



**Tribhuvan University**  
**Faculty Of Humanities and Social Science**

**MEDICINE RECOMMENDATION SYSTEM**  
**A PROJECT REPORT**

*Submitted to*  
**Department of Computer of Application**  
**SWSC - Basundhara, Kathmandu**  
*In partial fulfillment of the requirements for the Bachelor in Computer Application*

Submitted by  
Name: Naresh Khatri  
TU Reg No: 6-2-530-20-2020

Under the Supervision of  
**Bijay Babu Regmi**



**Tribhuvan University**  
**Faculty Of Humanities and Science**  
**Southwestern State College**

**SUPERVISOR'S RECOMMENDATION**

We hereby recommend that this project is prepared by **Naresh Khatri** under supervision by **Mr. Bijay Babu Regmi** entitled “**Medicine Recommendation System**” in partial fulfillment of the requirements for the degree of Bachelor of Computer Application be processed for the evaluation.

-----  
**Mr. Bijay Babu Regmi**  
**Supervisor**



**Tribhuvan University**  
**Faculty Of Humanities and Science**  
**Southwestern State College**

**LETTER OF APPROVAL**

This is to certify that this project is prepared by **Naresh Khatri (6-2-530-20-2020)** entitled “**Medicine recommendation system**” in partial fulfillment of the requirements for the degree of Bachelor in Computer Application has been evaluated. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

-----  
**Mr. Bijay Babu Regmi**  
**Project Supervisor**

-----  
**Mr. Kiran Ghimire**  
**Internal Supervisor**  
**BCA Department**  
**SWSC**

-----  
**Bikash Parajuli**  
**Internal Supervisor**

-----  
**Name**  
**External Examiner**  
**Tribhuvan University**

## **ACKNOWLEDGMENT**

For the partial fulfilment of the project this semester we would like to express our sincere gratitude to everyone who has directly and indirectly helped us to develop this project. There were times when we thought that this project was difficult or may be not possible to create but we are thankful to our respected supervisor **Mr. Bijay Babu Regmi** sir who continuously guided us to perform the project to complete in time. We could barely do anything without the help of teachers and friends who helped us throughout the whole journey and find and reconcile our mistakes.

I would also like to thank the Tribhuvan University for giving us this opportunity via the course of Computer Science and IT to help us understand the project ethics at this early stage and helped us to evaluate our knowledge and expand it a little more.

**With Respect,**  
**Naresh Khatri**  
**(6-2-530-20-2020)**

## ABSTRACT

The Medicine Recommendation System is made to provide individualized medical guidance, making it easier for people to manage their medical requirements. The Support Vector Machine (SVM) algorithm is used to analyze a patient's symptoms using the Support Vector Classifier (SVC) model. This model allows the system to identify possible diseases and suggest customized therapies, including diets, exercise regimens, and medication. This system simplifies the healthcare procedure in a similar way to how a travel booking system makes holiday planning easier by arranging all the important details. The system can precisely assess symptoms and pinpoint potential medical issues thanks to the SVM algorithm, which is well-known for its resilience in classification tasks. Serving as a personal healthcare assistant that helps people navigate their health journey, the platform is easy to use and intuitive. By offering tailored, well-informed advice.

**Keywords:** *medicine, predict, recommendation, SVC, SVM, symptoms etc.*

# TABLE OF CONTENTS

<b>SUPERVISOR’S RECOMMENDATION .....</b>	<b>ii</b>
<b>LETTER OF APPROVAL .....</b>	<b>iii</b>
<b>ACKNOWLEDGMENT .....</b>	<b>iv</b>
<b>ABSTRACT.....</b>	<b>v</b>
<b>LIST OF FIGURES .....</b>	<b>viii</b>
<b>LIST OF TABLES.....</b>	<b>ix</b>
<b>LIST OF ABBREVIATIONS .....</b>	<b>x</b>
<b>CHAPTER1: INTRODUCTION.....</b>	<b>1</b>
1.1 Introduction.....	1
1.2 Problem Statement.....	1
1.3 Objective.....	2
1.4 Scope and Limitations.....	2
1.4.1. Scope.....	2
1.4.2. Limitations .....	2
1.5 Development Methodology .....	3
1.6 Report Organization.....	6
<b>CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW.....</b>	<b>8</b>
2.1 Background Study.....	8
2.2 Literature Review.....	9
<b>CHAPTER 3: SYSTEM ANALYSIS AND DESIGN.....</b>	<b>10</b>
3.1. System Analysis.....	10
3.1.1. Requirement Analysis .....	10
3.1.1.1. Functional Requirement.....	10
3.1.1.2. Non-functional Requirement .....	11
3.1.2. Feasibility Analysis.....	11
3.1.2.1. Technical Feasibility .....	11
3.1.2.2. Operational Feasibility.....	11

3.1.2.3. Economic Feasibility .....	11
3.1.2.4. Schedule Feasibility .....	12
3.1.3. System Modelling using class and object diagram .....	13
3.1.4 Dynamic modeling using State and Sequence Diagrams .....	14
3.1.5 Process modeling using Activity Diagrams .....	16
3.2 System Design .....	17
3.2.1. Refinement of Classes and object diagram .....	17
3.2.2 Component Diagram .....	19
3.3 Algorithm Description .....	20
<b>CHAPTER 4: IMPLEMENTATION AND TESTING .....</b>	<b>22</b>
4.1 Implementation .....	22
4.1.1 Tools Used.....	23
4.1.2 Implementation Details of Module .....	24
4.2 Testing .....	29
4.2.1 Test cases for Unit Testing .....	29
4.2.2 Test cases for System Testing .....	30
<b>CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATION .....</b>	<b>35</b>
5.1 Conclusion .....	35
5.2 Lesson Learnt/ Outcome .....	35
5.3 Future Recommendations .....	36
<b>REFERENCES.....</b>	<b>37</b>
<b>APPENDIX.....</b>	<b>38</b>

## LIST OF FIGURES

Figure 1: Agile Methodology.....	5
Figure 2: Use case diagram of medicine recommendation system.....	10
Figure 3: Gantt Chart .....	12
Figure 4: Class Diagram of Medicine Recommendation System.....	13
Figure 5: Sequence Diagram of Medicine Recommendation System .....	14
Figure 6: State Diagram of Medicine Recommendation System.....	15
Figure 7: Activity Diagram of Medicine Recommendation System.....	16
Figure 8: Refined Class Diagram of Medicine Recommendation System .....	17
Figure 9: Refined Sequence Diagram of Medicine Recommendation System.....	18
Figure 10: Component Diagram of Medicine Recommendation System.....	19
Figure 11: SVM Architecture.....	20
Figure 12: Imports and Setup for Medicine recommendation system.....	24
Figure 13: Flask Application Initialization for medicine recommendation system.....	24
Figure14: Dataset Loading and Preprocessing for medicine recommendation system .....	25
Figure 15: Label Encoding for Medicine Recommendation System.....	26
Figure 16: Homepage for Medicine Recommendations System. ....	28
Figure 17: Running Medicine Recommendation application .....	28
Figure 18:Result Analysis .....	34



## LIST OF TABLES

Table 1: Unit Testing for Input Field .....	29
Table 2: Testing for System .....	30

## **LIST OF ABBREVIATIONS**

<b>AI:</b>	Artificial Intelligence
<b>CSS:</b>	Cascading Style Sheet
<b>HTML:</b>	Hyper Text Markup Language
<b>SVC:</b>	Support Vector classifier
<b>ML:</b>	Machine Learning
<b>SVM:</b>	Support Vector Machine
<b>OOP:</b>	Object Oriented Programming
<b>UML:</b>	Unified Modeling Language

# **CHAPTER1: INTRODUCTION**

## **1.1 Introduction**

The Medicine Recommendation System is a user-friendly digital tool developed to make healthcare more personalized and accessible for everyone. By analyzing a patient's symptoms, it uses advanced algorithms to recommend the most suitable treatments, including medications, workout routines, and diet plans. Similar to how booking a trip has become easier with online platforms, this system simplifies the healthcare process, offering patients timely and accurate advice without the usual confusion or delays.

Many people face challenges when it comes to managing symptoms, finding a diagnosis, and choosing the right treatment options. This system eliminates the guesswork by streamlining everything into one simple process. It allows patients to easily track their symptoms and receive personalized health recommendations, tailored specifically to their needs. From workout routines to diet plans, it suggests ways to improve overall wellness, helping patients make informed choices about their health.

The ultimate goal of the Medicine Recommendation System is to empower patients, giving them more control over their health journey. By digitizing and optimizing the entire healthcare experience, it not only improves the accuracy of diagnosis and treatment but also ensures the process is easy to navigate, making personalized care more approachable for everyone.

## **1.2 Problem Statement**

The Medicine Recommendation System addresses several critical challenges in modern healthcare. Traditional treatment methods often struggle to deliver highly personalized care due to the diversity of patient profiles, symptom variations, and the complexity of diagnoses. This system leverages machine learning to analyze individual symptoms, providing tailored treatment recommendations that enhance the relevance and effectiveness of care. Additionally, it reduces the risk of adverse drug reactions by analyzing medical histories and potential medication interactions, minimizing harmful prescriptions. By rapidly processing large volumes of data, the system also improves the accuracy and efficiency of treatment advice, supporting healthcare providers in making prompt, data-driven decisions. The continuous learning capability of the system allows it to refine recommendations over time, adapting to

new medical findings and improving overall patient care quality. Through this, the Medicine Recommendation System transforms healthcare into a more personalized, efficient, and safer experience for patients and providers alike.

### **1.3 Objective**

- To predict the disease based on the Symptoms
- To recommend the medicine, workout, and diet based on the Symptoms.

### **1.4 Scope and Limitations**

#### **1.4.1. Scope**

The Medicine Recommendation System is an innovative healthcare tool designed to support individuals in understanding and addressing their health needs through advanced technology. The system functions by analyzing symptoms provided by users, allowing it to assess possible health concerns through the power of machine learning algorithms. These algorithms enable the system to make accurate predictions about potential diseases, leveraging data patterns to produce reliable diagnostic suggestions based on the information entered. Upon identifying the probable disease or condition, the system goes a step further by recommending appropriate medications, specifically chosen to address the individual's unique symptoms and condition. This capability helps users take targeted actions in managing their health, all within an accessible and easy-to-use digital interface.

#### **1.4.2. Limitations**

- The medicine recommendation system primarily focuses on backend processes such as data retrieval, predicting, and recommending, rather than on the front-end user interface. Consequently, the system prioritizes functionality over visual or interactive elements, resulting in a limited UI experience compared to systems designed with a stronger emphasis on user interface.
- The system's capacity to anticipate diseases and suggest treatments may be limited if it only responds to one symptom, encounters ambiguity, or falls outside of its taught knowledge set. To guarantee appropriate advice for difficult or extremely particular health concerns, the system may need to be further developed or healthcare professionals' help may be needed.

## 1.5 Development Methodology

- Develop the system using Bootstrap and JavaScript for frontend and python for backend.
- This project adopted agile technique
- The development of a recommendation system for intelligent medicine requires the use of Support Vector Machines (SVM). SVM allows the system to accurately analyze user-inputted symptoms and suggest suitable therapies, such as drugs and exercise regimens, by training it on a sizable dataset of diseases and their corresponding medications. For customers looking for advice on how to manage their health, this method guarantees that the recommendations are both pertinent and tailored to them.

### **Agile Methodology:**

This project adopted agile technique. The Agile methodology is a style of project management that divides a project into phases. It significantly enhances the development of recommendation system projects by promoting flexibility, collaboration, and iterative progress.

We have adopted this methodology as it helps to break down the project into manageable tasks and ensure that everyone on the team is working towards same goal, that is well suited for our final group project leading to maintenance in the team structure.

Here are the key steps followed to use agile model on the medicine recommendation system:

- **Planning:**  
I defined the core functionality of the recommendation system, such as predicting the condition based on symptoms entered by the user and recommending appropriate treatments. They divide the work into manageable assignments that can be finished in a few weeks.
- **Requirement Analysis:**  
I analyzed the key features that users might want from a medicine recommendation system. This system predict the disease on the basis of user-inputted symptoms and suggest appropriate treatment.
- **Design:**  
I planned the overall structure of the recommendation engine and how the user interface would present suggestions. The design aimed to balance ease of use with the complexity of the underlying recommendation algorithms.

- Coding:

I developed the recommendation engine first, followed by creating the user interface. The development process was iterative, with features being added gradually and tested frequently.

- Testing:

As each component was completed, I tested it to ensure the system generated relevant manga recommendations. I used feedback from initial tests to make adjustments and refine the recommendations.

- Deployment:

The system was deployed incrementally, with each version improving upon the last. The initial version provided basic functionality, with further enhancements added as the project progressed

In conclusion, Agile methodology enables the system project to be developed in small, manageable steps, ensuring quick delivery of essential features and continuous predict and recommend the medicine and workouts.

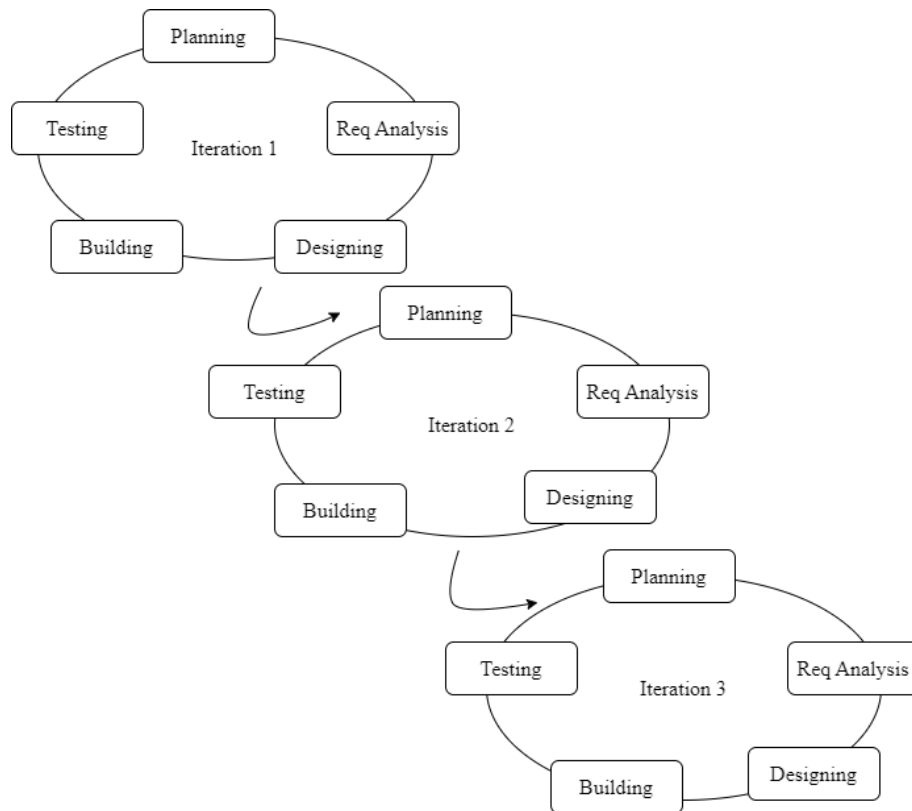


Figure 1: Agile Methodology

## **1.6 Report Organization**

### **Chapter 1**

This chapter has the following sections: Introduction, Problem Statement, Objectives, Scope, and Limitations. In this chapter, a clear idea of what the system aims to achieve is explained. The Medicine Recommendation System is developed as a virtual assistant aimed at streamlining the process of identifying and recommending appropriate medications based on symptoms, thereby reducing the reliance on manual research or consultations for minor ailments. The primary goal is to provide quick, accurate recommendations to users about suitable over-the-counter medicines while ensuring essential safety guidelines are followed.

The system addresses the inefficiencies of traditional methods by offering real-time, automated suggestions based on symptom inputs. The scope includes delivering essential information through an AI-driven conversational interface, though it is limited by its dependency on accurate symptom input and its focus on backend data processing rather than an enhanced user interface. The project follows the Agile development methodology, enabling iterative progress and continuous improvements based on user feedback.

### **Chapter 2**

In chapter 2, the background study explores the evolution of medicine recommendation system, This paper proposes a medicine recommendation system that uses AI and machine learning to predict diseases and medications based on symptoms entered by patients or users. AI and machine learning are multidisciplinary fields used in classification and predictive analysis, and medical professionals can use these systems to make more informed decisions. Advancements in AI and SVM technologies have also led to the rise of system, which can handle complex conversations, make recommendations, answer queries, and perform tasks for users. The system uses the Support Vector Machine (SVC) model to improve user experience, streamline processes, and reduce costs

### **Chapter 3**

This chapter analyzes system requirements, including functional and non-functional aspects, and feasibility analysis to ensure the project's technical, operational, and economic viability. System modeling is presented using diagrams like use-case, class, sequence, state, and activity diagrams. The chapter focuses on the Medicine Recommendation System's architecture, using refined UML diagrams and the algorithmic approach of Support Vector Classifier (SVC) and Support



Vector Machine (SVM). SVC model enhances the system's recommendation capabilities by identifying patterns in user inputs.

#### **Chapter 4**

This chapter outlines the implementation of the chatbot using tools like HTML, CSS, Python, flask, and Bootstrap for building the user interface, backend logic. Key modules include data cleaning, training the chatbot model, and managing user interactions. The testing phase involves unit testing of individual components and system testing to ensure reliable performance and accurate responses.

#### **Chapter 5**

The conclusion highlights the project's success in improving information the Medicine Recommendation System is a digital tool designed to make healthcare more personalized and accessible. It uses advanced algorithms to analyze patient symptoms and recommend suitable treatments, such as medications, workout routines, and diet plans. This system simplifies the healthcare process, providing timely and accurate advice without confusion. It empowers patients by empowering them with more control over their health journey and making personalized care more accessible.

# **CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW**

## **2.1 Background Study**

Most people tend to live a long and healthy life, but people are busy in their day to-day life and it is not possible for everyone to visit doctors for minor symptoms of a disease. Many people do not know about medicines and to visit a doctor and consult for minor symptoms for medicines is a time-consuming process. AI and machine learning like emerging technology can help us to create a recommended system that will prescribe medicine and this system can accurately predict a medicine to use. In this paper proposes the medicine recommendation system which will predict disease and medicine according to symptoms entered by patients/users [1].

uses of artificial intelligence that have drawn interest from scholars worldwide. RSs are developed using a variety of machine learning algorithms. Selecting, the most difficult task in the field of RSs is determining the optimal machine learning algorithm to offer people a good or service [3]. The computer science discipline of artificial intelligence gave rise to machine learning (ML). Combining computer science algorithms with statistics, machine learning (ML) is a multidisciplinary field that finds extensive use in classification and predictive analysis. Medical professionals can choose the right medication for their patients with the help of medicine recommender systems. Such recommendation systems can be developed with the aid of today's cutting-edge technologies, perhaps resulting in more succinct decisions [2]. Because of the rise With the advancements in AI and NLP technologies, Chatbot have become more sophisticated and can now handle complex conversations, make recommendations, answer queries, and even perform tasks for users. Today, chatbots are used in various industries, such as customer service, healthcare, finance, education, and entertainment, to enhance user experience, streamline processes, and reduce costs. In this system we use the SVC model that provides the Support vector machine algorithm [3].

## 2.2 Literature Review

A literature review for a medicine recommendation project would typically explore existing research, studies, and implementations related to system in medical fields.

The COVID-19 pandemic has exposed flaws in the medical and healthcare system, making it difficult to track infected individuals. A machine learning-based system based on symptoms is being proposed to address this issue. The system uses Rajam's inputs, random under-sampling and SMOTE to balance the data set. Machine learning techniques like KNN, Decision Tree, Naïve Bayes, Random Forest, SGD, and Support Vector Machine are used to categorize subjects based on age, comorbidity, and symptoms. The system is expected to be up to 99% accurate [4].

### Existing system

Diagnosis-based recommendation systems utilize diagnostic data, such as disease names and lab results, to suggest appropriate medications. For instance, if a patient is diagnosed with Type 2 Diabetes, the system might recommend Metformin. These systems often rely on machine learning techniques like classification algorithms, such as Random Forest or XGBoost, to predict suitable medicines based on the diagnostic input. Additionally, reinforcement learning is employed to refine recommendations over time by learning optimal prescriptions through trial and feedback, improving accuracy and adaptability in clinical scenarios [5].

Adverse drug interaction prediction systems aim to identify and avoid harmful combinations of medications. They analyze inputs such as a patient's current medication list and symptoms to recommend safe alternatives and appropriate dosages. Machine learning techniques like Graph Neural Networks are used to model complex relationships and interactions between drugs. Collaborative filtering, often utilized in recommendation systems, predicts potential interactions based on historical data of similar cases, ensuring personalized and safer medication plans. These methods enhance patient safety and treatment efficacy. Drug interactions occur when a medication doesn't mix well with another drug, food, or alcohol, causing it to stop working, become less effective, or trigger side effects. Understanding these interactions helps avoid them and improves overall health [6].

## CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

### 3.1. System Analysis

The project will be explained using dataflow diagrams, flowcharts, use-case diagrams, connection and entity diagrams, and so on.

#### 3.1.1. Requirement Analysis

##### 3.1.1.1. Functional Requirement

- The user shall be able to enter symptoms.
- The user shall be able to view potential diseases.
- The user shall be able to view descriptions of the diseases.
- The user shall be able to access information about recommended medicines.
- The user shall be able to access workout recommendations.
- The user shall be able to access dietary recommendations.
- The user shall be able to view suggested precaution.

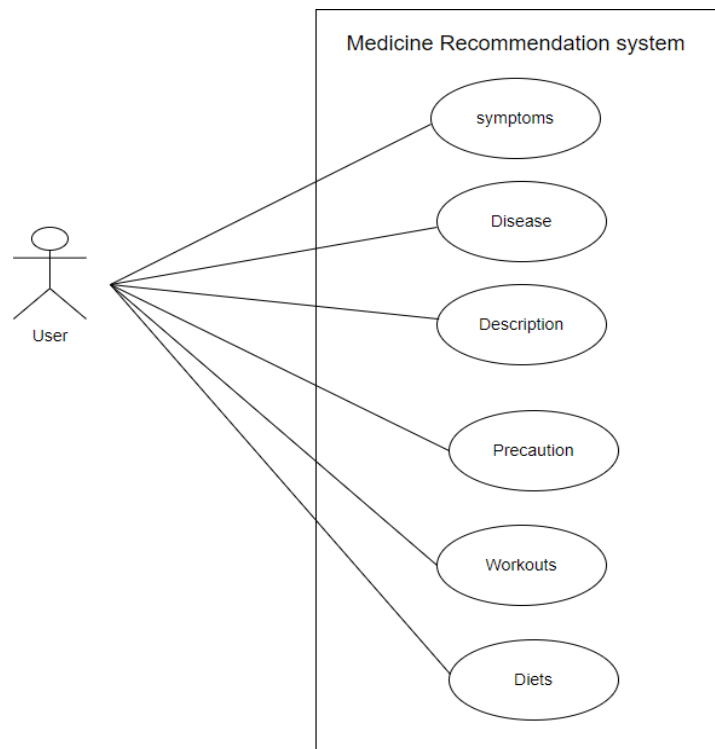


Figure 2: Use case diagram of medicine recommendation system

### **3.1.1.2. Non-functional Requirement**

- The system must be user-friendly.
- The system must provide recommendations quickly.
- The system must be compatible with different system devices.

### **3.1.2. Feasibility Analysis**

A feasibility study for the Southwestern College chatbot involves evaluating whether developing and deploying the chatbot is practical and beneficial.

#### **3.1.2.1. Technical Feasibility**

Support Vector Classifier (SVC) with a linear kernel. This is the algorithm that was initialized, trained, and used for prediction and recommendations. Libraries like sk-learn in Python provide efficient implementations of support Vector Classifier, accuracy-score, confusion-matrix.

#### **3.1.2.2. Operational Feasibility**

The operational implementation is simple and efficient. Once the system is made, the technical skills are not required to manage the web application on a daily basis. Simple knowledge about the system mentioned in the user manual and some simple guidance will be enough for the owner/admin to utilize the features and operate the system i.e, the user of the system can cope up with the system easily without being professional or trained in technical field.

#### **3.1.2.3. Economic Feasibility**

The project can be developed in a very cost-effective way because the project would be using open-source software like Python and Bootstrap which are available free online. Also, the benefits of the project outweigh the cost. Hence the project can be deemed economically feasible.

#### 3.1.2.4. Schedule Feasibility

It includes the total time for the completion of the project.

Gantt chart was used for scheduling the time for this project.

Activities	Week-1	Week-2	Week-3	Week-4	Week-5	Week-6	Week-7	Week-8	Week-9	Week-10	Week-11	Week-12
Requirement Gathering and Analysis												
Design												
Coding and Implementation												
Testing												
Deployment												
Project Documentation												

Figure 3: Gantt Chart

### 3.1.3. System Modelling using class and object diagram

In the figure 4, The class diagram shows how the four primary parts of a Flask-based illness prediction and health management system—\_App, Model, DiseaseHelper, and SymptomProcessor—interact with one another. As the main component of the system, the app acts as the point of entry and coordinates interactions with other parts to process user requests and deliver relevant answers. The machine learning model that predicts diseases is represented by the Model component, which has tools for loading the model and making predictions about diseases using input symptom data. By offering comprehensive details regarding diseases, including their descriptions, preventative measures, prescription drugs, dietary guidelines, and exercise suggestions, the DiseaseHelper component improves the system. Furthermore, the SymptomProcessor manages symptom processing by using a list of diseases and a lexicon of based on the user's input, the most likely.

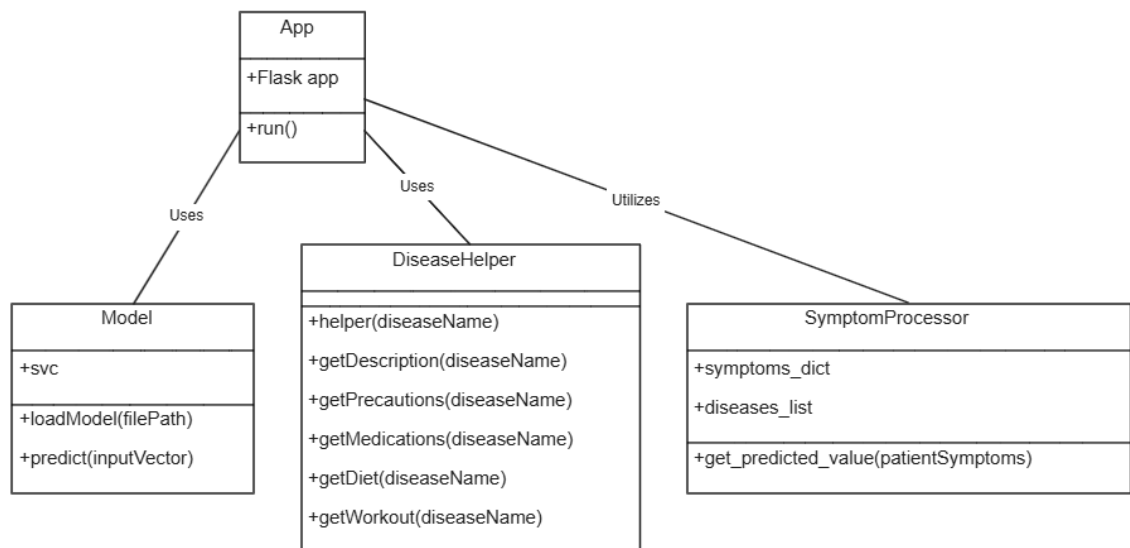


Figure 4: Class Diagram of Medicine Recommendation System

### 3.1.4 Dynamic modeling using State and Sequence Diagrams

#### Sequence Diagram

The sequence diagram illustrates the workflow of a Flask-based disease prediction system, detailing the interactions between the user, application, template, model, and helper components. The process starts when the user accesses the system and loads the index page, prompting the application to render the index.html template. The user then submits a form with their symptoms, which the application validates to ensure the input is accurate and complete. Once validated, the symptoms are passed to the machine learning Model, specifically a Support Vector Classifier (SVC), which processes the input and predicts the disease by returning a corresponding disease index. The application then interacts with the Helper component to retrieve additional information about the predicted disease, such as its description, precautions, medications, diet recommendations, and workout plans. Finally, the application incorporates these details into the index.html template, rendering it with the prediction and related information, which is displayed to the user. This structured and modular approach ensures a smooth flow of data between components, providing accurate predictions and comprehensive health information in a user-friendly manner.

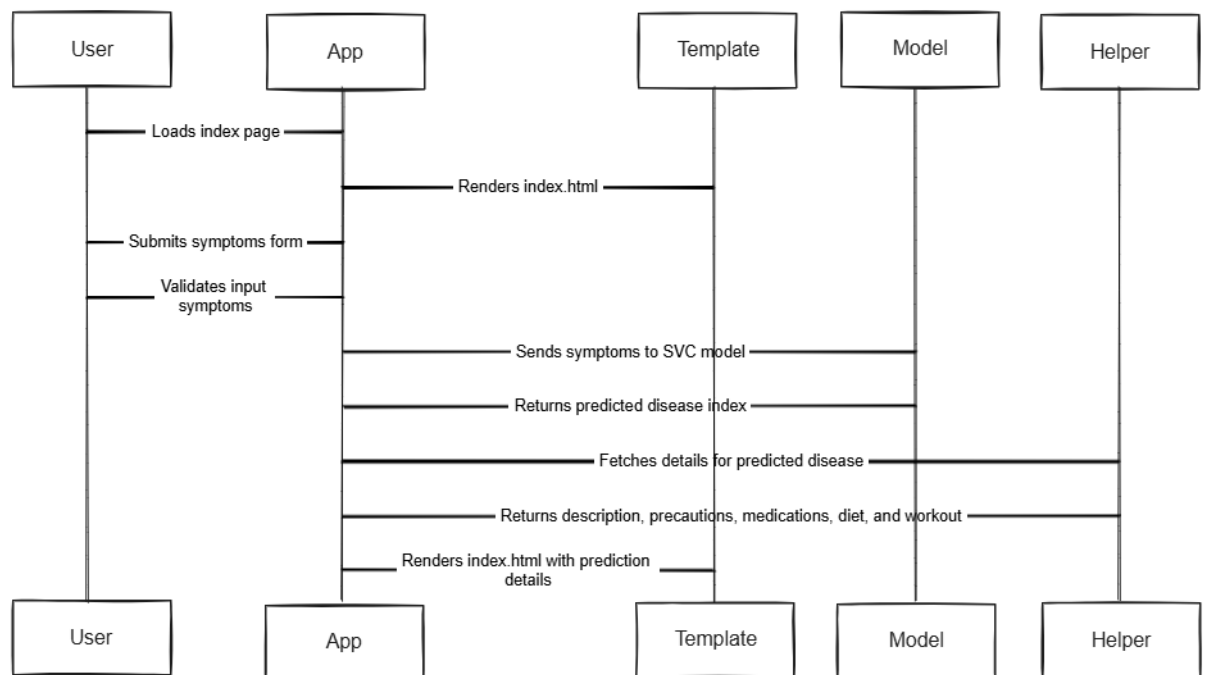


Figure 5: Sequence Diagram of Medicine Recommendation System



## State Diagram

In the below figure, the state diagram, the user requests the symptoms as input is processed with SVM which generates the output. At first user access home page, enter symptoms the system check entered symptoms are validated or not if symptoms are validated system make predict with the help of helper function and show the result on home page which is visible to user. If entered symptoms are invalidate the system redirect to again the enter symptoms page.

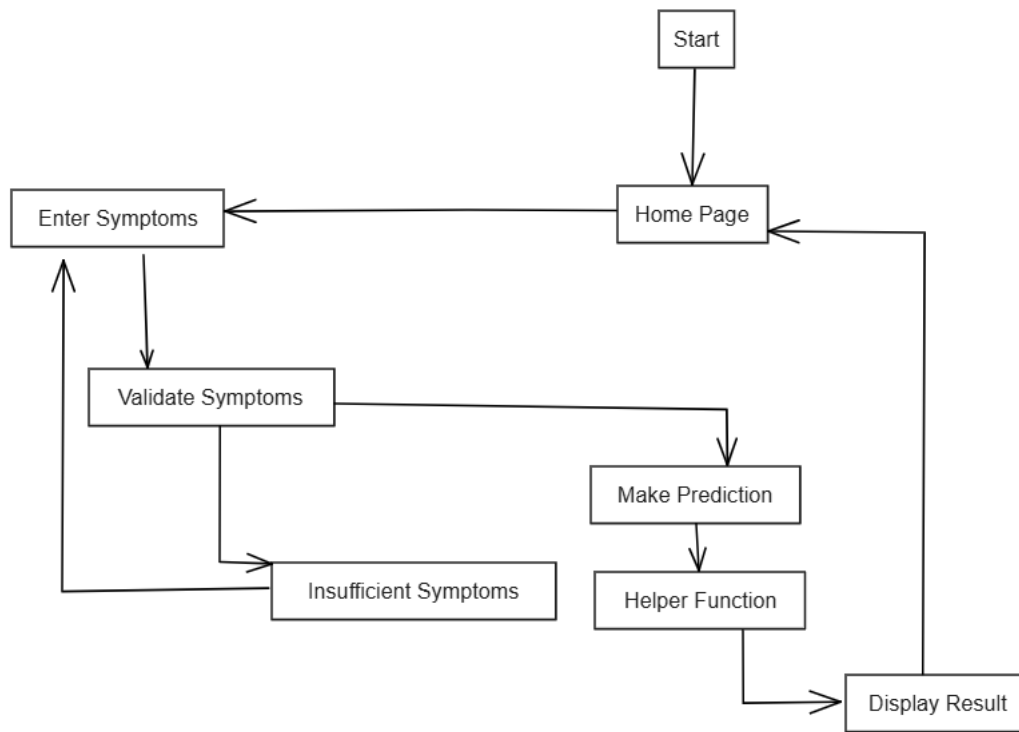


Figure 6: State Diagram of Medicine Recommendation System

### 3.1.5 Process modeling using Activity Diagrams

The below activity diagram shows the flow of the system where users enter the query which is pre-processed and bags of words are compared with existing tags to generate relevant output. On the above diagram user access the home page and input symptoms and system validate the system.

If symptoms are not validate system show error message elsewhere system prepare the input data To make disease prediction and also fetch disease details at the end display disease details and recommendations entity which are related to particular disease

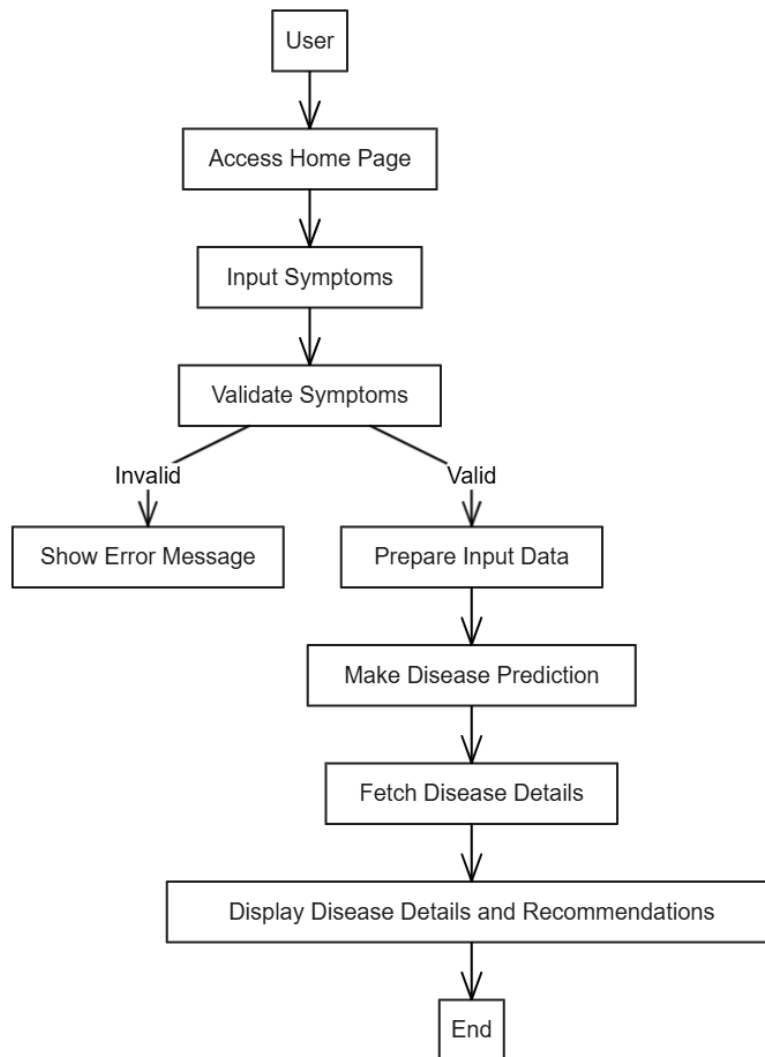


Figure 7: Activity Diagram of Medicine Recommendation System

## 3.2 System Design

System design is the process of representing architecture, interfaces, components that are included in the system. i.e., system design can be seen as the application of system theory to product development.

The UML diagrams are now refined to show more detailed description of the system component which makes it easier to understand the overall working of the system. Refined UML diagrams include class diagram, sequence diagram and activity diagram of different system modules.

### 3.2.1. Refinement of Classes and object diagram

The FlaskApp class serves as the core of the application, defining routes for handling user requests, initializing the app, and rendering templates to display results. It acts as the central orchestrator, coordinating interactions between the DiseaseModel and Dataset classes. The DiseaseModel class manages the machine learning aspects, including loading the predictive model through the `load_model()` method and generating predictions based on user-provided symptoms using the `predict(symptoms)` method. Meanwhile, the Dataset class handles data-related operations, such as loading datasets and fetching disease-specific information, including descriptions, precautions, diet, and workout recommendations. The FlaskApp uses the DiseaseModel to predict diseases and interacts with the Dataset to provide supplementary details, ensuring a comprehensive and user-friendly application. This modular design promotes maintainability, scalability, and efficient functionality.

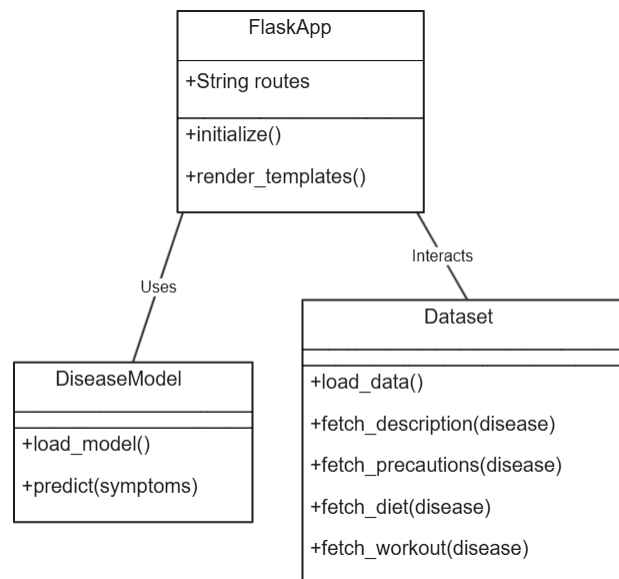


Figure 8: Refined Class Diagram of Medicine Recommendation System

## Refinement of Sequence diagram

The sequence diagram illustrates the workflow of a Flask-based disease prediction system, detailing the interactions between the user, application, template, model, and helper components. The process starts when the user accesses the system and loads the index page, prompting the application to render the index.html template. The user then submits a form with their symptoms, which the application validates to ensure the input is accurate and complete. Once validated, the symptoms are passed to the machine learning Model, specifically a Support Vector Classifier (SVC), which processes the input and predicts the disease by returning a corresponding disease index. The application then interacts with the Helper component to retrieve additional information about the predicted disease, such as its description, precautions, medications, diet recommendations, and workout plans. Finally, the application incorporates these details into the index.html template, rendering it with the prediction and related information, which is displayed to the user. This structured and modular approach ensures a smooth flow of data between components, providing accurate predictions and comprehensive health information in a user-friendly manner.

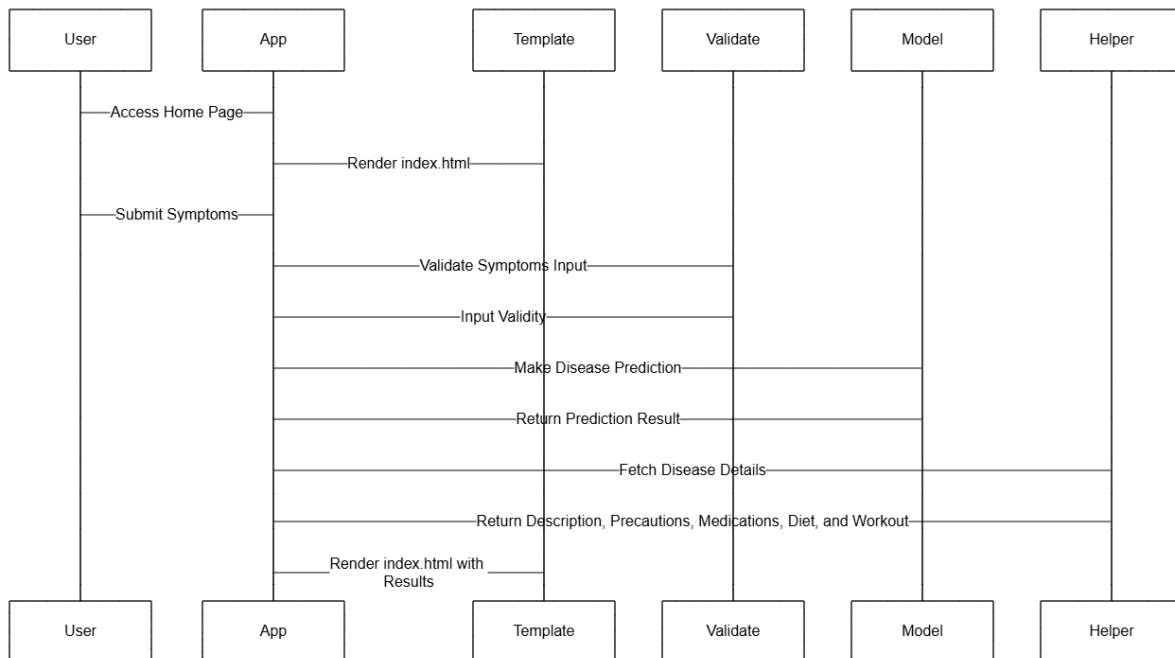


Figure 9: Refined Sequence Diagram of Medicine Recommendation System

### 3.2.2 Component Diagram

This diagram outlines the architecture of a Flask-based disease prediction system, showcasing the interaction between various components. The User interacts with the system through a User Interface, which serves as the front-end for the application. The interface connects to the Flask Application, which acts as the central controller for processing user inputs and orchestrating the backend functionalities. The Flask application interacts with three key modules: the Dataset Loader, the Model Manager, and the Helper Functions. The Dataset Loader is responsible for loading the required dataset files that support the predictive model. The Model Manager loads the trained prediction model, which is used to analyze user-provided symptoms and predict possible diseases. Meanwhile, the Helper Functions fetch supplementary details, such as disease descriptions, recommended medications, and other relevant health information. This modular design ensures that the Flask application serves as a seamless bridge between user interactions, model predictions, and additional information retrieval, providing a comprehensive and user-friendly experience

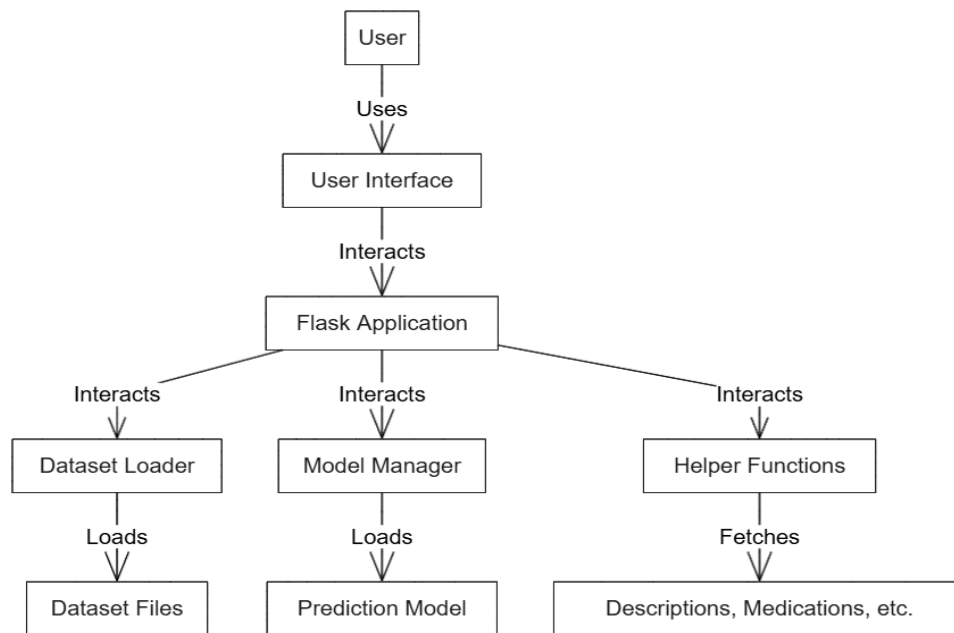


Figure 10: Component Diagram of Medicine Recommendation System

### 3.3 Algorithm Description

#### Support Vector Machine

Support Vector Machine (SVM) is a powerful and versatile supervised machine learning algorithm used for classification, regression, and outlier detection tasks. It works by finding the optimal hyperplane that best separates data points of different classes in a feature space.[3]

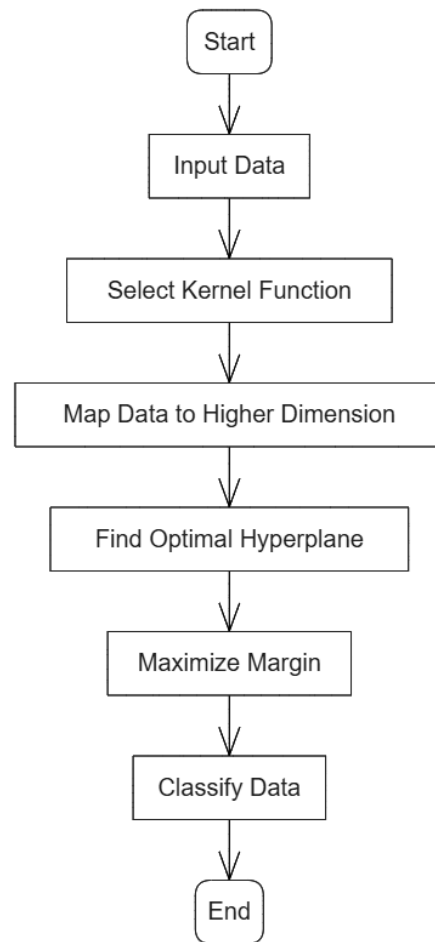


Figure 11: SVM Architecture

Support Vector Machines (SVM) can handle situations where there are more dimensions than samples and are very effective in high-dimensional domains. SVM is therefore especially well-suited for intricate datasets with a large number of characteristics. Furthermore, because SVM uses only a subset of training points—referred to as support vectors—in the decision function, it uses less memory. This methodical methodology guarantees computational effectiveness while preserving sound decision-making. Additionally, SVM is flexible since it can employ several

kernel functions for the decision boundary. Common kernels like linear, polynomial, and RBF are easily accessible, but SVM also enables the specification of unique kernels to meet the needs of particular problems.[3]

The Support Vector Machine (SVM) algorithm works systematically to classify data points or perform regression tasks. Below are the detailed steps:

Input the Dataset:

- Collect and preprocess the dataset.
- Ensure data is clean, scaled, and normalized if necessary to improve the performance of SVM.

Select the Kernel Function:

- Choose a kernel function based on the nature of the data:
- Linear Kernel: For linearly separable data.
- Non-Linear Kernels: For complex relationships (e.g., Polynomial, RBF)

Map Data to a Higher Dimensional Space

- For non-linear data, use the kernel trick to transform the data into a higher-dimensional space where it becomes linearly separable.
- SVM does not compute this transformation explicitly but relies on kernel functions to calculate dot products in the transformed space.

Find the Optimal Hyperplane:

- Identify the hyperplane that separates the classes with the maximum margin.[7]
- The margin is the distance between the hyperplane and the closest data points from each class.

Determine Support Vectors:

- Identify the data points closest to the hyperplane; these are the support vectors.
- These points define the margin and influence the position of the hyperplane.

## CHAPTER 4: IMPLEMENTATION AND TESTING

### 4.1 Implementation

The Medicine Recommendation System is designed to streamline the process of accessing medical information and recommendations, making it more efficient and user-friendly. It features a comprehensive architecture that includes frontend, backend, and database components.

The frontend is built using HTML, CSS, and JavaScript, which together create a visually appealing and interactive user interface. This interface allows users to type in queries and receive responses from the system in a conversational format. JavaScript enhances this interactivity by enabling real-time communication with the backend without needing to refresh the page. The backend is developed in Python and serves as the brain of the system. It uses advanced machine learning techniques to understand user queries. Specifically, the recommendation system employs Support Vector Classifier (SVC), a supervised learning algorithm based on Support Vector Machines (SVM), which excels at classification tasks. This enables the system to analyze user input accurately and recommend appropriate medicines. Python libraries such as pandas and scikit-learn might be used for data processing and building the SVM model.

The development process adheres to the Agile methodology, which breaks the project into manageable tasks and promotes iterative progress. This approach allows for continuous feedback and improvements, ensuring the chatbot evolves based on real user needs and feedback. Testing is a critical phase, involving unit testing of individual components, integration testing to ensure all parts work together smoothly, and user testing to gather feedback on the chatbot's effectiveness and usability. Similarly, Deployment involves releasing the chatbot to users, starting with core features and gradually adding enhancements based on feedback. Future improvements might include multilingual support to cater to a diverse user base, voice interaction capabilities for hands-free use, and personalized responses to enhance user experience.

Overall, the Medicine Recommendation System represents a significant step towards improving medical information accessibility and efficiency, leveraging cutting-edge technologies to offer a more responsive and user-friendly experience.



#### 4.1.1 Tools Used

- JavaScript

The main purpose of JavaScript, a popular and flexible programming language, is to give web pages more interaction. Along with HTML and CSS, it is a crucial component of the contemporary web development stack. By working with HTML components, modifying the Document Object Model (DOM), managing events, and much more, JavaScript gives developers the ability to provide dynamic content and improve user experience.

- PYTHON

Python is a popular, high-level programming language that is widely utilized in diverse applications like scientific computing, web development, data analysis, artificial intelligence, and more. It supports a wide range of object-oriented programming (OOP) concepts, including encapsulation, inheritance, and polymorphism. Python also boasts a rich collection of third-party frameworks and standard libraries for various applications, making it a popular choice for developers and organizations.

- Bootstrap

Bootstrap is a popular open-source front-end framework for building responsive, mobile-first websites and web applications. Initially developed by Twitter in 2011 by Mark Otto and Jacob Thornton, it provides a set of tools, including HTML, CSS, and JavaScript components, that help developers quickly design and customize web interfaces.

- Visual Studio Code

This is our code editor where we have written our all of codes. This tool is very user friendly and have lots of extensions which helps for making the coding process more efficient.

- Snipping Tool

This is a pre-installed tool on our Windows Machine which helps to take the screenshots of all required figures, documents and user interfaces.

- Microsoft Word

This tool is used to do all the documentation of our project from the scratch to the very end.

- Microsoft PowerPoint

This tool is used to make the PowerPoint slides to do presentation of our project.

### 4.1.2 Implementation Details of Module

This model can be decomposed into following modules:

#### Imports and Setup

In this module the code imports necessary libraries for data processing (pandas, numpy), building the machine learning model (SVC,SVM), handling web requests (Flask).

```
from flask import Flask, request, render_template, jsonify # Import jsonify
import numpy as np
import pandas as pd
import pickle
import warnings
from sklearn.exceptions import DataConversionWarning
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import MultinomialNB "Multinomial": Unknown word.
from sklearn.metrics import accuracy_score, confusion_matrix
```

Figure 12: Imports and Setup for Medicine recommendation system

#### Flask Application Initialization

The line `app = Flask(__name__)` initializes a Flask web application by creating an instance of the Flask class. The argument `__name__` helps Flask determine the root path of your application, allowing it to locate resources like templates and static files. This `app` object serves as the core of your web app, enabling you to define routes, handle HTTP requests, and control responses.

```
10 # Flask app
11 app = Flask(__name__)
12
```

Figure 13: Flask Application Initialization for medicine recommendation system

## Dataset Loading and Preprocessing

This part of the code loads multiple CSV files that contain datasets essential for the system's functionality, including symptoms, precautions, disease descriptions, medications, diets, and workout recommendations. It then loads a pre-trained machine learning model (svc.pkl) using the pickle library. Additionally, the helper() function is designed to retrieve and organize specific information for a given disease. It extracts the disease description, a list of precautions, recommended medications, dietary suggestions, and workout plans from their respective datasets. This structured information is returned as a tuple to be used further in the application, ensuring that all relevant data for the disease is efficiently gathered and ready for use.

```
sym_des = pd.read_csv("datasets/symptoms_df.csv")    "symptoms": Unknown word.
precautions = pd.read_csv("datasets/precautions_df.csv")
workout = pd.read_csv("datasets/workout_df.csv")
description = pd.read_csv("datasets/description.csv")
medications = pd.read_csv('datasets/medications.csv')
diets = pd.read_csv("datasets/diets.csv")

# Load model
svc = pickle.load(open('models/svc.pkl', 'rb'))

# Helper function
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]

    wrkout = workout[workout['disease'] == dis]['workout']    "wrkout": Unknown word.

    return desc, pre, med, die, wrkout    "wrkout": Unknown word.
```

Figure14: Dataset Loading and Preprocessing for medicine recommendation system

## Label Encoding

The preprocessing procedures for getting a dataset ready for machine learning are shown in the code sample. The first step is to extract the dataset's characteristics and target variable. While the target variable (y) is extracted as the prognosis column, the features (X) are extracted by dropping the prognosis column. The sklearn.preprocessing module's LabelEncoder is used to convert the target variable, which is categorical, into numerical values. The transform approach creates the encoded target Y by converting the distinct categories found in the target variable into numerical representations, which are then learned by the fit method. Lastly, the train\_test\_split function is used to divide the data into training and testing sets. Here, the random\_state is set to 20 and 70% of the data is used for training and 30% for testing.

```
X = dataset.drop('prognosis', axis=1)
y = dataset['prognosis']

# encoding prognonsis      "ecoding": Unknown word.
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)
```

Figure 15: Label Encoding for Medicine Recommendation System

## Model Training

A supervised machine learning approach for classification tasks, the Support Vector Classifier (SVC) is based on Support Vector Machines (SVM). Finding the best hyperplane to divide data points into distinct classes with the largest margin is how it operates. This margin, which is crucial in establishing the location of the decision boundary, is the separation between the hyperplane and the closest data points from each class, known as support vectors. Because it ignores other data points and concentrates only on these support vectors, SVC is resilient and less susceptible to noise. SVC employs kernels to translate input data into a higher-dimensional space where a linear hyperplane may successfully divide the classes when working with non-linearly separable data. Linear, polynomial, and radial basis functions are examples of common kernels.

```
models = {  
    'SVC': SVC(kernel='linear'),  
}  
  
# Loop through the models, train, test, and print results  
for model_name, model in models.items():  
    # Train the model  
    model.fit(X_train, y_train)  
  
    # Test the model  
    predictions = model.predict(X_test)  
  
    # Calculate accuracy  
    accuracy = accuracy_score(y_test, predictions)  
    print(f"{model_name} Accuracy: {accuracy}")  
  
    # Calculate confusion matrix  
    cm = confusion_matrix(y_test, predictions)  
    print(f"{model_name} Confusion Matrix:  
    print(np.array2string(cm, separator=', '))  
  
    print("\n" + "="*40 + "\n")
```

Figure 14: Model Definition for Medicine Recommendation System

## Web Application

The code defines a Flask web application with routes for handling user input and returning system responses. The web application uses AJAX to send asynchronous requests to the server and update the system response on the web page

```
@app.route('/predict', methods=['POST'])
def predict():
    symptoms = request.form.getlist('symptoms[]')

    # Ensure there are at least three unique symptoms
    unique_symptoms = set(symptoms)
    if len(unique_symptoms) < 3:
        return render_template(
            'index.html', message="Please enter at least three unique symptoms for accurate prediction"
        )

    input_vector = np.zeros(len(symptoms_dict))
    for symptom in unique_symptoms:
        if symptom in symptoms_dict:
            input_vector[symptoms_dict[symptom]] = 1
    input_vector = input_vector.reshape(1, -1)

    try:
        # Make prediction and map it to the disease name
        disease_index = svc.predict(input_vector)[0]
        disease_name = diseases_list[disease_index]
        dis_des, my_precautions, medications, my_diet, workout = helper(disease_name)

        return render_template(
            'index.html', predicted_disease=disease_name, dis_des=dis_des,
            my_precautions=my_precautions, medications=medications, my_diet=my_diet, workout=workout
        )
    except Exception as e:
        return render_template('index.html', message="An error occurred: " + str(e))
```

Figure 16: Homepage for Medicine Recommendations System.

## Running Medicine Recommendation application

This code ensures that the Flask application runs only when the script is executed directly, not when imported. The server runs in debug mode, making it easier to develop and troubleshoot your web application.

```
103     if __name__ == '__main__':
104         app.run(debug=True)
105
```

Figure 17: Running Medicine Recommendation application

## 4.2 Testing

Testing is the process of evaluating and verifying whether the developed software or application works properly or not i.e., whether there is match between the actual results and expected results or not. Testing is carried out during the development of the software.

### 4.2.1 Test cases for Unit Testing

Unit testing is the part of the testing methodology which includes testing of individual software modules as well as the components that make up the entire software. The purpose is to validate each unit of the software code so that it performs as expected.

Table 1: Unit Testing for Input Field

Test Case ID	Testcase Description	Test Step	Test Data	Excepted Outcome	Actual Outcome	Remarks
01	Checking response of the single symptoms	Enter the symptoms	Fever	Enter multiple symptoms	Urinary tract infection	Failed
02	Checking response of the same symptoms multiple time	Enter the multiple same symptoms	Fever, fever, fever	Please enter the unique symptoms	Urinary tract infection	Failed
03	Checking t response of the different symptoms	Enter the different symptoms	itching, skin_rash, stomach_pain, spotting_ urination	Drug Reaction	Drug Reaction	pass

#### 4.2.2 Test cases for System Testing

System testing is a process of verifying that a software system meets the specified requirements and works as intended. It evaluates the system as a whole and ensures its correct functioning.

Table 2: Testing for System

Test Case ID	Testcase Description	Test Step	Test Data	Excepted Outcome	Actual Outcome	Remarks
01	User should enter the symptoms that they have.	<ul style="list-style-type: none"><li>• Open the system</li><li>• Enter the symptoms</li><li>• Click on predict button</li></ul>	itching, skin_rash, stomach_pain, spotting_ urination	Drug Reaction	Drug Reaction	pass
02	User should enter the symptoms that	<ul style="list-style-type: none"><li>• Open the system</li></ul>	fatigue, mood_swings, weight_loss, restlessness	Hyperthyroidism	Hyperthyroidism	pass



	they have.	<ul style="list-style-type: none"> <li>• Enter the symptoms</li> <li>• Click on predict button</li> </ul>				
03	User should enter the symptoms that they have.	<ul style="list-style-type: none"> <li>• Open the system</li> <li>• Enter the symptoms</li> <li>• Click on predict button</li> </ul>	<ul style="list-style-type: none"> <li>• Fever</li> <li>• Fever</li> <li>• Fever</li> <li>• fever</li> </ul>	Please enter the three unique symptoms	Urinary tract infection	fail
04	User should enter the symptoms that	<ul style="list-style-type: none"> <li>• Open the system</li> </ul>	<ul style="list-style-type: none"> <li>• Fever</li> <li>• Headache</li> </ul>	Please enter the three At least three symptoms	Urinary tract infection	fail

	they have.	<ul style="list-style-type: none"> <li>• Enter the symptoms</li> <li>• Click on predict button</li> </ul>				
05	User should enter the symptoms that they have.	<ul style="list-style-type: none"> <li>• Open the system</li> <li>• Enter the symptoms</li> <li>• Click on predict button</li> </ul>	<ul style="list-style-type: none"> <li>• continuous_sneezing, shivering, chills, watering_from_eyes</li> </ul>	Please remove the special symbol(,)	Allergy	fail
06	User should enter the symptoms that	<ul style="list-style-type: none"> <li>• Open the system</li> </ul>	<ul style="list-style-type: none"> <li>• continuous_sneezg</li> <li>• shivering</li> <li>• chills</li> </ul>	Allergy	Allergy	pass

	they have.	<ul style="list-style-type: none"> <li>• Enter the symptoms</li> <li>• Click on predict button</li> </ul>	<ul style="list-style-type: none"> <li>• watering_from_eyes</li> </ul>			
07	User should not enter the symptoms that they have.	<ul style="list-style-type: none"> <li>• Open the system</li> <li>• Enter the symptoms</li> <li>• Click on predict button</li> <li>•</li> </ul>	<ul style="list-style-type: none"> <li>• No symptoms</li> </ul>	Please enter symptoms	Urinary tract infection	fail

## Result Analysis

Demonstration of the project shows that the system is successful in providing transparent medical information and recommendations to users. Both the functional and non-functional requirements were fulfilled during testing. The results showed that the project was able to meet its goals, but there is still room for improvement in terms of expanding the system's capabilities and providing even more accurate information.

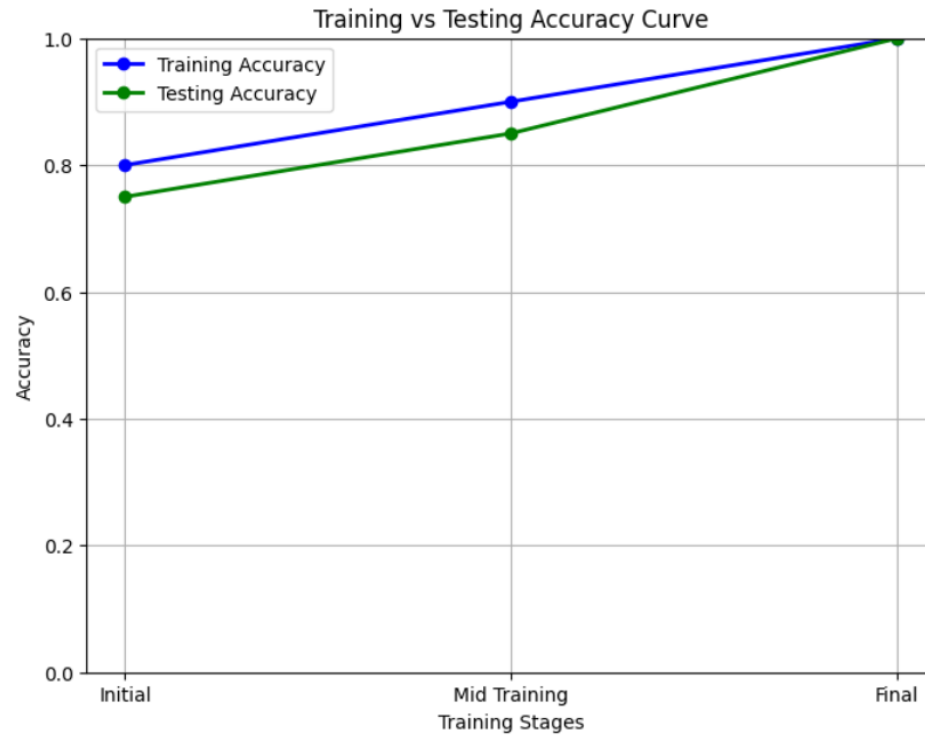


Figure 18:Result Analysis

## **CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATION**

### **5.1 Conclusion**

The Medicine Recommendation System is a digital tool designed to make healthcare more personalized and accessible. It uses advanced algorithms to analyze patient symptoms and recommend suitable treatments, such as medications, workout routines, and diet plans. This system simplifies the healthcare process, providing timely and accurate advice without confusion. It empowers patients by empowering them with more control over their health journey and making personalized care more accessible.

### **5.2 Lesson Learnt/ Outcome**

- Understanding of AI and SVC
- Developing this chatbot provided practical experience with AI technologies, particularly SCV and SVM networks. The project team gained a deeper understanding of how these technologies can be used to process and respond to user inputs effectively.
- Agile Methodology: The use of Agile methodology emphasized the importance of iterative development, collaboration, and continuous feedback. This approach allowed the team to adapt to changes and improve the system incrementally based on real-time feedback.
- Problem-Solving Skills: The team encountered various challenges related to training the SVC model, handling SVM words, and optimizing response accuracy. Overcoming these issues led to improved problem-solving skills and a better understanding of model tuning.\
- Technical Integration: The project demonstrated the importance of integrating multiple technologies (Python, Bootstrap, CSS, JavaScript ) to build a comprehensive solution. This integration enabled a seamless flow between frontend, backend connection use flask.

### **5.3 Future Recommendations**

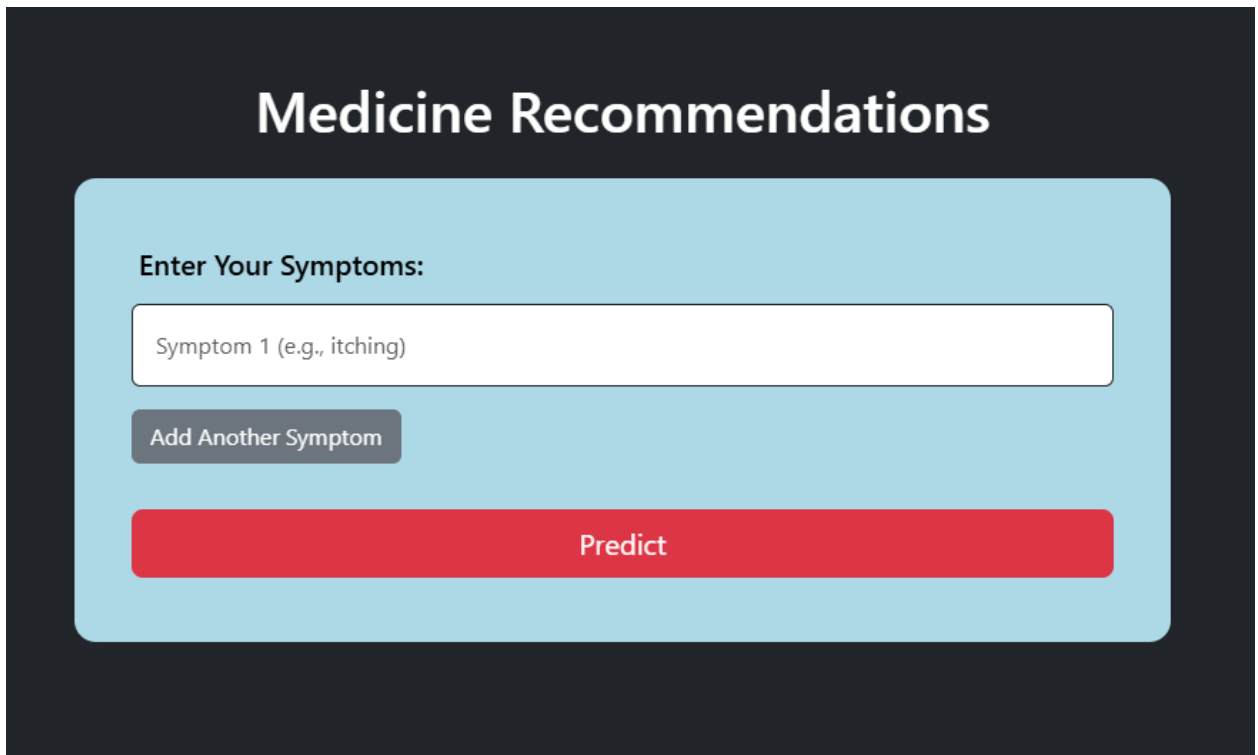
The project can work on many things which will enhance the use of this system. Some modifications that can make are listed below:

- It is also possible to develop a system that operates through voice commands.
- Its accuracy may be improved by productive labor to a certain degree, where it can determine the precise output without error and respond in accordance with the associated symptoms.

## REFERENCES

- [1] P. Khairnar<sup>1</sup>, Medicine Recommend System Using Machine Learning, Maharashtra, India: Aditya Hargane, 24 May 2022.
- [2] A. K. Binda, Medicine Recommender System: a Machine Learning Approach, Haryana, India: Global University, Saharanpur, 2022.
- [3] D. Cournapeau, Machine Learning in Python, scikit-learn, February 1st 2010.
- [4] Gloria Phillips-Wren, Manuel Mora, Fen Wang and Jorge Marx Gomez, Symptom based COVID-19 test recommendation system using machine learning technique, 18 April 2022.
- [5] J. Chen, A disease diagnosis and treatment recommendation system based on big data mining and cloud computing, Information Sciences, April 2018.
- [6] E. Starkman, Drugs Interaction Checker, Internet Brands , 2005 - 2024.
- [7] ResearchGate, Hybrid Techniques to Predict Solar Radiation Using Support Vector Machine and Search Optimization Algorithms: A Review - Scientific Figure on ResearchGate. Available from: <https://www.researchgate.net/figure/General-architecture-of-a-support-vector-machin>.

## APPENDIX



# Medicine Recommendations

Enter Your Symptoms:

Add Another Symptom

Predict

Home Page of medicine recommendation system