Tribhuvan University

Institute of Science and Technology

A Final Year Project Report

On

“MediAssist”

Submitted to:

Department of Computer Science and Information Technology

Academia International College

Gwarko, Lalitpur, Nepal

Submitted by:

Sharada Luitel (26517/077)

Krish Kumar Shrestha (26493/077)

Under the supervision of

Mr. Ananda Adhikari

Tribhuvan University

Institute of Science and Technology

Academia International College

Department of Computer Science and Information Technology

Email: mail@academiacollege.edu.np

Supervisor's Recommendation

The project work report entitled 'MediAssit" submitted by Sharada Luitel and Krish Kumar Shrestha, of Academia International College, is prepared under my supervision as per the procedure and format requirements laid by the Faculty of B.Sc. CSIT, Tribhuvan University, as partial fulfillment of the requirements for Bachelor of Science in Computer Science and Information Technology (B.Sc. CSIT). We, therefore, recommend the project work report for evaluation.

.....

Mr. Ananda Adhikari
Academia International
College B.Sc. CSIT

Tribhuvan University

Faculty of Computer Science Information Technology

Academia International College

Letter of Approval

This is to certify that this project prepared by Sharada Luitel and Krish Kumar Shrestha entitled "MediAssist" in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology has been well studied. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

…….. Program Coordinator Academia International College	…….. Project Supervisor Academia International College
…….. Internal Examiner Academia International College	…….. External Examiner

ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude to all those who have supported and guided us throughout the development of this project.

First and foremost, we are deeply thankful to our project supervisor **Mr. Ananda Adhikari**, whose expertise, encouragement, and suggestive feedback were precious throughout this research journey. His guidance has greatly enhanced our understanding of the subject matter and ensured the successful completion of this work. We also pay our grateful thanks to CSIT coordinator **Mr. Bishwas Mathema**, for his encouragement and feedback which helped us to track and schedule the process effectively. We also extend our sincere thanks to all the faculty members and the non-teaching staff for creating the best environment to carry out the project smoothly.

We are also grateful to our institution, **Academia International College** for providing the necessary resources and a helpful environment for learning and exploration.

A special thanks to our friends and peers for their invaluable discussions and encouragement, which helped shape this project in meaningful ways.

Finally, we extend our deep appreciation to our family, whose unwavering support and motivation have been our greatest strength throughout this process.

Thank you all for being an integral part of this journey.

ABSTRACT

The MediAssist is a web-based system used to assist user in identifying probable health diseases before it's too late. The frontend of the system is powered by HTML, CSS and the backend is built with flask while different Python libraries is also included during the development cycle, Support Vector Classifier (SVC) is used for efficient disease prediction and classification of the input data. The process of getting started is easy as users have to complete a basic registration form with their personal data. The user inputs the symptoms, and are matched by the system against a dataset of disease and predicts probable disease and recommends suitable precautious measures. This application aims on preventing users from possible health hazards that they might get themselves into if early diagnosis of those is ignored. With the technological combination of different web development tools, machine learning algorithms and user-friendly design it ensures easy accessibility, efficiency and accuracy.

Keywords: Flask, Support Vector Classifier, Classification, Early Diagnosis, Machine Learning

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATION

AI: Artificial Intelligence

CSS: Cascading Style Sheet

DFD: Data Flow Diagram

ERD: Entity Relation Diagram

HTML: Hyper Text Markup Language

ML: Machine Learning

SVC: Support Vector Classifier

UI: User Interface

CHAPTER 1: INTRODUCTION

1.1 Introduction

Healthcare is an essential service that impacts the lives of people worldwide. It is critical to provide the right treatment to patients as fast as possible, and the success of such treatment depends on the accuracy of the medication prescribed. Over the years, the healthcare industry has made significant progress in medical research, diagnostic equipment, and other medical technologies. However, one area that has been overlooked is the health care system, which can improve healthcare outcomes and reduce healthcare costs.

In the context of Nepal there are some special challenges for the Nepalese healthcare system especially those that are quite remote and rural they lack doctors. Moreover, there is a great divide and differentiation on the healthcare system some areas have advanced hospitals, while others have mere clinics. These factors create a problem in the provision of standard and quality treatment across the country. The deliver of standardized care is already challenging because of the disease type and the patient's cultural background, economic status, and overall health.

MediAssist is a web-based application that aims to provide medical facility to their user with better awareness and predicting possible disease that user maybe suffering from based on the symptoms that user have provided. It offers a comprehensive guide to managing health effectively.

MediAssist aims to serve as a virtual healthcare companion for Nepalese people, particularly in rural areas where professional medical advice is scarce. It helps user aware about the possible danger in future and offers different suggestions like diets, workouts, precautions and medications with the help of available and accessible internet and technology.

1.2 Problem Statement

There are several head-scratching issues that prevails in Nepalese healthcare system that makes it hard in delivering uniform and qualitative treatment and care across the nation. Some of these factors include limited health workforce, especially in the rural setting, poor density of qualified doctors, lack of fundamental and emergency drugs, and inadequate medical facilities.

The common problem that people are facing in Nepal are as follow:

- Limited access to healthcare facilities in rural and remote areas.
- Shortage of medical professionals, including doctors and nurses.
- Lack of awareness about preventive healthcare and early diagnosis.

1.3 Objectives

The main objective of MediAssist can be summarized as follow:

- To analyze user-provided symptoms using machine learning algorithm and accurately predict potential disease.

- To improve patient safety by awaking users of what they might be suffering from.
- To elevate healthcare quality by suggesting diets and workouts to the users.

1.4 Scope and Limitation

1.4.1 Scope

The scope of MediAssist aims at enhancing health and spread awareness amongst the people of Nepal via a web-based application. The goal of this project is to help people to forecast what diseases can they get depending on their symptoms and give proper health advices.

MediAssist will deliver the following functionalities:

- Disease prediction based on symptoms provided by the user.
- Recommendations for preventative steps such as diet and exercises for the sick depending on symptoms.
- Useful knowledge about various sign/symptoms and precautions of the same disease.

1.4.2 Limitation

The limitation of this project are as follow:

- **Limited dataset:** The system works with a small number of symptoms, 131 of them, and 41 diseases which limits its potential.
- **Lack of professional insight:** While system predicts and recommends, it can't replace actual doctors and their diagnosis. Misinterpretation of systems suggestions without expert's consultation is not recommended.
- **Technological limitation:** The platform faces difficulty if there is no stable internet connection as it is web-based application.
- **Dependence on user engagement**: The systems accuracy depends on user's willingness to provide accurate symptoms.

1.5 Development Methodology

The project's development and implementation were guided by the Agile methodology [1]. This approach emphasized iterative progress, continuous feedback, and flexibility, allowing us to adapt to changing requirements and evolving needs. By integrating Agile practices, we ensure the system evolves effectively, promptly addresses emerging issues, and consistently meets user needs throughout the development lifecycle.

Requirement Gathering and Analysis

In the initial phase of project requirements are collected from various sources. For the functioning and features of the system requirements were essential to be defined earlier. Requirements were datasets, different technologies used in project, precautious dietary and workout advices inputs from the user. The main goal is to be accurate as possible and suggest best possible medications, diets, workouts ensuring user safety.

Design

After requirement gathering phase was completed design phase began. Structuring the system into functional modules and outlining system architecture and detailed database

design. This phase ensures that all components work together seamlessly to deliver an efficient system. The design focuses on defining the core architecture, detailing the interactions between different modules such as data processing, recommendation generation, and user management. The database design is carefully planned to store and retrieve information efficiently. Additionally, the user interface (UI) is designed to be simple and accessible for various users. Overall, the design phase lays the foundation for a system where all components integrate seamlessly, ensuring a smooth and efficient user experience.

Development

In development phase the actual coding takes place. In this phase each and feature is developed in small and manageable iterations. Like building UI, training model, creating recommendation system, connection to database and many more. This approach of working in small iterative module helps ensuring functioning and testing of components easy and fast.

Testing and Integration

In this phase we integrate the developed components into a system and test the functionality of the system to ensure that our system is working properly. Unit Testing and System Testing were carried out simultaneously as every feature is implemented and integrated. Testing ensures that system works effectively and efficiently under various conditions.

Deployment

After completion of the testing and integrating next phase namely deployment begins. This phase includes hosting the developed system making sure it is accessible for all the users. The system is implemented for actual use and feedback is collected from users. This feedbacks works as information regarding system's future improvement.

Maintenance

After deployment we need to update and fix bugs regularly. Maintenance helps keep system functionality up and running. It also includes adding new features, optimizing system, fixing bad user experience.

1.6 Report Organization

The report has been prepared by following the guidelines as per Tribhuvan University. This report is separated into six different chapters. Each chapter is divided into sub chapters.

The preliminary section includes Title page, Acknowledgement, Abstract, Table of Contents, List of Abbreviations, List of Figures and List of Tables.

The main report is divided into six chapters, which includes:

Chapter 1: Introduction

The first chapter begins with giving information about our project, its background, the healthcare situations in Nepal and the problem faced. It states the objectives and defines scopes and limitations of project. It also explains the methods used while working on its creation. Moreover, it helps to determine the structure of the report.

Chapter 2: Background Study and Literature Review

This chapter includes study of current scenarios and trends as well as study of systems and similar works of another researcher. it also focuses on providing theoretical background of the project. It explains how the MediAssist project integrates with prior work and aims at enhancing health care availability in Nepal.

Chapter 3: System analysis

This chapter focuses on the system analysis phase where the functional and non-functional requirement is discussed. It also includes feasibility analysis like technical and operational, cost and time, which serve as key success points.

Chapter 4: System design

Chapter 4 focuses on the design phase, where the architecture of the system is planned. This includes the database design, the transformation of ER diagrams, and the design of user interfaces. The chapter also covers the algorithms used in the project that enable prediction, recommendations, and suggestions in the project.

Chapter 5: Implementation and Testing

In this chapter, the implementation process is discussed in detail, covering the tools, technologies, and platforms used to build. It provides an overview of the development process involved. Furthermore, the chapter describes the testing process, including unit testing, integration testing, and system testing, along with an analysis of the results to ensure the system functions as expected.

Chapter 6: Conclusion and Future Recommendations

The final chapter provides a summary of the key findings and accomplishments of the project. It reflects on whether the project objectives were met. The chapter also outlines recommendations for future work and improvements to enhance the system's capabilities, expand its reach, and ensure continued effectiveness in improving health outcomes.

The final part of the report consists of References and appendices. The references are listed in accordance to IEEE referencing standards and appendices includes the screenshots of the system and major code snippets.

CHAPTER 2: BACKGROUND STUDY AND LITEREATURE REVIEW

2.1 Background Study

In today's fast-moving world, many individuals face critical health risks due to delays in diagnosis and inadequate access to medical care. Hectic schedules often cause people to dismiss early warning signs of illness, while those living in rural or underserved areas encounter significant challenges such as limited healthcare facilities and the need to travel long distances. Even in urban settings, overcrowded hospitals and extended waiting times deter people from seeking medical attention promptly. These delays often result in diseases being detected at advanced stages, leading to more complicated treatments, higher costs, and preventable fatalities. For patients and their loved ones, this can lead to deep emotional distress, regret, and frustration, underscoring the pressing need for solutions that make healthcare more accessible.

Historically, receiving medical care involved in-person visits, prolonged waits for appointments, and drawn-out diagnostic procedures. These processes were further hampered by obstacles such as geographical limitations, irregular doctor availability, and inflexible schedules. Manual systems often worsened these issues, making timely treatment difficult.

Everywhere like in Nepal, the healthcare could be a nightmare especially in rural and isolated regions. Nepal consists of different geographical terrains a number of which include mountainous areas and the far western region villages where there is little access to even basic health care provisions. In these places, it is a norm for people be speaking hours or even days, on whatever transport they can access, just to be able reach a heath facility. This inaccessibility to appropriate healthcare, the relative scarcity of health and medical services human resource and facilities is a major hindrance to early access to health services.

According to the study, nearly **66,000** people in Nepal die [2] every year due to insufficient basic health care facilities. They die due to diseases that, if diagnosed early would have been prevented or that could have been controlled. For example, sicknesses such as hypertension, diabetes and respiratory disorders, which plague the population in the region, would easily manageable once diagnosed. However, lack of health-care and health information leaves those people untreated and their conditions deteriorate. In many cases, they die as well. Also, approximately **19000** people die every year from being unable to access and receive proper medical care.

Yet another major factor that increase mortality rates is most patients avoid consulting a doctor thinking that minor symptoms should derogate on their own. They are especially common in the rural settings as people there have low concerns regarding health disorders and need for early treatment. However, not giving attention to these signals may be dangerous. Lots of diseases that could have been averted eventually turn into severe complications as a result of this delay in diagnosis. For example, a child with flu, cough, or tiredness could mean the child has developed a severe disease like tuberculosis or pneumonia. Without intervention, the above-mentioned conditions are some of the deadly that can be confronted.

This ignorance of early symptoms is not only wanting in diseases like respiratory infections but in chronic diseases like cardiovascular diseases and cancer where symptoms often manifest themselves faintly. These diseases are in most cases diagnosed at a later stage when normal treatment can hardly be done, and therefore the prospects of the patient's survival are slim. In such circumstances, there are those who would have otherwise been treated if they were attended to early are left to die. Using the phrase "Prevention is better than cure" the above-mentioned situations support this phrase. Most of these cases could be averted by early intervention or routinely examination and increase in physical activities, but since people are not well informed and cannot afford hospital bills, this kills them.

Today, however, technology is transforming healthcare. Digital tools now provide faster, more convenient access to medical resources, enabling individuals to monitor their symptoms, take charge of their health, and identify potential illnesses at an early stage. Machine learning, especially through the use of Support Vector Classifier, is playing a essential role in improving disease prediction. By analyzing symptoms reported by users, these algorithms generate precise and data-driven diagnostic insights, paving the way for earlier interventions and better overall health outcomes. Although it can't provide user with healthcare services it can provide diet suggestions and workout suggestions which up to some extent can reduce the severity of the disease and make people safe and prevent from untimely death.

2.2 Literature Review

The adoption of machine learning and artificial intelligence in healthcare has seen remarkable advancements over the years, offering solutions to some of the industry's most pressing issues, such as the early detection of diseases, efficient management of healthcare services, and greater accessibility to medical support. Among these advancements, disease prediction systems have emerged as critical tools, designed to analyze symptoms, patient medical history, and other health-related inputs to accurately predict potential health conditions before they become severe. These systems empower healthcare professionals to make informed decisions, leading to better patient outcomes.

A wide range of machine learning algorithms is used to power these systems. For instance, Support Vector Machines [3] excel in classifying complex datasets by finding the optimal decision boundaries. Naïve Bayes classifiers are also known for their simplicity and effectiveness in handling categorical data, making them suitable for symptom-based predictions. Decision Trees provide a transparent and interpretable way to make predictions by following a logical flow of decisions based on input features. Meanwhile, Neural Networks leverage their ability to recognize intricate patterns in large datasets, making them particularly effective in more complex diagnostic scenarios. These algorithms collectively contribute to building robust and reliable disease prediction systems, transforming the way healthcare is delivered and improving early intervention opportunities.

Due to flexibility that allows it to learn from different datasets, machine learning models have bear great promise for disease prediction. For example, Balakrishnan Duraisami , in their study in 2024 showed that the proposed model, Support Vector Machines (SVMs) are useful tools in identifying chronic diseases including heart disease. Working on the same, their study pointed out that SVMs not only provide high accuracy but also outcompeted

other statistical methods for diagnosing the primary stage of heart-related complications [4]. This credit goes to the fact that SVM has the capability to determine the best decision planes in most of the cases as well as in the higher order feature space.

Likewise, Himesh et al. [5] employed the Naïve Bayes classifiers to establish diabetes predetermination by entering user's symptoms. While explaining the algorithm, the authors noted that categorical data was easy to work with and the manipulation of the algorithm was easy to accomplish and the accuracy which reached 85% spoke for itself. This makes it useful for application in symptom-based prediction systems.

Lisha Kurian conducted significant research on medicine recommendation systems [6] using machine learning algorithms, emphasizing patient-specific factors. Their study utilized machine learning to extract patient data and provide personalized health advice, highlighting the importance of tailoring recommendations to individual patient needs. However, the study's evaluation of algorithmic explainability was limited. Ms. Silpa and her team [7] developed a system designed for use in medical emergencies, employing a machine learning approach similar to ours. Their system utilized a Random Forest classifier to provide real-time medical assistance, especially during emergencies.

In addition, more complex machine learning techniques known as the Convolutional Neural Networks (CNNs), and Long Short-Term Memory (LSTM) networks have advanced the ways in which complex medical data such as medical images and sequential health records can be processed. Chen et al Zhang et al. [8] provided more evidence to what these models are capable of in their study by training the model to obtain a stunning a 92% accuracy on lung diseases. The system that was developed meant that patient reported symptoms was linked with images and this was shown that deep learning was capable of bridging to structured and unstructured data in order to provide correct diagnoses. Each of these advancements is indicative of the revolutionary role that machine learning algorithm may play in present day healthcare delivery.

Another approach designed based on Patel et al. function in an original method that involves the use of an ontology-based system to map the symptoms reported by the patient to the right medical specialist. This system utilized a taxonomic model [9] built from medical database knowledge that directly mapped signs into the related fields of specialty. This paper noted that the given approach helped to help users increase satisfaction with the site since the recommendations given were correct and up to date. They are especially helpful when patients cannot decide which specialist to see in that it helps reduce the time taken to seek medical attention and make decisions. Jianguo Chen and his team proposed a disease diagnosis and treatment system that utilizes data sharing and treatment intelligence. Their work focuses on the integration of data-driven insights to enhance treatment accuracy and efficiency, contributing to the broader field of intelligent healthcare systems.

Collaborative filtering systems leverage patient feedback and historical data to make doctor recommendations, analyzing patterns from user experiences to identify suitable healthcare providers. Hybrid systems, on the other hand, integrate collaborative filtering with content-based approaches, combining patient preferences with specific doctor attributes, such as specialization or experience. Gupta and Singh [10] demonstrated the superiority of hybrid systems, showing that these approaches achieved higher accuracy and patient satisfaction compared to standalone systems. By blending personalized insights with comprehensive

medical knowledge, hybrid systems provide recommendations that are not only accurate but also tailored to individual patient needs.

IBM Watson for Oncology [11] is a prime example of a real-world implementation where machine learning is used to recommend cancer treatment options. The system analyzes patient data against a vast database of medical literature to suggest treatment plans, helping oncologists make informed decisions. Similarly, AiCure [12] is an AI-based platform used in real-world clinical trials to ensure medication adherence. It uses computer vision and machine learning to monitor patients' adherence to prescribed medications, thus indirectly supporting personalized medicine recommendation by ensuring the correct implementation of treatment plans.

CHAPTER 3: SYSTEM ANALYSIS

3.1 System Analysis

This section outlines the purpose and functionality of our system, emphasizing the critical role of requirements in shaping its design and capabilities. The requirements define the system's structure, features, and different constraints, serving as the foundation for the entire development process. Identifying and gathering requirements is often difficult due to their changing nature. As development progresses, user needs may differ from initial needs, leading to adjustments in requirements. Moreover, the interdependencies among requirements creates difficulties as modifications to lower-level requirements can influence higher-level ones and vice versa.

3.1.1 Requirement analysis

The system requirement is classified into two types functional requirements and non-functional requirements.

I. Functional Requirements

- **User Registration and login**
  User can register using their email and create account once they create account, they can login using same credentials and only then they can use the system.
- **Search for Symptoms**
  Once user is logged in one can select symptoms from the list and then they can request model to recommend possible disease they've been suffering from.
- **Information about medication, diet, workout**
  Once disease is diagnosed then user can view related precaution, workouts and diets related to the disease.
- **Contact developer**
  User can send message to the admin simply by visiting contact us page. Users can provide with their valuable feedbacks and suggestions for future improvements.
- **Managing users and feedbacks**
  Admin can view list of the users from the admin dashboard which is accessible to admin only and admin can also view the messages from the different users.
- **Log out**
  User can log out securely once they are done using the system by terminating the user session.

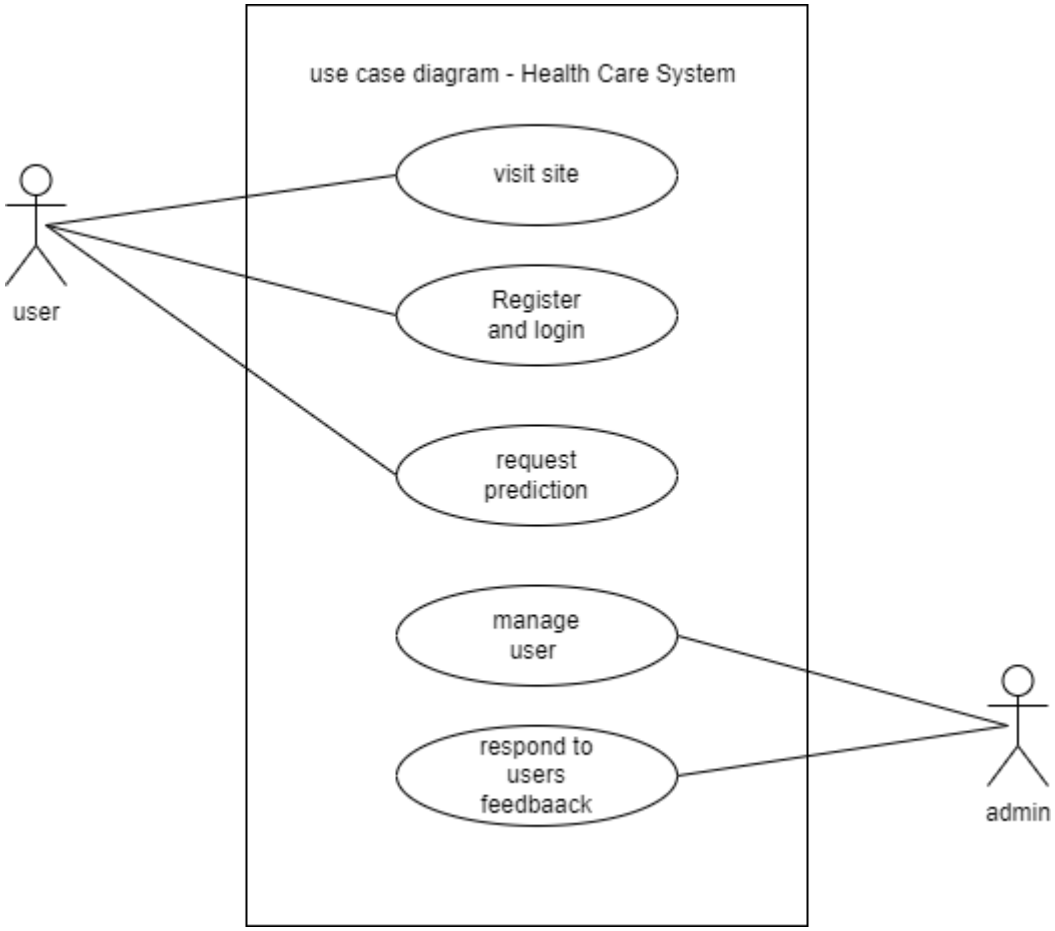

Figure 3.1.1: Use Case Diagram for MediAssist

II. Non-Functional Requirements

- **User friendly**
  The system is user-friendly enough to such extent that user with basic knowledge about how to use and surf internet can use this system smoothly.
- **Speed**
  The system's response time is fast. During the time of testing, we felt no lagging in between users input submission and result generation.
- **Availability**
  The possibility of system failure is very low and the system is available all the time as required by the user.
- **Security**
  This system is protected by firebase authentication making user login and accessing system secure and reliable.
- **Scalability**
  We can integrate more features and moules later on in this project making it flexible and scalable.

3.1.2 Feasibility Analysis

Feasibility analysis is a critical step in project planning and decision-making. It involves systematically analysis of likelihood of a project's success by examining various factors

like technical, operational, economic, schedule that could impact its implementation. The goal of feasibility analysis is to ensure that the proposed project is realistic, achievable, and worth pursuing before committing significant resources. Following feasibility analysis were performed before the working on the project:

I. Technical Feasibility

The technical portion of a MediAssist involves assessing the availability of necessary hardware, software and algorithms, to develop and operate the system effectively. This includes evaluating the technology stack, such as python, JavaScript, html, CSS, flask, firebase, machine learning algorithm such as support vector classifier which assists in separating data using linear regression allowing to predict more accurately, ensuring access to reliable of model and application. This web application is supported by almost every latest web browser and most of all it can run on any modern operating system.

II. Operational Feasibility

MediAssist setup is operationally feasible since it is in accordance with the increasing the probability of positive outcomes. The system can be easily implemented within the current environments without disruption. Its friendly user interface means easy to use for everyone. Furthermore, the system covers vital functional requirements as mention in above chapter. This system seems useful in actual health care sector and that is why the paper sees it as useful and it may be helpful to improve the health care delivery.

III. Economic Feasibility

The MediAssist is economically appealing for user because it has the potential to reduce healthcare costs while improving efficiency. For users, this means access to more affordable and effective healthcare solutions. The system's initial setup costs are kept low since it leverages open-source datasets and software, which helps minimize development expenses. For users, this means to a free and sustainable solution that prioritizes user health without compromising quality.

IV. Schedule

Table 3.1.1: Schedule for Project

Activity	Start Date	End Date
Planning and Requirement Gathering	8/4/2024	8/8/2024
System Design	8/9/2024	8/15/2024
Research and Collect Data	8/9/2024	8/26/2024
Coding	8/27/2024	12/25/2024
Testing	10/21/2024	10/26/2024
Documentation	8/4/2024	12/28/2024

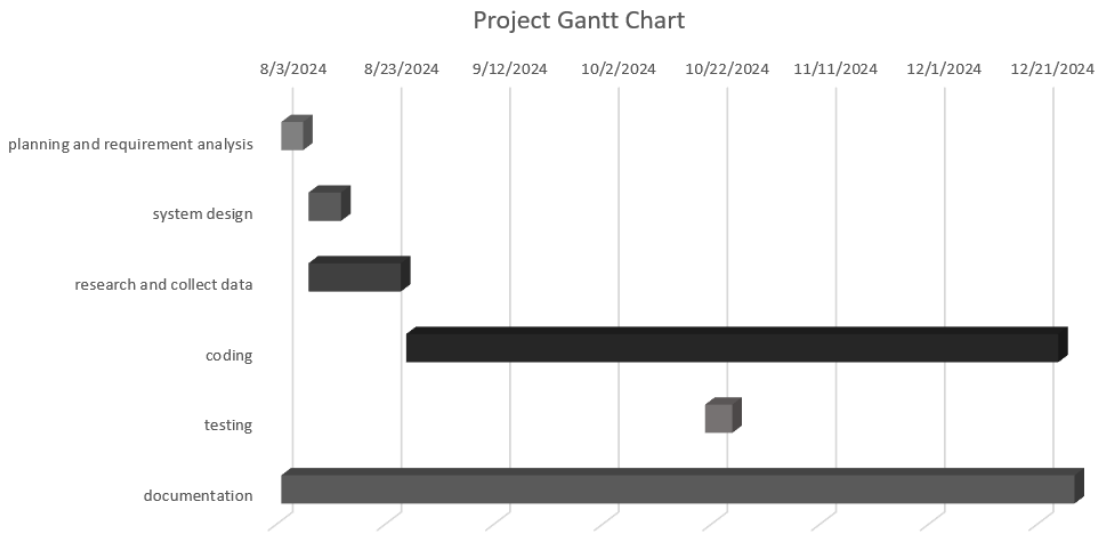

Figure 3.1.2: Project Gantt Chart

3.1.3 Analysis

- **Data modeling using ER diagram:**

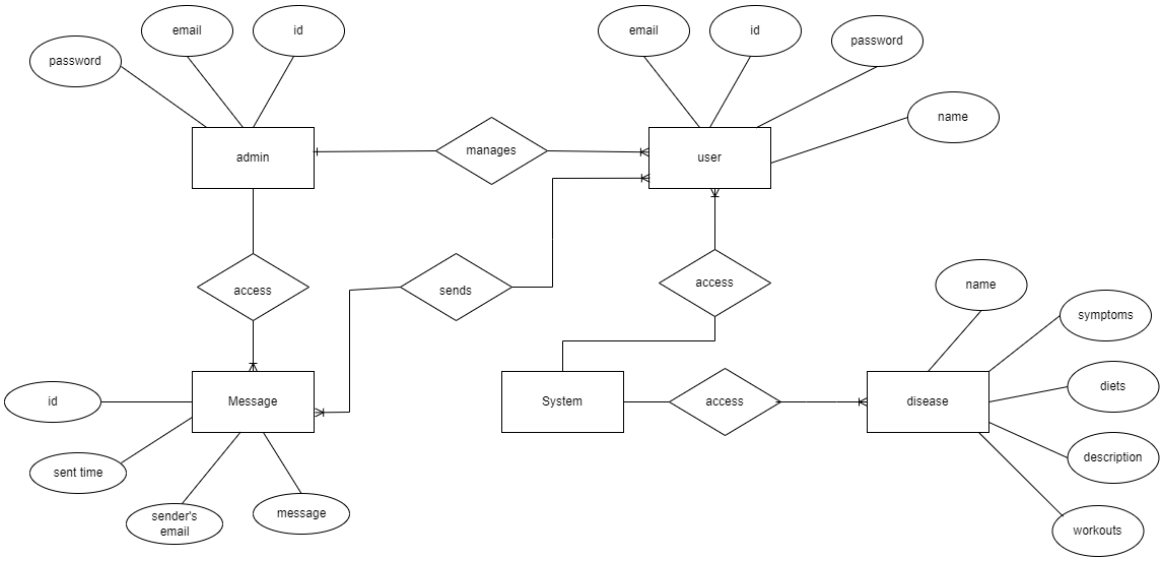

Figure 3.1.3: ER Diagram of MediAssist

- **Process modelling using DFD:**
  Level 0 DFD:

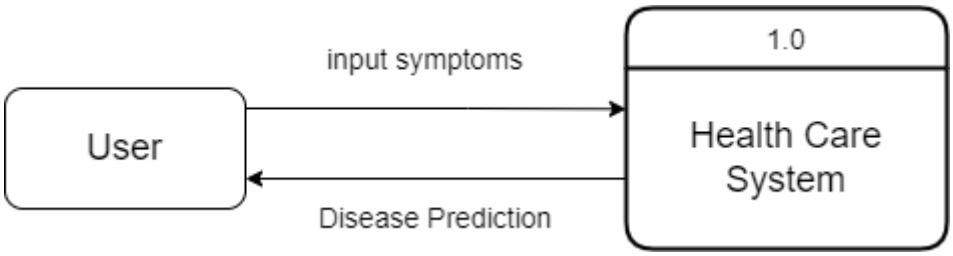

Figure 3.1.4: Level 0 Data Flow Diagram

Level 1 DFD:

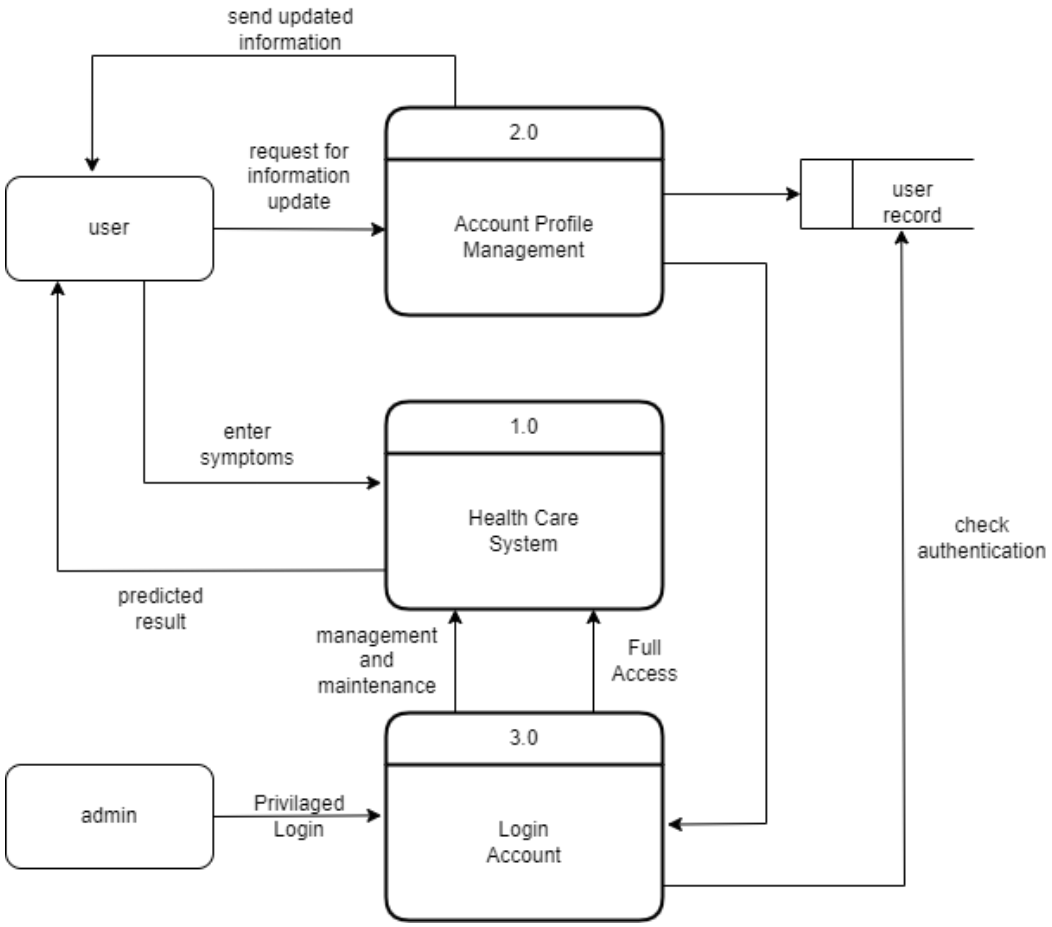

Figure 3.1.5: Level 1 Data Flow Diagram

CHAPTER 4: SYSTEM ANALYSIS

4.1 Design

System design refers to the process of defining the architecture, components, modules, interfaces, and data structures for a software system. It involves creating a detailed blueprint that outlines how the system will be constructed, how it will operate, and how it will fulfill the specified requirements. This process ensures that the system is designed to be scalable, maintainable, and efficient, while addressing both functional and non-functional requirements. System design serves as a bridge between the initial requirements gathering phase and the actual implementation, providing a clear roadmap for developers to follow.

4.1.1 Architectural Design

The MediAssist is designed in 3-tier architecture, dividing system into three layers: presentation layer, logical layer and data layer. This design separates data from logic and user interface making design modular, scalable and maintainable.

- **Presentation layer**
  Presentation layer provides the user interface, that allows user to enter symptoms that user has been suffering from in the form of checkbox. It is responsible for all the UI component and handling user interaction and communicating with logical layer in order to display result.
- **Logical layer**
  The logical layer is core component that has trained model which is responsible for classification using support vector classifier. It processes user input from presentation layer and then process from data in data layer.
- **Data layer**
  The data tier stores information of complete system from user details to user feedbacks to models and datasets everything is stored in database which is used in future analysis.

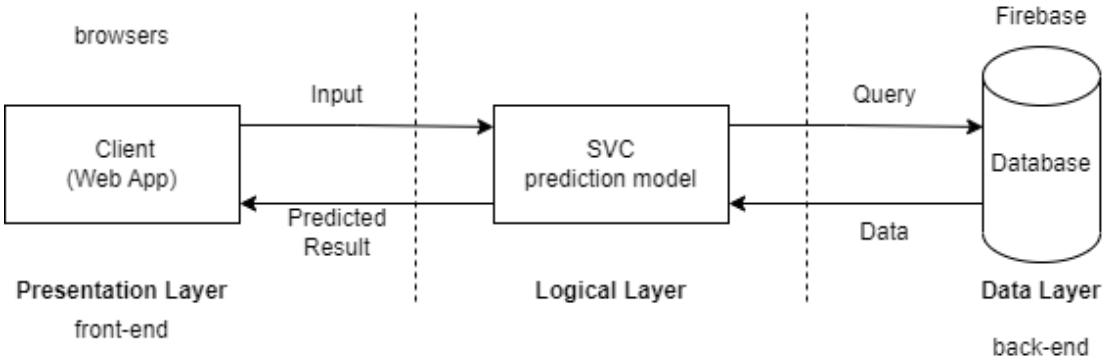

Figure 4.1.1: 3-tier client-server architecture

4.1.2 System Flowchart

The flowchart outlines the sequence of activities from user entering into our system to registering, logging in, using the system and logging out from the system. While admin can manage users and feedbacks from those users.

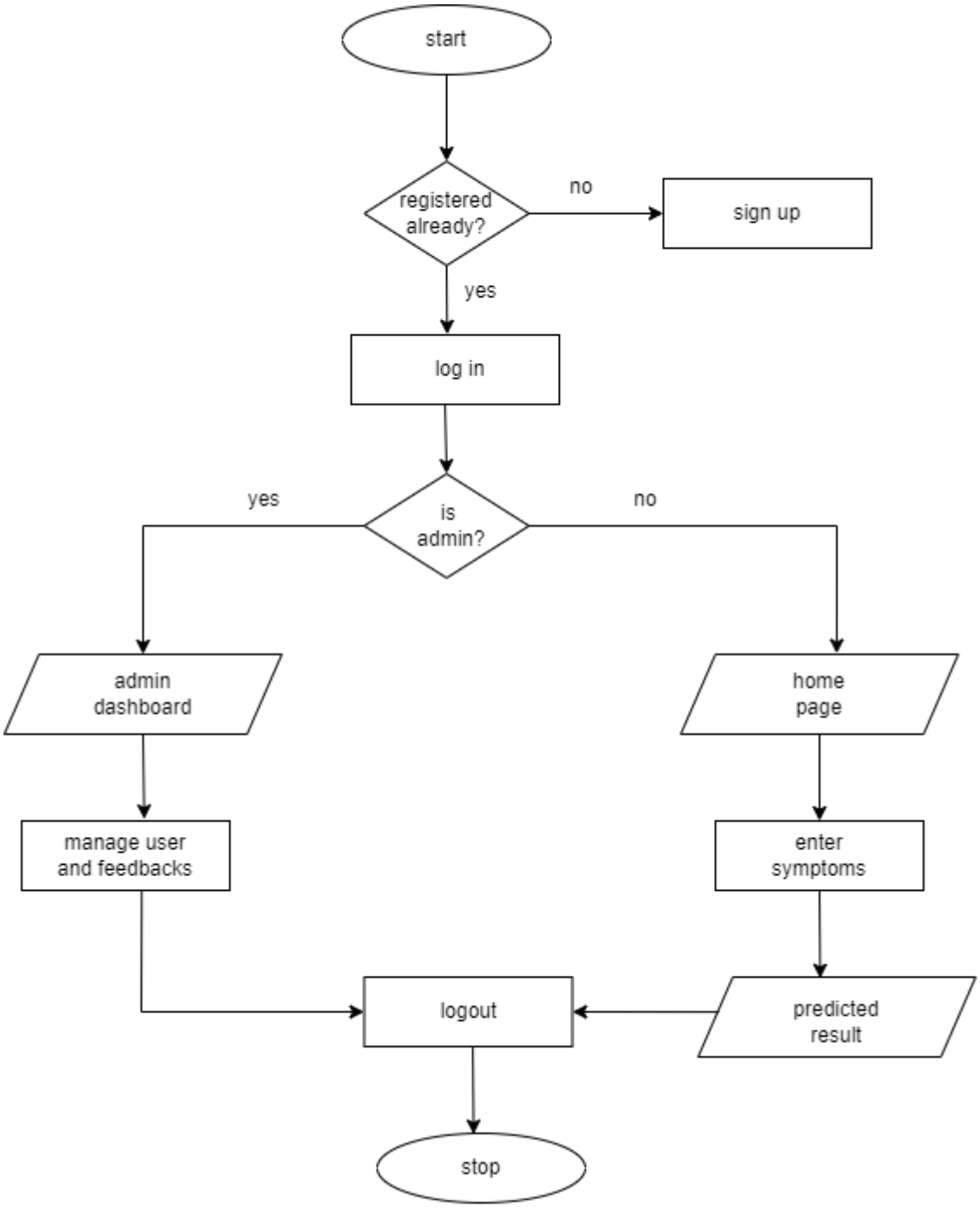

Figure 4.1.2: Flowchart of MediAssist

4.1.3 Database Design

Database Design shows how our data is stored in the database. We have tables like authentication, users, messages and datasets. Data are stored and retrieved from this table when required.

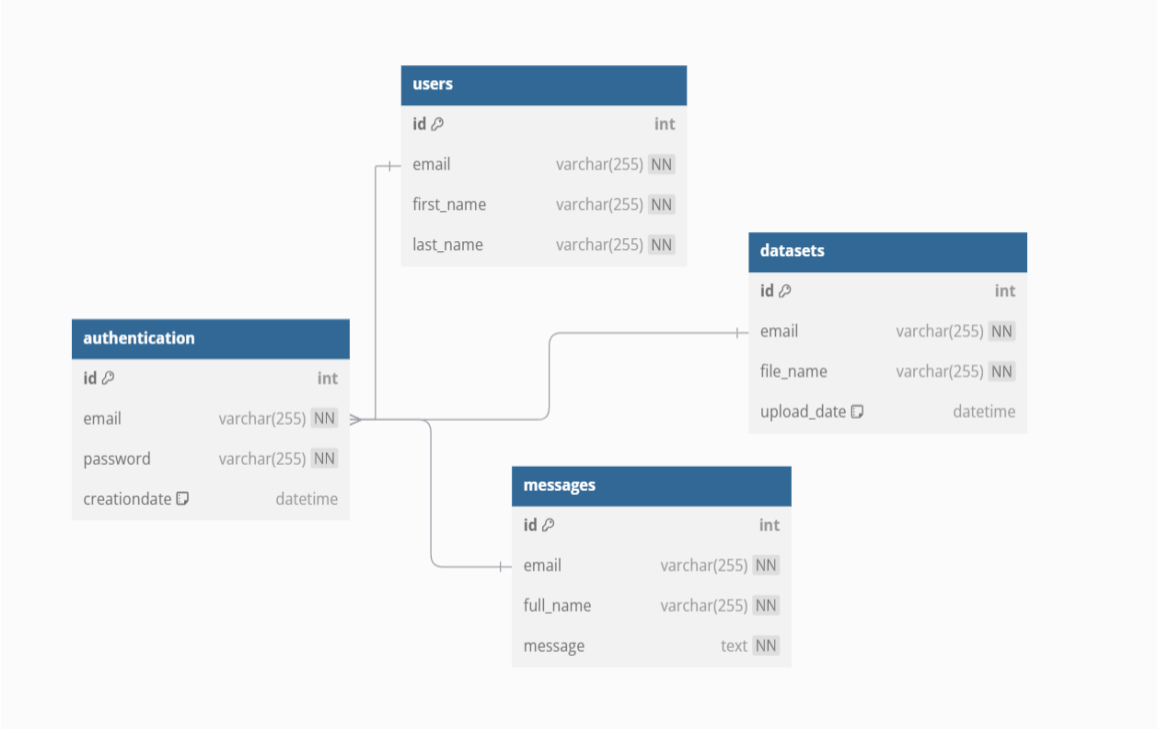

Figure 4.1.3: Database Design

4.2 Algorithm Details

Support Vector Classification (SVC) is a machine learning algorithm used for binary and multi-class classification tasks. It works by finding the optimal hyperplane that maximizes the margin between different classes in the feature space. SVC is effective in high-dimensional spaces and can be extended to handle non-linear data using kernel functions. For a linear Support Vector Classifier (SVC), the decision boundary is defined by a linear equation. The formula for the decision function f(x) is:

$$f(x)=w\cdot x+b$$

Where:

w is the weight vector (normal to the hyperplane).

x is the input feature vector.

b is the bias term (also known as the intercept).

· denotes the dot product.

Decision Rule:

If $f(x) \geq 0$, classify the input as one class (e.g., +1).

If $f(x) < 0$, classify the input as the other class (e.g., -1).

Basic concepts of svc

Support Vectors:

These are the data points closest to the decision boundary (hyperplane) and are critical in defining the position of the hyperplane. The classifier is named after these support vectors because they "support" the optimal boundary.

Hyperplane:

In an SVC, the goal is to find the optimal hyperplane that separates different classes with the maximum margin. In higher dimensions, this hyperplane becomes a decision boundary.

Margin:

The distance between the hyperplane and the nearest data point from either class. SVC aims to maximize this margin.

SVC can be of two types:

Linear svc

When data is linearly separable, the SVC finds a straight line (in 2D), plane (in 3D), or hyperplane (in higher dimensions) that best separates the classes.

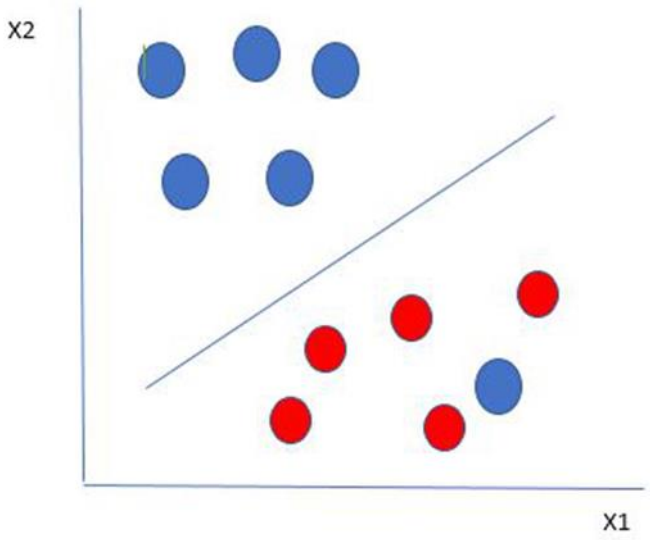

Figure 4.2.1: Linear SVC

Nonlinear svc

When data is not linearly separable, SVC uses a kernel trick to map the data into a higher-dimensional space where a linear separation is possible.

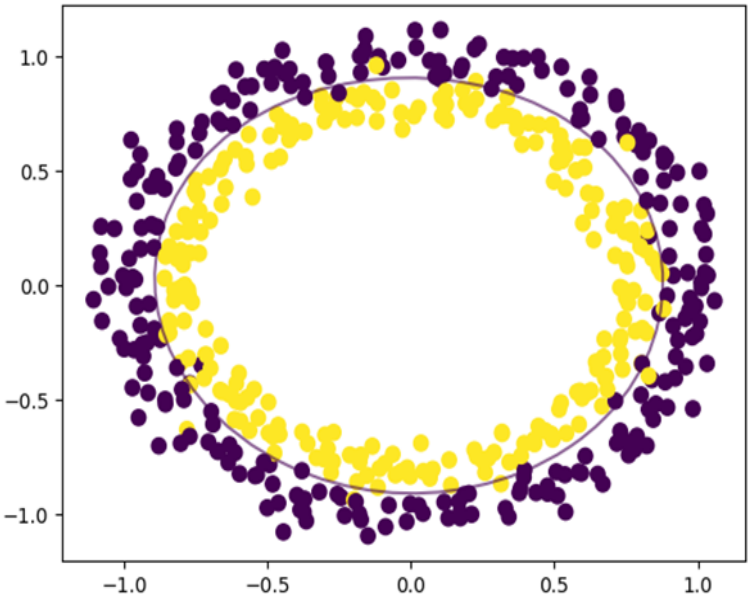

Figure 4.2.2: Non-Linear SVC

4.2.1 Evaluation of algorithm

Evaluating a system using metrics like a confusion matrix, accuracy, precision, recall, and F1-score provides insights into its performance. Here's an explanation and process:

Confusion Matrix

A confusion matrix is a table used to evaluate the performance of a classification model by comparing predicted and actual values.

Table 4.2.1: Confusion Matrix

	Predicted: Positive	Predicted: Negative
Actual: Positive	True Positive (TP)	False Negative (FN)
Actual: Negative	False Positive (FP)	True Negative (TN)

Accuracy:

Measures proportion of correct prediction.

$$\textbf{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

Precision:

Focuses on quality of positive prediction.

$$\textbf{Precision} = \frac{TP}{TP+FP}$$

Recall (Sensitivity):

Measures how well mode captures actual positives.

$$\textbf{Recall} = \frac{TP}{TP+FN}$$

F1-Score:

Harmonic mean of precision and recall, balancing the two.

$$\textbf{F1-Score} = 2.\frac{Precision.Recall}{Precision+Recall}$$

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1 Implementation

This chapter is all about tools and technologies used throughout the project.

5.1.1 Tools Used

- **Flask:**
  Python is the major programming language used in MediAssist. Mainly we used flask to build our web applications quickly and efficiently. It is used as backend framework for handling server-side of the project. It is designed for building web application.
- **HTML:**
  HTML is used to create skeleton structure of this project. Every front-end functionality's structure is created using this markup language.
- **Tailwind CSS:**
  This CSS framework is used to style the structure developed using HTML in order to make UI part attractive and appealing to the users. Tailwind CSS helped positioning images, texts, buttons, and other components in their respective place as well as design them to look beautiful.
- **JavaScript:**
  JavaScript is used to make the web application interactive without the JavaScript it was not possible to make things like submitting form and getting results and many other tasks related to the user interaction.
- **Firebase:**
  Firebase is used to authenticate user login as well as storing user information, datasets, and user messages/responses. Overall firebase is used as backend or database as well as login authenticator provider.
- **Microsoft Excel:**
  Microsoft Excel is used to create Gantt chart as well as comma separated value used in our datasets. Without tool like excel it would have been very difficult to manage datasets.
- **Microsoft Word:**
  Microsoft Word is used as documenting tool. Overall documentation of this project was written and documented using Microsoft word.
- **VS Code:**
  VS Code is and IDE used to write code and manage them. We used vs code to develop our system various extensions provided by vs code made writing code easier and faster.
- **Draw.io:**
  We use this draw.io tool for creating flowcharts, wireframes, and other visual representations that we planned in planning phase and used to guide how our system should be it also provided guideline so we don't deviate from objective.
- **dbdiagram.io:**
  This tool is used to create high level database design structure.

5.1.2 Implementation Details of Module

Modules are the separation of the project into smaller subproject or tasks such that it is easier to manage and scale up later. Different modules are designed so that debugging and different phases are easier to implement. The different modules included in system are:

- User management module
  User management module consist of pages like register and signup page. This module handles user registration and login. Once successfully registered and logged in it then redirects user to home page.
- Admin module
  Admin module is responsible for handling user information. Admin can view lists of users, add user, delete user and view feedbacks and messages sent by the user.
- Landing page
  The landing page provides user with little description of the project and disclaimer notice to use the system responsibly. It also features image demo of the project and allows to move forward into the system's functionalities.
- Home page
  Home page features list of symptoms which when selected by the user and submitted to the system provides result. Here support vector machine uses its logic and then predicts the possible disease user might be suffering from.
- Contact Us Page
  Contact Us page allows users to send message to the admin. This helps in receiving review and feedbacks that will be helpful in future improvement of the future as well as future maintenance and update. User can report bugs about the system as well.

5.2 Testing

Testing refers to the process of evaluating a system, component, or product to identify any defects or issues and ensure it functions as intended. In the context of software, testing involves running the program to detect bugs, errors, or inconsistencies, and verify that it meets specified requirements.

5.2.1 Test cases for unit testing

Unit Testing tests individual components or functions in isolation to ensure they work correctly. The purpose is to validate that each unit of the software code performs as expected. The following test scenarios were used to test the system after the build is completed.

Table 5.2.1: Test cases for Sign Up

Id	Test Scenario	Input Data	Expected Result	Observed Result	Test Status
1	Sign Up Testing	Enter new Email and Password	Account created successfully	Account created successfully	Pass
2	Sign Up Testing	Using Invalid Email	Account can't be created	Error message shown because of invalid email format	Pass

3	Sign Up Testing	Missing Credentials	Account can't be created	Error message shown because of missing value	Pass
4	Sign Up Testing	Enter already used email	Account can't be created	Error message shown because of duplication of email	Pass

Table 5.2.2: Test cases for User Log In

Id	Test Scenario	Input Data	Expected Result	Observed Result	Test Status
1	Login Testing	Correct Credentials	User login successful	User was logged in Successfully	Pass
2	Login Testing	Incorrect Credentials	User login should be unsuccessful	Error message due to incorrect credentials	Pass
3	Login Testing	Missing Credentials	User login should be unsuccessful	Error message due to missing credentials	Pass

Table 5.2.3: Test cases for Input data

Id	Test Scenario	Input Data	Expected Result	Observed Result	Test Status
1	Input Testing	User Input less than 4 Symptoms	Message to select 4-6 symptoms is shown	Message was displayed to select 4-6 symptoms	Pass
2	Input Testing	User Input more than 6 Symptoms	Message to select 4-6 symptoms is shown	Message was displayed to select 4-6 symptoms	Pass
3	Input Testing	User Input 4 to 6 symptoms	Result section visible	Result section was visible under input section	Pass

Table 5.2.4: Test cases for models' prediction

Id	Test scenario	Input Data	Expected Outcome	Observed Outcome	Status
1	Model Testing	itching, skin rash, nodal skin eruptions, skin peeling	Fungal Infection	Fungal Infection	Pass
2	Model Testing	extra marital contacts, receiving blood transfusion, receiving unsterile injection, family history	AIDS	AIDS	Pass
3	Model Testing	blackhead, skin peeling, skin rash, pus filled pimples	ACNE	ACNE	Pass
4	Model Testing	vomiting, dehydration, diarrhea, sunken eyes	Gastroenteritis	Gastroenteritis	Pass
5	Model Testing	acidity, headache, blurred vision, excessive hunger	Migraine	Migraine	Pass
6	Model Testing	vomiting, yellowish skin, dark urine	Hepatitis A	Hepatitis D	Fail

5.2.2 Test cases for system testing

Once all unit testing was performed completely then we integrated the system and tested it completely ensuring that every functionality works together as well. Following tests were performed during the system testing phase:

Table 5.2.5: Test cases for Responsiveness

Id	Test Scenario	Expected Result	Observed Result	Test Status
1	Responsiveness	Response to different device height and width	Web application was responsive to change in height and width of device	Pass
2	Clicking on button and icons	Should perform respective icons when clicked or pressed	Buttons and icons were functioning properly	Pass

Table 5.2.6: Test cases for System Usability Testing

Id	Test Scenario	Expected Result	Observed Result	Test Status
1	Click on various link on the system	Users should be taken into respective pages	Users were able to get the expected web pages after clicking the link	Pass
2	Send feedback from contact us page	Message should be delivered to the admin	Message was delivered	Pass

5.3 Result Analysis

Through analysis of the result, we found every developed module of the MediAssist is working fine and the output that a system produces is in line with the intended requirement. The application developed here includes features like entering from the list of symptoms, finding the list of medications suggested, and getting the diet and workout recommendations.

The application uses various complex computations such as the machine learning algorithms to determine medicines based on the inputs the user has made. It also authenticates, communicates with backend securely, and provides almost accurate result. In general, the MediAssist works fine and corresponds to the users' expectations providing accurate, fast and easy-to-navigate features without any drawbacks.

CHAPTER 6: CONCLUSION AND FUTURE RECOMMENDATION

6.1 Conclusion

In conclusion, the MediAssist is an intelligent tool that utilizes high-end machine learning technique called Support Vector Machines (SVM) along with different technologies like HTML, CSS, JavaScript, Firebase and different tools, to facilitate correct prediction and fully meet the objectives of the system for which it has been developed. Through its simple user interface, users can directly search for symptoms and get a list of suggested medications, precautions, diets and workouts. This is a unique source for patient, carers, and other health professionals. By testing it properly in unit level and system level we confirmed that our system is functional and working properly.

6.2 Future Recommendation

Although our MediAssist is complete web-application we can scale its functionalities furthermore by integrating various features adding many functionalities. This system can be further improved by considering following recommendation.

- Improving datasets, adding symptoms and diseases.
- Testing efficiency with real doctors.
- Making system more accurate, faster and efficient.
- Saving users search history.
- Adding nearby hospital’s location and contacts of doctors and nurses.

REFERENCES

[1] S. Laoyan, "What is Agile methodology? (A beginner's guide)," 02 02 2024. [Online]. Available: https://asana.com/resources/agile-methodology.

[2] P. Sunar, "Around 66,000 die every year due to lack of health care in Nepal," nagarik network, 12 12 2024. [Online]. Available: https://myrepublica.nagariknetwork.com/news/around-66000-die-every-year-due-to-lack-of-health-care-in-nepal-675a796f2250f.html.

[3] "javatpoint," [Online]. Available: https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm.

[4] R. S. K. S. V. R. M. S. Balkrishnan Duraisami, "Heart disease prediction using support vector machine," 2024 dec 14.

[5] R. e. a. Himesh, "International Journal of Medical Informatics," *Application of Naive Bayes in Diabetes Prediction,* vol. 12, no. 3, pp. 45-50, 2021.

[6] E. K. J. J. K. U. J. Mrs. Lisha Kurian, "Disease Prediction and Medicine Recommendation Systems: A Comparative Analysis on learning algorithms," 2023.

[7] B. S. D. V. C. M. K. P. Silpa, "Drug Recommendation System in Medical Emergencies using Machine Learning," 2023.

[8] Y. e. a. Zhang, "IEEE Transactions on Healthcare Informatics," *Deep Learning for Lung Disease Prediction: A Case Study,* vol. 29, no. 4, pp. 325-334, 2022.

[9] M. e. a. Patel, "Healthcare Systems Research Journal," *Ontology-Based Doctor Recommendation System,* vol. 10, no. 1, pp. 5-22, 2019.

[10] S. K. G. A. Gupta, "Journal of Computational Healthcare," *Improving Doctor Recommendations with Hybrid Models,* vol. 7, no. 2, pp. 101-113, 2021.

[11] D. B. E. C.-C. J. F. J. G. D. K. A. &. N. E. Ferrucci, *Building Watson: An overview of the DeepQA project,* vol. 31, no. 3, pp. 59-79, 2010.

[12] A. S. &. C. D. R. Bhagavathula, "AiCure: Harnessing artificial intelligence to improve medication adherence," vol. 2, no. 12, 2020.

APPENDICES

SNAPSHOTS

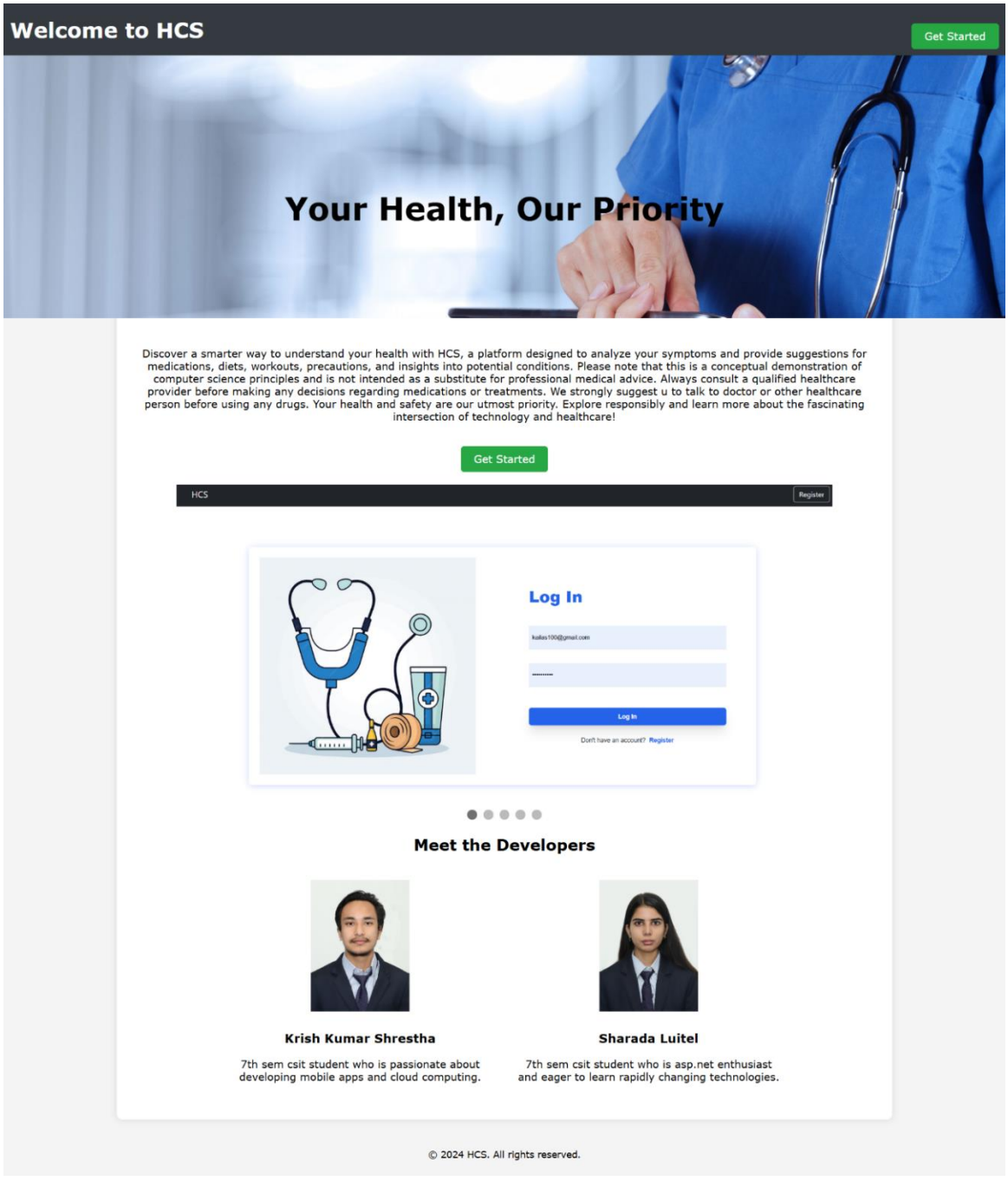

Figure 0.1: Landing Page

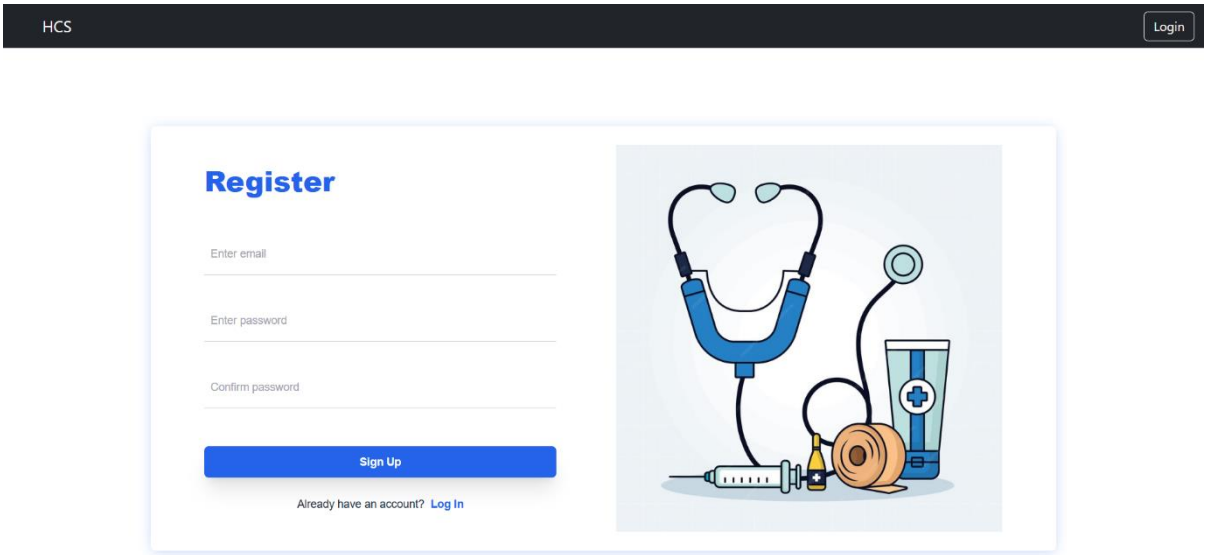

Figure 0.2: Register Page

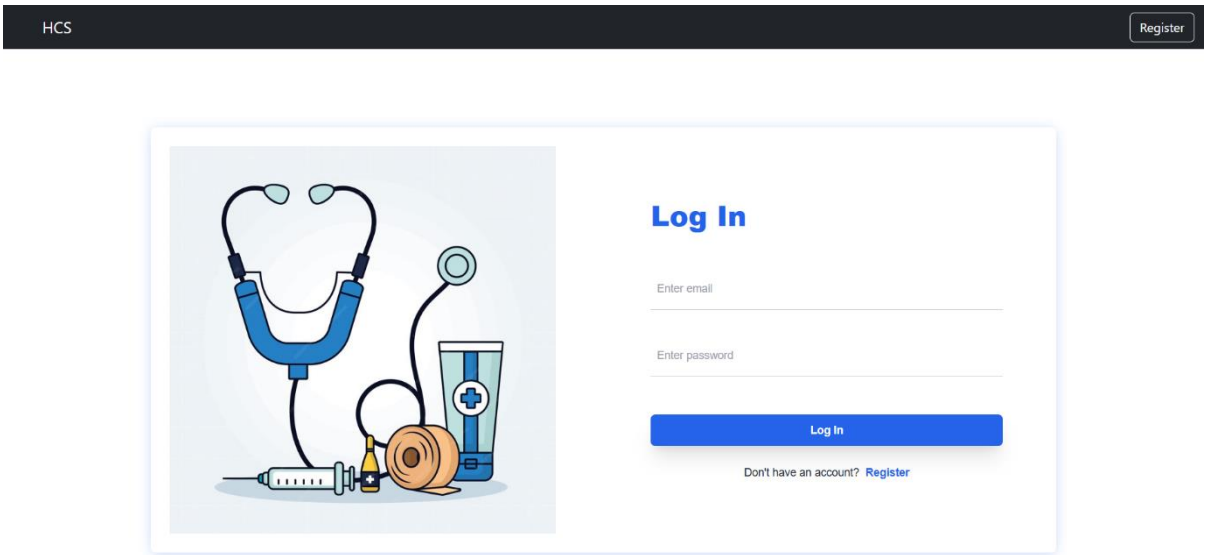

Figure 0.3: Login Page

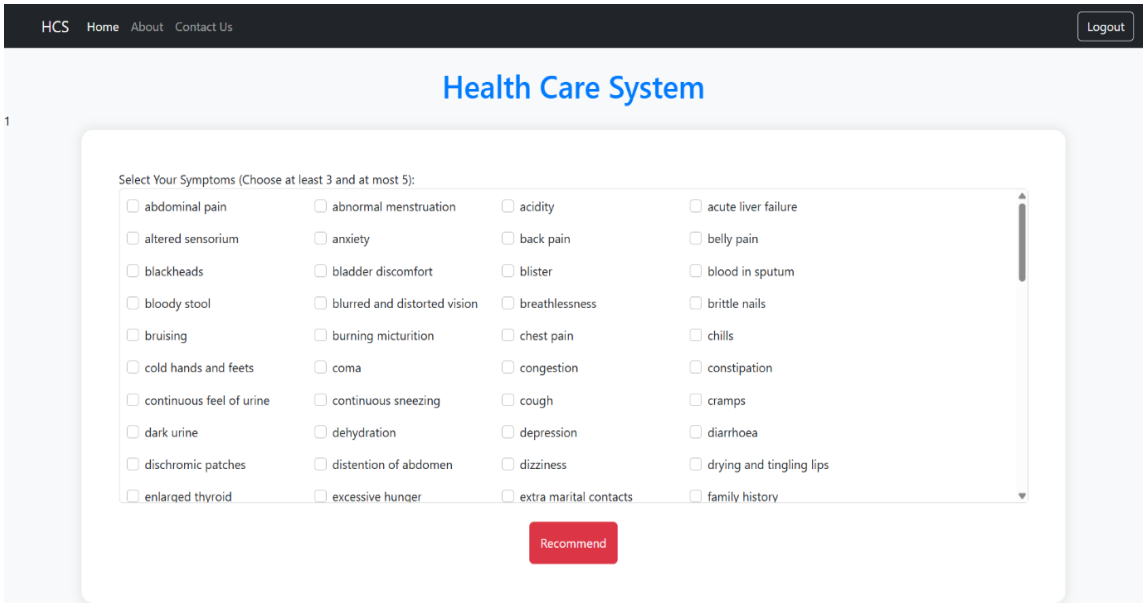

Figure 0.4: Home Page

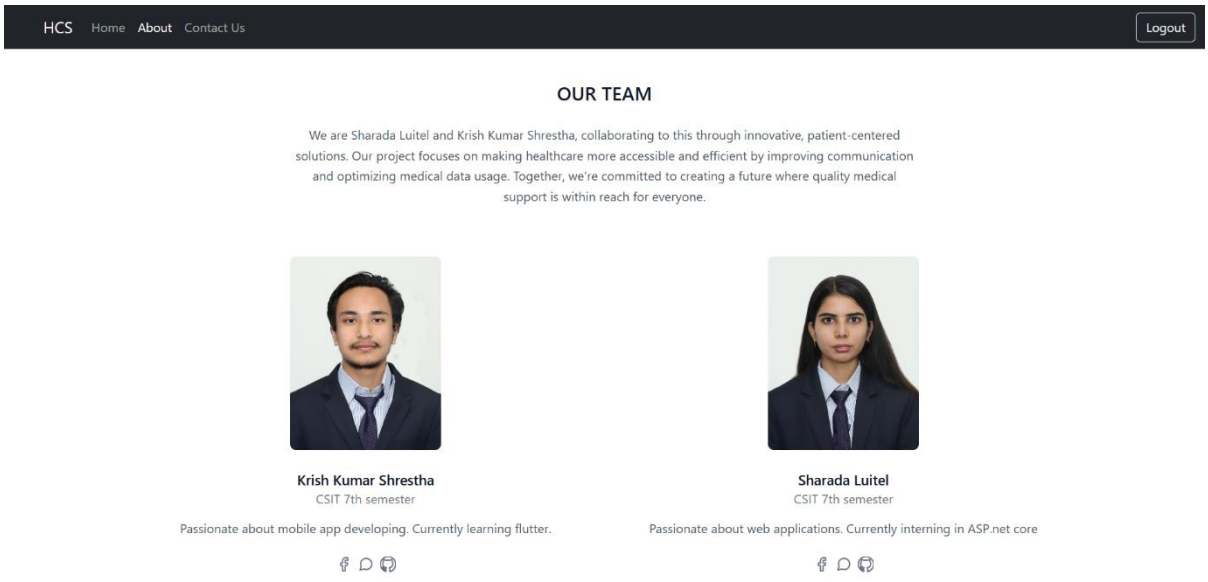

Figure 0.5: About us Page

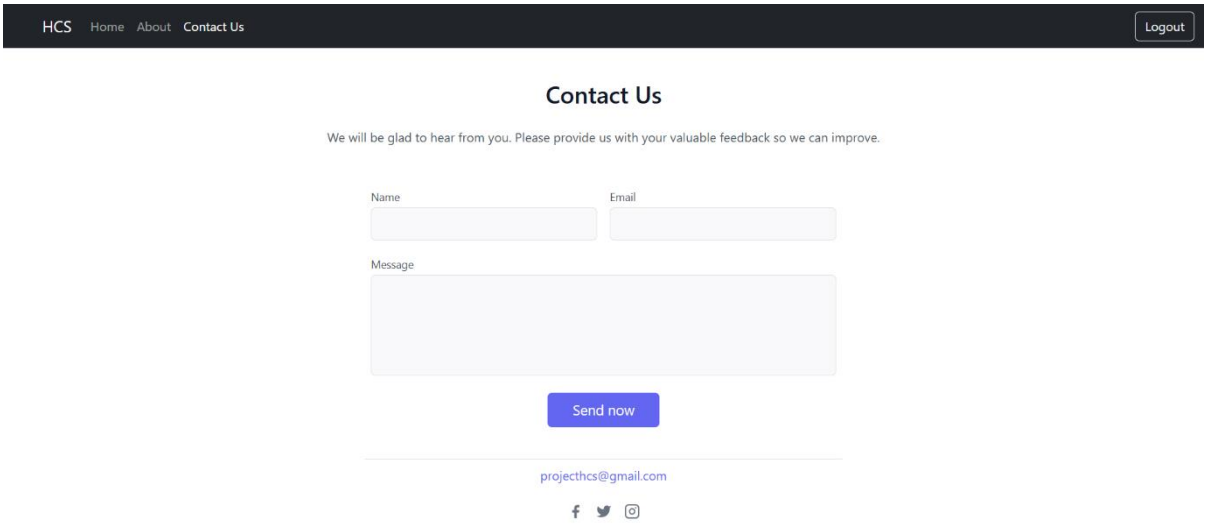

Figure 0.6: Contact us Page

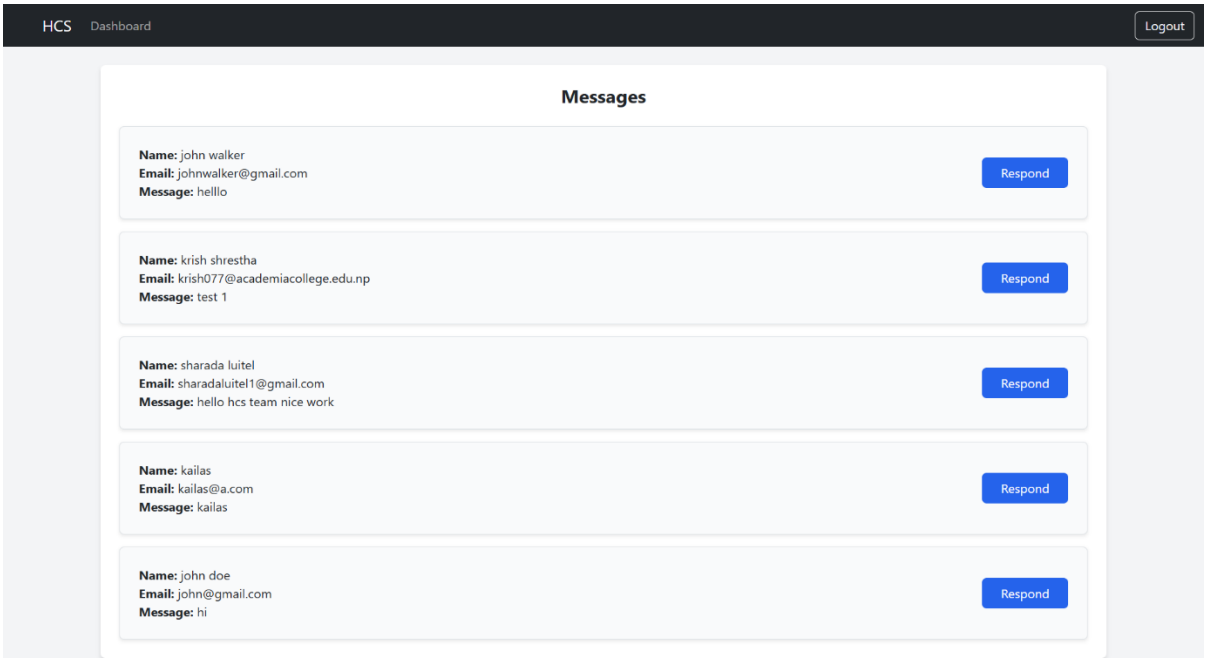

Figure 0.7: User Message Page

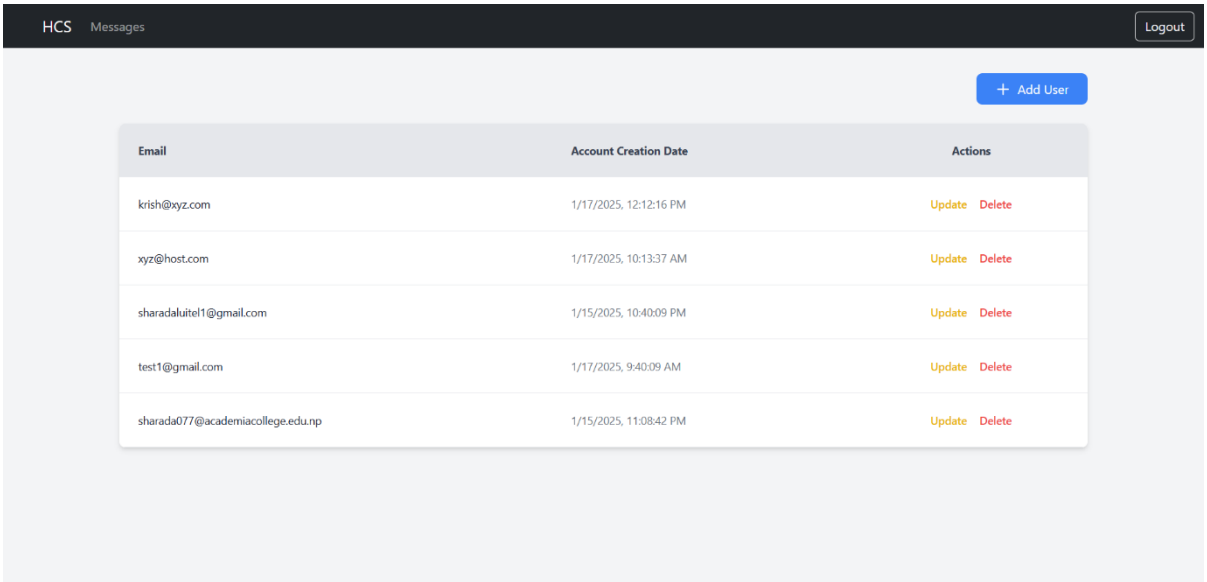

Figure 0.8: Admin dashboard Page

Source Code

Splitting dataset into training and testing data

```
X = df.drop('prognosis', axis = 1)
Y = df['prognosis']
le = LabelEncoder()
le.fit(Y)
Y = le.transform(Y)
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.3, random_state=20)
```

Training the Model

```
models = {
   "SVC" : SVC (kernel='linear')
}
for model_name, model in models.items():
   #train model
   model.fit(X_train, Y_train)
   #test model
```

```
    predictions = model.predict(X_test)
    #calculate accuracy
    accuracy = accuracy_score(Y_test, predictions)

    #confusion matrix
    cm = confusion_matrix(Y_test, predictions)
    print(f"{model_name} Accuracy:  {accuracy}")
    print(f"{model_name} Confusion Matrix:")
    print(np.array2string(cm, separator=', '))
```

Loading Dataset

```
symptoms = pd.read_csv('path')
precautions = pd.read_csv('path)
workout = pd.read_csv('path)
description = pd.read_csv('path)
medications = pd.read_csv('path)
diets = pd.read_csv('path)
def helper(dis):
    desc = description[description['Disease']==dis]['Description']
    desc = " ".join([w for w in desc])
    pre = precautions[precautions['Disease']==dis][['Precaution_1', 'Precaution_2', 'Precaution_3', 'Precaution_4']]
    pre = [col for col in pre.values]
    med = medications[medications['Disease']== dis]['Medication']
    med = [med for med in med.values]
    die = diets[diets['Disease'] == dis]['Diet']
    die = [die for die in die.values]
    workt = workout[workout['disease']== dis]['workout']
    return desc, pre, med, die, workt
symptoms_dict = {}
diseases_list = {}
def get_prediction(patient_symptoms):
```

```
    input_vector = np.zeros(len(symptoms_dict))
    for item in patient_symptoms:
        input_vector[symptoms_dict[item]]= 1
    return diseases_list[svc.predict([input_vector])[0]]
```

Test

```
symptomss = input("Enter your symptoms: ")
user_symptoms = [s.strip() for s in symptomss.split(',')]
user_symptoms = [sym.strip("[]' ") for sym in user_symptoms]
predicted_disease = get_prediction(user_symptoms)
desc, pre, med, die, workt = helper(predicted_disease)
#print results
```