Faculty of Computer Science

*Information Technology (IT)*

**Egyptian E-Learning University (EELU)**

A Step Towards Innovation in Predictive Healthcare

*Heart Failure Prediction System*

Using Machine Learning Technique

Heart failure is a life-threatening condition affecting millions worldwide. This project aims to develop a Heart Failure Prediction System leveraging machine learning to predict the risk of heart failure using clinical data. By integrating a web-based interface with advanced AI models, the system offers a user-friendly platform for healthcare professionals to make data-driven decisions. Key features include CSV-based batch predictions, a responsive web interface, and seamless integration with existing healthcare workflows. This system aspires to enhance diagnostic accuracy, promote early intervention, improve patient outcomes through innovative and accessible technology, and aims to make diagnosis faster, more accurate, and more efficient by combining artificial intelligence with healthcare. This system is designed to support medical professionals, reduce complications, and improve the quality of life for patients through early detection and intervention.

Step Towards Innovation in Predictive Healthcare

Heart Failure Prediction System

This project focuses on building a system to predict the risk of heart failure using machine learning techniques. It analyzes patient data, such as age, body mass index (BMI), physical activity, and medical history, to identify individuals who may be at risk. The goal is to provide early warnings that help doctors take preventive measures and improve patient care.

The project aims to make diagnoses faster, more accurate, and more efficient by combining artificial intelligence with healthcare. This system is designed to support medical professionals, reduce complications, and improve the quality of life for patients through early detection and intervention.

Under the Supervision of :

**Dr. Alaa Zaki**

Electronics Research Institute

Guided by :

**Dr. Rehab Abo Al-Hassan**

**Project Team :**

|  |  |
| --- | --- |
| **Ahmed Mohamed Elroby** | 21-01842 |
| **Abanoub Emad Roshdy** | 21-01808 |
| **Shaimaa Mohmed Zaki** | 21-01946 |
| **Ahmed Gomaa Saleh** | 21-01818 |
| **Ahmed Samir Darwish** | 21-01830 |
| **Aya Mohamed Hassan** | 21-02876 |
| **Alaa Nasser Mohamed** | 21-01860 |

Dedication

This project is wholeheartedly dedicated to my beloved **parents**, whose boundless love, continuous encouragement, and unwavering faith in me have been the backbone of my journey. Your sacrifices, prayers, and constant motivation have guided me through every challenge and fueled my determination to persevere. Without your presence, this achievement would not have been possible.

I also extend my deepest gratitude and dedicate this work to my **esteemed supervisors**, **Dr. Alaa** and **Eng. Rehab Abo Al-Hassan**, whose exceptional guidance, insightful feedback, and continuous support were crucial to the success of this project. Their dedication to excellence and passion for knowledge have left a lasting impact on my academic and personal growth.

To my **dear friends and colleagues**—**Ahmed Mohamed Elroby, Abanoub Emad Roshdy, Shaimaa Mohmed Zaki, Ahmed Gomaa Salah, Ahmed Samir Darwish, Aya Mohamed Hassan, and Alaa Nasser Mohamed**—this journey would not have been the same without your friendship, teamwork, and words of encouragement. Your shared enthusiasm and collaborative spirit turned challenges into learning opportunities and moments of stress into shared laughter.

I also dedicate this project to every **aspiring researcher, engineer, and healthcare innovator** striving to bridge the gap between technology and medicine. May this work stand as a small step in the long path toward making intelligent healthcare accessible, efficient, and impactful for all.

Finally, I dedicate this to **my future self**, as a reminder that perseverance, learning, and collaboration can transform ideas into reality—and that this is only the beginning.

Acknowledgment

First and foremost, I am profoundly grateful to **Allah Almighty**, whose endless blessings, guidance, and mercy have illuminated my path throughout this journey. It is only by His will and grace that I was granted the strength, patience, and clarity to complete this project.

I would like to extend my heartfelt thanks and deep appreciation to my **supervisor, Dr. Alaa**, for his continuous guidance, expert knowledge, and unwavering encouragement. His insightful direction and belief in my potential have played a pivotal role in shaping the project’s structure and outcome. I am equally indebted to **Eng. Rehab Abo Al-Hassan**, whose meticulous attention to detail, thoughtful feedback, and constructive support provided the momentum I needed at every critical stage. Their mentorship has been both intellectually enriching and personally inspiring.

I am also incredibly thankful to my **friends and colleagues** — **Ahmed Mohamed Elroby, Abanoub Emad Roshdy, Shaimaa Mohmed Zaki, Ahmed Gomaa Salah, Ahmed Samir Darwish, Aya Mohamed Hassan, and Alaa Nasser Mohamed** — for being more than just teammates. Your companionship, collaboration, encouragement, and shared moments of challenge and celebration made this journey more meaningful and memorable.

My sincere gratitude also goes to the **faculty and staff at the Egyptian E-Learning University**, whose commitment to academic excellence and innovation in education provided me with a strong foundation in the fields of technology and engineering. Their dedication to student success created a motivating environment that allowed me to grow and strive for excellence.

To my **family**, I owe my deepest appreciation and love. Your steadfast support, prayers, and belief in me—even during the most difficult times—have been my greatest source of strength. Your sacrifices and love gave me the confidence to keep moving forward, and for that, I will be forever grateful.

Lastly, I would like to acknowledge everyone—mentors, peers, educators, and supporters—who, directly or indirectly, played a role in the development of this project. This work is a reflection not only of my effort but of the collective influence of everyone who inspired, taught, and supported me.

Table of Contents

Contents

[chapter 1: INTRODUCTION 1](#_Toc200301225)

[1.1 Preamble : 2](#_Toc200301226)

[1.2 Problem Background : 2](#_Toc200301227)

[1.3 Problem Statement : 2](#_Toc200301228)

[1.4 Significance of the Project : 2](#_Toc200301229)

[1.5 Project Aim and Objectives : 3](#_Toc200301230)

[1.6 Project Scope : 3](#_Toc200301231)

[1.7 Project Software and Hardware Requirements : 3](#_Toc200301232)

[1.8 Project Limitations : 4](#_Toc200301233)

[1.9 Project Expected Output : 4](#_Toc200301234)

[1.10 Project Schedule : 5](#_Toc200301235)

[1.11 Report Outline : 5](#_Toc200301236)

[CHAPTER 2: RELATED EXISTING SYSTEMS 1](#_Toc200301237)

[2.1 Introduction 2](#_Toc200301238)

[2.2 Existing Systems 2](#_Toc200301239)

[2.3 Overall Solution Approach : 8](#_Toc200301240)

[2.4 Summary: 8](#_Toc200301241)

[chapter 3: SYSTEM REQUIREMENTS ENGINEERING AND PLANNING 1](#_Toc200301242)

[3.1 Introduction: 2](#_Toc200301243)

[3.2 Feasibility Study: 2](#_Toc200301244)

[3.3 Requirements Elicitation Techniques: 2](#_Toc200301245)

[3.4 Targeted Users: 4](#_Toc200301246)

[3.5 Functional Requirements : 4](#_Toc200301247)

[3.6 Non-Functional Requirements: 5](#_Toc200301248)

[chapter 4: SYSTEM DESIGN 1](#_Toc200301249)

[4.1 Introduction: 2](#_Toc200301250)

[4.2 UML Use Case Diagram: 2](#_Toc200301251)

[4.3 UML Class Diagram: 4](#_Toc200301252)

[4.4 UML Sequence Diagram: 5](#_Toc200301253)

[4.5 UML Activity Diagram: 8](#_Toc200301254)

[4.6 Graphical User Interface (GUI) Design: 10](#_Toc200301255)

[4.7 Summary: 11](#_Toc200301256)

[chapter 5: SYSTEM IMPLEMENTATION 1](#_Toc200301257)

[5.1 Introduction: 2](#_Toc200301258)

[5.2 Database Implementation: 2](#_Toc200301259)

[5.3 Graphical User Interface Implementation: 6](#_Toc200301260)

[5.4 Other Components Implementation: 10](#_Toc200301261)

[5.5 Summary 16](#_Toc200301262)

[chapter 6: SYSTEM TESTING AND INSTALLATION 1](#_Toc200301263)

[6.1 Introduction : 2](#_Toc200301264)

[6.2 Heuristic Evaluation : 2](#_Toc200301265)

[6.3 Cooperative Evaluation : 4](#_Toc200301266)

[6.4 Requirements Validation and Completeness : 6](#_Toc200301267)

[6.5 System Installation : 7](#_Toc200301268)

[6.6 Summary : 8](#_Toc200301269)

[chapter 7: PROJECT CONCLUSION AND FUTURE WORK 1](#_Toc200301270)

[7.1 Introduction : 2](#_Toc200301271)

[7.2 Overall Weaknesses : 2](#_Toc200301272)

[7.3 Overall Strengths : 4](#_Toc200301273)

[7.4 Future Work : 5](#_Toc200301274)

[7.5 Summary : 7](#_Toc200301275)

[chapter 8: REFERENCES 8](#_Toc200301276)

[8.1 APPENDIX A: Flask API Sample Response : 9](#_Toc200301277)

[8.2 APPENDIX B: .env Sample Configuration : 1](#_Toc200301278)

[8.3 APPENDIX C: Directory Structure : 1](#_Toc200301279)

List of Table of Contents

[Table ‎1‑1: Project Expected Output 5](#_Toc200301281)

[Table ‎3‑1: Functional Requirement 4](#_Toc200301282)

[Table ‎6‑1: Requirement Validation 6](#_Toc200301283)

List of Figures

[Figure ‎1‑1: Project Schedule 5](#_Toc200301284)

[Figure ‎2‑1: related system 1 2](#_Toc200301285)

[Figure ‎2‑2: related system 2 4](#_Toc200301286)

[Figure ‎2‑3: related system 3 6](#_Toc200301287)

[Figure ‎3‑1: Elicitation Flowchart 3](#_Toc200301289)

[Figure ‎3‑2: Non-Functional Requirement 5](#_Toc200301290)

[Figure ‎4‑1: Use case diagram 2](#_Toc200301291)

[Figure ‎4‑2: Class diagram 4](#_Toc200301292)

[Figure ‎4‑3: Sequence diagram 5](#_Toc200301293)

[Figure ‎4‑4: Activity diagram 8](#_Toc200301294)

[Figure ‎5‑1: Database Implementation 3](#_Toc200301295)

[Figure ‎5‑2: Create\_tables.sql 4](#_Toc200301296)

[Figure ‎5‑3: Create\_prediction\_tables 5](#_Toc200301297)

[Figure ‎5‑4: GUI Components 7](#_Toc200301298)

[Figure ‎5‑5:predictionform.tsx 8](#_Toc200301299)

[Figure ‎5‑6: Dashboard.tsx 9](#_Toc200301300)

[Figure ‎5‑7:app.py 10](#_Toc200301301)

[Figure ‎5‑8: ECG\_Service.py 13](#_Toc200301302)

[Figure ‎5‑9:Chatbot 14](#_Toc200301303)

[Figure ‎5‑10: Chatbot\_Service.py 15](#_Toc200301304)

[Figure ‎6‑1: Heuristic Evaluation 3](#_Toc200301305)

[Figure ‎6‑2: Cooperative Evaluation 5](#_Toc200301306)

[Figure ‎7‑1: Overall Weaknesses 3](#_Toc200301307)

[Figure ‎7‑2: Overall Strengths 5](#_Toc200301308)

[Figure ‎7‑3: Future Work 6](#_Toc200301309)

Abstract

Heart failure is a critical medical condition that affects millions of people globally and poses a significant challenge to healthcare systems due to its high prevalence and mortality rates. The timely prediction and prevention of heart failure are crucial for improving patient outcomes, reducing hospitalizations, and lowering healthcare costs. This project focuses on developing an advanced predictive system for heart failure using machine learning techniques, which are renowned for their ability to uncover patterns in complex datasets and provide accurate predictions.

The primary goal of this system is to analyze patient data, including demographic details, clinical metrics, and lifestyle factors, to identify individuals at risk of heart failure before severe symptoms arise. Key features such as age, body mass index (BMI), physical health, diabetes status, mental health, and smoking habits are considered to build a comprehensive model that accurately evaluates the likelihood of heart failure. This system provides actionable insights that support healthcare professionals in making informed decisions and adopting timely preventive measures, ultimately saving lives and enhancing the quality of care.

The development process involves a thorough exploration of the dataset, including data preprocessing, visualization, and feature selection, to ensure that the model is built on reliable and meaningful data. Various machine learning algorithms are evaluated to identify the most effective approach for achieving high prediction accuracy. The project also integrates a user-friendly web interface powered by a Flask API, enabling healthcare providers to interact seamlessly with the system and access predictions in real-time.

The system not only emphasizes predictive accuracy but also contributes to the broader field of healthcare innovation by demonstrating the transformative potential of artificial intelligence in clinical applications. By bridging the gap between advanced technology and practical healthcare needs, this project addresses one of the most pressing challenges in modern medicine. It serves as a step toward the future of predictive and preventive medicine, where early detection and intervention become the cornerstone of effective healthcare delivery.

# INTRODUCTION

A top view of a desk with computer and various objects

Description automatically generated

## Preamble :

Heart failure is a complex medical condition that occurs when the heart cannot pump blood effectively to meet the body's needs. It affects millions of individuals globally, leading to severe health complications and placing a significant burden on healthcare systems. The growing prevalence of heart failure necessitates innovative approaches to enhance early detection and intervention. This project aims to address this challenge by leveraging the capabilities of machine learning to develop a reliable predictive system for heart failure.

## Problem Background :

Heart failure is a leading cause of hospitalization and mortality worldwide. Its diagnosis is often delayed until the condition reaches an advanced stage, reducing the effectiveness of treatment options. This delay is partly due to the complexity of the condition, which involves multiple risk factors such as age, body mass index (BMI), diabetes, smoking, and physical health. Traditional diagnostic methods often fail to identify early warning signs, leaving patients at greater risk of severe complications.

## Problem Statement :

The lack of efficient, data-driven tools for predicting heart failure limits the ability of healthcare providers to intervene at an early stage. Current systems rely heavily on manual analysis and general risk assessments, which are prone to error and subjectivity. There is a critical need for a system that uses machine learning to analyze patient data and provide accurate, actionable predictions for heart failure risks.

## Significance of the Project :

This project is significant because it introduces a data-driven approach to healthcare, enabling the prediction of heart failure risks with greater accuracy. By providing early warnings, the system empowers healthcare professionals to implement timely preventive measures, reducing complications and improving patient outcomes. Additionally, it demonstrates the potential of integrating artificial intelligence into medical applications, paving the way for advancements in predictive healthcare.

## Project Aim and Objectives :

* This project aims to design and implement a predictive system for heart failure using machine learning techniques. The specific objectives include:

### Analyzing and preprocessing patient data to ensure its quality and relevance.

### Developing a machine learning model capable of predicting heart failure risks.

### Creating a user-friendly web interface for healthcare providers to access predictions.

### Demonstrating the system’s effectiveness through testing and validation.

## Project Scope :

The project focuses on heart failure prediction by analyzing a predefined dataset containing patient demographic, clinical, and lifestyle information. It involves integrating a machine learning model with a web interface, using a Flask API to enable seamless interaction. The scope is limited to designing a prototype system that can be scaled for practical healthcare use.

## Project Software and Hardware Requirements :

### Software Requirements:

* Python Programming Language
* Scikit-learn library
* Flask framework
* HTML, CSS, and JavaScript for frontend development
* MySQL for database management

### Hardware Requirements:

* Personal computer with at least 8GB RAM and 500GB storage
* Internet connection for accessing external libraries and APIs

## Project Limitations :

* This project has several limitations, including:

### Reliance on the quality and completeness of the dataset.

### The machine learning model’s accuracy depends on the diversity and size of the training data.

### The system is a prototype and not intended for direct clinical deployment without further validation.

## Project Expected Output :

* The expected outputs of the project include:

### A trained and tested machine learning model capable of predicting heart failure risks on patient data.

### A web interface that allows healthcare providers to input patient data and receive predictions.

### Data visualization components for insights into key risk factors and their relationships to heart failure.

### Comprehensive documentation detailing the system’s architecture, implementation process, and usage instructions for users

### Validation reports showcasing the performance, accuracy, and reliability of the machine learning model.

### Recommendations for scaling the system to include additional healthcare datasets and potential applications in broader predictive medicine.

### Integration suggestions for incorporating the system into existing healthcare infrastructures, making it accessible and practical for real-world usage.

## Project Schedule :

A diagram of a project schedule

Description automatically generated

Figure ‎1‑1: Project Schedule

## Report Outline :

Table ‎1‑1: Project Expected Output

|  |  |
| --- | --- |
| **Chapter One** | Provides an overview of the project, including the background problem, significance, objectives, scope, limitations, and expected outcomes. |
| **Chapter Two** | Analyzes existing systems related to heart failure prediction, highlights their limitations, and presents the overall approach for the proposed solution. |
| **Chapter Three** | Requirements elicitation techniques, identification of targeted users, and detailed specification of functional and non-functional requirements. |
| **Chapter Four** | Including context diagrams, data flow diagrams (DFD), entity relationship diagrams (ERD), and UML diagrams such as use case, and sequence diagram. |
| **Chapter Five** | Describes the implementation phase, including database development, GUI, and integration of system components using APIs. |
| **Chapter Six** | Focuses on the testing process, validation of requirements, and the installation |
| **Chapter Seven** | Discusses the strengths and weaknesses of the system, and future work. |

# RELATED EXISTING SYSTEMS

A group of people working on a computer

Description automatically generated

## Introduction

This chapter explores the existing systems and technologies used for heart failure prediction and diagnosis. By analyzing the current approaches, we can identify their limitations and the opportunities for improvement. The aim is to provide a comprehensive understanding of the research gaps and justify the development of the proposed system.

## Existing Systems

Several systems have been developed to aid in the prediction and diagnosis of heart failure. These systems utilize a combination of traditional statistical methods and modern machine-learning techniques:

### First related work

#### Title: Cardiac Failure Forecasting Based on Clinical Data Using a Lightweight Machine Learning Metamodel

#### Methodology and Materials:

The methodology employed in this study aims to develop and evaluate a machine learning metamodel for predicting heart failure based on clinical test data. The flow of the proposed framework is depicted in the figure. The first step involves data collection to create the dataset. Next, significant variables are extracted, and the data are prepared accordingly. Subsequently, the dataset is divided into training and testing sets. The training data are then utilized to train the proposed metamodel. Finally, the metamodel is generated and tested to obtain the output results

A diagram of a model

Description automatically generated

Figure ‎2‑1: related system 1

#### Advantages:

* **Focus on Clinical Data**: The use of clinical data directly aligns with your project goal of predicting heart failure based on real-world medical data. This makes it highly relevant for your research.
* **Lightweight Metamodel**: A lightweight approach suggests that the model is optimized for efficiency, which can be crucial for deployment in real-time applications or resource-constrained environments.
* **Use of Metamodeling**: Metamodeling can help in combining multiple models to improve prediction accuracy, which might enhance your system’s performance by using various machine learning techniques for heart failure prediction.
* **Applicability to Practical Systems**: The emphasis on clinical data ensures that the model can be tested and applied in a clinical setting, potentially bridging the gap between theoretical machine learning models and real-world healthcare applications.

#### Limitations:

* **Model Complexity**: While lightweight models are efficient, they may be less accurate or may miss certain complex patterns in data compared to more complex models like deep learning. This could affect prediction precision.
* **Data Availability**: Clinical data, especially medical records, can sometimes be limited, incomplete, or challenging to acquire, especially in different formats, which might be an obstacle in implementing this model.
* **Generalization**: A lightweight metamodel might be overfitted to specific datasets and may not generalize well across varied clinical environments or different patient populations.
* **Integration Challenges**: The integration of such models into existing healthcare systems might face challenges, such as regulatory approvals and healthcare professionals’ adoption.

### Second related work:

#### Title: Prediction of Heart Disease Based on Machine Learning Using Jellyfish Optimization Algorithm

#### Methodology and Materials:

This paper presents a performance analysis of different ML techniques based on selecting the meaningful features of the dataset in the hope of improving heart disease prediction accuracy. In this study, the performance of different ML models such as ANN, DT, Adaboost, and SVM using the Jellyfish algorithm and feature selection for the prediction of heart disease was compared, aiming at obtaining the highest performance model. The Cleveland dataset used in this study was obtained from the Kaggle Machine Learning repository.

A diagram of a machine learning process

Description automatically generated

Figure ‎2‑2: related system 2

#### Advantages:

* **Jellyfish Optimization Algorithm**: This algorithm is inspired by jellyfish behavior and is a relatively new and unique approach to optimization, which may offer novel solutions to improving prediction accuracy, especially in highly variable data like heart disease prediction.
* **Improved Accuracy**: Optimization algorithms, especially novel ones like Jellyfish, can improve the efficiency of machine learning models by tuning hyperparameters, which might result in more accurate predictions for heart failure.
* **Adaptability**: The optimization algorithm can be tailored to different machine learning models, allowing for flexibility in the choice of algorithms used in your prediction system.
* **Novel Approach**: Using newer, lesser-known optimization algorithms can give your project a unique edge compared to conventional approaches, possibly making it stand out in terms of innovation

#### Limitations:

* **Lack of Proven Widespread Use**: While the Jellyfish Optimization Algorithm is promising, it may not be as well-established or validated in heart disease prediction compared to more traditional optimization algorithms (like Genetic Algorithms or Particle Swarm Optimization).
* **Complexity**: The algorithm’s novelty might come with a steeper learning curve. Understanding and implementing this optimization method could require deeper knowledge, which may take extra time and resources, especially for your first project involving machine learning.
* **Computational Cost**: Optimization algorithms can sometimes be computationally expensive, particularly with large datasets, leading to longer training times or the need for more powerful hardware.
* **Data Sensitivity**: The performance of the optimization algorithm heavily depends on the quality and structure of the input data. Inconsistent or noisy data might result in less effective predictions, requiring extensive preprocessing.

### Third related work:

#### Title: A proposed technique for predicting heart disease using machine learning algorithms and an explainable AI method

#### Methodology and Materials :

Figure shows the proposed system’s sequences for predicting heart diseases. We first gathered and preprocessed the dataset to remove any necessary inconsistencies, such as replacing null occurrences with average values. We divided the dataset into two distinct groups, named the test dataset and the training dataset, respectively. Next, we implemented several distinct classification algorithms to determine which one achieved the highest accuracy for these datasets.

A diagram of a process flow

Description automatically generated

Figure ‎2‑3: related system 3

#### Advantages:

* **Incorporation of Explainable AI (XAI):** One of the key strengths of this technique is its focus on **explainability**, which is critical in healthcare applications. XAI can help healthcare professionals understand and trust the model’s predictions, improving the likelihood of real-world adoption. This is particularly important in medical fields, where transparency and interpretability of AI systems are crucial for patient care and clinical decision-making.
* **Machine Learning Algorithms Integration:** The use of machine learning algorithms for heart disease prediction suggests that the technique leverages established methods (such as decision trees, SVM, etc.) that can be highly accurate for classification tasks. These methods have been widely studied and are known for their strong performance in predicting heart disease based on clinical data.
* **Improved Decision Support for Healthcare Professionals:** By combining traditional machine learning with explainable AI, this approach can serve as a decision-support tool for healthcare providers. XAI allows users to interpret why the model made a certain prediction, helping them make informed decisions based on model insights rather than treating it as a "black box."
* **Accuracy and Transparency:** Machine learning models can provide highly accurate predictions when trained on large datasets, while XAI adds a layer of transparency that helps overcome the common skepticism around AI-based decision-making in medicine.

#### Limitations:

* **Trade-off Between Accuracy and Explainability:** While explainability is important, it may come at the cost of model accuracy. More complex models, such as deep learning or ensemble models, can achieve higher prediction accuracy, but they often lack explainability. This creates a trade-off where using more interpretable models may reduce prediction performance slightly, depending on the dataset.

## Overall Solution Approach :

1. Leverage Advanced Machine Learning:

* Utilize interpretable algorithms and feature importance analysis to enhance trustworthiness.

1. Ensure Accessibility:

* Develop a web-based interface to make the system widely available to healthcare professionals.

1. Use Diverse and Representative Data:

* Train the model on datasets that include patients from different demographics and medical histories.

1. Enable Seamless Integration:

* Design the system to be compatible with electronic health records and hospital management systems.

1. Provide Actionable Insights:

* Deliver clear and concise predictions to support clinical decision-making in real time.

## Summary:

This chapter reviewed existing systems for heart failure prediction and diagnosis, highlighting their strengths and limitations. By identifying the gaps in these systems, we have established the need for an innovative approach that addresses these challenges. The proposed solution aims to fill these gaps and offer a practical, efficient, and accessible tool for healthcare professionals.

# SYSTEM REQUIREMENTS ENGINEERING AND PLANNING

A group of people working on a server

Description automatically generated

## Introduction:

This chapter focuses on the engineering and planning aspects of the system requirements. It outlines the feasibility of the project, the methods used to gather requirements, the identified users, and the specifications of both functional and non-functional requirements. A well-defined requirements engineering process ensures that the developed system meets the intended goals and provides a clear path for its design and implementation.

## Feasibility Study:

* The feasibility study evaluates the project in terms of technical, operational, and economic factors:

### Technical Feasibility: Ensures the availability of required technologies, including Python, machine learning libraries, and web development frameworks such as Flask.

### Operational Feasibility: Confirms that healthcare professionals can seamlessly integrate the system into their workflow.

### Economic Feasibility: Demonstrates the cost-effectiveness of using open-source tools and the potential savings achieved through early heart failure prediction.

## Requirements Elicitation Techniques:

* To define system requirements accurately, the following elicitation techniques were employed:

### Interviews: Conducted with healthcare providers to understand their needs for a heart failure prediction system.

### Questionnaires: Distributed to gather insights about the features and functionalities desired in the system.

### Document Analysis: Reviewed medical literature and existing systems to identify critical features.

### Brainstorming Sessions: Facilitated discussions with the project team to align technical capabilities with user expectations.

A diagram of a process

Description automatically generated

Figure ‎3‑1: Elicitation Flowchart

## Targeted Users:

* The system is designed for the following user groups:

### Healthcare Providers: Doctors and medical staff who will input patient data and utilize predictions for clinical decision-making.

### Healthcare Administrators: To monitor system performance and integrate its insights into hospital workflows.

### Researchers: To analyze the system’s output for studies related to heart failure and preventive healthcare.

## Functional Requirements :

Table ‎3‑1: Functional Requirement

|  |  |
| --- | --- |
| **Functional Requirements Definition** | **Functional Requirements Specification** |
| The functional requirements define the core operations of the system, including:   * Ability to accept patient data through a web interface. * Processing patient data using the machine learning model. * Displaying prediction results with risk levels and key factors contributing to the risk. * Generating data visualizations for insights into heart failure trends. | a document that outlines the functionality and behavior of a system includes:   1. The system shall allow users to input patient details (e.g., age, BMI, health metrics). 2. The system shall predict heart failure risk with a confidence score. 3. The system shall provide visualizations of the predictions and contributing factors. 4. The system allows healthcare professionals to download or print reports of the predictions. |

## Non-Functional Requirements:

### Performance: The system shall process predictions within 3 seconds for each input.

### Scalability: The system shall support up to 500 concurrent users without performance degradation.

### Usability: The web interface shall be intuitive and accessible to non-technical healthcare professionals.

### Security: All patient data shall be encrypted during transmission and storage to ensure confidentiality.

### Availability: The system shall maintain 99.9% uptime for consistent usage in clinical settings.

A diagram of a computer program

Description automatically generated with medium confidence

Figure ‎3‑2: Non-Functional Requirement

# SYSTEM DESIGN

A computer with a puzzle piece

Description automatically generated

## Introduction:

The system design phase plays a crucial role in transforming the requirements into a structured blueprint that guides the implementation of the system. This chapter outlines the design strategies and methodologies for developing the Heart Failure Prediction System. By leveraging design principles and visual modeling techniques, the aim is to ensure that the system is intuitive, scalable, and meets functional and non-functional requirements.

The design includes various Unified Modeling Language (UML) diagrams, such as use case, activity, sequence, and class diagrams, to provide a detailed representation of system functionalities, workflows, and interactions. Additionally, the graphical user interface (GUI) design emphasizes user-friendliness, ensuring that healthcare providers can seamlessly interact with the system.

## UML Use Case Diagram:

A diagram of a diagram

Description automatically generated

Figure ‎4‑1: Use case diagram

### Diagram Explanation:

1. Actors:

* **Healthcare Provider (HCP):** The primary user who interacts with the system to input patient data, view predictions, and analyze insights.
* **Administrator (Admin):** Manages system users and monitors system usage.

1. Use Cases:

* **Input Patient Data:** Allows the healthcare provider to input details like age, BMI, and medical history.
* **Predict Heart Failure Risk:** Processes patient data through the machine learning model.
* **Generate Risk Report:** Creates detailed reports for clinical documentation.
* **View Risk Factors and Insights:** Displays key factors contributing to the predicted risk.
* **Manage User Access:** Enables the administrator to manage access permissions.
* **Monitor System Usage:** Provides usage analytics for system performance evaluation.

1. Additional Actor:

* **Researcher**: Added a new actor to reflect the potential use of the system by researchers for data analysis and study purposes.

1. Relationships Between Actors:

* **Healthcare Provider ↔ Administrator**:  
  Represents interactions where providers report system issues, and administrators resolve them.
* **Administrator ↔ Researcher**:  
  Demonstrates how administrators grant researchers access to system data upon request.

1. Additional Use Case:

* **Export Data for Research**:  
  Allows researchers to request and export anonymized system data for analysis.

## UML Class Diagram:

A diagram of a computer

Description automatically generated

Figure ‎4‑2: Class diagram

### Diagram Explanation:

1. Classes:

* **User**: The parent class for HealthcareProvider, Administrator, and Researcher.
* **HealthcareProvider**: Manages patient data, views predictions, and generates reports.
* **Administrator**: Handles user management and monitors system performance.
* **Researcher**: Exports and analyzes system data for research purposes.
* **Patient**: Represents patient data, including attributes like age, BMI, and healthMetrics.
* **PredictionSystem**: Processes patient data, predicts risk, and generates visualizations.
* **Database**: Handles data storage, retrieval, and updates.

1. Relationships:

* **Inheritance**: HealthcareProvider, Administrator, and Researcher inherit from the User class.
* **Associations**:
* HealthcareProvider manages patients.
* PredictionSystem analyzes Patient data and interacts with the Database.
* Administrator manages Database access, while Researcher exports data from it.

1. Attributes and Methods:

* Each class includes relevant attributes (e.g., userID, patientID) and methods (e.g., login(), predictRisk()).

## UML Sequence Diagram:

A screenshot of a computer screen

Description automatically generated

Figure ‎4‑3: Sequence diagram

### Explanation of the Sequence Diagram:

1. Actors and Participants:

* **Healthcare Provider (HCP):** Initiates actions like logging in, entering patient data, and requesting reports.
* **Web Interface (UI):** Acts as a mediator between the healthcare provider and the system's backend.
* **Prediction System (PS):** Processes data, performs analysis, and generates predictions.
* **Database (DB):** Stores and retrieves data related to users, patients, and predictions.

1. Steps in the Sequence:

* Login Process:
* The healthcare provider logs in through the web interface.
* The web interface validates credentials with the database.
* If successful, access is granted to the provider.
* Patient Data Input:
* The provider inputs patient data through the web interface.
* The interface sends this data to the prediction system.
* The prediction system fetches additional relevant data from the database and processes it to predict heart failure risk.
* Risk Report Request:
* The provider requests a detailed risk report.
* The system generates the report, stores it in the database, and displays it to the provider.

1. Multiple Actors:

* **Healthcare Provider (HCP):** Interacts with the system to log in, input patient data, view predictions, and request reports.
* **Administrator (Admin):** Manages user access and monitors system performance.
* **Researcher (Res):** Requests anonymized data for research purposes.

1. Detailed Interactions:

* For Healthcare Providers:
* Includes full prediction and report generation workflow.
* For Administrators:
* User management actions (e.g., adding/removing users).
* System monitoring through usage data retrieval and analytics.
* For Researchers:
* Requests anonymized patient data for analysis.
* Interaction ensures data privacy and security during processing.

1. Database Interactions:

 The database handles user authentication, patient data storage, usage tracking, and anonymized data retrieval.

1. Prediction System Logic:

* Processes patient data, analyzes it and predicts heart failure risk.
* Generates and stores detailed reports for later access.

## UML Activity Diagram:

A screenshot of a computer screen

Description automatically generated

Figure ‎4‑4: Activity diagram

### UML Activity Diagram Explanation:

1. Swimlanes for Actors:

* **Healthcare Provider**: Logs into the system, inputs patient data, views predictions, and requests/downloads reports.
* **Administrator**: Manages users and monitors system performance through analytics.
* **Researcher**: Requests anonymized data for research purposes.

1. Branches and Decisions:

* Validation of patient data:
* If valid, data is processed by the prediction system.
* If invalid, an error message is displayed, and the process stops.

1. System Interactions:

* **Web Interface**: Handles data validation and interaction between the user and backend components.
* **Prediction System**: Processes patient data, predicts risk, and generates reports.
* **Database**: Stores reports, manages users, and retrieves anonymized data.

1. Parallel Workflows:

* The diagram shows distinct workflows for each actor, allowing the system to handle multiple user roles concurrently.

**Why is This Diagram Efficient and Complex?**

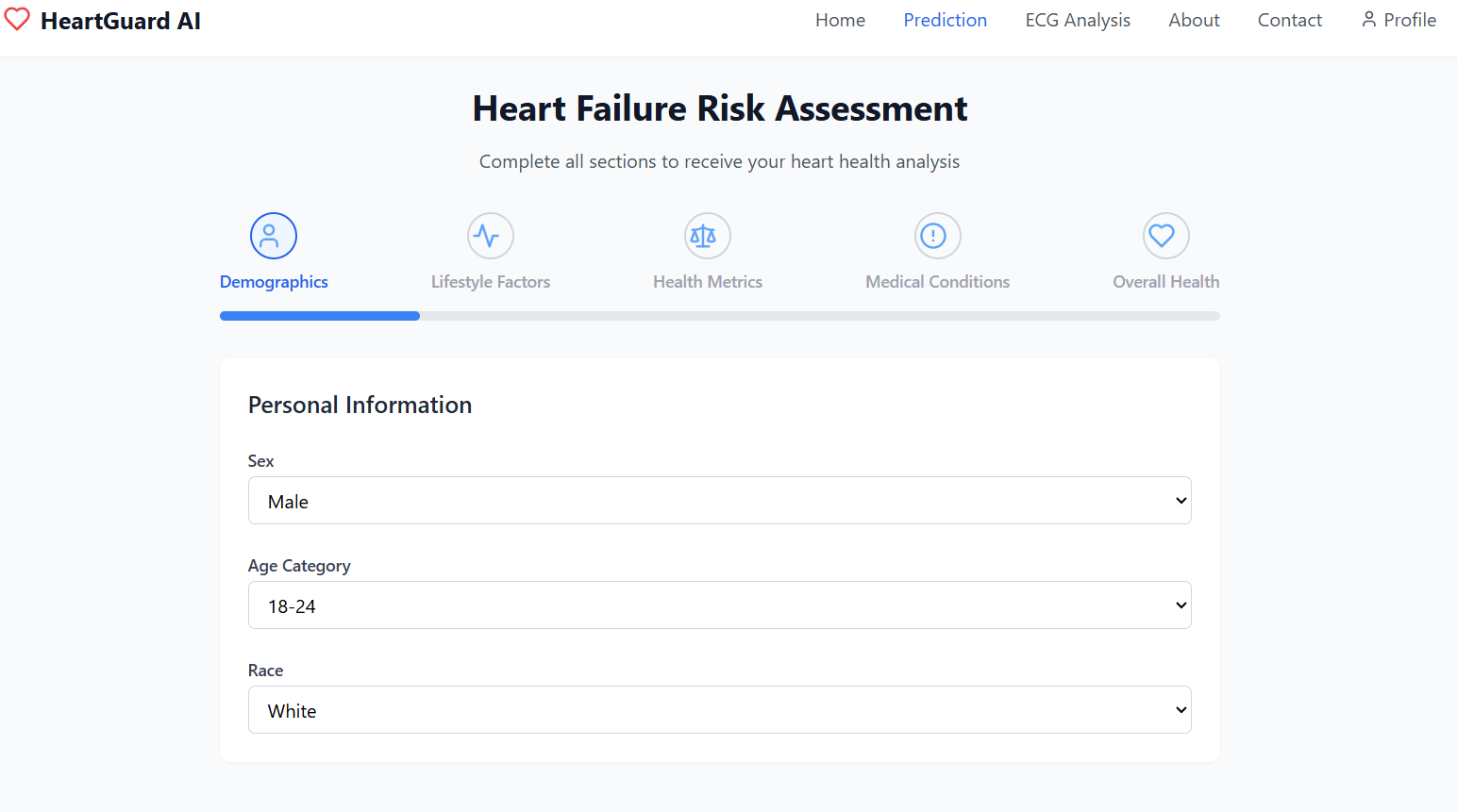
* **Efficiency:**
* **Logical flow minimizes redundancy and highlights key decision points.**
* **Represents system interactions in a clear, step-by-step manner.**
* **Complexity:**
* **Incorporates multiple actors (Healthcare Provider, Administrator, Researcher).**

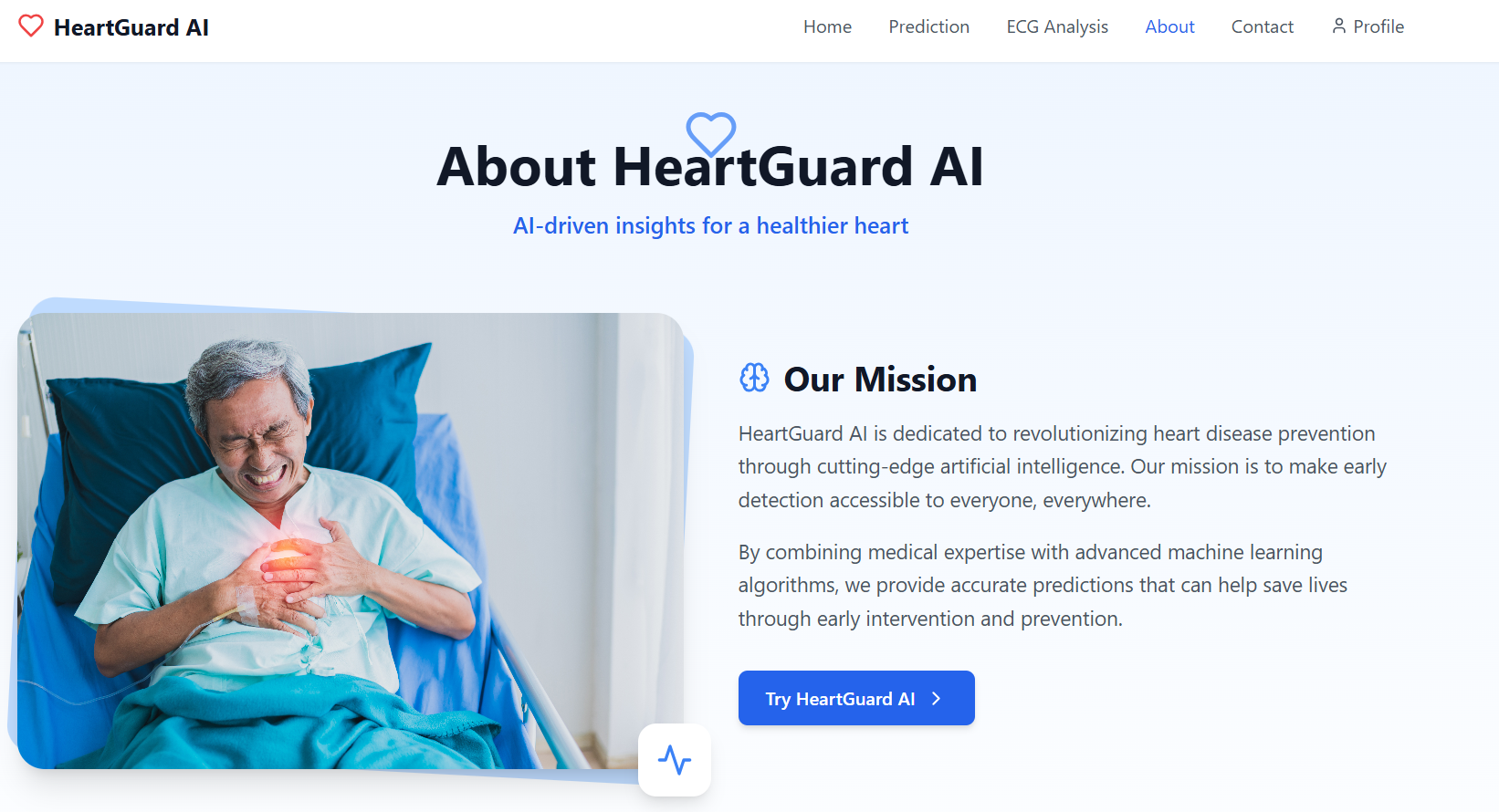
## Graphical User Interface (GUI) Design:

The Graphical User Interface (GUI) is a crucial component of the Heart Failure Prediction System, serving as the primary medium through which users interact with the application. It is designed to be simple, user-friendly, and accessible, enabling healthcare providers, administrators, and researchers to perform essential tasks efficiently.

A screenshot of a computer

Description automatically generated

****

****

**A screenshot of a computer

Description automatically generated**

## Summary:

This chapter presented the system design approach, emphasizing the use of UML diagrams to define system functionality and user interactions. The design ensures that the system is both functional and accessible, addressing the needs of healthcare professionals effectively. By aligning with the requirements, the design phase sets a strong foundation for the subsequent implementation and testing phase

# SYSTEM IMPLEMENTATION

## Introduction:

This chapter presents the practical implementation of the Heart Failure Prediction System, transforming the system’s conceptual design into a fully functioning application. The system integrates various components, including the front-end user interface, back-end server logic, machine learning model, and supporting services. The implementation used modern web development technologies such as **React** and **TypeScript** for the frontend, **Supabase** for user authentication and cloud-based database storage, and **Flask** for building the API and integrating machine learning models.

In addition to the core prediction functionality, the system was enhanced with advanced features, including an **ECG-based analysis module**, which processes electrocardiogram data to support diagnostic insights, and a **chatbot assistant** that provides a basic level of interaction and guidance for the user. These modules were developed and deployed alongside the main system to enrich the functionality and increase usability.

The implementation phase focused on ensuring seamless communication between system components, robustness in performance, and scalability for future enhancements. Careful attention was paid to modular coding practices, environment configuration, and error handling. This chapter outlines how each part of the system was developed, connected, and validated to form a cohesive, intelligent prediction platform accessible via the web.

## Database Implementation:

The database component of the Heart Failure Prediction System was implemented using **Supabase**, an open-source backend-as-a-service (BaaS) platform that offers real-time PostgreSQL databases, user authentication, and RESTful APIs out-of-the-box. Supabase was chosen for its seamless integration with JavaScript-based frontends like React and its ease of use for rapid prototyping and deployment.

The core data storage revolves around the **predictions** table, which is responsible for storing the results generated by the machine learning model. Each record in this table includes:

* The **user identifier (user\_id)** linked to Supabase authentication,
* The **input features** submitted by the user (e.g., BMI, age category, smoking status, physical activity),
* The **prediction result** (e.g., high or low risk of heart failure),
* A **timestamp** for tracking when the prediction was made.

This schema was designed to support future data analysis, performance monitoring, and continuous improvement of the model by collecting real user data. Supabase also securely manages user sessions and account creation via its built-in authentication module, enabling the system to maintain a personalized experience for each user.

The integration with Supabase was established through its client SDK, where data is sent and fetched directly from the frontend using asynchronous functions. This allows the system to dynamically store prediction results and later retrieve them for visualization or audit purposes in the dashboard interface.

Overall, Supabase offered a scalable, secure, and efficient solution for managing both structured data and user authentication, significantly accelerating backend development



Figure ‎5‑1: Database Implementation

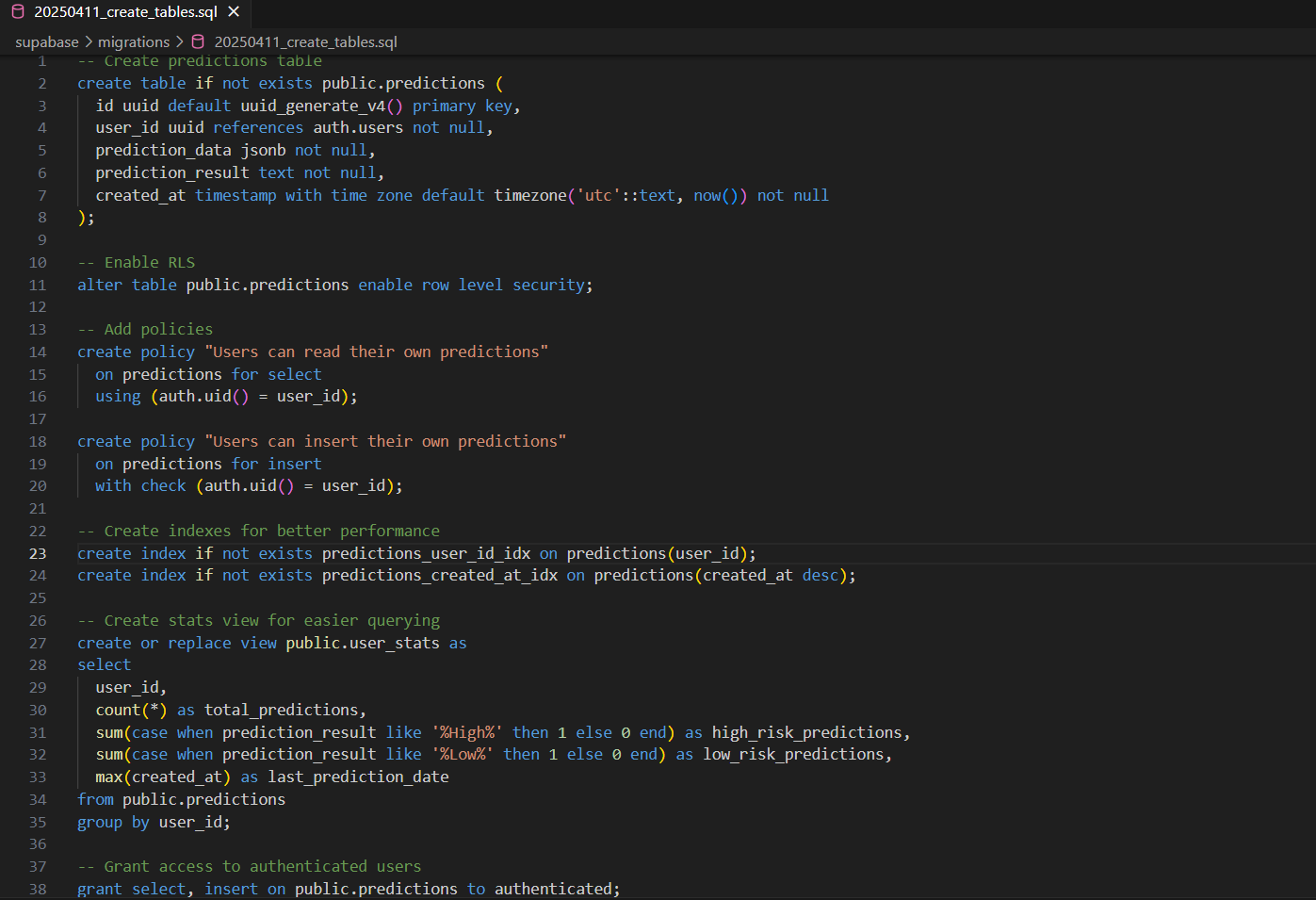


Figure ‎5‑2: Create\_tables.sql

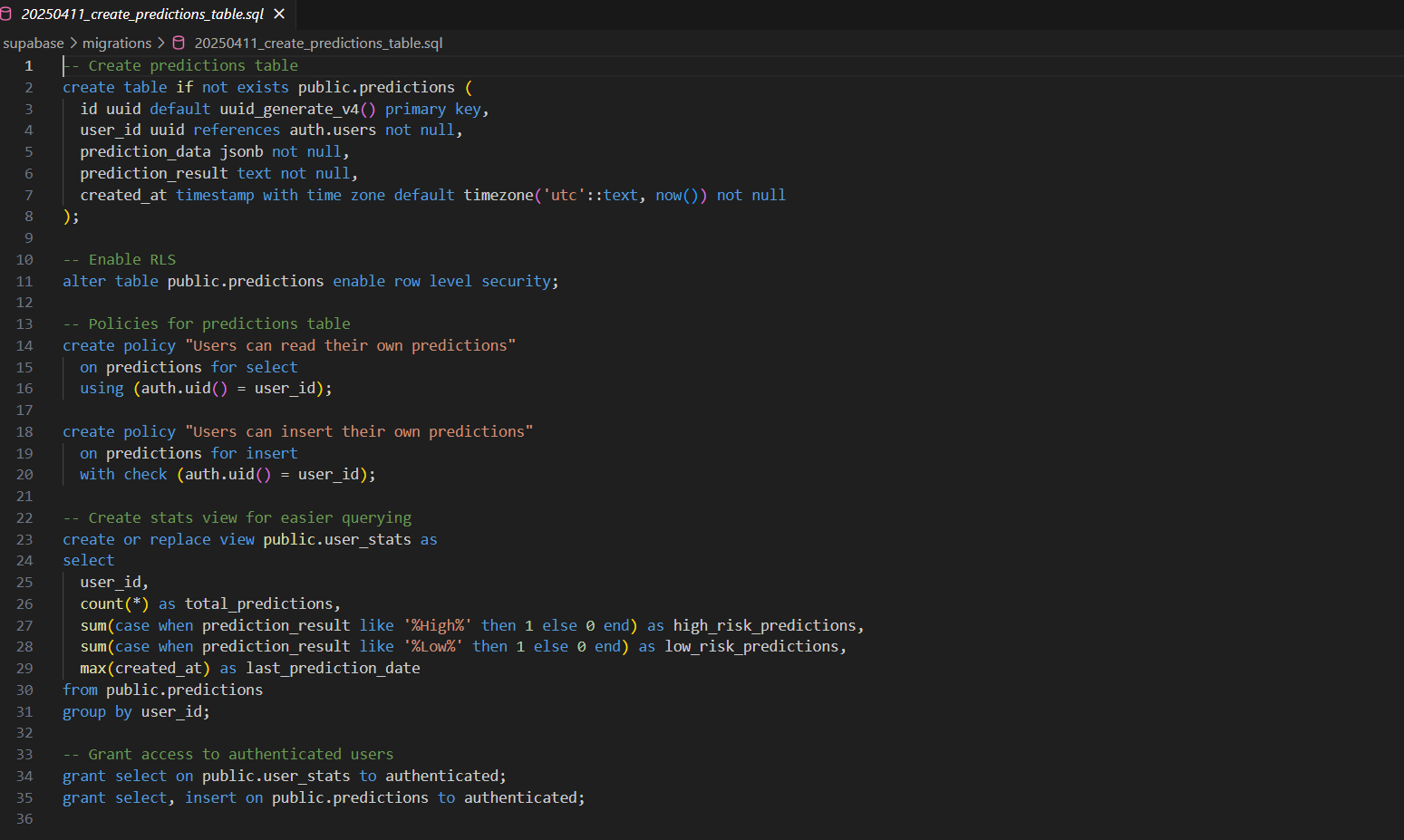


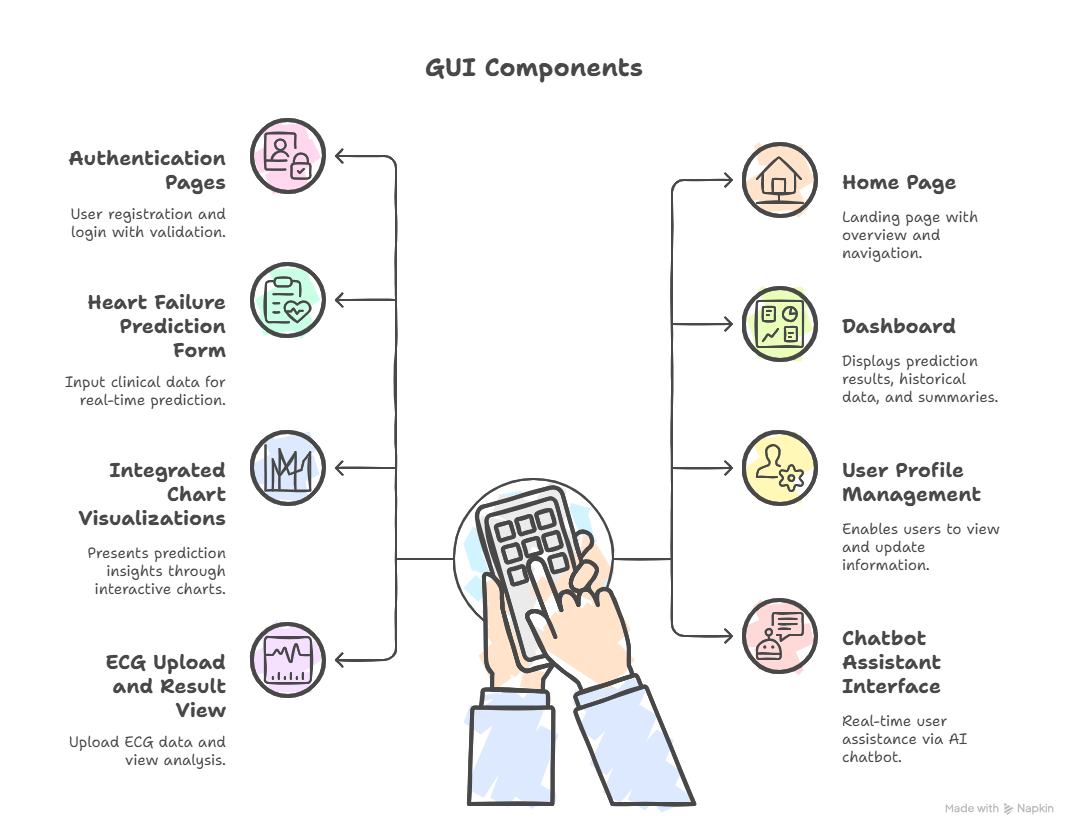
Figure ‎5‑3: Create\_prediction\_tables

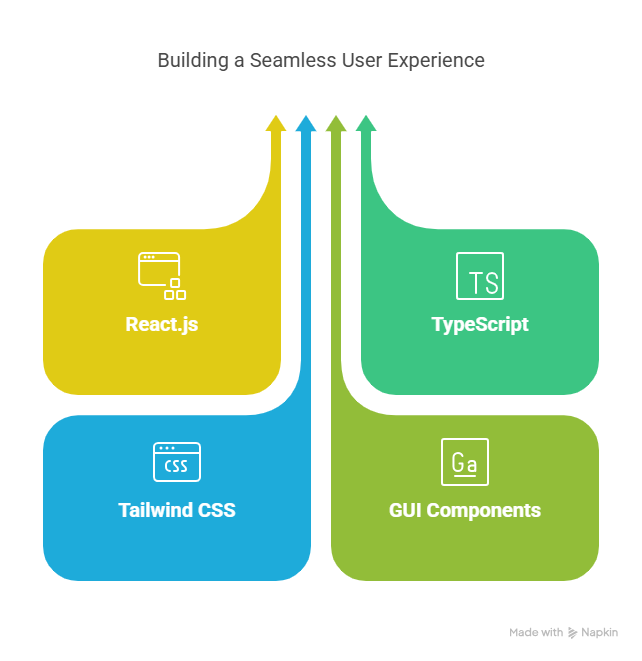
## Graphical User Interface Implementation:

The system's frontend was developed using **React.js** with **TypeScript** for type safety and enhanced code maintainability. Styling was achieved using **Tailwind CSS**, enabling a responsive and modern user interface design. The graphical user interface (GUI) encompasses the following key components:

* **Authentication Pages**: Includes user registration and login functionalities with secure form validation.
* **Home Page**: Serves as the landing page, providing a general overview and navigation to core features.
* **Heart Failure Prediction Form**: A structured form allowing users to input clinical data for real-time prediction.
* **Dashboard**: Displays the prediction results, including historical data and analysis summaries.
* **Integrated Chart Visualizations**: Presents prediction insights and data trends through interactive charts.
* **User Profile Management**: Enables users to view and update their personal information securely.
* **ECG Upload and Result View**: Allows users to upload ECG data files and view corresponding analysis results.
* **Chatbot Assistant Interface**: Offers real-time user assistance and guidance through an AI-powered chatbot.

This structured and user-friendly interface ensures a seamless user experience, supporting both functionality and accessibility across devices.

Figure ‎5‑4: GUI Components



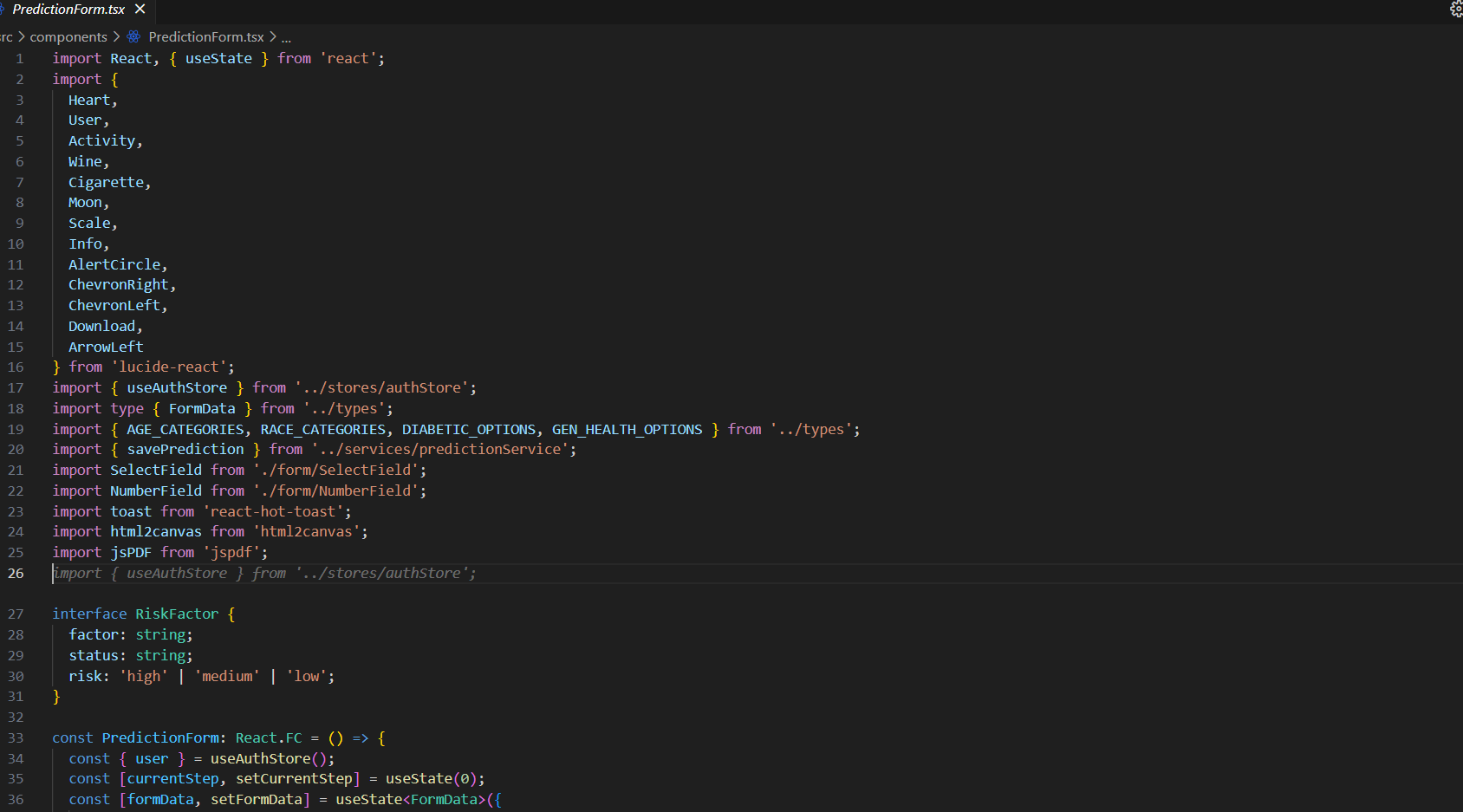
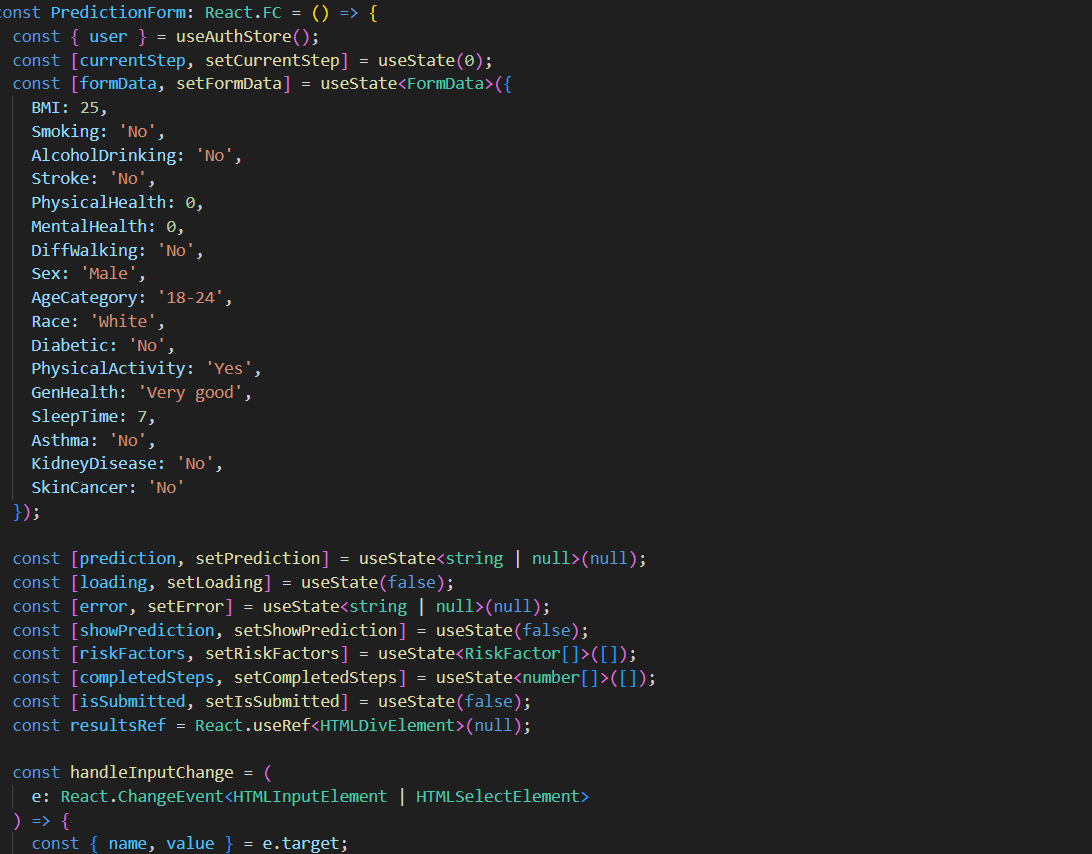


Figure ‎5‑5:predictionform.tsx



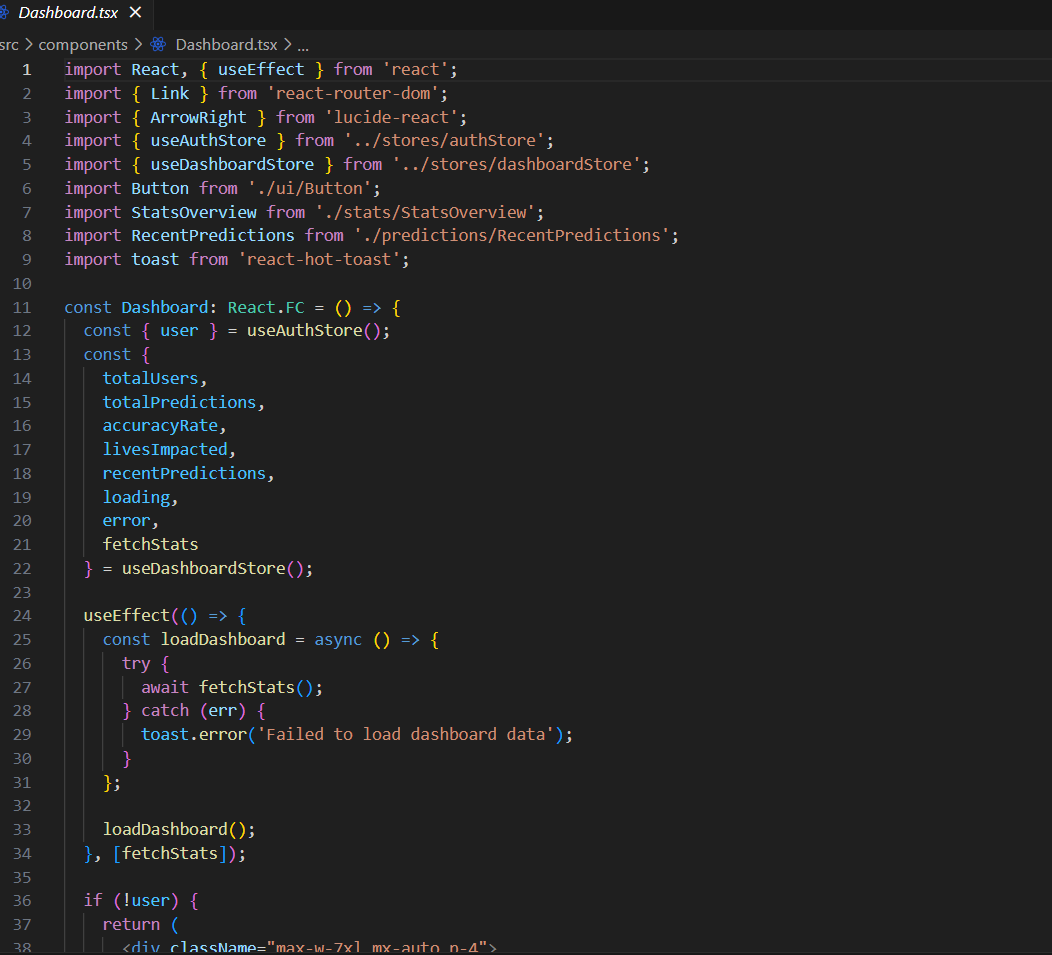


Figure ‎5‑6: Dashboard.tsx

## Other Components Implementation:

This section describes the implementation of backend services and machine learning components used in the system.

* **✅ ML Prediction API (Flask)**

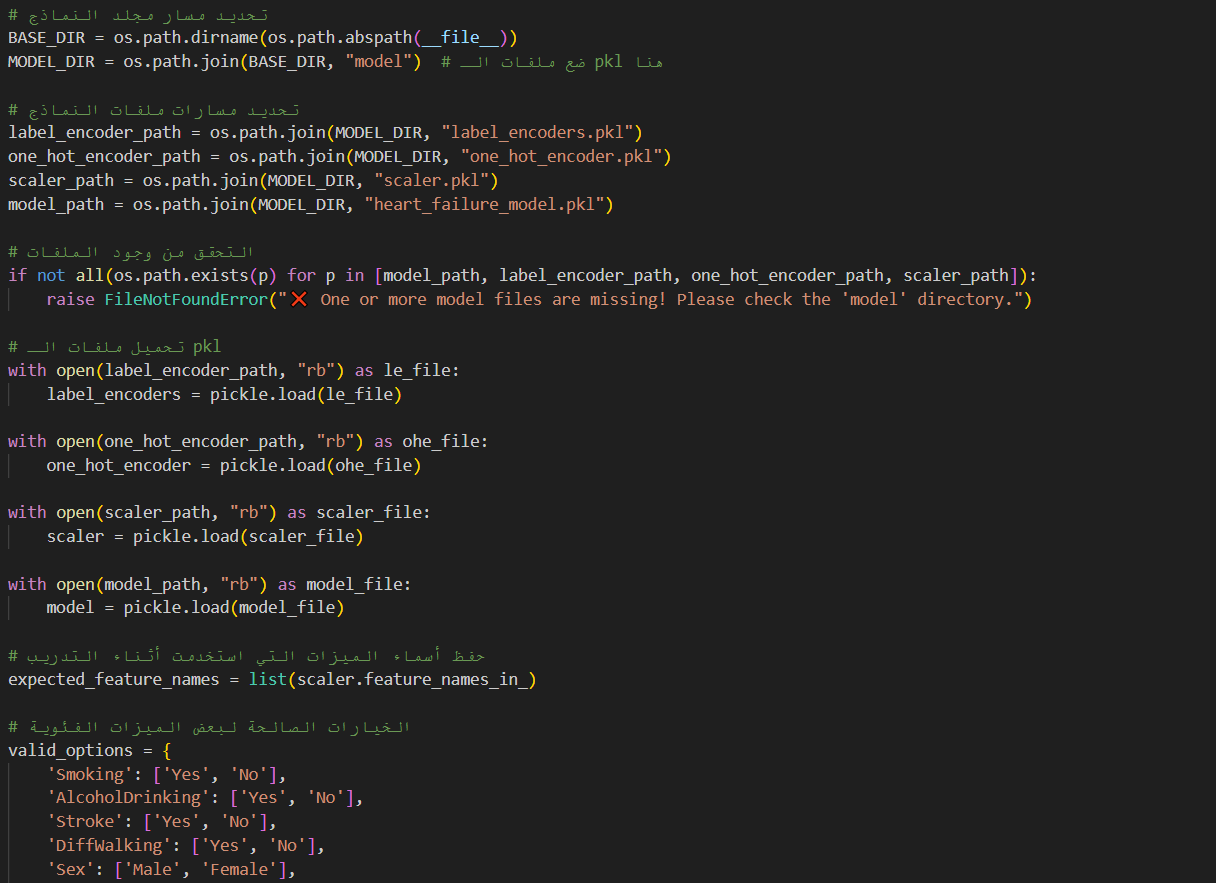
The heart failure prediction model is deployed using **Flask** and includes the following components:

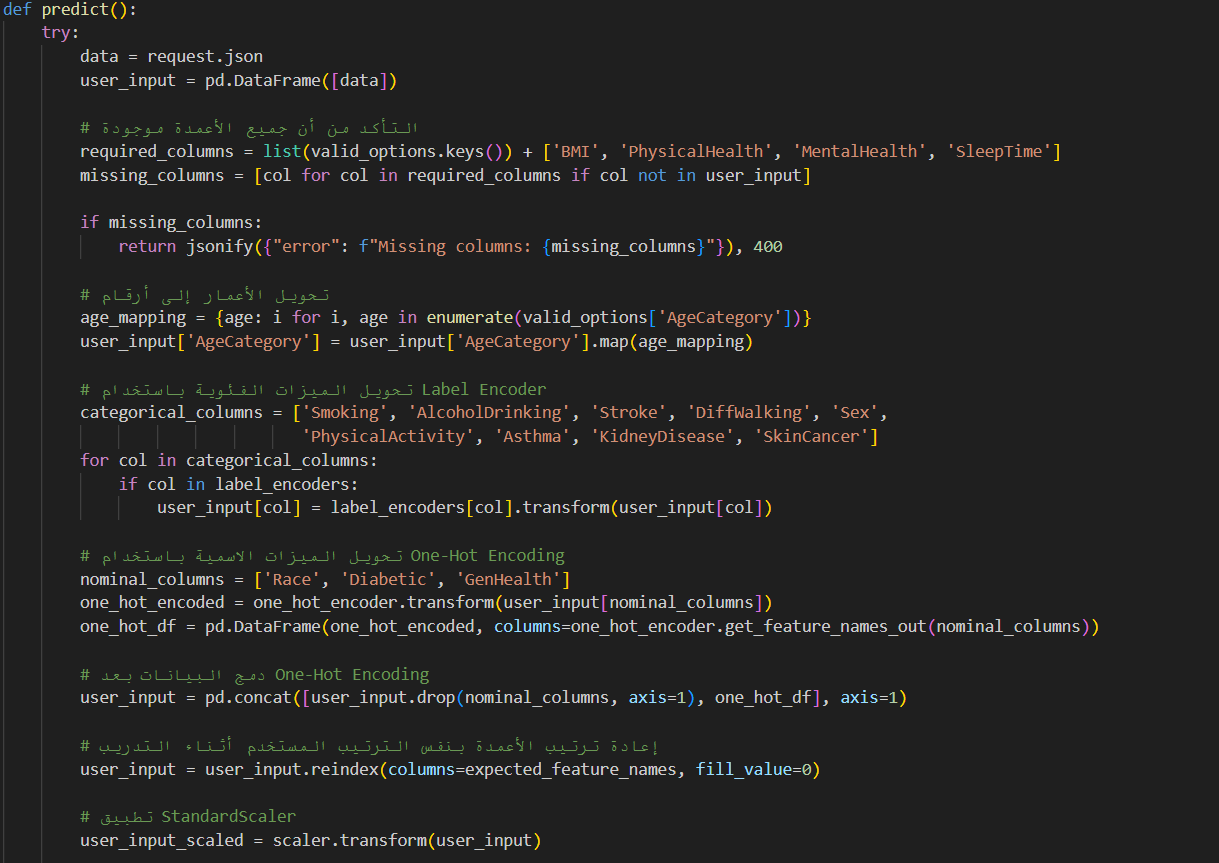
* heart\_failure\_model.pkl (trained ML model)
* scaler.pkl, label\_encoder.pkl, one\_hot\_encoder.pkl (preprocessing tools)

**API Endpoint:**  
POST /predict — Receives patient features as JSON input, preprocesses them, and returns the predicted outcome.



Figure ‎5‑7:app.py





* **✅ ECG Model Service**

The file ecg\_service.py handles the processing of uploaded ECG CSV files. It:

* Preprocesses the ECG waveform data
* Feeds it into a dedicated deep learning model
* Returns the classification result

This service supports ECG data visualization and is optimized for real-time analysis



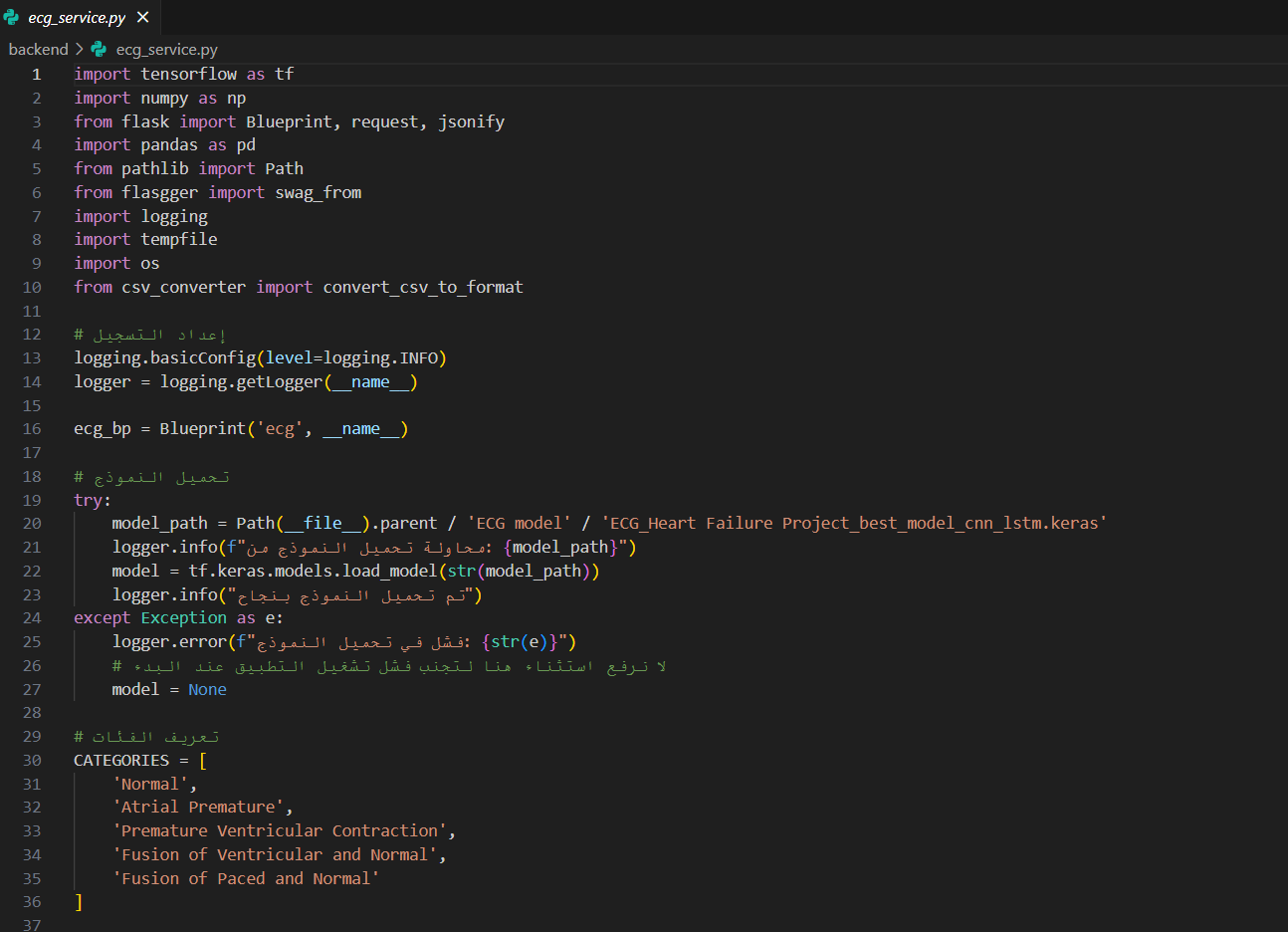
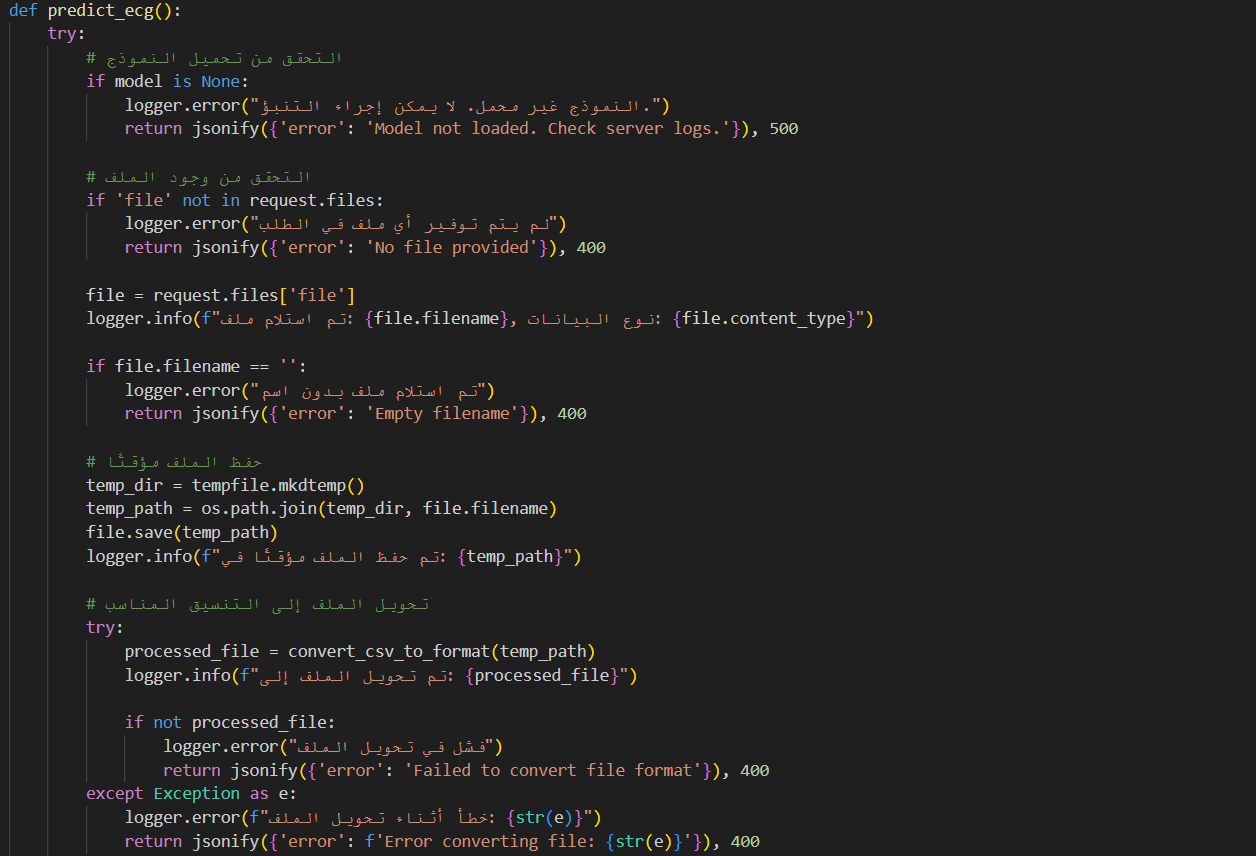


Figure ‎5‑8: ECG\_Service.py



* **✅ Chatbot Integration**

A lightweight chatbot is integrated using chatbot\_service.py. It:

* Responds to user queries with health-related advice
* Uses a rule-based engine or a pre-trained language model
* Handles predefined intents and simple prompts to provide informative replies

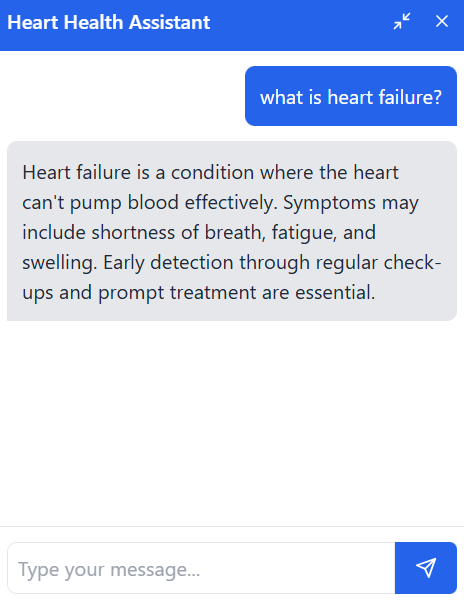


Figure ‎5‑9:Chatbot

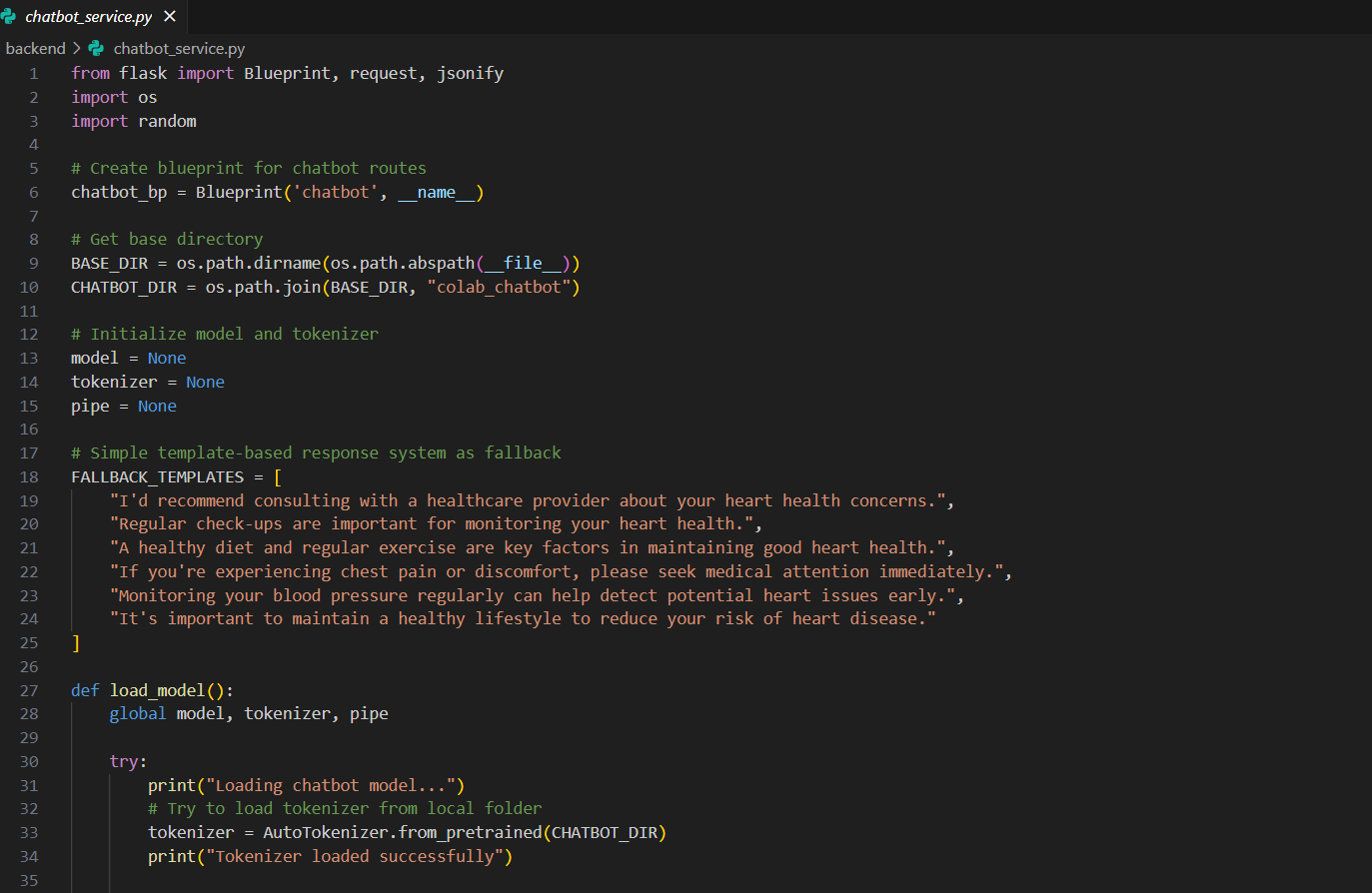


Figure ‎5‑10: Chatbot\_Service.py



## Summary

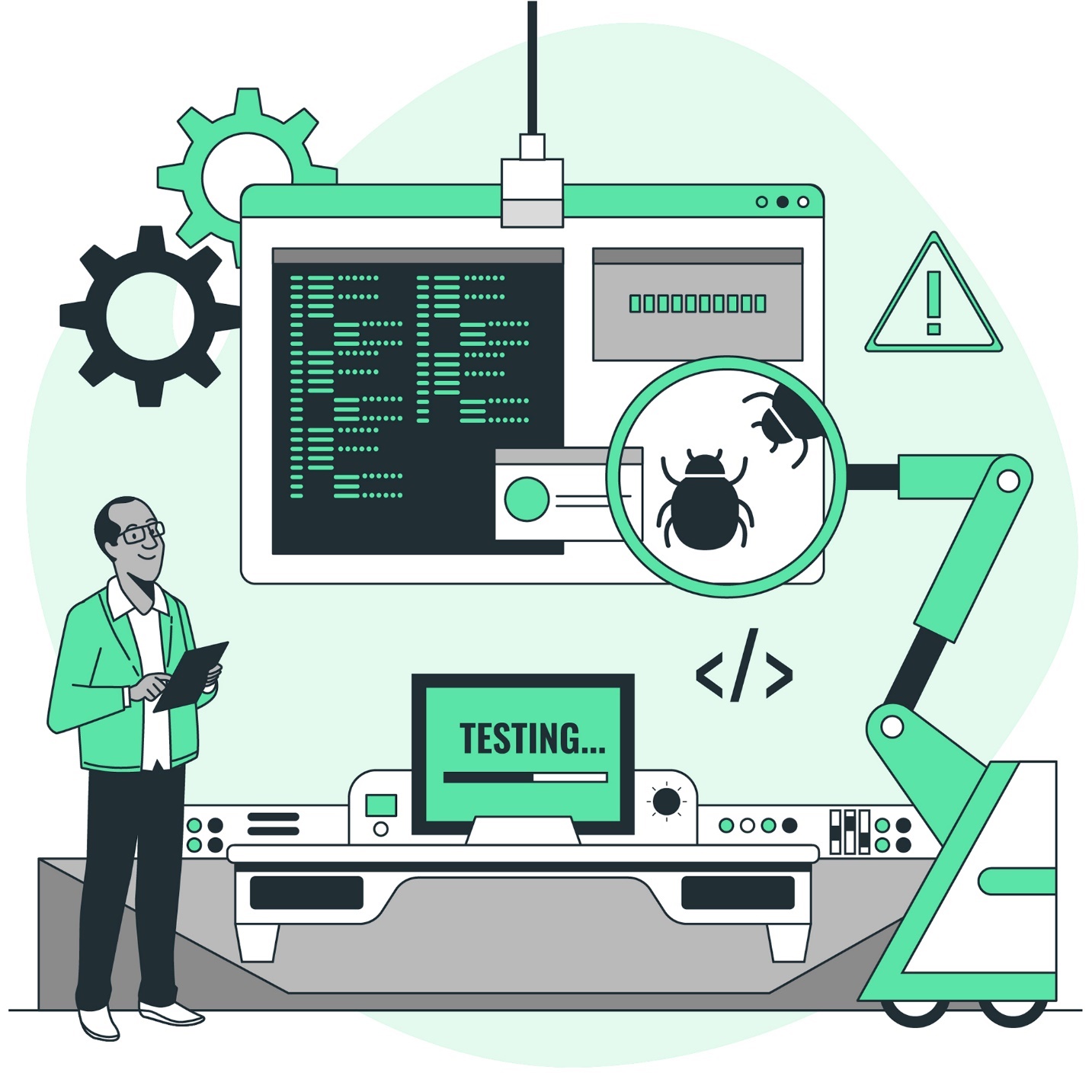
This chapter details the full implementation of the Heart Failure Prediction System, covering the frontend development, backend integration, database setup, and additional intelligent modules. The system was successfully built using a modern web development stack, including React and TypeScript for creating a responsive and user-friendly interface, Supabase for managing user authentication and cloud database storage, and Flask for serving machine learning models through secure APIs.

The implementation phase not only focuses on the core functionality of heart failure risk prediction based on clinical and lifestyle data but was also expanded to incorporate new components such as an ECG-based prediction service and a basic chatbot assistant, enhancing the system’s usability and practical value. Each part of the system was carefully integrated, ensuring seamless communication between the frontend, backend, and the machine learning components.

Special attention was given to designing a scalable and modular architecture to facilitate future improvements, such as adding more sophisticated models, integrating external health databases, or expanding the chatbot's capabilities. The use of technologies like Supabase and Flask contributed to rapid development and deployment while maintaining reliability, security, and performance.

In conclusion, the system now represents a complete, intelligent web-based application that not only predicts the risk of heart failure but also provides users with insightful interaction through advanced features. This lays a strong foundation for further developments in the second phase of the project.

# SYSTEM TESTING AND INSTALLATION



## Introduction :

System testing and installation are among the most critical phases of the software development lifecycle, especially in applications related to the medical and healthcare domains. For the Heart Failure Prediction System, thorough testing was essential to verify that the system performs its intended tasks correctly, delivers accurate predictions, and provides a user-friendly interface. Moreover, installation procedures were designed to ensure smooth deployment on any compatible environment, making the system accessible and easy to run for users and developers alike.

This chapter provides a comprehensive overview of the system’s evaluation process, focusing on two key aspects: heuristic evaluation and cooperative evaluation. These testing methods help assess the usability, effectiveness, and functionality of the system. Additionally, the chapter verifies whether all specified requirements—both functional and non-functional—have been successfully met. Lastly, this chapter documents the detailed installation and deployment procedures used to make the system operational in a local environment.

## Heuristic Evaluation :

Heuristic evaluation is a usability inspection method in which the system is reviewed by the developers or experts based on a set of well-known usability principles. For this project, the system interface, responsiveness, feedback mechanisms, and error handling capabilities were evaluated based on Nielsen’s 10 usability heuristics.

Some of the main heuristics considered during this evaluation included:

* **Visibility of system status**: The system provides clear feedback during user interactions, such as form submissions, loading states, and prediction responses.
* **Match between system and real world**: The language used in the interface is simple and familiar to users, avoiding technical or medical jargon where unnecessary.
* **User control and freedom**: The system includes options for resetting forms, editing input, and logging out, providing users with control over their actions.
* **Error prevention and handling**: Error messages are displayed clearly if required fields are missing or invalid data is entered.
* **Aesthetic and minimalist design**: The interface was designed using Tailwind CSS with minimal clutter, clean colors, and organized sections to enhance usability.

The heuristic evaluation helped identify small issues related to spacing, responsiveness on mobile devices, and label clarity. These were fixed iteratively based on testing feedback, improving the overall user experience

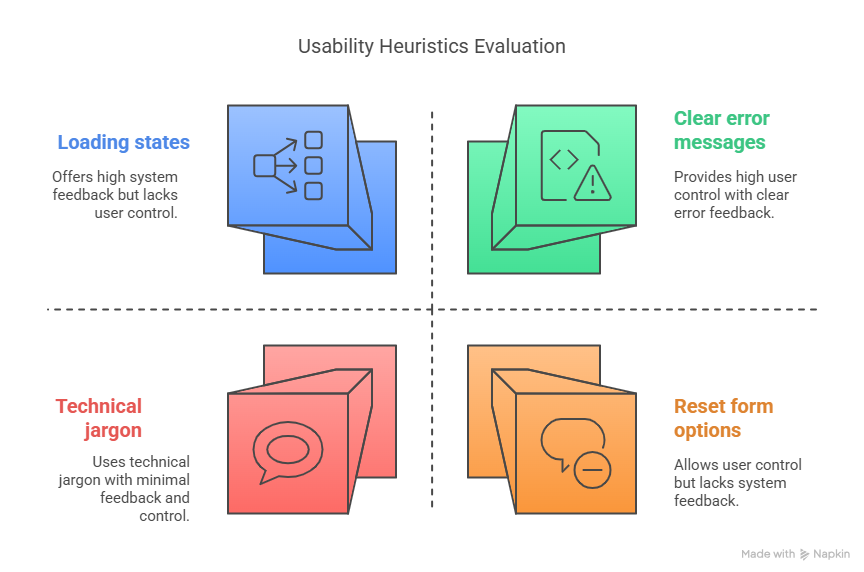


Figure ‎6‑1: Heuristic Evaluation

## Cooperative Evaluation :

Cooperative evaluation involves observing real users while they interact with the system and collecting their feedback and comments. For this system, testing was conducted with a small group of students and instructors who fit the profile of potential users. The users were asked to perform specific tasks such as registering, logging in, submitting prediction data, uploading ECG files, and using the chatbot.

The observations from the sessions revealed several important insights:

* Users found the system intuitive and easy to navigate without prior training.
* The prediction feedback was perceived as useful, especially the clear message (e.g., high or low risk).
* Users appreciated the additional features such as ECG integration and the chatbot for assistance.
* A few participants recommended adding more explanation next to each form field to help users understand medical terms like BMI, Physical Health, etc.
* One user noted a desire for downloadable reports after prediction.

Based on this evaluation, minor enhancements were made, including tooltip additions, better form spacing, and improving button labels to enhance clarity.

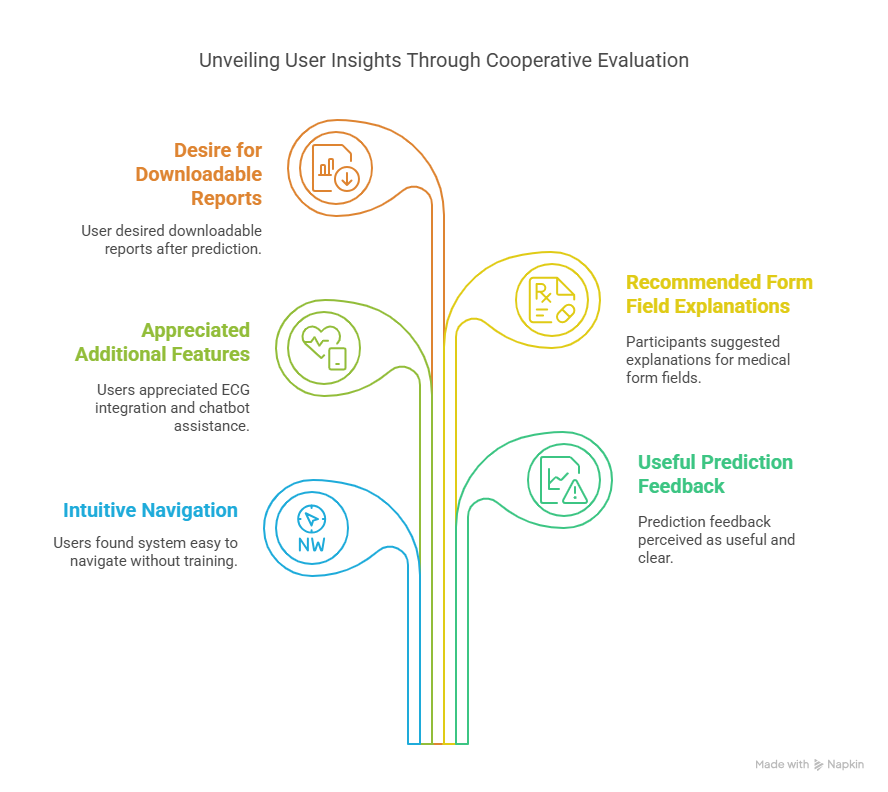


Figure ‎6‑2: Cooperative Evaluation

## Requirements Validation and Completeness :

To validate the project requirements, the original functional and non-functional requirements specified in Chapter 3 were reviewed. The following table summarizes the verification of each requirement:

Table ‎6‑1: Requirement Validation

|  |  |  |
| --- | --- | --- |
| Status | Requirement Description | Requirement Type |
| ✅ Implemented | |  | | --- | | User registration and login |  |  | | --- | |  | | Functional |
| ✅ Implemented | Submit health-related data and receive a prediction | Functional |
| ✅ Implemented | Store prediction in the database | Functional |
| ✅ Implemented | Allow ECG file upload and analysis | Functional |
| ✅ Implemented | |  | | --- | | Provide chatbot assistance |  |  | | --- | |  | | Functional |
| ✅ Implemented | Web-based, responsive UI | |  | | --- | | Non-functional |  |  | | --- | |  | |
| ✅ Implemented | |  | | --- | | Secure authentication |  |  | | --- | |  | | |  | | --- | | Non-functional |  |  | | --- | |  | |
| ✅ Implemented | Fast prediction response | |  | | --- | | Non-functional |  |  | | --- | |  | |
| ✅ Implemented | Scalable architecture for future updates | |  | | --- | | Non-functional |  |  | | --- | |  | |

## System Installation :

The Heart Failure Prediction System was implemented as a full-stack web application and deployed locally for testing purposes. The following steps were followed for setting up and running the system:

1. **Frontend Installation (React + TypeScript)**
2. **Prerequisites**: Install Node.js and npm.
3. **Navigate to the frontend project directory**
4. Run the project:

npm install

npm run dev

The system runs on: <http://localhost:5173>

**Navigate to the backend folder** and create a virtual environment :

python -m venv venv

.\venv\Scripts\activate

pip install -r requirements.txt

Run the Flask server:

python app.py

The API runs on: <http://127.0.0.1:5000>

1. **Supabase Configuration**

* Environment variables were added in the .env file.
* Tables were created via the Supabase dashboard to store predictions and user data.

**2. Chatbot and ECG Services**

* Implemented as separate Flask routes and integrated into the main API.
* Each service loads its corresponding .pkl model and handles input independently.

The entire system is modular, so each component (frontend, backend, Supabase) can be updated or replaced independently as needed.

## Summary :

This chapter has comprehensively presented the testing and installation phases of the Heart Failure Prediction System. Through a structured and systematic evaluation process, the system's functionality, usability, and performance were thoroughly examined using both **heuristic** and **cooperative** evaluation techniques. The heuristic evaluation allowed developers to inspect the system’s interface against established usability principles, identifying areas for improvement in navigation, feedback, and form design. Meanwhile, the cooperative evaluation enabled real users to interact with the system, providing valuable insights from a practical usage perspective, which led to enhancements that significantly improved the overall user experience.

Additionally, the chapter focused on the validation and completeness of the system’s requirements. By revisiting the functional and non-functional requirements defined earlier in the project lifecycle, it was confirmed that the implemented system successfully met all specifications. This verification step was crucial in ensuring that the project objectives were not only understood but accurately delivered.

The chapter also documented the full installation process for the system, detailing the setup of both frontend and backend components, as well as the integration with Supabase for database and authentication services. The installation instructions were designed to be reproducible and user-friendly, making it easy for developers or future contributors to deploy and run the system in a local environment. It also covered the steps required to launch the Flask server, React development server, and configure the environment variables, ensuring smooth interaction between all system modules.

# PROJECT CONCLUSION AND FUTURE WORK



## Introduction :

This chapter provides a comprehensive reflection on the entire project lifecycle, summarizing the achievements, challenges, strengths, and limitations encountered during the development of the Heart Failure Prediction System. It also outlines the possible directions for future enhancements and extensions. The aim is to consolidate the knowledge gained and emphasize how the work contributes to the healthcare and medical technology domains. The conclusion is drawn based on practical implementation, user evaluation, technical performance, and the success of integrating machine learning with a real-time web-based platform

## Overall Weaknesses :

Despite the successful implementation of the core system functionalities, several limitations were identified throughout the project:

* **Limited dataset control**: The dataset used was sourced from a public domain (Kaggle) and lacked real-time or region-specific medical data, which may impact the precision of the prediction for specific populations.
* **Binary prediction outcome**: The current model provides a binary result (low/high risk), which may oversimplify complex heart conditions. A more nuanced, multi-class classification or probabilistic prediction might improve decision support for clinicians.
* **Chatbot capabilities**: The chatbot assistant is rule-based and lacks natural language understanding. It can handle predefined inputs but does not use advanced NLP techniques to interpret open-ended queries.
* **No cloud deployment yet**: The system was tested and deployed in a local environment. Without deployment on a scalable cloud infrastructure, real-world accessibility and performance testing in live conditions were limited.
* **ECG model is standalone**: While an ECG-based module was implemented, it functions independently from the main clinical prediction model and has not yet been integrated into a unified decision-making process.

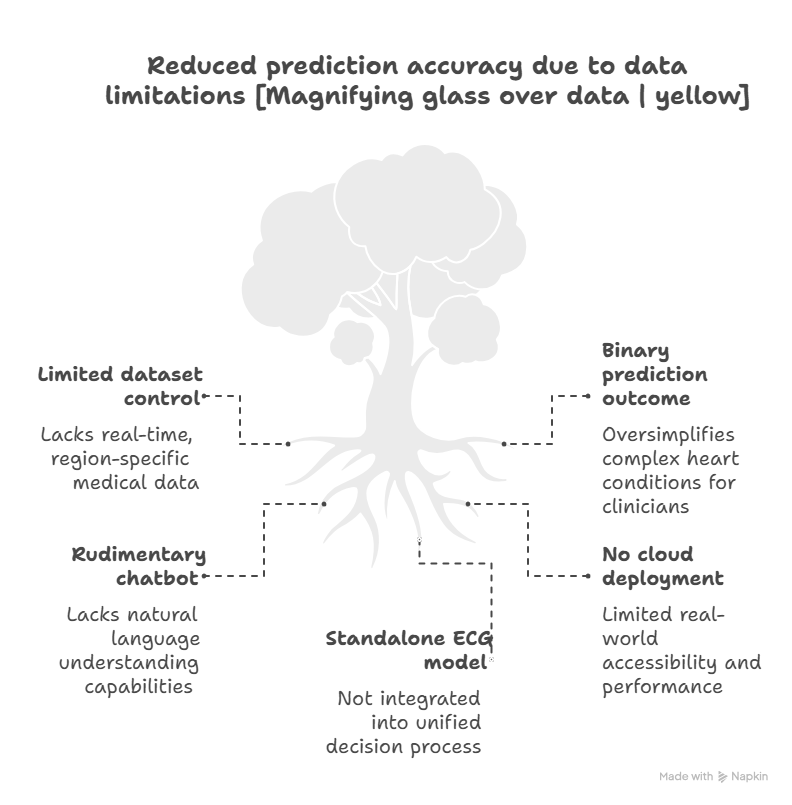


Figure ‎7‑1: Overall Weaknesses

## Overall Strengths :

Despite the aforementioned weaknesses, the project has demonstrated several notable strengths:

* **Full system integration**: The project successfully combined frontend (React + TypeScript), backend (Flask), database (Supabase), and machine learning models into a unified web-based platform.
* **User-centered design**: Emphasis was placed on creating a clean, intuitive, and responsive interface using Tailwind CSS, enhancing usability for both medical professionals and general users.
* **Modular architecture**: The system was built in a modular fashion, making it easy to maintain, debug, and extend. Each component can be updated independently without affecting the entire application.
* **Inclusion of intelligent components**: The integration of an ECG analysis tool and a chatbot marks a significant enhancement beyond basic predictive systems, bringing additional depth and interaction to the application.
* **Security and authentication**: Supabase was effectively used to manage user accounts and ensure secure data access, a critical factor in handling sensitive health-related information.
* **Data Visualization Integration**: The use of Power BI during the data exploration phase allowed for clear, interactive visual insights into patient data, supporting better understanding and feature selection for model training
* **Scalability Potential**: The system was designed with future scalability in mind, making it suitable for cloud deployment, additional data sources, or further medical functionalities with minimal architectural changes

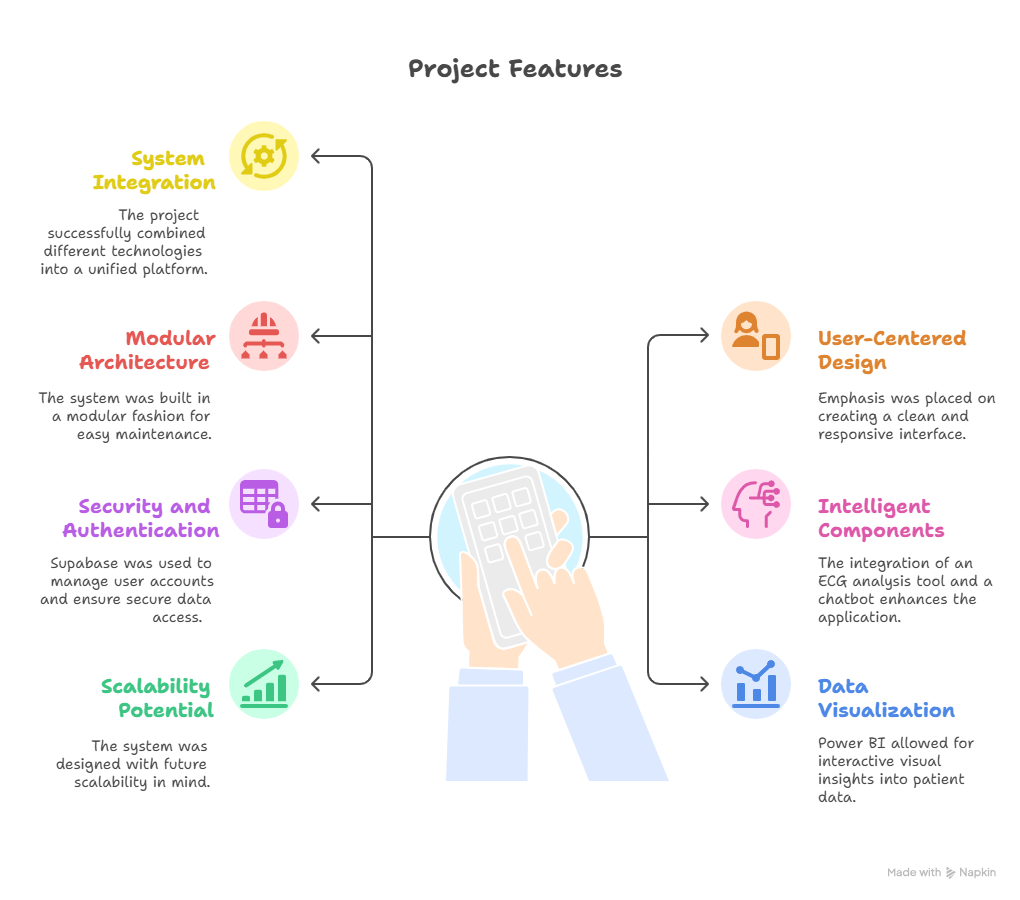


Figure ‎7‑2: Overall Strengths

## Future Work :

There are several directions for future development and enhancement of the system:

* **Cloud Deployment**: Hosting the system on platforms like AWS, Render, or Heroku to make it publicly accessible and allow for real-world usage and data collection.
* **Improved ML Model**: Enhancing the machine learning pipeline by using more sophisticated algorithms, such as ensemble models, or incorporating time-series data from ECGs.
* **Explainable AI (XAI)**: Integrating explainable AI techniques to provide rationale behind predictions, which would improve transparency and clinical trust.
* **Real-time ECG Processing**: Developing support for real-time ECG signal input and analysis, potentially from wearable devices or hospital equipment.
* **Natural Language Processing (NLP)** Chatbot: Upgrading the chatbot assistant using NLP frameworks like Rasa or spaCy to improve human-like interaction and understanding of user queries.
* **Multilingual Interface**: Offering the system in multiple languages to support broader accessibility, especially for non-English speaking users or patients.
* **Integration with EHR systems**: Exploring integration with Electronic Health Records (EHR) to automatically pull clinical data and enhance prediction accuracy based on longitudinal health data.

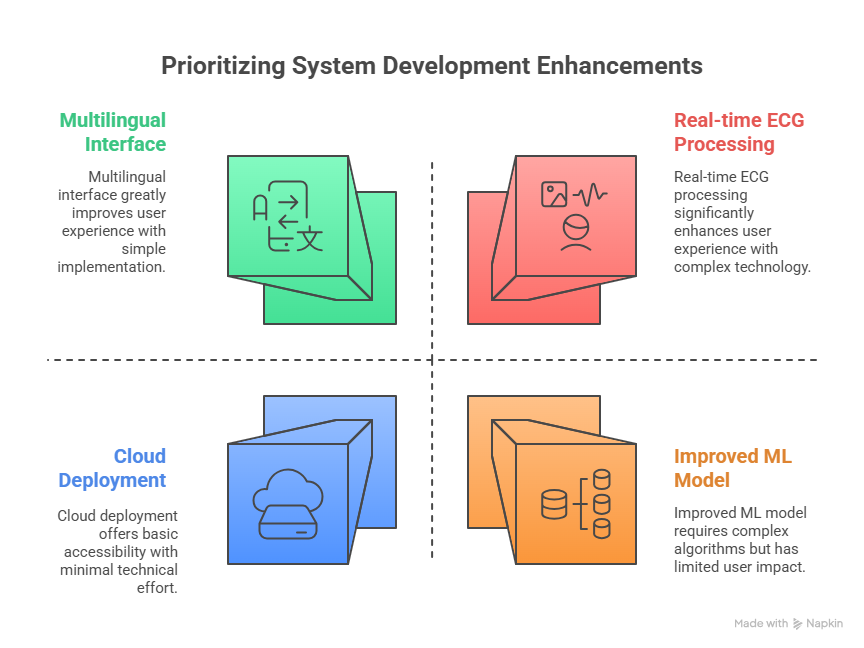


Figure ‎7‑3: Future Work

## Summary :

This chapter has served as a comprehensive reflection on the Heart Failure Prediction System, outlining the key achievements, limitations, and future directions of the project. It began by revisiting the initial objectives and analyzing the system’s real-world readiness through a critical evaluation of both technical performance and user experience.

The discussion of weaknesses revealed several areas where the system can be improved, such as enhancing the intelligence of the chatbot, integrating more nuanced prediction outputs, and expanding beyond a local deployment environment. These limitations, however, are expected and natural in the context of a first-phase prototype and offer valuable opportunities for targeted improvements.

Conversely, the strengths of the system demonstrated how the combination of a modern frontend (React, TypeScript, Tailwind CSS), a lightweight and effective backend (Flask), and reliable backend services (Supabase) can lead to the creation of a modular, scalable, and user-friendly web application. The addition of intelligent features like ECG prediction and a chatbot assistant further elevated the system's capabilities and positioned it as a forward-looking healthcare tool.

In terms of future work, the project holds strong potential for continued development. Suggestions include deploying the system on the cloud, integrating real-time ECG data, applying explainable AI techniques, and strengthening the chatbot with natural language processing capabilities. These enhancements would not only improve technical functionality but also extend the practical usefulness of the system in clinical or telemedicine settings.

Overall, the project succeeded in building a solid foundation for a machine learning-based medical decision support system. It brought together multiple technologies, adhered to user-focused design principles, and maintained clear goals toward improving healthcare prediction. This achievement marks a major milestone in the first phase of the project and opens doors for further innovation in the second semester and beyond.

# REFERENCES



1. Pytlak, Kamil. *Personal Key Indicators of Heart Disease Dataset*, Kaggle. Available at: <https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease>
2. Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825-2830.
3. Hunter, J. D. (2007). *Matplotlib: A 2D Graphics Environment*. Computing in Science & Engineering, 9(3), 90-95.
4. Flask Documentation. https://flask.palletsprojects.com
5. Supabase Documentation. https://supabase.com/docs
6. React Documentation. <https://react.dev>
7. Tailwind CSS Documentation. <https://tailwindcss.com/docs>
8. XGBoost: A Scalable Tree Boosting System. (2016). *Proceedings of the 22nd ACM SIGKDD*. <https://arxiv.org/abs/1603.02754>
9. Power BI Official Docs. <https://learn.microsoft.com/en-us/power-bi/>

*Additional references, such as articles, lecture slides, or tools used in development, may be added here*

## APPENDIX A: Flask API Sample Response :

POST /predict

Request Body:

{

"features": [50, 25, 1, 0, 0, 3, 5, 1, 1, 2, 1, 0, 1, 1, 6, 0, 0, 1]

}

Response:

{

"prediction": 1

}

## APPENDIX B: .env Sample Configuration :

VITE\_SUPABASE\_URL=https://your-project.supabase.co

VITE\_SUPABASE\_ANON\_KEY=your\_supabase\_anon\_key

## APPENDIX C: Directory Structure :

## 

project/

├── backend/

│ ├── app.py

│ ├── model/

│ │ ├── heart\_failure\_model.pkl

│ │ ├── scaler.pkl

│ │ ├── label\_encoder.pkl

│ │ └── one\_hot\_encoder.pkl

│ └── requirements.txt

├── src/

│ ├── components/

│ ├── pages/

│ └── ...

├── .env

├── package.json

└── index.html