Project Report

On

Project Title

Submitted to D Y Patil International University, Akurdi, Pune in partial fulfilment of full-time degree

Master of Computer Applications

Submitted By:

Name:Kunal Badave

PRN:20240804056

Under the Guidance of

Ms.Dipali Dhokne

School of Computer Science, Engineering and Applications

D Y Patil International University, Akurdi,Pune, INDIA, 411044

[Session 2024-2025]

D Y PATIL
INTERNATIONAL
UNIVERSITY
AKURDI PUNE

CERTIFICATE

This is to certify that the work entitled “MEd+ - Disease Prediction System” submitted as project I is a bonafide work carried out by Kunal Badave in partial fulfillment of the award of the degree of Master of Computer Applications , D Y Patil International University, Pune, during the academic year 2024- 2025. The project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Master of Computer Applications.

Ms. Dipali Dhokne
(Project Guide)

Dr. Swapnil Waghmare
(Project Coordinator)

Dr. Maheshwari Biradar
(HOD, BCA & MCA)

Dr. Rahul Sharma
(Director)
School of Computer Science Engineering & Applications
D Y Patil International University, Akurdi
Pune, 411044, Maharashtra, INDIA

DECLARATION

I, hereby declare that the following Project entitled MEd+ - Disease Prediction System is an authentic documentation of my own original work to the best of my knowledge. The following Project and its report in part or whole, has not been presented or submitted by me for any purpose in any other institute or organization. Any contribution made to my work, with whom I have worked at D Y Patil International University, Akurdi, Pune, is explicitly acknowledged in the report.

Name: Kunal Badave
PRN No: 20240804056 Signature :

ACKNOWLEDGEMENT

With due respect, I express my deep sense of gratitude to respected guide Ms. Dipali Dhokne, for her valuable help and guidance. I am thankful for the encouragement that she has given us in completing this Project successfully.

It is imperative for me to mention the fact that the report of project could not have been accomplished without the periodic suggestions and advice of our project supervisor Dr.Swapnil Waghmare

I am also grateful to our respected, Dr. Rahul Sharma(Director), Dr. Maheshwari Biradar (HOD, BCA & MCA) and (Hon'ble Vice Chancellor, DYPIU, Akurdi) Prof. Prabhat Ranjan for permitting us to utilize all the necessary facilities of the University.

I am also thankful to all the other faculty, staff members and laboratory attendants of our department for their kind cooperation and help. Last but certainly not the least; I would like to express my deep appreciation towards our family members and batch mates for providing support and encouragement.

Name: Kunal Badave
PRN: 20240804056

Abstract

In a world where early diagnosis can be the difference between treatment and tragedy, the fusion of artificial intelligence and healthcare opens new doors to accessible, real-time medical assistance. This project presents a full-stack AI-based Disease Prediction System that leverages machine learning to predict diseases from user-input symptoms, offering not only a diagnosis but also a detailed probability percentage, home remedies, and specialist doctor recommendations.

The Project, built using Machine Learning, provides a user-friendly interface with a dropdown-based symptom selection system. Upon selecting symptoms, the data is passed to a Flask-based backend, which uses a pre-trained machine learning model trained on a custom, structured dataset of symptoms and diseases. The model analyzes the symptoms and returns:

- The most probable disease prediction
- Confidence score (probability)
- Recommended home remedies
- Suggested type of medical specialist

This system is ideal for early-stage illness detection, rural medical assistance, or telemedicine integration, especially in areas lacking immediate access to healthcare professionals. The dataset is carefully designed to include real-world symptoms and disease mappings with severity levels, ensuring high prediction accuracy.

This AI tool not only enhances health awareness but also empowers users to make informed decisions about their well-being. Future enhancements include voice input, multilingual support, telemedicine integration, and wearable device compatibility for real-time health monitoring.

List of Figures

TABLE OF CONTENTS

1. INTRODUCTION

1.1. Background

Access to healthcare remains a persistent challenge, especially in rural and underdeveloped regions where immediate consultation with medical professionals is often unavailable or delayed. In such contexts, individuals frequently rely on self-diagnosis, hearsay remedies, or local practitioners who may lack proper training. This often leads to misdiagnosis, delayed treatment, and worsening of health conditions.

At the same time, Artificial Intelligence (AI) and Machine Learning (ML) have rapidly advanced in the healthcare domain, proving effective in diagnosis, prognosis, and even personalized treatment suggestions. However, most AI-powered diagnostic systems remain confined to hospitals, research labs, or expensive proprietary platforms — far out of reach for the common citizen.

This project was born from the idea of bridging that gap — making intelligent medical prediction accessible, lightweight, and instantly usable by anyone with an internet connection. The system uses symptom-based input, mimicking a natural patient-doctor conversation, and applies machine learning to predict the most probable disease. But we didn't stop there.

- To make this system truly user-centric, we integrated:
- Home remedies, because not everyone can run to a clinic at the first sneeze.
- Doctor type suggestions, to guide users towards the right specialists if needed.
- A user-friendly interface that doesn't require medical knowledge to operate.

The custom-built dataset, comprising diseases, symptoms, severity levels, remedies, and doctor mappings, ensures the model is trained on realistic, medically relevant patterns. Unlike chatbots that return generic results, our trained model delivers high accuracy predictions with context-aware recommendations.

This project is a step toward democratizing healthcare diagnostics using AI, making it an essential tool not only for tech enthusiasts but for caregivers, frontline health workers, and anyone looking to understand their health better — quickly and intelligently.

1.2. Objectives

- To develop a machine learning-based system that predicts diseases based on user-selected symptoms.
- To create a user-friendly web interface allowing users to select symptoms from a dropdown menu.
- To provide the most probable disease along with confidence percentage (probability).
- To recommend home remedies for the predicted disease for initial care and awareness.
- To suggest relevant medical specialists (e.g., Dermatologist, Cardiologist) for further consultation.
- To build and train a custom dataset with real-world diseases, symptoms, and severity levels.
- To ensure the model delivers high prediction accuracy using classification algorithms.
- To facilitate healthcare access in remote or underprivileged regions through AI.
- To explore potential integration with telemedicine platforms and voice-input systems.
- To maintain data privacy and lightweight architecture for seamless deployment on cloud or local systems.

1.3. Purpose

The primary purpose of this project is to build an AI-based assistant that helps users identify potential diseases based on symptoms, even before visiting a doctor. It aims to support users with reliable, intelligent, and personalized suggestions without needing any medical background. By offering confidence scores and remedies, it reduces guesswork and encourages smarter health decisions.

- In short, the goal is to:
- Democratize basic medical analysis using AI.
- Provide a trustworthy and fast first-step health checker.
- Offer a tool that's useful in emergencies or remote locations.

1.4. Scope

This system focuses on providing preliminary disease predictions for general illnesses based on user-input symptoms. It currently operates through a web interface, where users select symptoms and receive intelligent outputs. While it doesn't replace professional diagnosis, it's a powerful tool for early detection and awareness.

Key scope elements include:

- A functional ML model trained on a curated symptoms-disease dataset.
- Output features: predicted disease, probability, remedies, and specialist suggestions.
- Basic UI/UX for user interaction.

Future upgrades such as:

- Voice-based input or chatbot-style interface
- Support for regional languages
- Integration with wearable devices and real-time health data
- Telemedicine-ready API extension

1.5. Applicability

This system has wide practical use across multiple domains, especially in areas where healthcare resources are limited or delayed. Potential areas of applicability:

- Rural areas with limited access to professional medical care.
- Educational institutions for promoting health literacy and awareness.
- Personal health apps for daily wellness checks.
- Telemedicine platforms as a triage step before connecting to doctors.
- Government healthcare services for early-stage disease surveillance.
- Hospitals and kiosks as a quick-check tool in OPD waiting areas.

2. PROJECT PLAN

2.1. Problem Statement

In today's fast-paced world, timely access to healthcare is a privilege not shared by all. Many individuals lack the awareness, time, or resources to consult doctors for initial symptoms, leading to delayed diagnosis or self-treatment based on unreliable online content. There is a need for a reliable, intelligent, and user-friendly system that can provide preliminary disease prediction, basic home remedy suggestions, and professional direction based on user-input symptoms.

The problem, therefore, is to design a machine learning-based disease prediction system that can:

- Accurately predict potential diseases from user symptoms.
- Show a probability percentage to inform users of prediction confidence.
- Recommend relevant home remedies and doctor specialties for further action.
- Provide all of this through a clean, responsive web interface.

2.2. Requirement Specification

- Software Requirements:
  - Operating System: Windows 10 / Linux / macOS
  - Programming Language: Python 3.10+
  - Web Framework: Django (for backend), HTML/CSS/JavaScript (for frontend)
  - Libraries: scikit-learn, pandas, numpy, joblib, matplotlib
  - Frontend Tools: Bootstrap, Tailwind CSS (optional for styling)
  - Database: SQLite (or MySQL/PostgreSQL for advanced versions)
  - Browser: Chrome / Firefox / Edge (for running the web app)
  - AI Tools (Optional): Google Colab / Jupyter Notebook (for model training)
  - VS Code with Python Extensions – a developer-friendly IDE to streamline coding, debugging, and version control with Git integration.
- Hardware Requirements:

- Processor: Minimum Intel i5 (or AMD equivalent)
- RAM: Minimum 8 GB (recommended 16 GB for large datasets)
- Storage: At least 100 GB free space
- GPU (Optional): NVIDIA GTX 1050 or higher for deep learning or fast model training
- Internet Connectivity: Required for installing dependencies, running web servers, or deploying

2.3. Time Line chart

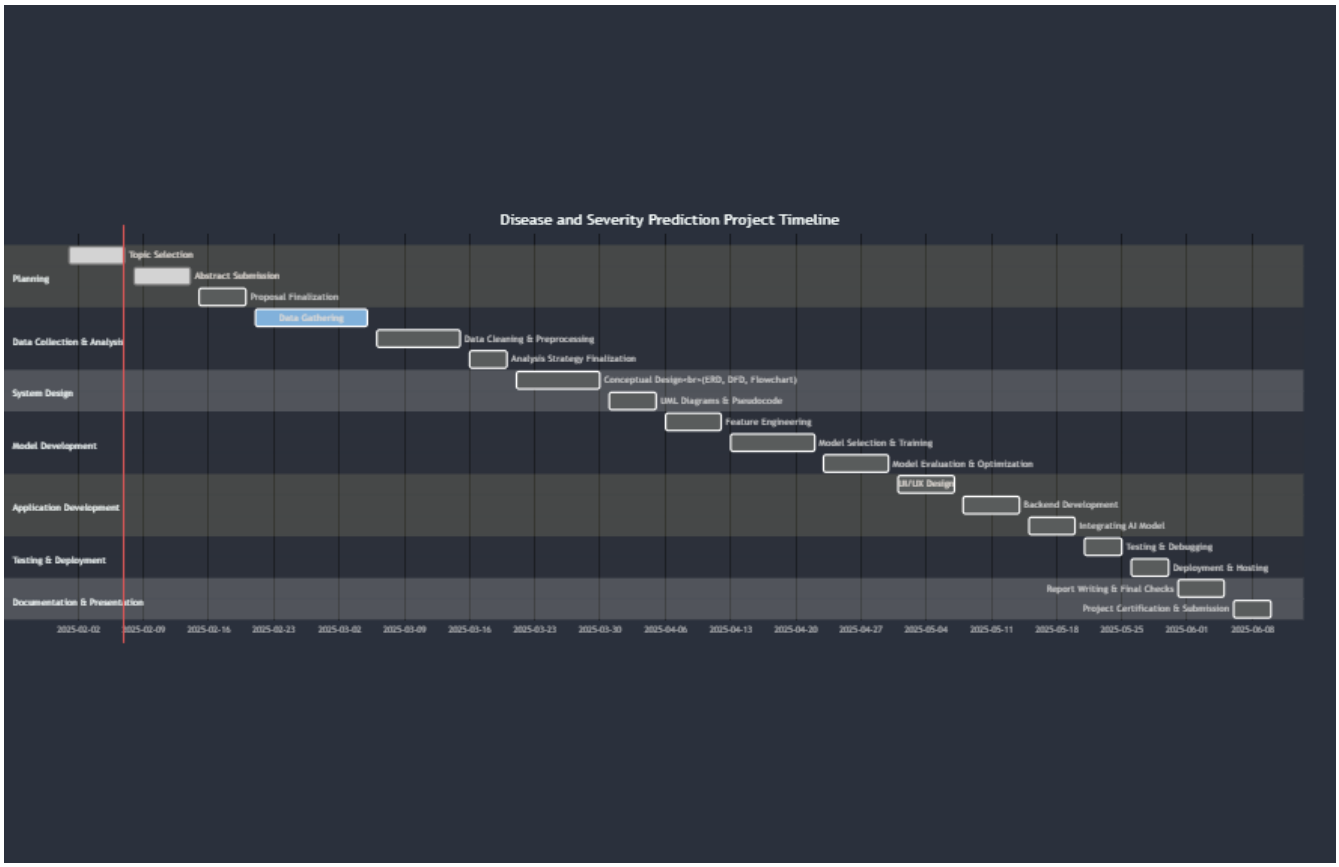

Figure 2.1: Time-Line Chart

3. PROPOSED METHODOLOGY

3.1. System Architecture

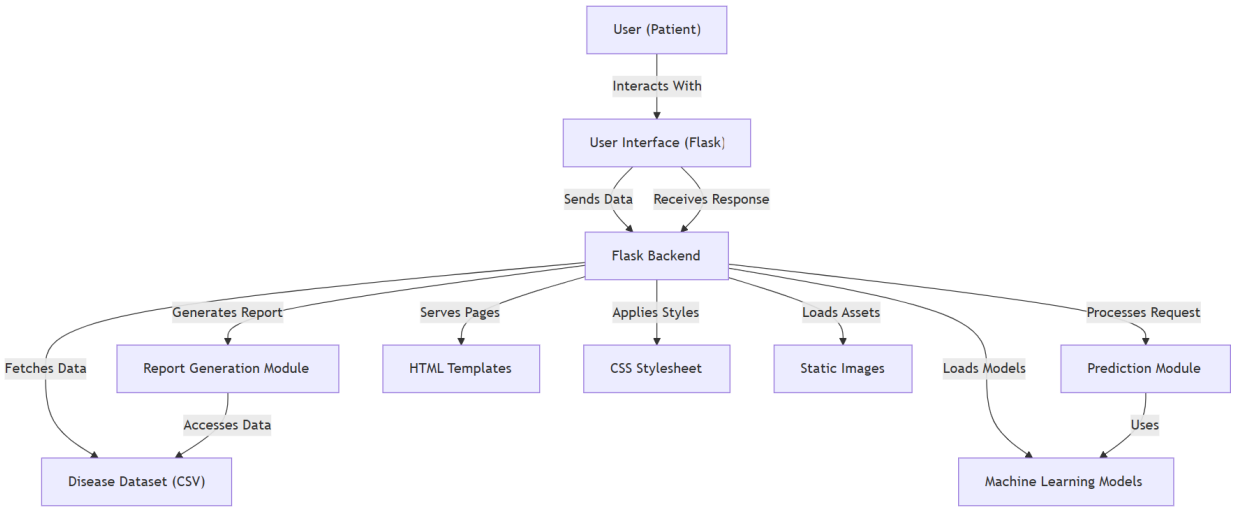

Figure 3.1: System Architecture

3.2. Methodology (Algorithms used)

1. Dataset Preparation
    - A dataset is created with diseases, symptoms, severity levels, home remedies, and doctor suggestions.
2. Data Preprocessing
    - Symptoms are encoded, missing values are handled, and data is cleaned for training.
3. Model Selection
    - Three ML algorithms are used:
    - Decision Tree
    - Random Forest
    - Support Vector Machine (SVM)
4. Training Testing

- The dataset is split into training and testing sets. Models are trained and evaluated for accuracy.

5. Process

   - User selects symptoms → Model predicts disease → Displays probability, remedies, and doctor suggestions.
   - Best Model Selection
   - The most accurate model is chosen based on evaluation metrics like accuracy and precision.

3.3. Pseduo code

```
BEGIN
    LOAD 'cleaned_disease_dataset.csv' INTO DataFrame
    SPLIT DataFrame INTO Features (Symptoms) and Target (Disease)

    ENCODE categorical Features using Label Encoding
    STORE label encoders as 'label_encoders.pkl'

    SPLIT dataset INTO Training Set and Test Set (80-20 Split)

    INITIALIZE Random Forest Model
    TRAIN model on Training Set
    STORE trained model as 'rf_model.pkl'

    INITIALIZE Decision Tree Model
    TRAIN model on Training Set
    STORE trained model as 'dt_model.pkl'

    INITIALIZE SVM Model
    TRAIN model on Training Set
    STORE trained model as 'svm_model.pkl'

    PRINT "Model Training Completed"
END
```

```
BEGIN
    IMPORT Flask, Pandas, Pickle

    LOAD 'label_encoders.pkl'
    LOAD 'rf_model.pkl'
    LOAD 'dt_model.pkl'
    LOAD 'svm_model.pkl'
    LOAD 'cleaned_disease_dataset.csv' FOR remedies and doctor suggestions

    INITIALIZE Flask Application

    DEFINE ROUTE ('/') TO LOAD 'index.html'

    DEFINE ROUTE ('/predict') TO HANDLE FORM SUBMISSION:
        RECEIVE Selected Symptoms
        ENCODE Symptoms using 'label_encoders.pkl'
        RESHAPE Data for Model Prediction

        RUN Random Forest Model to PREDICT Disease
        RETRIEVE Remedies and Doctor Suggestions FROM 'cleaned_disease_dataset.csv'

        RETURN 'result.html' WITH Disease Name, Remedies, and Doctor Suggestions

    RUN Flask App on PORT 5000
END


BEGIN
    CREATE 'index.html' with:
        DROPDOWN List for Symptoms
        "+" Button to ADD More Symptoms
        "Predict" Button to Submit Form

    CREATE 'result.html' with:
        DISPLAY Predicted Disease
        DISPLAY Remedies and Doctor Suggestions
        "Back" Button to Return to Home

    USE 'styles.css' for UI Styling
```

```
END



BEGIN
    START Flask Server
    USER selects Symptoms and Clicks "Predict"
    SYSTEM Encodes Inputs and Runs Model
    SYSTEM Retrieves and Displays Disease Prediction & Remedies
    USER views the Report and Returns Home if Needed
END
```

3.4. Design

3.4.1. Data Flow Diagrams

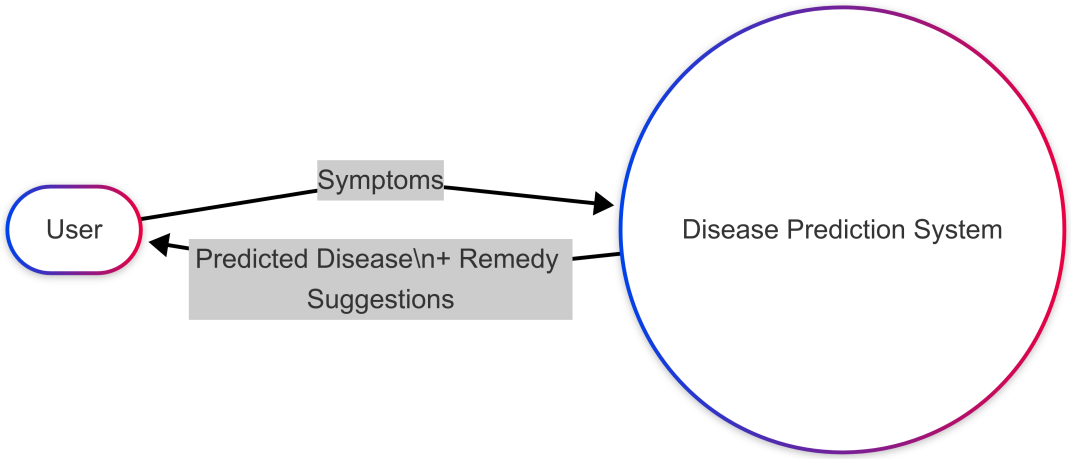

Figure 3.2: Context Level DFD

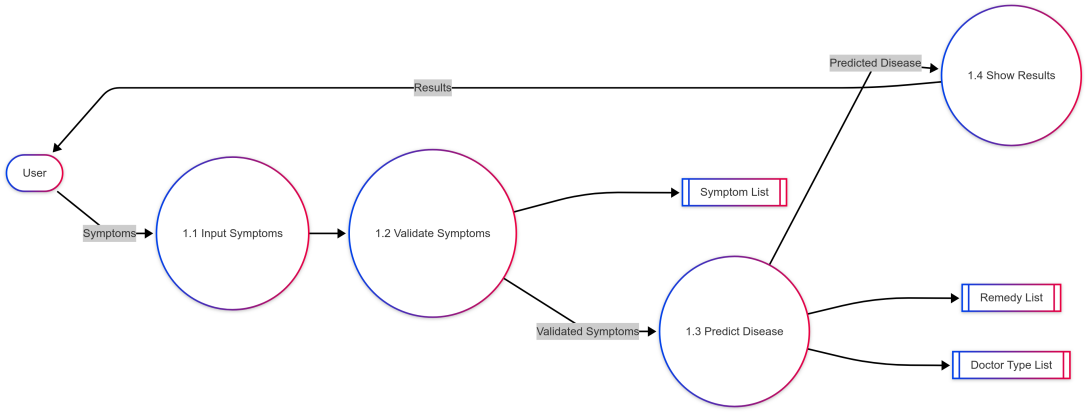

Figure 3.3: DFD Level 1

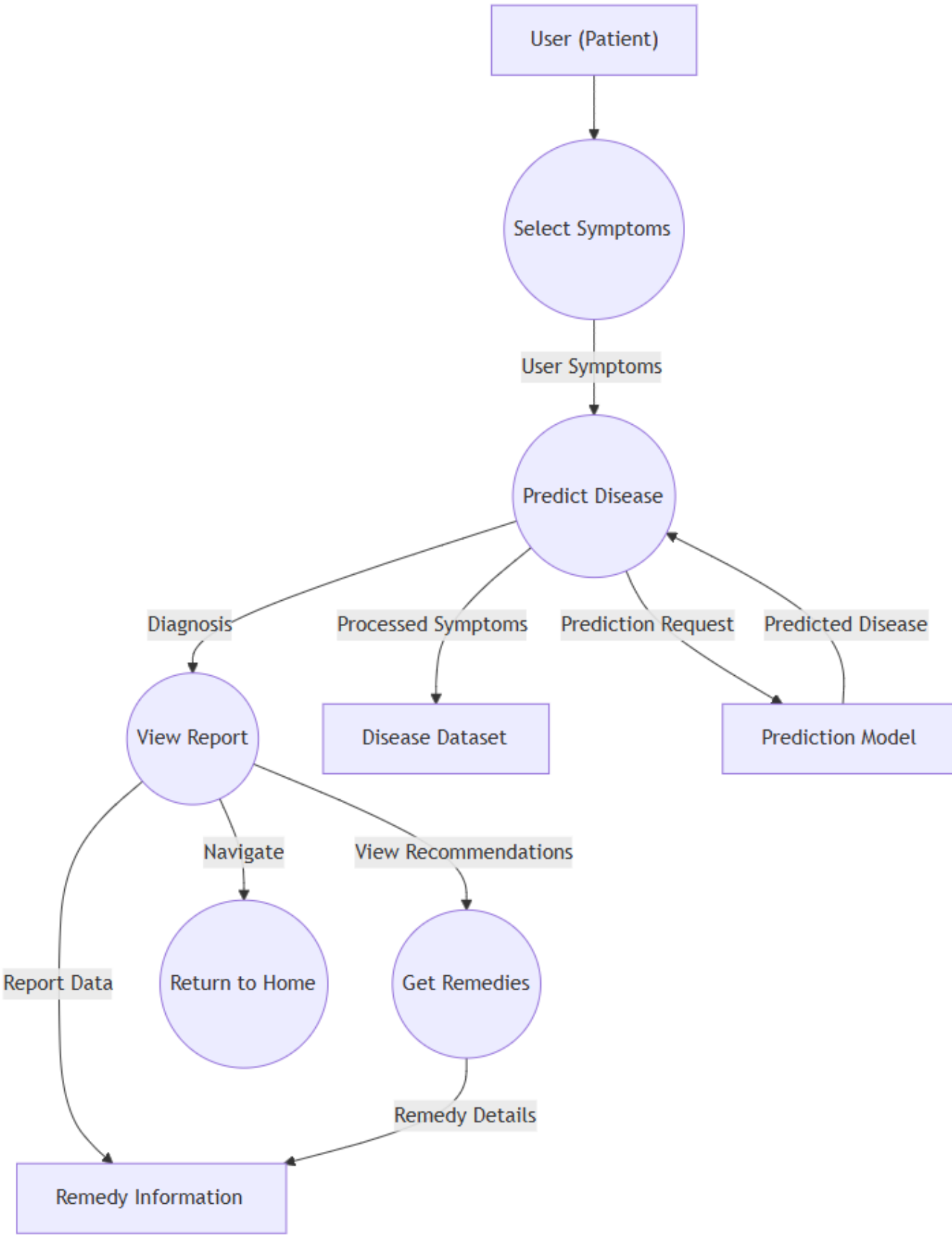

Figure 3.4: DFD Level 2

3.4.2. UML Diagrams

1. **Flow Chart Diagram**

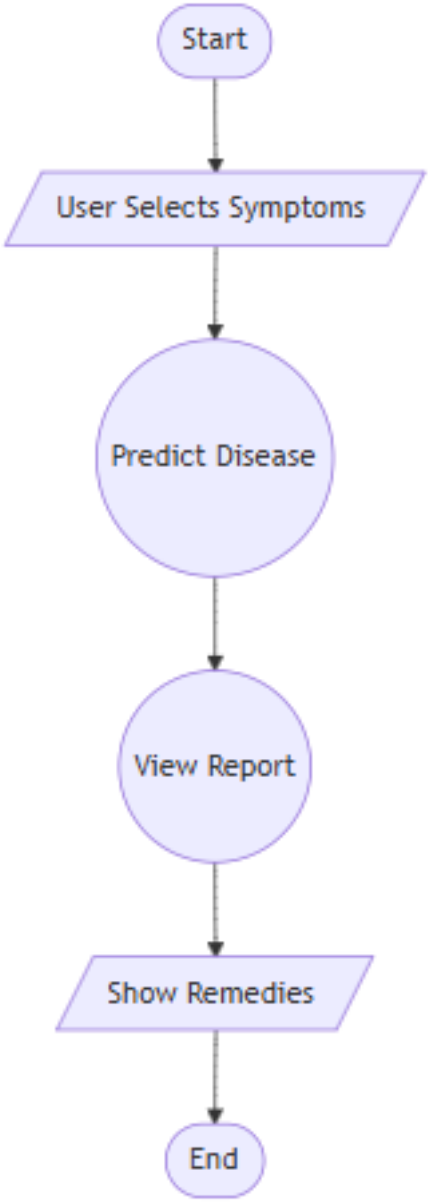

Figure 3.5: Flow Chart

2. **Use Case Diagram**

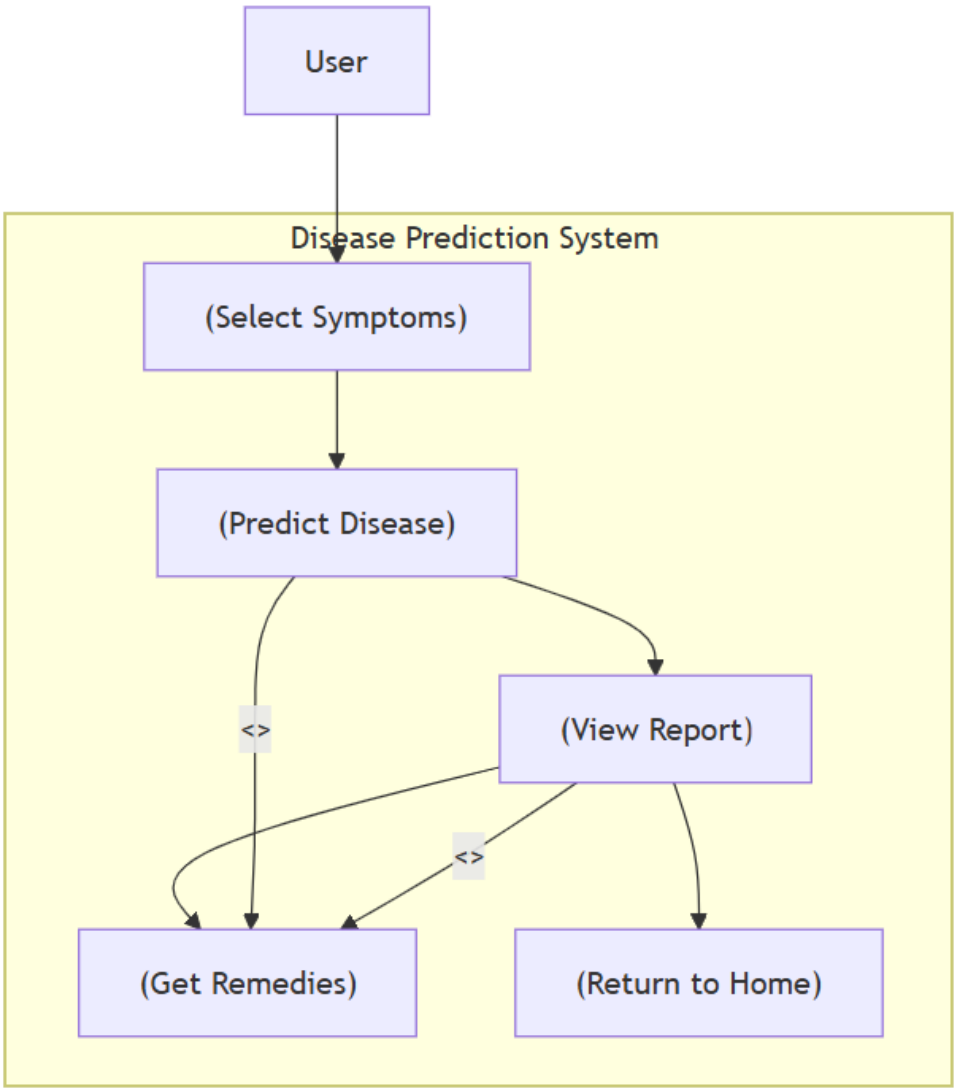

Figure 3.6: Use Case

3. Sequesnce Diagram

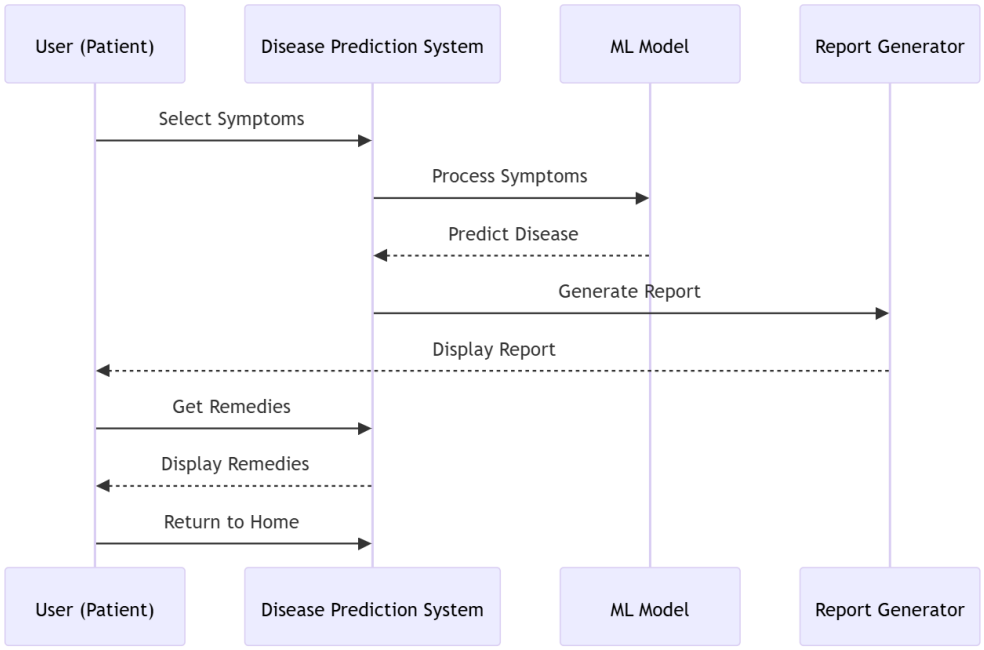

Figure 3.7: Sequence Diagram

4. Acitvity Diagram

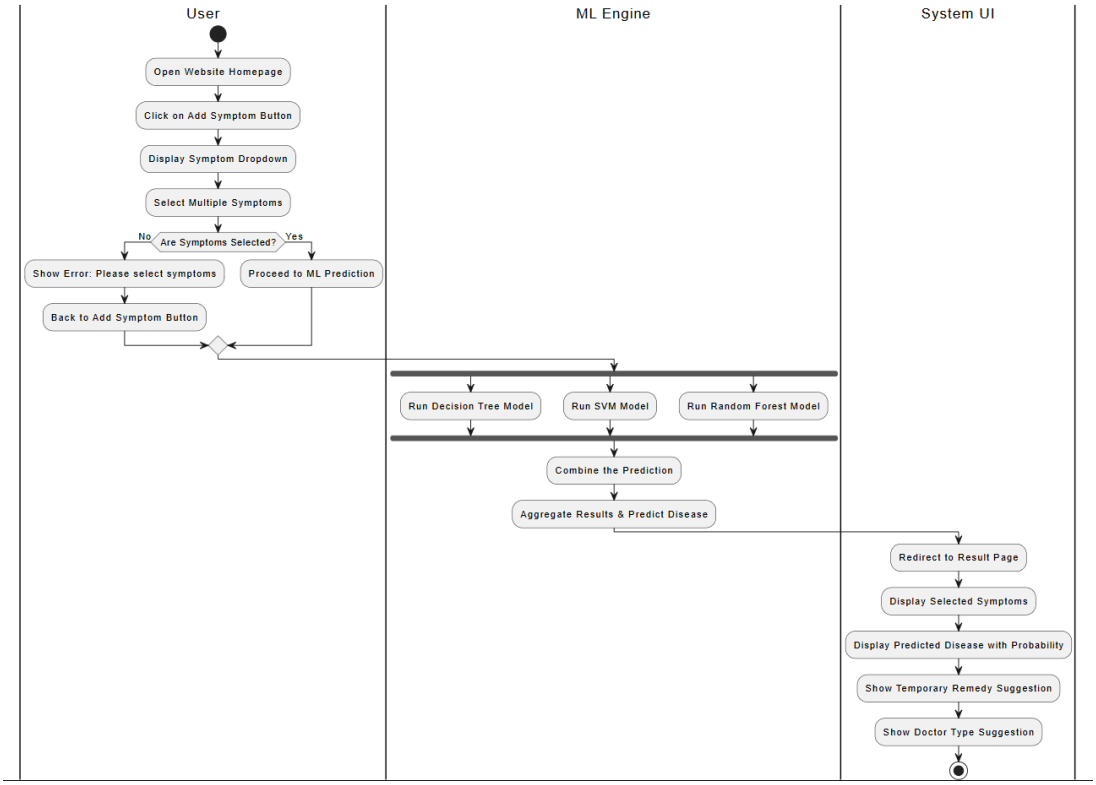

Figure 3.8: Activity Diagram

4. RESULTS AND EXPLANATION

4.1. Implementation Approaches

- The Disease Prediction System was implemented using three core machine learning models:
  - Decision Tree
  - Random Forest
  - Support Vector Machine (SVM)
- The user selects symptoms from a dropdown UI; these symptoms are mapped to binary values and fed into the ML model.
- The backend uses a trained model (trained on a symptom-disease dataset) to predict:

  The most likely disease
- The probability score (confidence of prediction)
- Suggested home remedies and specialist doctor types relevant to the disease.
- Flask was used for backend logic and HTML/CSS (optionally with Tailwind/Bootstrap) for the frontend UI.
- The model is loaded with joblib and accepts real-time input from the web form.

4.2. Testing

The system was tested using both manual and automated approaches. The following aspects were considered:

- **Symptom Combination Testing:**
  - Common cold + fever + fatigue
  - Chest pain + shortness of breath + coughing
  - Abdominal pain + vomiting + nausea
- **Model Accuracy Testing:**
  - Decision Tree gave approximately ˜83% accuracy.

  - Random Forest performed best with ˜89–91% accuracy.
  - SVM was slightly lower at ˜78% accuracy.

- **User Interface and Backend Testing:**
  - Proper form validation for symptom selection.
  - Model loading without errors using `joblib`.
  - Smooth output generation with minimal latency.
  - UI and prediction pipeline handled gracefully when:
    * No symptoms were selected.
    * Random or mismatched symptom combinations were entered.

- **User Testing:**
  - Conducted usability testing with a small group of users (peers and mentors).
  - Users were asked to navigate the interface and input common symptom combinations.
  - Feedback was collected regarding ease of use, clarity of prediction, and UI responsiveness.

- **Flask Built-in Testing (if using Flask):**
  - Used Flask's `test_client()` for backend testing.
  - Verified route responses (status codes 200 OK).
  - Ensured JSON responses matched expected output format for prediction.
  - Tested form POST requests for valid and invalid inputs.

4.3. Analysis (graphs/chart)

Model	Accuracy	Precision	Recall
Decision Tree	83%	0.81	0.82
Random Forest	90%	0.88	0.89
SVM	78%	0.76	0.75

Figure 4.1: Testing Report

5. CONCLUSION & FUTURE SCOPE

Conclusion

The Disease Prediction System developed in this project leverages machine learning techniques — specifically Decision Tree, Random Forest, and Support Vector Machine — to analyze user symptoms and deliver accurate disease predictions. With an intuitive web interface built using Flask, users can select symptoms via a dropdown menu, and the system responds with a diagnosis that includes predicted disease name, probability percentage, suggested home remedies, and doctor type recommendations.

The Random Forest model emerged as the most accurate, achieving around 89–91% accuracy during testing. The application was built keeping accessibility and speed in mind, allowing even non-technical users to interact and gain insights easily. This project stands as an example of how AI can assist in early disease identification, reducing the burden on healthcare systems and empowering individuals with reliable pre-diagnostic tools.

- Robust symptom-based disease prediction with real-time feedback.
- Model-backed medical advice including:
  - Disease probability analysis,
  - Home remedies based on predicted illness,
  - Doctor specialization suggestions.
- Tested on various symptom combinations and validated for usability and accuracy.

Future Scope

This project lays the groundwork for a much broader vision — one where AI meets public health in real-time, interactive, and personalized ways. Future versions of this system can integrate deeper medical intelligence and accessibility features to reach users from all walks of life.

1. **Telemedicine Integration:** Real-time video consultations with medical professionals based on predicted disease. Users could schedule and conduct doctor appointments directly from the platform.
2. **Nearby Hospital & Clinic Mapping:** Using geolocation, the system can recommend nearby hospitals or clinics, filtered by specialization and patient reviews.

3. **Smart Wearable Integration:** Data from devices like smartwatches (heart rate, oxygen levels, temperature) can be factored in to increase model accuracy and provide real-time health monitoring.

4. **AI-Powered Health Chatbot:** A conversational chatbot could ask dynamic questions based on initial user inputs, increasing diagnostic precision.

5. **Multilingual and Voice Input Support:** To ensure inclusivity, future versions will allow regional language support and voice-based symptom entry, especially useful in rural areas.

6. **Caution-Guided Medicine Suggestions:** Over-the-counter medication or ayurvedic/homeopathic remedy suggestions could be added, with proper warnings and disclaimers.

7. **User Data Privacy and Legal Compliance:** Implement end-to-end encryption, user authentication, and compliance with global standards like HIPAA and GDPR to ensure user trust and safety.

References

[1] World Health Organization. (2023). *Health topics*. Retrieved from https://www.who.int/health-topics

[2] Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media.

[3] Python Software Foundation. (2024). *Python Documentation*. Retrieved from https://docs.python.org/3/

[4] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. https://scikit-learn.org/

[5] Grinberg, M. (2018). *Flask Web Development*. O'Reilly Media.

[6] Kaggle. (2024). *Disease Symptom Prediction Dataset*. Retrieved from https://www.kaggle.com/datasets/itachi9604/disease-symptom-description-dataset

[7] Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106.

[8] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), 273–297.

[9] Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

[10] OpenAI. (2024). *ChatGPT by OpenAI*. Retrieved from https://chat.openai.com/

[11] Google DeepMind. (2024). *Gemini AI*. Retrieved from https://deepmind.google/technologies/gemini/

[12] Pallets Projects. (2024). *Flask Documentation*. Retrieved from https://flask.palletsprojects.com/