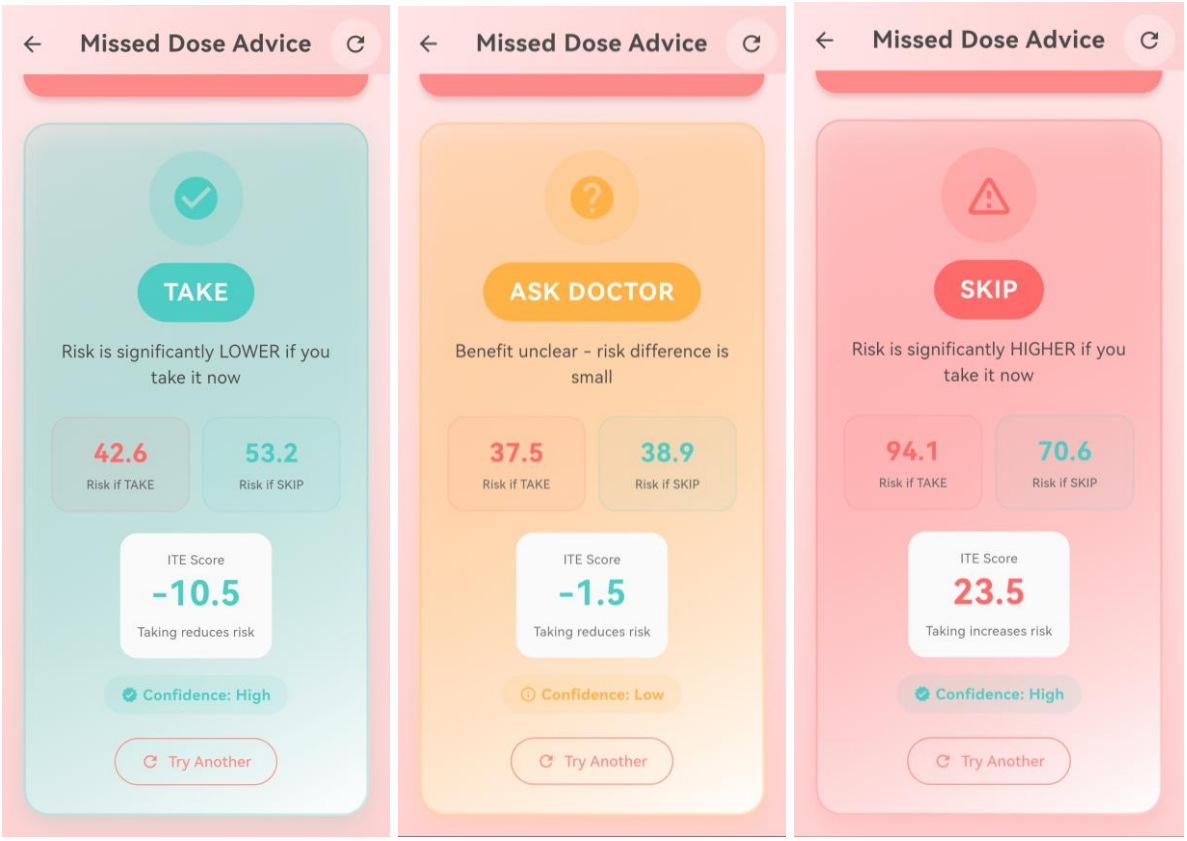


Figure 5-40: Missed Dose Decision Outcomes Based on User Input

Figure 5-40 illustrates the decision outcomes generated after the user selects the "Get Decision" function within the Missed Dose Decision module. Based on the user-provided information, the system presents one of three possible recommendations: Take Now, Consult a Doctor, or Skip Dose. The system processes the input data by transforming it into a set of derived features, which are then evaluated using a trained machine learning model to determine the most appropriate decision. This feature provides users with clear and actionable guidance to support safe decision-making when a dose is missed.

5.4 Implementation Issues and Challenges

Pill Image Recognition Module

The pill image recognition module faced significant challenges in achieving robust feature extraction under real-world conditions. Traditional computer vision methods were extensively tested but showed poor reliability. Colour-based features (RGB, HSV histograms) were highly sensitive to lighting variations, causing confusion between similar colours such as beige and yellow-orange. Shape descriptors (Hu moments, contour features) were unstable due to changes in scale, orientation, and distance. Texture and edge-based methods (GLCM, LBP, HOG, Canny) were affected by background noise and the inherently low-texture nature of pills. Keypoint-based methods (SIFT, SURF, ORB) also failed due to insufficient distinctive features.

Domain-specific techniques such as OCR (Tesseract, EasyOCR) were unreliable due to low-contrast pill imprints, while score line detection using Hough Transform was inconsistent for faint or occluded lines. Colour segmentation (k-means) and geometric features (circularity, size) were similarly unstable under varying lighting and perspective conditions. Machine learning classifiers (SVM, Random Forest, KNN, shallow neural networks, ensemble models) performed poorly due to weak handcrafted features. Integration with DrugBank API was also limited by access restrictions.

To overcome these issues, a CNN-based deep learning approach was adopted using 10 drug classes with ~1,000 images per class. Although accuracy improved, training was computationally expensive (6–7 hours per session without GPU), slowing development. In addition, public datasets lacked consistent metadata mapping, and the NDC dataset had structural inconsistencies, making integration difficult. Due to these limitations, the module was deprioritised in favour of more feasible components.

Smart Scheduling Module

The smart scheduling module was complex due to multiple interacting constraints, including drug–drug interactions, varying half-lives, meal timing requirements, and user schedules. Developing a safe scheduling algorithm required extensive rule design and iterative refinement, particularly when resolving conflicts between interaction gaps and user availability. Edge cases where required gaps exceeded waking hours required fallback strategies to balance safety and practicality, increasing implementation complexity and development time.

Missed Dose Module

The missed dose module required a T-learner model, but real patient data was unavailable due to privacy constraints. A synthetic dataset of 10,000 records was generated using clinically realistic ranges, including missed dose duration (0.5–48 hours), age (20–90), renal function (1–5), symptom severity (1–5), and adherence rate (0.5–1.0).

Although synthetic data is a limitation, the model was evaluated using PEHE and decision accuracy metrics, showing that it provides a reasonable approximation for missed dose decision support.

5.6 Concluding Remark

The DrugSafe+ system has been successfully implemented, with all core modules developed and integrated according to the proposed design. The system combines mobile application features, backend services, database management, and machine learning components to provide a comprehensive medication safety solution.

All key functions, including drug interaction checking, drug information retrieval, allergy management, smart scheduling, and missed dose decision support, were implemented and tested within the development environment. The system demonstrated stable performance, with responsive API communication and consistent cross-module operation on the Android platform.

Although limitations exist in the pill image recognition module due to the use of a limited training dataset, the model may still produce similarity-based predictions for unseen classes, which can lead to potential misclassification. Despite this, the overall system satisfies the defined project objectives and functional requirements.

The system is now ready for further evaluation. The next chapter presents the testing methodology, performance analysis, and discussion of results to assess the effectiveness and reliability of the proposed solution.

Chapter 6

System Evaluation and Discussion

6.1 System Testing and Performance Metrics

6.1.1 CNN-Based Pill Image Classification (Training Phase)

This section describes the training phase of the Convolutional Neural Network (CNN) classification for the pill image recognition module.

Dataset Overview

The model was trained using a supervised dataset of 10 drug classes with a total of 10,000 images, as shown in Figure 6-1. Each class contains 1,000 images, ensuring balanced class distribution. The dataset includes variations in lighting, background, angle, and position to reflect real-world conditions. It was split into 80% training (8,000 images) and 20% validation (2,000 images), as shown in Figure 6-2. Data augmentation was applied to improve robustness and generalization.

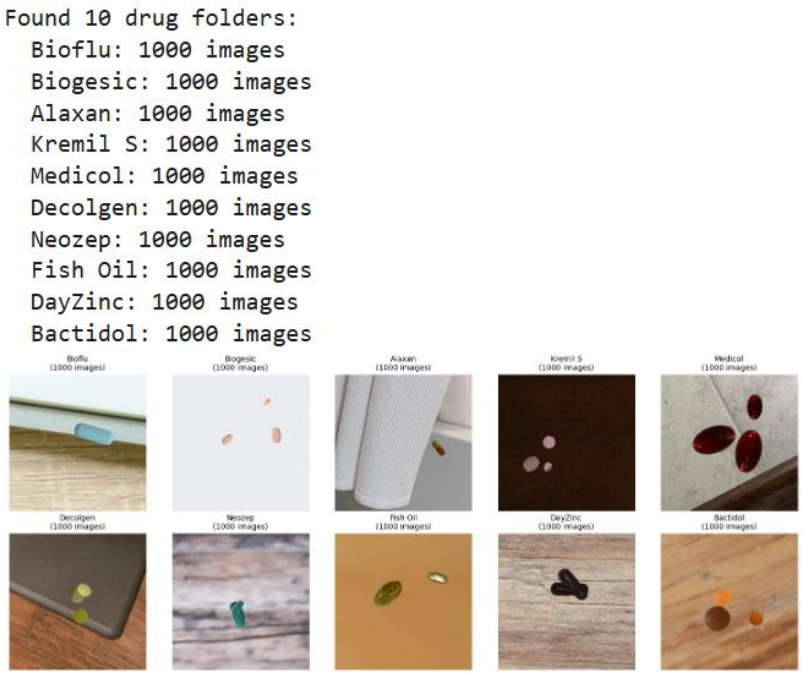


Figure 6-1: Dataset Distribution for Pill Image Recognition

```
🔧 Setting up data with advanced augmentations...
Dataset split:
Training: 8000 images
Validation: 2000 images
Data loaders created (batch size: 32)
```

Figure 6-2: Training and Validation Dataset Split

Model Configuration

The CNN model used a ResNet18 architecture based on transfer learning, as shown in Figure 6-3. The pre-trained model was fine-tuned for 10-class pill classification. The model contains approximately 11,445,322 parameters, with 9,252,874 parameters being trainable.

Training was performed using a CPU-based environment in Google Colab, which increased training duration but ensured stable execution.

```
Downloading: "https://download.pytorch.org/models/resnet18-f37072fd.pth" to /root/.cache/torch/hub/checkpoints/resnet18-
f37072fd.pth
100%|██████████| 44.7M/44.7M [00:00<00:00, 163MB/s]
Model created: ResNet18 (pretrained)
   Classes: 10
   Total params: 11,445,322
   Trainable params: 9,252,874 (80.8%)
```

Figure 6-3: ResNet18 Model Architecture for Pill Classification

Training Performance

The training process of the Convolutional Neural Network (CNN) model is illustrated in Figures 6-4 to 6-8. The model was trained with a maximum of 20 epochs, but early stopping was triggered after 15 epochs when validation performance did not improve for 5 consecutive epochs. Early stopping was implemented to prevent overfitting and to automatically terminate training when no further improvement in validation performance was observed.

During the initial training stage, the model demonstrated rapid learning, where the validation accuracy increased significantly from 94.65% as shown in Figure 6.4 in the first epoch. This indicates that the pre-trained weights provided a strong initial feature representation for pill image classification.

```
Epoch 1/20 [Train]: 100%|██████████| 250/250 [19:42<00:00,  4.73s/it, Loss=0.1600, Acc=86.6%]
💾 NEW BEST: 94.65% (saved to Google Drive)

📊 Epoch 1/20 (1373.7s)
  Train: Loss=0.4126, Acc=86.69%
  Val:   Loss=0.1572, Acc=94.65%
  Best:  Acc=94.65% (epoch 1)
  LR:    0.000488
```

Figure 6-4: Training Performance at Early Epoch

As training progressed, continuous improvement was observed across subsequent epochs, as shown in Figures 6-5 and 6-6. The validation accuracy steadily increased while the loss values decreased, indicating improved model convergence and better generalisation to unseen data.

```
Epoch 2/20 [Train]: 100%|██████████| 250/250 [18:30<00:00,  4.44s/it, Loss=0.3650, Acc=95.4%]
💾 NEW BEST: 96.15% (saved to Google Drive)

📊 Epoch 2/20 (1286.9s)
  Train: Loss=0.1494, Acc=95.30%
  Val:   Loss=0.1234, Acc=96.15%
  Best:  Acc=96.15% (epoch 2)
  LR:    0.000452
------------------------------------------------------------
Epoch 3/20 [Train]: 100%|██████████| 250/250 [18:07<00:00,  4.35s/it, Loss=0.0817, Acc=97.3%]
💾 NEW BEST: 96.20% (saved to Google Drive)

📊 Epoch 3/20 (1258.2s)
  Train: Loss=0.0848, Acc=97.31%
  Val:   Loss=0.1164, Acc=96.20%
  Best:  Acc=96.20% (epoch 3)
  LR:    0.000397
------------------------------------------------------------
Epoch 4/20 [Train]: 100%|██████████| 250/250 [18:28<00:00,  4.43s/it, Loss=0.0086, Acc=98.4%]
💾 NEW BEST: 96.90% (saved to Google Drive)

📊 Epoch 4/20 (1281.4s)
  Train: Loss=0.0466, Acc=98.49%
  Val:   Loss=0.0975, Acc=96.90%
  Best:  Acc=96.90% (epoch 4)
  LR:    0.000328
------------------------------------------------------------
Epoch 5/20 [Train]: 100%|██████████| 250/250 [18:28<00:00,  4.43s/it, Loss=0.0010, Acc=99.4%]

📊 Epoch 5/20 (1283.5s)
  Train: Loss=0.0212, Acc=99.39%
  Val:   Loss=0.1071, Acc=96.70%
  Best:  Acc=96.90% (epoch 4)
  LR:    0.000251
```

Figure 6-5: Training Performance Improvement

```
Epoch 6/20 [Train]: 100%|██████████| 250/250 [18:34<00:00,  4.46s/it, Loss=0.0016, Acc=99.6%]
💾 NEW BEST: 97.50% (saved to Google Drive)

📊 Epoch 6/20 (1287.8s)
  Train: Loss=0.0164, Acc=99.62%
  Val:   Loss=0.0977, Acc=97.50%
  Best:  Acc=97.50% (epoch 6)
  LR:    0.000173
------------------------------------------------------------
Epoch 7/20 [Train]: 100%|██████████| 250/250 [18:16<00:00,  4.38s/it, Loss=0.0119, Acc=99.8%]
💾 NEW BEST: 97.75% (saved to Google Drive)

📊 Epoch 7/20 (1269.2s)
  Train: Loss=0.0082, Acc=99.79%
  Val:   Loss=0.0795, Acc=97.75%
  Best:  Acc=97.75% (epoch 7)
  LR:    0.000104
------------------------------------------------------------
Epoch 8/20 [Train]: 100%|██████████| 250/250 [18:14<00:00,  4.38s/it, Loss=0.0014, Acc=99.9%]

📊 Epoch 8/20 (1270.3s)
  Train: Loss=0.0045, Acc=99.88%
  Val:   Loss=0.0713, Acc=97.65%
  Best:  Acc=97.75% (epoch 7)
  LR:    0.000049
------------------------------------------------------------
Epoch 9/20 [Train]: 100%|██████████| 250/250 [18:32<00:00,  4.45s/it, Loss=0.0012, Acc=99.9%]
💾 NEW BEST: 98.05% (saved to Google Drive)

📊 Epoch 9/20 (1288.0s)
  Train: Loss=0.0026, Acc=99.94%
  Val:   Loss=0.0676, Acc=98.05%
  Best:  Acc=98.05% (epoch 9)
  LR:    0.000013
```

Figure 6-6: Training Accuracy and Loss Trend

The model achieved its best performance at a validation accuracy of 98.10% in Figure 6-7, demonstrating strong classification capability across the 10 pill categories. After reaching this peak performance, no further significant improvement was observed in subsequent epochs.

```
Epoch 10/20 [Train]: 100%|██████████| 250/250 [18:19<00:00,  4.40s/it, Loss=0.0002, Acc=99.9%]
💾 NEW BEST: 98.10% (saved to Google Drive)

📊 Epoch 10/20 (1275.8s)
  Train: Loss=0.0027, Acc=99.94%
  Val:   Loss=0.0719, Acc=98.10%
  Best:  Acc=98.10% (epoch 10)
  LR:    0.000500
```

Figure 6-7: Best Model Validation Accuracy

Early stopping was triggered when the validation performance did not improve for a predefined number of epochs, as shown in Figure 6-8. This mechanism effectively prevented

unnecessary training and reduced the risk of overfitting, ensuring that the final model retained optimal generalisation performance.

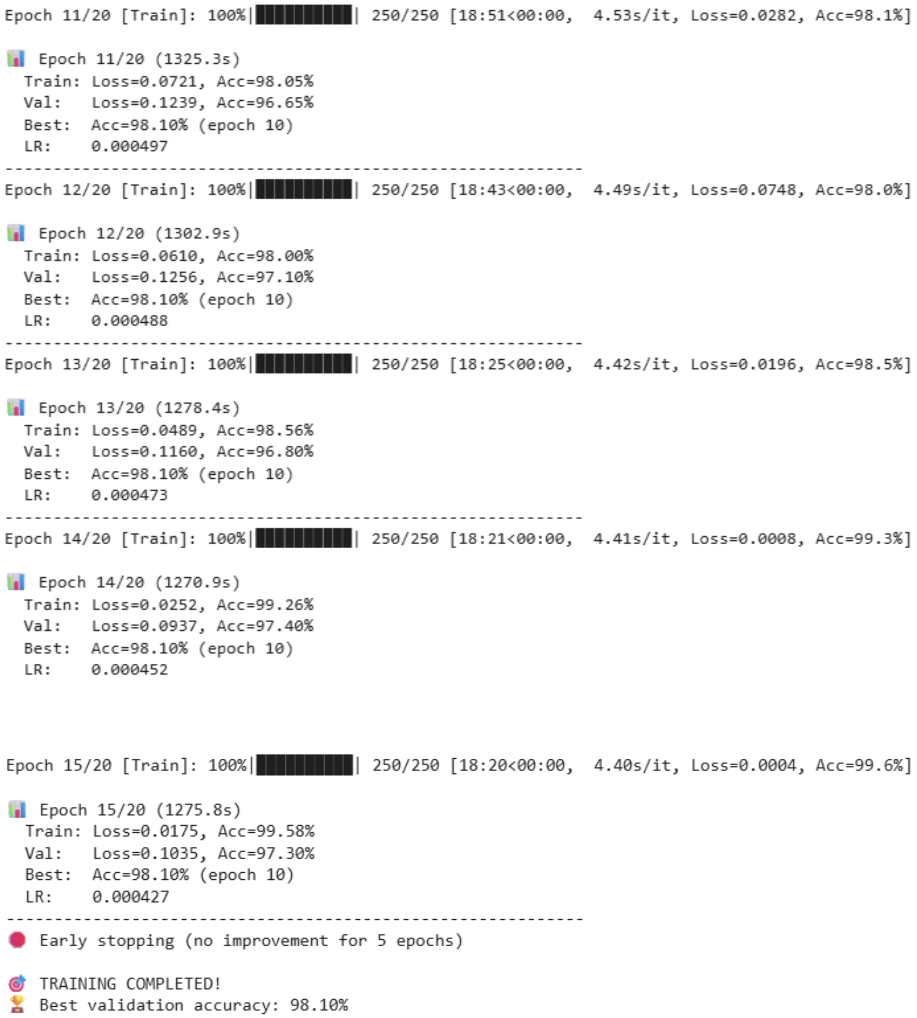


Figure 6-8: Early Stopping During Training

Performance Summary

The final performance of the CNN model is summarised in Table 6-1, where the system achieved a best validation accuracy of 98.10%, demonstrating strong classification capability for the trained dataset.

Performance Metric	Value
Model Architecture	ResNet18 (Transfer Learning)
Number of Classes	10 drug categories
Total Dataset Size	10,000 images
Training Set Size	8,000 images (80%)
Validation Set Size	2,000 images (20%)
Maximum Epochs Configured	20 epochs

Actual Training Epochs	15 epochs (Early stopping applied)
Early Stopping Patience	5 epochs
Optimizer	AdamW
Loss Function	Cross Entropy Loss
Best Validation Accuracy	98.10%
Final Training Accuracy	99.94%
Final Validation Loss	~0.07
Training Environment	Google Colab (CPU-based)

Table 6-1: Performance Summary of CNN Pill Image Recognition Model

6.1.2 CNN Embedding-Based Pill Image Representation (Training Phase)

This section describes the feature extraction and embedding-based representation approach used in the pill image recognition system. Unlike traditional classification, this method transforms each pill image into a high-dimensional feature vector using a pre-trained deep learning model.

Dataset Overview

The embedding system was developed using a supervised dataset of 118 pill images from a self-constructed structured pharmaceutical dataset, as shown in Figure 6-9. Due to the lack of suitable publicly available datasets with required format and annotation quality, the dataset was manually curated.

Each image represents a single pill instance and is linked with detailed metadata, including drug name, imprint, colour, shape, strength, and viewing side. The dataset includes variations in lighting, background, orientation, and viewing angle to simulate real-world conditions, improving the robustness of extracted embeddings and supporting reliable similarity-based matching.

```
Data shape: (118, 13)
Columns (13):
 1. drug_name
 2. Imprint
 3. Color
 4. Shape
 5. images
 6. pilltype_id
 7. label_code_id
 8. prod_code_id
 9. is_ref
10. is_front
11. is_new
12. image_path
13. label

Found 118 images in /content/pill_images/

Sample image files (first 5):
1. 0007-3650_0_1.jpg
2. 0007-4895_0_1.jpg
3. 0054-0097_0_1.jpg
4. 0049-2340_0_1.jpg
5. 0006-0575_0_1.jpg
```

Figure 6-9: Dataset Loading and CSV Structure for CNN Embedding System

Model Configuration

The embedding system utilises a ResNet50-based convolutional neural network as a feature extractor to generate 2048-dimensional feature embeddings, as shown in Figure 6-10. The pre-trained ResNet50 model is employed with its final fully connected classification layer removed, enabling the extraction of deep visual features instead of class predictions.

This configuration allows the model to capture essential visual characteristics such as pill texture, shape boundaries, imprint patterns, and colour distribution, which are critical for distinguishing visually similar pills.

```
Downloading: "https://download.pytorch.org/models/resnet50-0676ba61
100%|██████████| 97.8M/97.8M [00:01<00:00, 84.2MB/s]

PyTorch model created successfully!
Input size: 224x224 RGB
Output dimension: 2048 (ResNet50 feature dimension)
Testing with: 0002-3270_0_0.jpg
Test successful!
Embedding shape: (2048,)
Embedding norm: 1.000000
Embedding range: [0.000000, 0.139031]
```

Figure 6-10: ResNet50 Embedding Model Initialization in PyTorch Environment

Embedding Extraction Process

Each pill image is resized to 224 × 224 pixels and passed through the ResNet50 feature extractor to generate a fixed-length embedding vector, as shown in Figure 6-11. The resulting embeddings are L2-normalised to ensure a consistent scale, which is essential for accurate similarity comparison.

A total of 118 images were successfully processed, and each image was converted into a 2048-dimensional feature vector without any processing failure. This confirms the stability and reliability of the embedding extraction pipeline.

```
Extracting embeddings for 118 images...
Processing images: 100%|██████████| 8/8 [00:25<00:00,  3.21s/it]

RESULTS:
Successfully processed: 118 images
Failed: 0 images

Embedding extraction complete!
Embeddings shape: (118, 2048)
Each image represented by 2048-dimensional vector
Embedding dtype: float32
```

Figure 6-11: Embedding Extraction Process for All Pill Images

Database Construction and FAISS Indexing

After feature extraction, all embeddings were stored in a structured database together with their corresponding metadata attributes, including drug name, imprint, colour, shape, strength, and viewing side. To enable efficient similarity search, a FAISS (Facebook AI Similarity Search) index was constructed using the IndexFlatIP method with cosine similarity, as shown in Figure 6-12.

Furthermore, the complete embedding database structure, including metadata attributes and embedding index mapping, is illustrated in Figure 6-13. This structure establishes a direct linkage between each feature vector and its associated pill information, enabling accurate and efficient retrieval of visually similar pills.

```
Creating FAISS index for cosine similarity...
 3. Saved FAISS index: /content/pill_database_pytorch/faiss_index.bin
Index type: IndexFlatIP (cosine similarity)
Vectors in index: 118
 4. Saved config: /content/pill_database_pytorch/config.json
```

Figure 6-12: FAISS Index Construction and Database Creation Output

```
FINAL METADATA COLUMNS:
 1. pill_id
 2. image_filename
 3. full_path
 4. embedding_index
 5. drug_name
 6. imprint
 7. color
 8. shape
 9. strength
10. side
11. ndc
```

```
SAMPLE DRUG INFORMATION (first 3 pills):

Pill 1:
    Drug: Cymbalta
    Imprint: Lilly 3270 60mg
    Color: Blue / Green
    Shape: Capsule/Oblong
    Strength: 60mg

Pill 2:
    Drug: Cymbalta
    Imprint: Lilly 3270 60mg
    Color: Blue / Green
    Shape: Capsule/Oblong
    Strength: 60mg

Pill 3:
    Drug: Cialis
    Imprint: C 5
    Color: Yellow
    Shape: Oval
```

Figure 6-13: Final Embedding Database with Metadata Structure

Performance Summary

The embedding system successfully processed all 118 pill images and generated 2048-dimensional feature vectors for each sample. The FAISS index was successfully constructed and integrated with the metadata database, enabling efficient similarity-based retrieval of pill images. The results demonstrate that the embedding-based approach provides a robust and scalable solution for pill identification in real-world scenarios.

Performance Metric	**Value**
Model Architecture	ResNet50 (Feature Extractor)
Embedding Dimension	2048
Total Dataset Size	118 images
Unique Pill Instances	62
Unique Drug Names	59

Image Input Size	224 × 224 RGB
Feature Extraction Success Rate	100% (118/118 images)
Failed Processing	0 images
Embedding Data Type	float32
Embedding Normalisation	L2 Normalisation
Similarity Metric	Cosine Similarity
FAISS Index Type	IndexFlatIP
Total Vectors in Index	118
Database Storage	NumPy (.npy) + CSV Metadata
Metadata Attributes	drug_name, imprint, colour, shape, strength, side, ndc
Average Images per Pill	1.90
Execution Environment	Google Colab (CPU-based)

Table 6-2: Performance Summary of CNN Embedding-Based Pill Image Representation System

6.1.3 Performance Evaluation of Missed Dose Prediction Model

The missed-dose prediction system was evaluated using a T-learner causal inference model trained on 10,000 simulated patient records. The model estimates individualized treatment effects (ITE) by comparing predicted outcomes under both medication intake and non-intake scenarios, as shown in Figure 6-14. The model performance was further assessed using multiple evaluation metrics including PEHE, decision accuracy, ATE error, and risk reduction compared to a rule-based method. As shown in Figure 6-16, the model achieved a PEHE of 7.18 and a decision accuracy of 63.7%, demonstrating moderate performance in treatment decision support.

```
PEHE (lower is better): 7.18

Decision Accuracy (clear cases only): 63.7%
Clear cases: 1135 out of 2000

ATE Error: 0.68
True ATE: -0.22
Pred ATE: -0.90

Risk Reduction vs Rule-based: 5.2%
Rule-based avg risk: 46.98
```

Figure 6-14: Missed Dose Prediction Model Performance Evaluation Results

Reason for Moderate Accuracy

The moderate accuracy of 63.7% is attributable to three main factors. First, synthetic training data lacks the complexity of real-world clinical records. Second, ITE estimation is fundamentally challenging because only one potential outcome can ever be observed per patient. Third, intentional noise added during data generation simulates real-world unpredictability. Despite this, the PEHE of 7.18 confirms reasonable treatment effect estimation, while the 5.2% risk reduction demonstrates practical clinical value. The "ASK DOCTOR" option further mitigates low-confidence predictions by deferring to medical professionals when uncertainty is high.

6.2 Testing Setup and Result

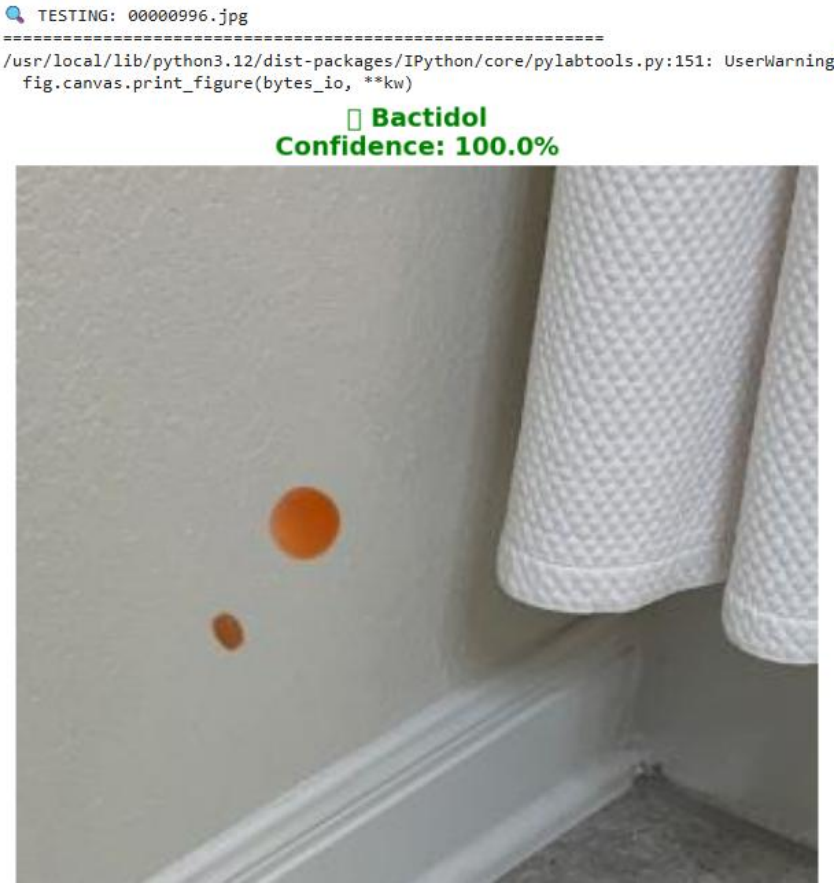
6.2.1 Testing for CNN Pill Image Classification

The testing phase of the CNN pill image recognition module was conducted to evaluate the performance of the trained ResNet18 model in a controlled inference environment. The model was deployed in Google Colab for evaluation.

Testing was performed using locally stored pill images uploaded to simulate real-world input conditions. Each image was processed by the CNN model to generate a predicted class label and corresponding confidence score.

The results show that the model produced highly accurate predictions under controlled conditions. For example, Bactidol was correctly identified with 100% confidence as shown in Figure 6-15, and Decolgen was also correctly classified with 100% confidence as shown in Figure 6-16.

Overall, the CNN model demonstrates strong and reliable classification performance on the tested samples within the trained dataset.

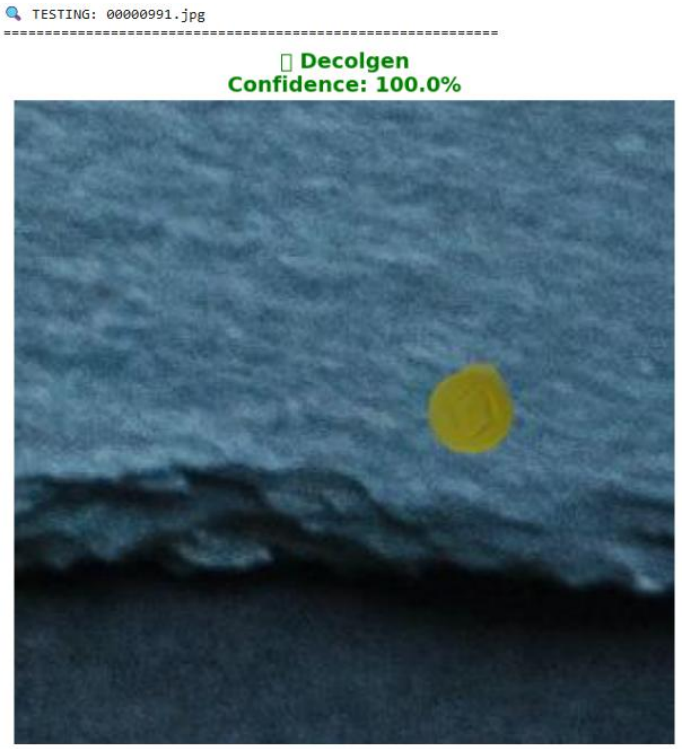


```
Source: YOUR_AI_MODEL
Status: HIGH_CONFIDENCE_MATCH
Pill: Bactidol
Confidence: 100.0%

 Message:  This is Bactidol (100.0% confidence)

 All AI Predictions:
  1. Bactidol: 100.0%
  2. Medicol: 0.0%
  3. Alaxan: 0.0%
```

Figure 6-15: Bactidol Pill Classification Result Using CNN Model




```
 IDENTIFICATION RESULTS:
Source: YOUR_AI_MODEL
Status: HIGH_CONFIDENCE_MATCH
Pill: Decolgen
Confidence: 100.0%

 Message:  This is Decolgen (100.0% confidence)

 All AI Predictions:
  1. Decolgen: 100.0%
  2. Fish Oil: 0.0%
  3. Bioflu: 0.0%
```

Figure 6-16: Decolgen Pill Classification Result Using CNN Model

6.2.2 Testing for CNN Embedding-Based Retrieval

The testing phase of the CNN embedding-based pill retrieval system was conducted using a trained ResNet50 feature extractor integrated with a FAISS similarity search index. The system was evaluated in a simulated inference environment using Google Colab, where a real pill image was uploaded to perform feature extraction and similarity matching against the existing database, as shown in Figure 6-17.

During this stage, the system first processed the uploaded pill image and displayed the image upload information, including file path, image size, format, and preprocessing status. This confirms that the input image was successfully received and prepared for feature extraction.

The system then extracted a 2048-dimensional embedding vector from the input image and performed similarity matching against the stored embedding database using cosine similarity. The retrieval results showing the top 5 most similar pill images, along with their confidence scores and drug information, are presented in Figure 6-18.

To further validate system consistency, a second test was conducted using a different input pill image. The system again successfully generated embeddings and returned accurate similarity-based results, demonstrating stable performance across different inputs. The output of this second test case is shown in Figure 6-19.

Overall, the results show that the system is capable of accurate and consistent similarity-based retrieval. High confidence scores in both test cases indicate strong feature representation and effective discrimination ability of the ResNet50 embedding model.

```
Uploaded: 0032-1224_0_0.jpg
Path: /content/0032-1224_0_0.jpg

Image size: (224, 224)
Image mode: RGB
Image format: JPEG

Loading database and searching for similar pills...
Loaded PyTorch database with 118 pill images
Unique drugs: 62

DRUG INFORMATION IN DATABASE:
• Drug Names: 59 unique
• Shapes: 8 unique
• Colors: 22 unique

Extracting features from your image using PyTorch...
Using device: cpu
PyTorch model created successfully!
Input size: 224x224 RGB
Output dimension: 2048 (ResNet50 feature dimension)
Features extracted successfully
Embedding shape: (2048,)
Embedding norm: 1.000000 (should be ~1.0)

SEARCH COMPLETE!
Found 10 potential matches
Best match score: 1.0000
High confidence matches: 10 (score ≥ 0.7)
```

Figure 6-17: Input Image Upload and Preprocessing Information

```
MATCH #1 (Confidence: 1.0000):
--------------------------------
DRUG NAME: Creon
IMPRINT: CREON 1224
SHAPE: Capsule/Oblong
COLOR: Orange / Clear
SIDE: front
Image: 0032-1224_0_0.jpg

MATCH #2 (Confidence: 0.9504):
--------------------------------
DRUG NAME: Creon
IMPRINT: CREON 1224
SHAPE: Capsule/Oblong
COLOR: Orange / Clear
SIDE: back
Image: 0032-1224_0_1.jpg

MATCH #3 (Confidence: 0.8910):
--------------------------------
DRUG NAME: Effexor XR
IMPRINT: W Effexor XR 150
SHAPE: Capsule/Oblong
COLOR: Orange
SIDE: front
Image: 0008-0836_0_0.jpg

MATCH #4 (Confidence: 0.8501):
--------------------------------
DRUG NAME: Cymbalta
IMPRINT: Lilly 3270 60mg
SHAPE: Capsule/Oblong
COLOR: Blue / Green
STRENGTH: 60mg
SIDE: front
Image: 0002-3270_0_0.jpg

MATCH #5 (Confidence: 0.8475):
--------------------------------
DRUG NAME: Dyazide
IMPRINT: DYAZIDE SB DYAZIDE S...
SHAPE: Capsule/Oblong
COLOR: Red & White
SIDE: back
Image: 0007-3650_0_1.jpg
```

Figure 6-18: Top-5 Similarity Search Results (FAISS Output)

Figure 6-19: Second Test Retrieval Results

6.2.3 Testing for Missed Dose Decision Module

The testing phase of the missed-dose decision module was conducted in a simulated environment using a separate test set of 2,000 patient records. During inference, the model predicts outcomes under both treatment and control conditions to compute individualized treatment effects (ITE) for each patient. The prediction results are presented in Figure 6-20. The results demonstrate that the system is capable of generating stable predictions and identifying variations in treatment response across patients.

```
Predicted ITE range: -31.32 to 37.19
Predicted ITE mean: -0.90
```

Figure 6-20: Missed Dose Model Prediction Inference Results

6.3 Project Challenges

The project faced several key challenges that affected system scope and implementation feasibility. A major limitation was the lack of access to real clinical patient data due to privacy and ethical constraints, requiring the use of simulated datasets for training and evaluation, which may reduce real-world generalisability.

Computational limitations also impacted development, as CPU-based processing and Google Colab environments led to longer training times and restricted scalability for large datasets and complex models.

In addition, the system remains a prototype and has not been deployed in a production environment. Therefore, aspects such as security, robustness, optimisation, and large-scale user handling were not addressed.

These challenges highlight the gap between prototype development and real-world clinical deployment.

6.4 Objectives Evaluation

This section evaluates the extent to which each objective defined in Chapter 1 has been achieved. Each objective is restated, followed by its achievement status and a brief evaluation based on system functionality.

Objective 1: Enable accurate identification of medications using pill images
Status: Achieved
A pill image recognition module has been successfully developed. The system is capable of analysing visual features such as shape, colour, and imprint to generate matching results. The module functions as intended and supports users in identifying medications under testing conditions.

Objective 2: Detect and assess potential drug–drug interactions
Status: Achieved
A drug–drug interaction checking module has been implemented to evaluate medication combinations and classify interaction severity. The system provides warnings to highlight potential risks and supports safer medication usage.

Objective 3: Provide centralized access to comprehensive drug information
Status: Achieved
A drug information module has been established to deliver structured and accessible medication details. Users are able to retrieve relevant information efficiently, improving understanding of medication usage and precautions.

Objective 4: Support management and monitoring of medication allergies

Status: Achieved

An allergy management module has been developed to allow users to record known allergies and view active ingredients, add personal notes, and access potential cross-reactivity information. The module improves awareness of allergy-related risks and reduces the likelihood of adverse reactions.

Objective 5: Generate optimized medication schedules based on safety constraints

Status: Achieved

A missed dose decision support module has been developed to assist users in deciding whether to take or skip a missed dose. The system evaluates multiple parameters, including timing conditions and safety considerations, to support user decision-making.

Objective 6: Provide decision support for handling missed medication doses

Status: Achieved

A missed dose decision support module has been developed to guide users in handling missed doses. The system provides recommendations based on timing conditions, assisting users in making safer decisions.

Overall Conclusion

All objectives have been successfully achieved through the implementation and functional testing of the respective modules. The developed system demonstrates its capability as a context-aware mobile medication management platform that integrates multiple medication-related functions into a unified system to enhance medication safety and support informed decision-making.

6.5 Concluding Remark

This chapter presented the evaluation and testing of all major modules in the proposed system, including pill image recognition, embedding-based retrieval, and missed-dose decision support.

Overall, the system demonstrates effective performance under controlled conditions. The CNN models achieved strong classification and feature extraction results, while the FAISS retrieval system provided reliable similarity matching. The missed-dose module also showed consistent predictive behaviour.

Although evaluated using simulated data and prototype-level deployment, the results indicate that the proposed system is feasible and functionally effective. Identified limitations, such as data constraints and computational limitations, provide directions for future improvement and real-world deployment.

Chapter 7
Conclusion and Recommendation

7.1 Conclusion

This project successfully developed a context-aware mobile medication management system that integrates multiple medication-related functions into a unified platform. It addresses limitations of existing systems that provide fragmented services by combining drug reference, interaction checking, allergy management, scheduling, and adherence monitoring into one application.

The system includes modules for pill image recognition, drug–drug interaction checking, allergy management, smart medication scheduling, and missed-dose decision support. These modules support users across the full medication lifecycle, from identification to safety checking and adherence management, with consistent data flow and integrated system design.

From a technical perspective, the project demonstrates the integration of deep learning and data-driven decision support in healthcare applications. The image recognition and embedding-based retrieval modules show strong performance, while the missed-dose model demonstrates the feasibility of causal inference for personalised decision support. Rule-based components further enhance medication safety through structured risk assessment.

Overall, the system meets its main objective of developing an integrated and intelligent medication management platform. Although evaluated in a simulated environment, the results indicate that the approach is effective and provides a solid foundation for future real-world deployment.

7.2 Recommendation

While the current system performs well as a prototype, several improvements are recommended for future development.

Firstly, larger and more diverse real-world datasets should be used, especially for pill recognition and missed-dose prediction. Access to clinical data, with proper ethical approval and anonymisation, would improve model generalisation and reliability.

Secondly, deployment in a cloud-based production environment should be considered to support scalability, real-time processing, and a larger user base. Security improvements such as encryption and secure authentication are also important for real-world use.

Thirdly, the pill recognition module can be improved using more advanced deep learning models and expanded datasets covering more medication types to increase classification accuracy.

In addition, future enhancements such as multilingual support, voice interaction, and wearable device integration can improve accessibility, especially for elderly users.

Finally, continuous learning strategies such as periodic retraining and feedback-based updates can help maintain system accuracy over time.

Overall, the system provides a strong foundation for context-aware medication management, with clear pathways for future improvement toward clinical application.

REFERENCES

[1] S. García-Sánchez, B. Somoza-Fernández, A. de Lorenzo-Pinto, C. Ortega-Navarro, A. Herranz-Alonso, and M. Sanjurjo, "Mobile health apps providing information on drugs for adult emergency care: Systematic search on app stores and content analysis," *JMIR Mhealth Uhealth*, vol. 10, no. 4, p. e29985, 2022. [Online]. Available: https://mhealth.jmir.org/2022/4/e29985

[2] B. Y. Kim, A. Sharafoddini, N. Tran, E. Y. Wen, and J. Lee, "Consumer Mobile Apps for Potential Drug-Drug Interaction Check: Systematic Review and Content Analysis Using the Mobile App Rating Scale (MARS)," JMIR mHealth and uHealth, vol. 6, no. 3, p. e74, Mar. 2018.

[3] Aungst T. D., "Medical Applications for Pharmacists Using Mobile Devices," *Annals of Pharmacotherapy*, vol. 47, no. 7-8, pp. 1088–1095, 2013.

[4] N. Bricon-Souf and C. R. Newman, "Context awareness in health care: A review," International Journal of Medical Informatics, vol. 76, no. 1, pp. 2–12, 2007.

[5] "Pill Identification Tool," Drugs.com. [Online]. Available: https://www.drugs.com/pill_identification.html. Accessed: 20 March 2026.

[6] V. Law, C. Knox, Y. Djoumbou, T. Jewison, A. C. Guo, Y. Liu, A. Maciejewski, D. Arndt, M. Wilson, V. Neveu, A. Tang, G. Gabriel, C. Ly, S. Adamjee, Z. T. Dame, B. Han, Y. Zhou, and D. S. Wishart, "DrugBank 4.0: shedding new light on drug metabolism," *Nucleic Acids Research*, vol. 42, no. D1, pp. D1091–D1097, Jan. 2014.

[7] P. J. Pitts, "openFDA: an open question," *Ther. Innov. Regul. Sci.*, vol. 49, no. 2, pp. 254–255, 2015.

[8] S. Liu, W. Ma, R. Moore, V. Ganesan, and S. Nelson, "RxNorm: prescription for electronic drug information exchange," *IT Professional*, vol. 7, no. 5, pp. 17–23, Sept.-Oct. 2005.

[9] Z. Yaniv, J. Faruque, S. Howe, K. Dunn, D. Sharlip, A. Bond, P. Perillan, O. Bodenreider, M. J. Ackerman, and T. S. Yoo, "The National Library of Medicine Pill Image Recognition Challenge: An Initial Report," 2016 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), Washington, DC, USA, 2016, pp. 1–9.

[10] K. Al-Hussaeni, I. Karamitsos, E. Adewumi, and R. M. Amawi, "CNN-Based Pill Image Recognition for Retrieval Systems," Applied Sciences, vol. 13, no. 8, p. 5050, 2023.

[11] Y. Qiu, Y. Zhang, Y. Deng, S. Liu, and W. Zhang, "A Comprehensive Review of Computational Methods For Drug-Drug Interaction Detection," IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 19, no. 4, pp. 1968-1985, July-Aug. 2022.

[12] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani, “Robust and efficient fuzzy match for online data cleaning,” in Proc. ACM SIGMOD Int. Conf. Management of Data, San Diego, CA, USA, 2003, pp. 313–324.

[13] H. Huang, L. Zhang, Y. Yang, L. Huang, X. Lu, J. Li, H. Yu, S. Cheng, and J. Xiao, “Construction and application of medication reminder system: intelligent generation of universal medication schedule,” BioData Mining, vol. 17, no. 23, 2024.

[14] Bindoff, I. K., “An Intelligent Decision Support System for Automated Medication Review,” University of Tasmania, Hobart, Australia, 2005.

[15] S. Bhattacharyya, “A brief survey of color image preprocessing and segmentation techniques,” J. Pattern Recognit. Res., vol. 1, pp. 120–129, 2011.

[16] C. Shorten and T. M. Khoshgoftaar, “A survey on image data augmentation for deep learning,” J. Big Data, vol. 6, no. 60, 2019.

[17] L. Yujian and L. Bo, “A normalized Levenshtein distance metric,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 29, no. 6, pp. 1091–1095, Jun. 2019.

[18] K. Pykes, “Cross-Entropy Loss Function in Machine Learning: Enhancing Model Accuracy,” DataCamp, Feb. 27, 2026. [Online]. Available: https://www.datacamp.com/tutorial/the-cross-entropy-loss-function-in-machine-learning. Accessed: May 1, 2026.

[19] [Discussion] Why normalise according to ImageNet mean and std dev for transfer learning?,” PyTorch Forums, Mar. 23, 2021. [Online]. Available: https://discuss.pytorch.org/t/discussion-why-normalise-according-to-imagenet-mean-and-std-dev-for-transfer-learning/115670/. Accessed: May 1, 2026.

[20] Softmax Activation Function in Neural Networks,” GeeksforGeeks, Nov. 17, 2025. [Online]. Available: https://www.geeksforgeeks.org/deep-learning/the-role-of-softmax-in-neural-networks-detailed-explanation-and-applications/. Accessed: May 1, 2026.

[21] “Explain meaning and purpose of L2 normalization,” Cross Validated (StackExchange), 2017. [Online]. Available: https://stats.stackexchange.com/questions/331926/explain-meaning-and-purpose-of-l2-normalization. Accessed: May 1, 2026.

[22] “Cosine Similarity,” GeeksforGeeks, Jul. 15, 2025. [Online]. Available: https://www.geeksforgeeks.org/dbms/cosine-similarity/. Accessed: May 1, 2026.

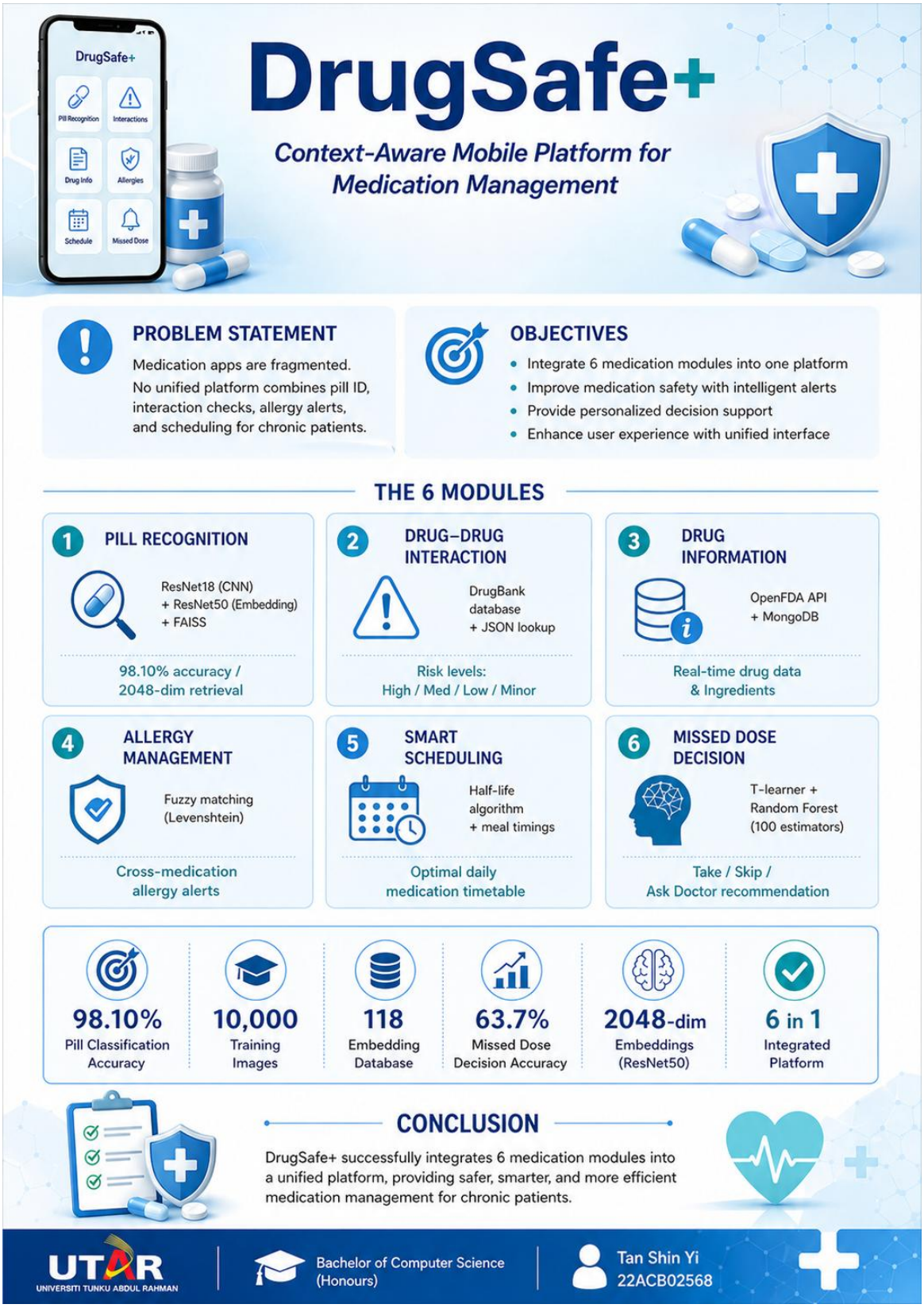
[23] “Chapter 6 – Pharmacokinetic Modelling,” Anesthesia Key. [Online]. Available: https://aneskey.com/chapter-6-pharmacokinetic-modelling/. Accessed: May 1, 2026.

[24] J. Hallare and V. Gerriets, “Elimination Half-Life of Drugs,” StatPearls, updated May 3, 2025.

[25] National Kidney Foundation Patient Education Team, “Estimated Glomerular Filtration Rate (eGFR),” Feb. 5, 2026.

[26] A. Bansal, K. Balaji, and Z. Lalani, “Temporal Encoding Strategies for Energy Time Series Prediction,” arXiv preprint arXiv:2503.15456, Mar. 2025.

POSTER



Bachelor of Computer Science (Honours)
Faculty of Information and Communication Technology (Kampar Campus), UTAR