Vel Tech
Rangarajan Dr. Sagunthala
R&D Institute of Science and Technology
(Deemed to be University Estd. u/s 3 of UGC Act, 1956)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING
10214CS601 MINOR PROJECT -1
SUMMER SEMESTER(2024-2025)
REVIEW-II

"AI – DRIVEN HEALTH MANAGEMENT"

PRESENTED BY

C.NAGENDRA (VTU22016)(22UECM0043)
B.RAJU (VTU21815)(22UECM0022)
P.DURGA PRASAD (VTU21988)(22UECM0207)

SUPERVISED BY

Dr. R.ROSELIN KIRUBA, M.E , PhD
Assistant Professor.

OVERVIEW

- ❑ABSTRACT
- ❑OBJECTIVE
- ❑INTRODUCTION
- ❑LITERATURE REVIEW (SOFT COPY OF PAPERS TO BE LINKED AS HYPERLINK)
- ❑DESIGN AND METHODOLOGIES
- ❑IMPLEMENTATION
- ❑TESTING
- ❑INPUT AND OUTPUT
- ❑INCLUDE DEMO VIDEO-1 (Till REVEW-1)
- ❑INCLUDE DEMO VIDEO-2(Complete Implementation of Project)
- ❑CONCLUSION
- ❑WEB REFERENCES LINK (TILL REVIEW DATE ALL LINKS TO BE INCLUDED DAY WISE)
- ❑PLAGIARISM REPORT OF PPT
- ❑REFERENCES

ABSTRACT

- AI-Driven Health Management help users analyze their symptoms and receive preliminary diagnoses.
- This project focuses on building a rule-based Symptom Analyzer that offers users healthcare advice in multiple languages, making it more accessible to diverse populations.
- The goal of the project is to create a web-based application where users can input symptoms, get potential disease predictions based on a pre-built dataset, and retrieve detailed disease information (including causes, treatments, and remedies) in various languages.
- This enhances healthcare accessibility and provides basic guidance for early detection.
- The system employs a rule-based approach for matching user symptoms against a Symptom-Disease Dataset.
- The backend is built using Python frameworks , and a relational database is used to store multilingual disease information.
- The frontend interface allows users to input symptoms and switch between languages via a user-friendly design.

- The disease information is stored in multiple languages under the same disease key in the database, and users can switch between different translations seamlessly.
- The system successfully matches symptoms to diseases and retrieves accurate, detailed information about each identified disease in the user's preferred language.
- It ensures accessibility for non-English-speaking users, and the language-switching feature works seamlessly across various devices.
- The project offers basic healthcare assistance, helping users make informed decisions about their health.
- The Symptom Checker provides a functional solution for delivering healthcare information in a user-friendly and multilingual format.
- While the rule-based system effectively addresses common symptoms, future enhancements can include Natural Language Processing (NLP) and machine learning for improved accuracy and flexibility.
- This system forms a foundation for more advanced healthcare diagnostic tools.

OBJECTIVES

1. Aim of the Project

•The aim of this project is to develop a rule-based Symptom Checker that allows users to input their symptoms, receive a potential disease diagnosis, and access detailed disease-related information (such as causes, treatments, and remedies) in multiple languages.

• The project focuses on providing a simple yet accessible healthcare tool for users to gain insight into their health conditions, particularly enhancing usability through multilingual support for diverse audiences.

2. Scope of the Project

The scope of this project includes:

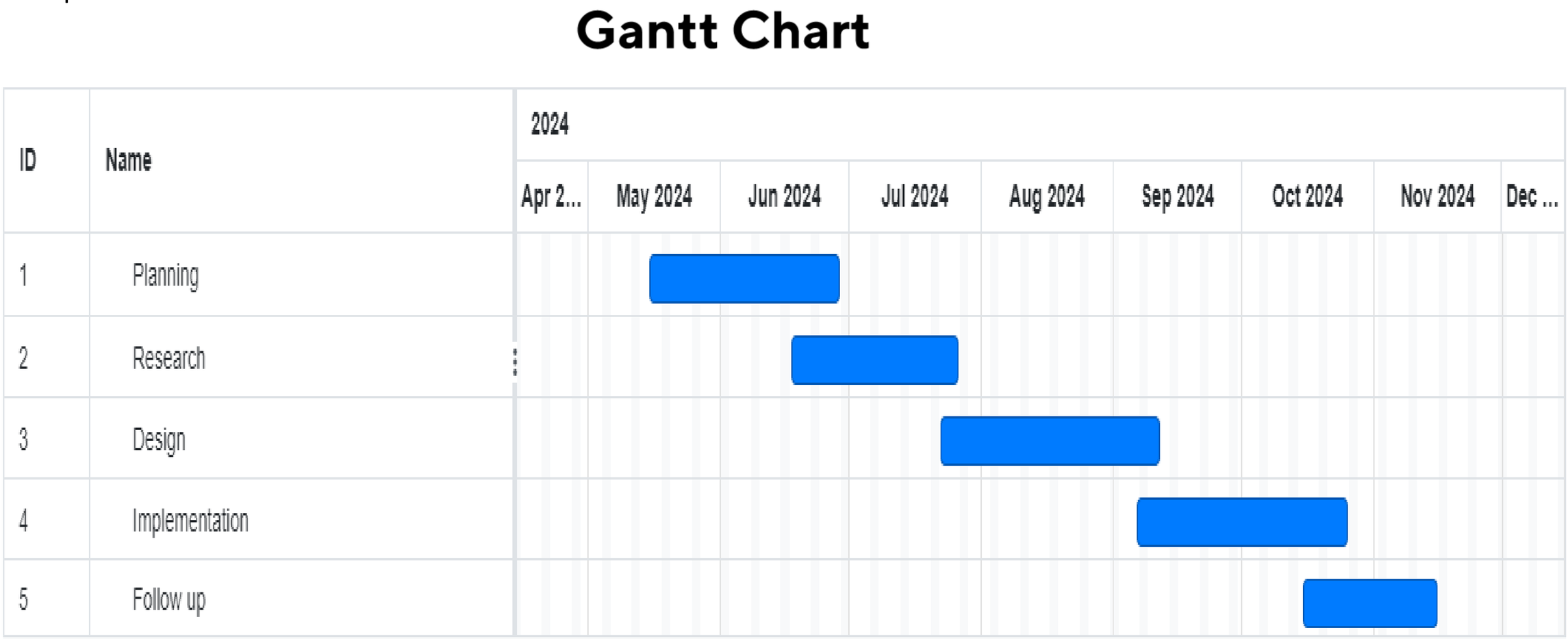
•Building a web-based application with a user-friendly interface for symptom input.

•Implementing a rule-based logic system to match user symptoms to diseases based on a predefined symptom-disease dataset.

- Storing detailed disease information (cause, treatment, home remedies) in a multilingual database, with each disease's information available in multiple languages.
- Providing language-switching functionality, allowing users to easily view disease information in their preferred language.

TIMELINE OF THE PROJECT

Sample Gantt Chart



Include timeline from October 2023 to January 2024.

INTRODUCTION

- In today's digital age, accessibility to reliable and user-friendly healthcare information is more important than ever.
- A Symptom Checker offers a solution by providing users with an easy way to input their symptoms and receive potential disease diagnoses.
- The aim of this project is to create a rule-based web application that not only predicts diseases based on symptoms but also provides detailed disease information in multiple languages, enhancing its usability for non-English speakers.
- With the rise in the use of technology in healthcare, such tools can play a crucial role in early disease detection and promoting health awareness.
- This project addresses the challenge of healthcare accessibility by offering multilingual support, allowing users from diverse linguistic backgrounds to access healthcare information.
- The system will store detailed disease information such as causes, treatments, and home remedies, which users can view in their preferred language at the click of a button.

LITERATURE REVIEW

Sl.No	Author's Name	Paper name and publication details	Year of publication	Main content of the paper
1	S Aboueid	The **Use** of Artificially Intelligent **Symptom checkers by** University Students–An Exploratory Sequential Mixed Methods Study	2021	Rising healthcare costs, wait times, unnecessary care, and lack of access to a primary care provider, are pressing issues encountered by various health systems and in part, are a result of misinformed patient demand. The literature suggests that one way to economize the healthcare system is to provide patients with reliable tools to inform better decisions on when to seek care. The Internet is often used as a source of health information. University students, a population group considered to be technology savvy, often browse the Internet for health topics and self-diagnosis

Sl.No	Author's Name	Paper name and publication details	Year of publication	Main content of the paper
2	CH Tsai, Y You, X Gui, Y Kou, JM Carroll	Exploring and promoting diagnostic transparency and explainability in online **symptom checkers**	2021	Online symptom checkers (OSC) are widely used intelligent systems in health contexts such as primary care, remote healthcare, and epidemic control. OSCs use algorithms such as machine learning to facilitate self-diagnosis and triage based on symptoms input by healthcare consumers.
3	C Woodcock, B Mittelstadt, D Busbridge	The impact of explanations on layperson trust in **artificial intelligence**–driven **symptom checker apps**: experimental study	2022	The aim of this study is to ascertain whether explanations provided by a symptom checker affect explanatory trust among laypeople and whether this trust is impacted by their existing knowledge of disease.

Sl.No	Author's Name	Paper name and publication details	Year of publication	Main content of the paper
4	You, Yue	Self-**diagnosis** through chatbot-based **symptom checkers**: user experiences and design considerations	2020	Developing Artificial Intelligence (AI)-based chatbots in the healthcare market, which use human-like conversations to interact with users
5	Y You, R Ma, X Gui	User experience of **symptom checkers**: a systematic review	2022	aiming to characterize users studied in the existing literature, identify the aspects of user experience of symptom checkers that have been studied, and offer design suggestions.

Sl.No	Author's Name	Paper name and publication details	Year of publication	Main content of the paper
6	Nikita Vijay Shinde; Aniket Akhade; Pranali Bagad; Harshit Bhavsar; S.K. Wagh; Amol Kamble	Healthcare chatbot system **using artificial intelligence**	2021	This paper presents Healthcare Chatbot using Artificial Intelligence that can make a human-system interaction to resolve basic queries regarding health parameters before consulting a doctor. The actual purpose behind this work is to work on the user's symptoms and to provide medical suggestions according to it, to reduce the time and cost required for the process.
7	Berkay Murat; Arda Oğulcan Üzer; Sarp Ketenci; Sinan Yaşbek; İlker Korkmaz	A **Symptom** Evaluation System on Medical **Diagnosis**	2023	The Symptom Evaluation System (SES) to be used on medical diagnosis is a software application that offers time-saving solutions to the problems regarding the communication and the relationship between patients and doctors.

Sl.No	Author's Name	Paper name and publication details	Year of publication	Main content of the paper
8	Lekha Athota; Vinod Kumar Shukla; Nitin Pandey; Ajay Rana	Chatbot for healthcare system **using artificial intelligence**	2020	Healthcare is very important to lead a good life. However, it is very difficult to obtain the consultation with the doctor for every health problem. The idea is to create a medical chatbot using Artificial Intelligence that can diagnose the disease and provide basic details about the disease before consulting a doctor.

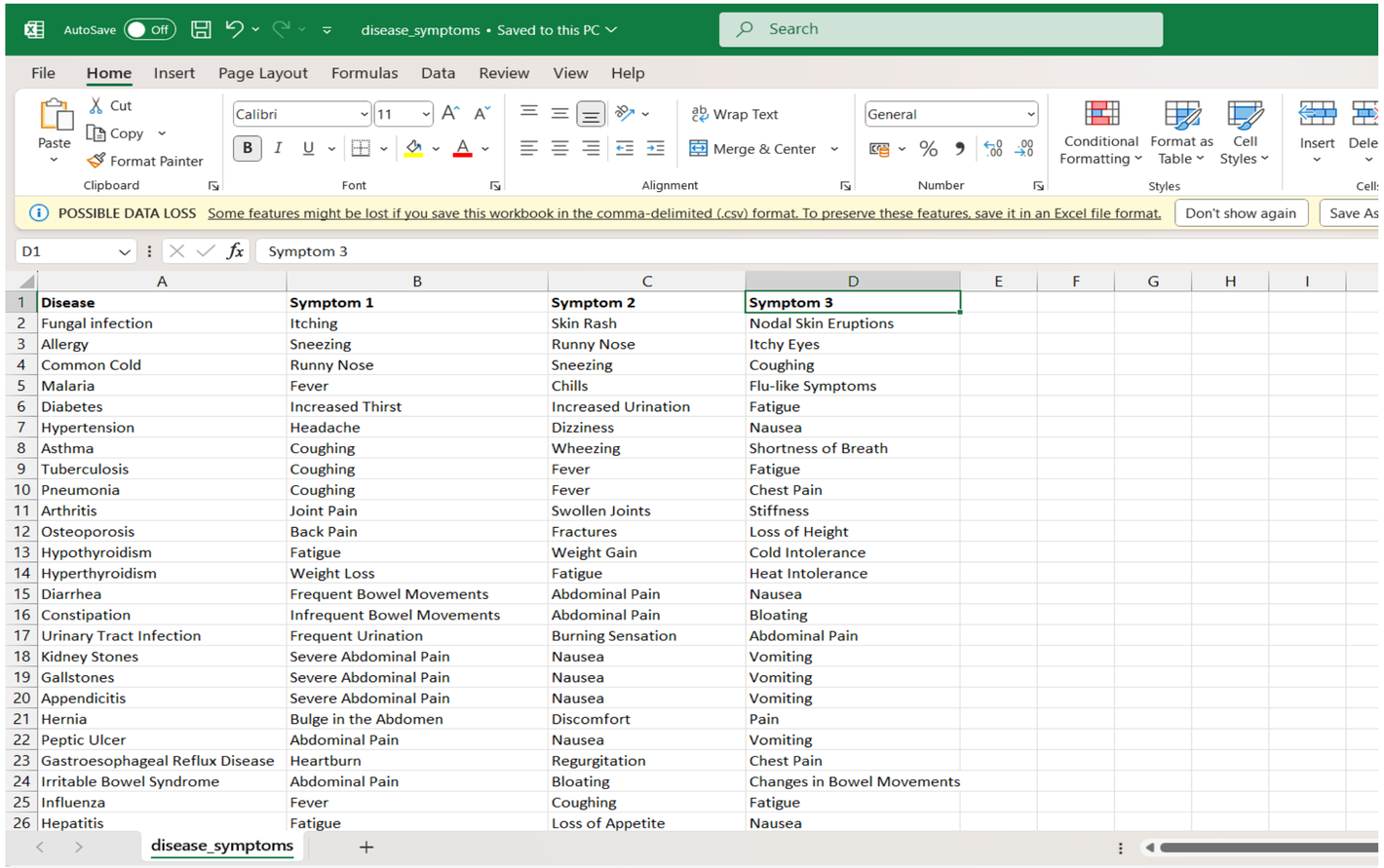
DESIGN AND METHODOLOGIES

- MODULE 1: **Data Collection**
- MODULE 2: **Algorithms**
  - Rule-Based Matching Algorithm
  - Pattern Matching (String Matching)
  - Natural Language Processing (NLP) Techniques
  - Database Querying Algorithm

MODULE 1: Data Collection

Step:1 Collection of data

Symptom – Disease data set which is having number of diseases and their respective symptoms



	A	B	C	D
1	**Disease**	**Symptom 1**	**Symptom 2**	**Symptom 3**
2	Fungal infection	Itching	Skin Rash	Nodal Skin Eruptions
3	Allergy	Sneezing	Runny Nose	Itchy Eyes
4	Common Cold	Runny Nose	Sneezing	Coughing
5	Malaria	Fever	Chills	Flu-like Symptoms
6	Diabetes	Increased Thirst	Increased Urination	Fatigue
7	Hypertension	Headache	Dizziness	Nausea
8	Asthma	Coughing	Wheezing	Shortness of Breath
9	Tuberculosis	Coughing	Fever	Fatigue
10	Pneumonia	Coughing	Fever	Chest Pain
11	Arthritis	Joint Pain	Swollen Joints	Stiffness
12	Osteoporosis	Back Pain	Fractures	Loss of Height
13	Hypothyroidism	Fatigue	Weight Gain	Cold Intolerance
14	Hyperthyroidism	Weight Loss	Fatigue	Heat Intolerance
15	Diarrhea	Frequent Bowel Movements	Abdominal Pain	Nausea
16	Constipation	Infrequent Bowel Movements	Abdominal Pain	Bloating
17	Urinary Tract Infection	Frequent Urination	Burning Sensation	Abdominal Pain
18	Kidney Stones	Severe Abdominal Pain	Nausea	Vomiting
19	Gallstones	Severe Abdominal Pain	Nausea	Vomiting
20	Appendicitis	Severe Abdominal Pain	Nausea	Vomiting
21	Hernia	Bulge in the Abdomen	Discomfort	Pain
22	Peptic Ulcer	Abdominal Pain	Nausea	Vomiting
23	Gastroesophageal Reflux Disease	Heartburn	Regurgitation	Chest Pain
24	Irritable Bowel Syndrome	Abdominal Pain	Bloating	Changes in Bowel Movements
25	Influenza	Fever	Coughing	Fatigue
26	Hepatitis	Fatigue	Loss of Appetite	Nausea

Module 2- Mention the algorithm

1. Rule-Based Matching Algorithm

- **Description**: This is the core algorithm used to match user-inputted symptoms to a set of predefined symptoms for each disease in the dataset. It works by checking for exact or close matches between the symptoms provided by the user and the stored dataset.

2. Pattern Matching (String Matching)

- **Description**: This technique is used to process user input and identify symptoms even if they are phrased differently or contain slight variations.

3. Natural Language Processing (NLP) Techniques (Optional Future Enhancement)

- **Description**: If the system is upgraded, **NLP** can be used to process and interpret free-text symptom inputs, rather than requiring structured inputs.

4. Database Querying Algorithms

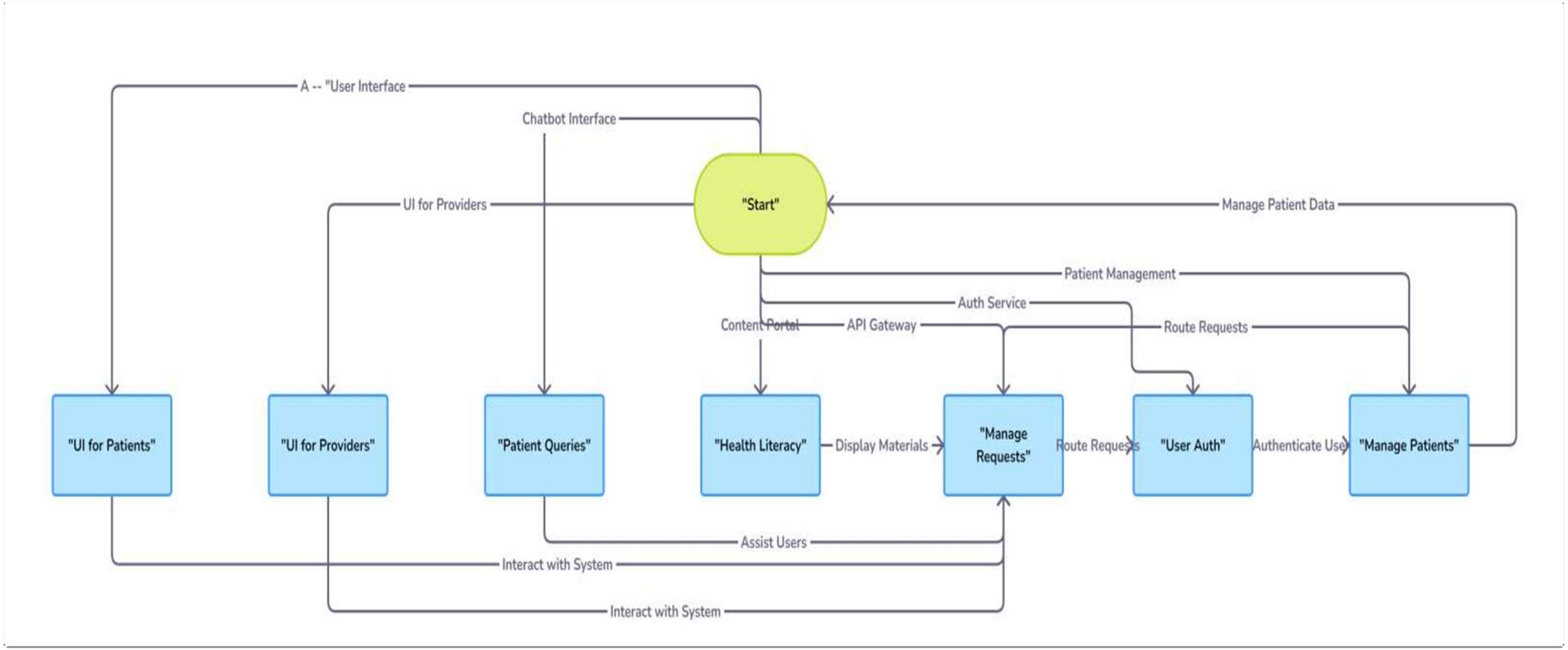
- **Description**: This algorithm retrieves disease information (e.g., causes, treatments, and remedies) from a **multilingual database** after the symptom match is made.

IMPLEMENTATION

- Architecture Diagram
- Data –Flow Diagram
- Use Case Diagram
- Class Diagram
- Activity Diagram
- Sequence Diagram
- Collaboration Diagram(If applicable)
- E-R Diagram

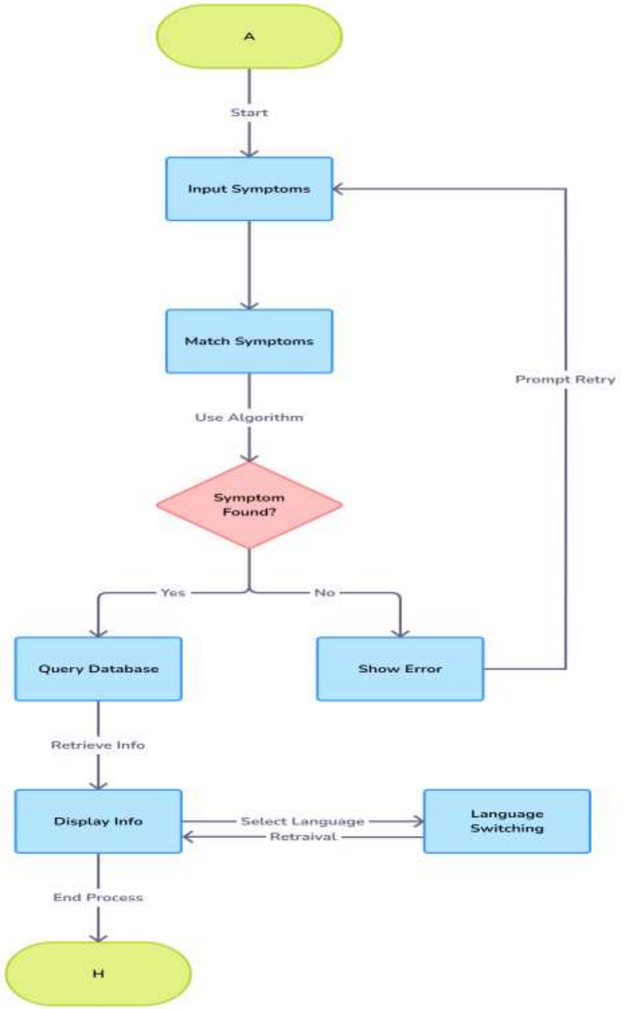
It’s a sample only and may vary according to the project

Architecture Diagram



- **User Interface:** Allows users to input symptoms and select a preferred language.
- **Backend:** Processes the input, matches symptoms to diseases using the rule-based algorithm, and queries the database.
- **Symptom-Disease Dataset:** Contains the predefined symptom-disease mappings for rule-based matching.
- **Disease Information Database:** Stores detailed disease information in multiple languages.
- **Process Flow:** Starts with symptom input, followed by language selection, symptom matching, and the retrieval of disease information from the database, which is displayed in the chosen language.

Data –Flow Diagram



•**User Interface**: Allows users to interact the system ,they can input their symptoms and select a preferred language that the information should be displayed.

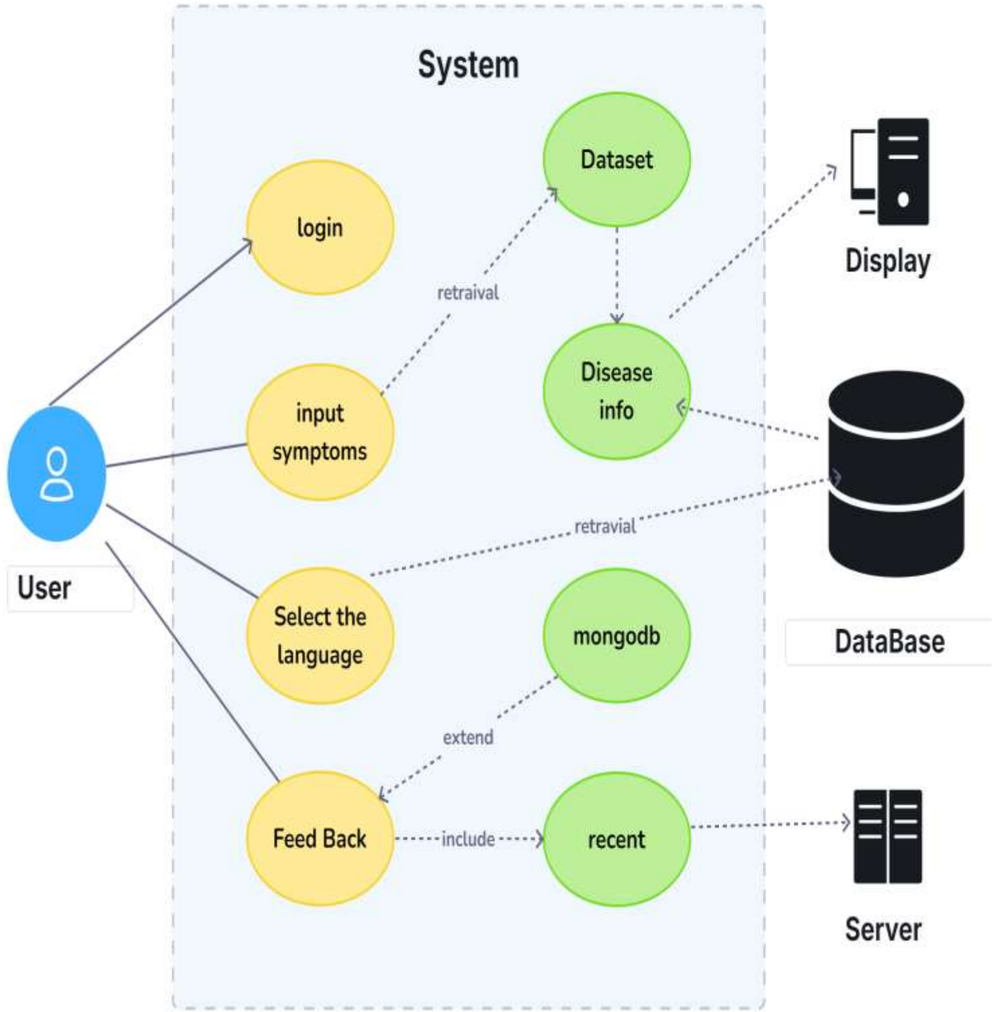
•**Backend**: Processes the input, matches symptoms to diseases using the rule-based algorithm, and queries the database for information.

•**Symptom-Disease Dataset**: Contains the predefined symptom-disease mappings for rule-based matching.

•**Disease Information Database**: Stores detailed disease information in multiple languages.

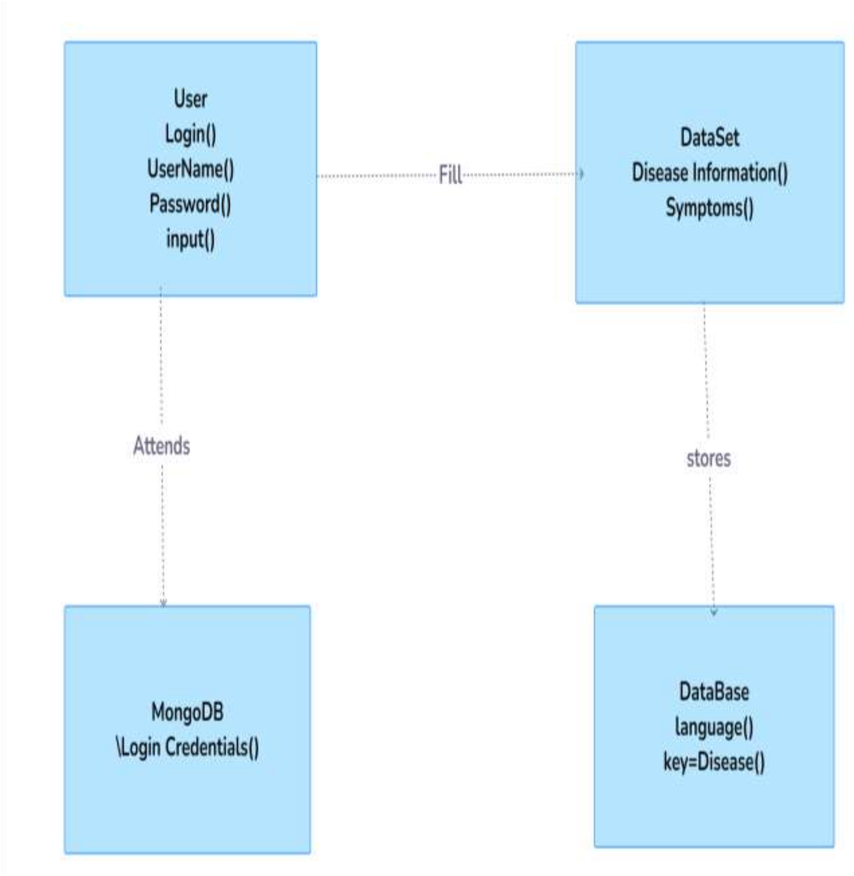
•**Process Flow**: Starts with symptom input, followed by language selection, symptom matching, and the retrieval of disease information from the database, which is displayed in the chosen language.

Use Case Diagram



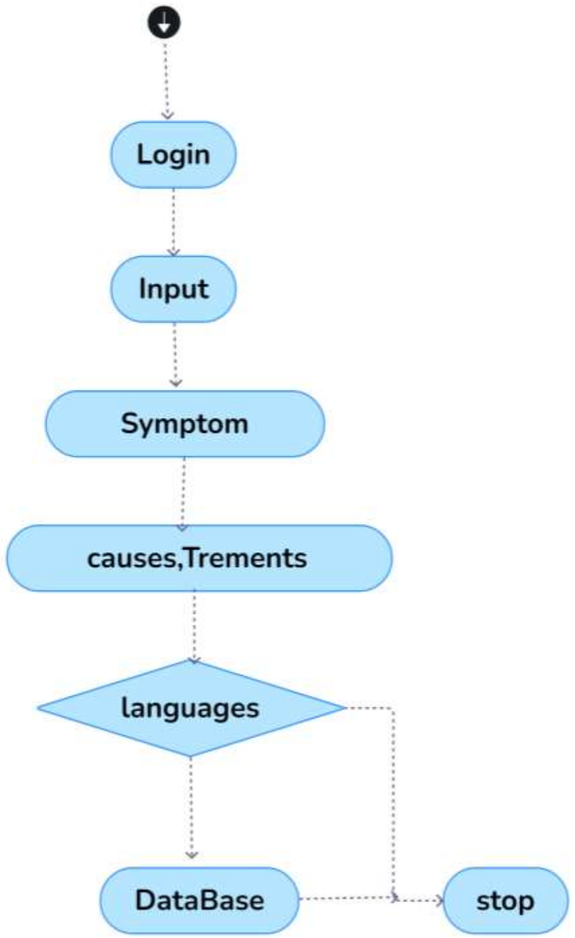
- Inputs symptoms into the system.
- Selects a preferred language for viewing disease information.
- Receives a potential diagnosis from the symptom matching engine.
- Views detailed disease information, including causes and treatments.
- Switches between languages to view the disease information in their preferred language

Class Diagram



- **User**: Represents the user of the system, who inputs symptoms and selects a language.
- **Symptom Checker(Data Set)**: The main engine that interacts with the user input and performs operations like symptom matching.
- **Symptom Matcher**: A class responsible for comparing user input with the symptom-disease dataset.
- **Disease Database**: Stores disease information in multiple languages.
- **Disease**: Contains detailed disease data, including causes, treatments, and remedies.
- **Language (database)**: Handles multilingual support, allowing disease information to be presented in different languages.
- **Mongo DB:**To verify the login (user name and password)

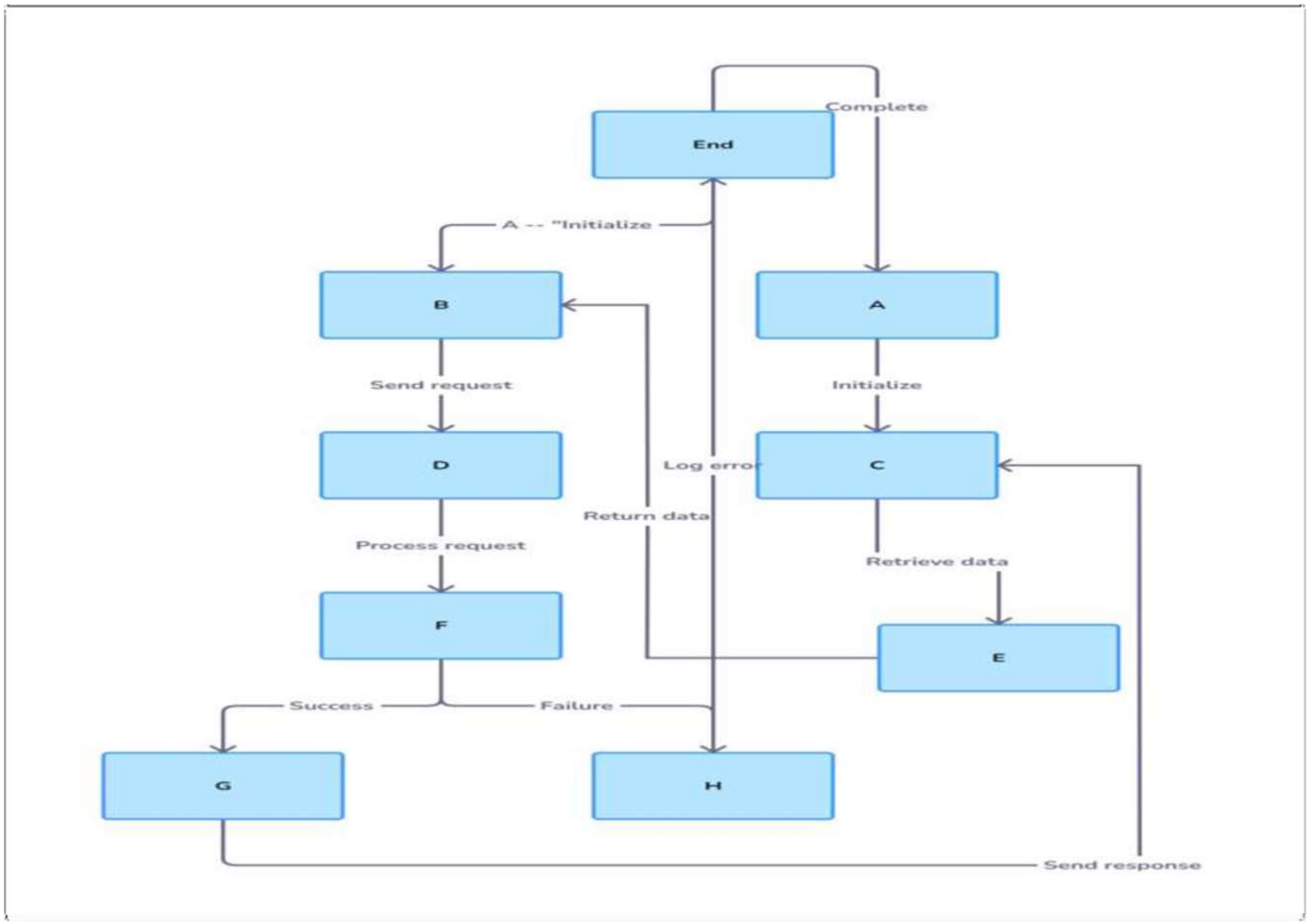
Activity Diagram



- **User Inputs Symptoms**: The user starts by entering their symptoms into the system. This triggers the backend process to handle the input.
- **Symptom Processing**: The system receives the symptoms and begins processing them. This may involve basic pattern matching or rule-based checks to match symptoms with potential diseases.
- **Symptom-Disease Matching**: The system compares the user's symptoms with a predefined dataset of symptoms and their associated diseases. If a match is found, the system retrieves the relevant disease(s).
- **Language Selection**: If multilingual support is enabled, the user selects their preferred language. The system checks the database for the corresponding disease information in the chosen language.
- **Retrieve Disease Information**: Once the matching disease is found, the system retrieves detailed information from the database in the selected language.

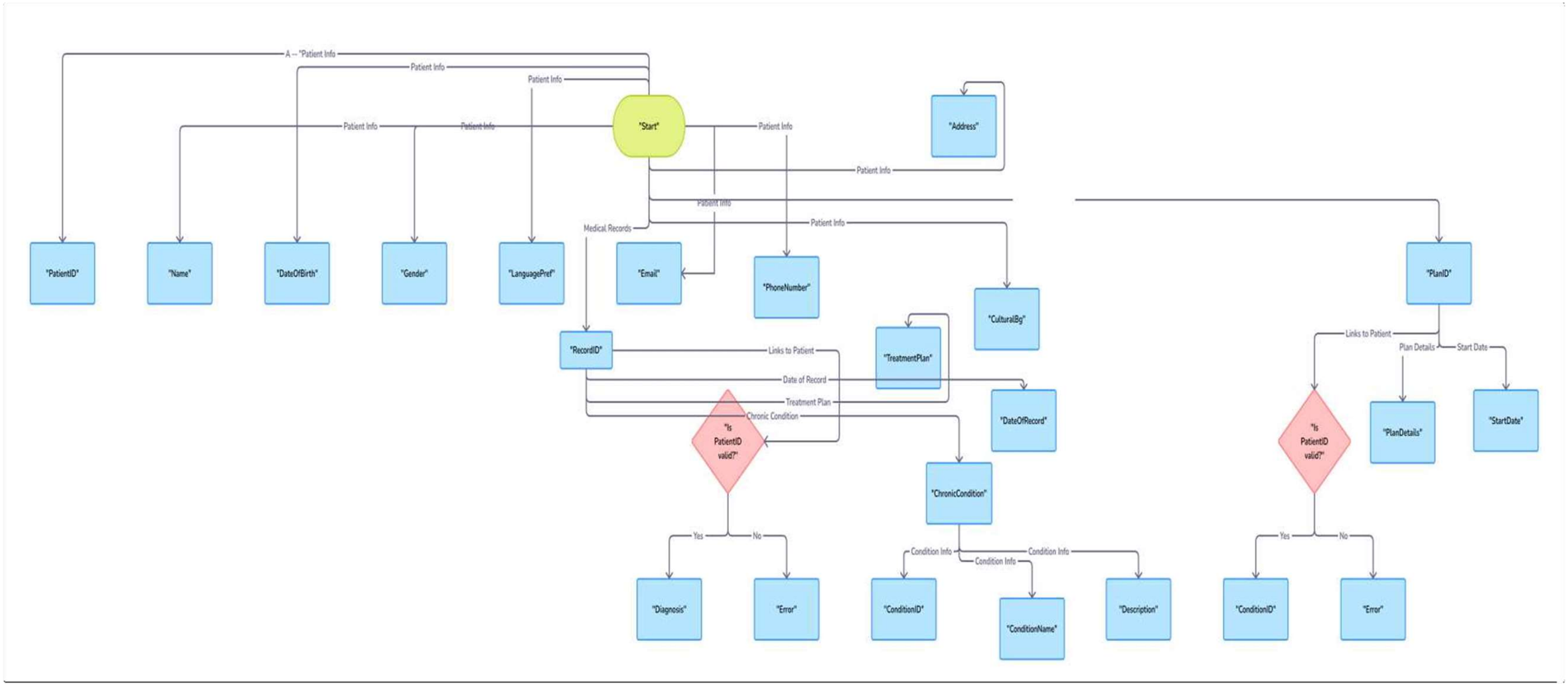
Sequence Diagram

,



- **User Input**: The sequence starts when the user inputs their symptoms into the system via the user interface. This action initiates a request for symptom analysis.
- **Symptom Checker (Front-end)**: The user input is sent to the Symptom Checker engine. This component handles the request and forwards the symptoms to the backend for processing.
- **Symptom Matching (Backend)**: The backend matches the input symptoms to diseases by querying the Symptom-Disease Dataset. It uses a rule-based algorithm or pattern matching to find potential matches based on the symptoms.
- **Disease Database Query**: Once a disease match is found, the backend retrieves the corresponding disease information from the Disease Information Database. The database stores disease-related information such as causes, treatments, and home remedies, with multilingual support.
- **Language Selection**: If the user has selected a preferred language, the system retrieves the information in the selected language from the database. If no language is selected, the default language is used.
- **Optional Language Switch**: The user can optionally switch the language at this point, triggering a new request to retrieve the same disease information in another language. The system queries the database again for the appropriate language version and displays it to the user.

E-R Diagram


A -- "Patient Info
Patient Info
Patient Info
Patient Info
Patient Info
"Start"
Patient Info
"Address"
Patient Info
Patient Info
Patient Info
Medical Records
Patient Info
"PatientID"
"Name"
"DateOfBirth"
"Gender"
"LanguagePref"
"Email"
"PhoneNumber"
"CulturalBg"
"PlanID"
"RecordID"
Links to Patient
"TreatmentPlan"
Links to Patient
Plan Details
Start Date
Date of Record
Treatment Plan
Chronic Condition
"DateOfRecord"
"Is PatientID valid?"
"PlanDetails"
"StartDate"
"ChronicCondition"
"Is PatientID valid?"
Yes
No
Condition Info
Condition Info
Condition Info
Yes
No
"Diagnosis"
"Error"
"ConditionID"
"ConditionName"
"Description"
"ConditionID"
"Error"

Entities:

- **User**: Represents the user who interacts with the system by entering symptoms and selecting a preferred language.
- **Symptom**: Captures the symptoms entered by the user. Each symptom is related to one or more diseases.
- **Disease**: Stores information about diseases, including causes, symptoms, treatments, and home remedies.
- **Language**: Contains the various languages available for viewing disease information.
- **Disease Information**: Includes the detailed descriptions of causes, symptoms, treatments, and remedies, available in multiple languages.
- **Symptom-Disease Mapping**: An associative entity that maps symptoms to potential diseases based on predefined rules.

Relationships:

- **User Input**: A user inputs symptoms, which are matched to diseases using the rule-based engine.
- **Has Symptom**: Each disease is linked to one or more symptoms.

- **Has Disease Info**: Each disease has corresponding disease information (cause, treatment, home remedies) stored in multiple languages.
- **Select Language**: Users can select a language, and the system retrieves the disease information in the chosen language.

Cardinality:

- A **User** can input multiple symptoms.
- A **Symptom** can be related to multiple **Diseases**.
- A **Disease** has corresponding information in one or more **Languages**.
- The **Symptom-Disease Mapping** connects symptoms to diseases in a many-to-many relationship.

TESTING

- UNIT TESTING
- INTEGRATION TESTING
- WHITE BOX TESTING
- BLACK BOX TESTING

UNIT TESTING

```
import unittest

class TestSymptomInput(unittest.TestCase):
    def test_valid_symptom_input(self):
        symptoms = ["fever", "cough"]
        self.assertTrue(all(s in symptom_dataset for s in symptoms))  # Assume s

    def test_invalid_symptom_input(self):
        symptoms = ["", "asdf"]
        self.assertFalse(all(s in symptom_dataset for s in symptoms))

if __name__ == '__main__':
    unittest.main()
```

```
class TestSymptomMatching(unittest.TestCase):
    def test_symptom_disease_matching(self):
        symptoms = ["fever", "cough"]
        matched_disease = symptom_checker(symptoms)  # Assume symptom_checker is
        self.assertIn("Flu", matched_disease)

    def test_no_match_found(self):
        symptoms = ["unknown symptom"]
        matched_disease = symptom_checker(symptoms)
        self.assertEqual(matched_disease, [])

if __name__ == '__main__':
    unittest.main()
```

INTEGRATION TESTING

```
import unittest

class TestSymptomMatchingIntegration(unittest.TestCase):
    def test_symptom_input_and_matching(self):
        symptoms = ["fever", "headache"]
        result = symptom_checker(symptoms)  # Integrating symptom input with sym
        self.assertIn("Flu", result)  # Assumes 'Flu' is in the matching disease

if __name__ == '__main__':
    unittest.main()
```

```
class TestSymptomMatcherAndDatabaseIntegration(unittest.TestCase):
    def test_symptom_to_database_retrieval(self):
        symptoms = ["fever", "sore throat"]
        matched_disease = symptom_checker(symptoms)
        disease_info = query_disease_db(matched_disease[0])  # Matching disease
        self.assertIsNotNone(disease_info)  # Check if info is retrieved from D

if __name__ == '__main__':
    unittest.main()
```

SYSTEM TESTING

Functionality Testing :

Objective: Ensure that all features of the system work as intended

Usability Testing :

Objective: Ensure the system is user-friendly and easy to navigate

Performance Testing :

Objective: Ensure the system performs efficiently under normal and heavy loads

Security Testing :

Objective: Ensure that the system is secure and protects user data.

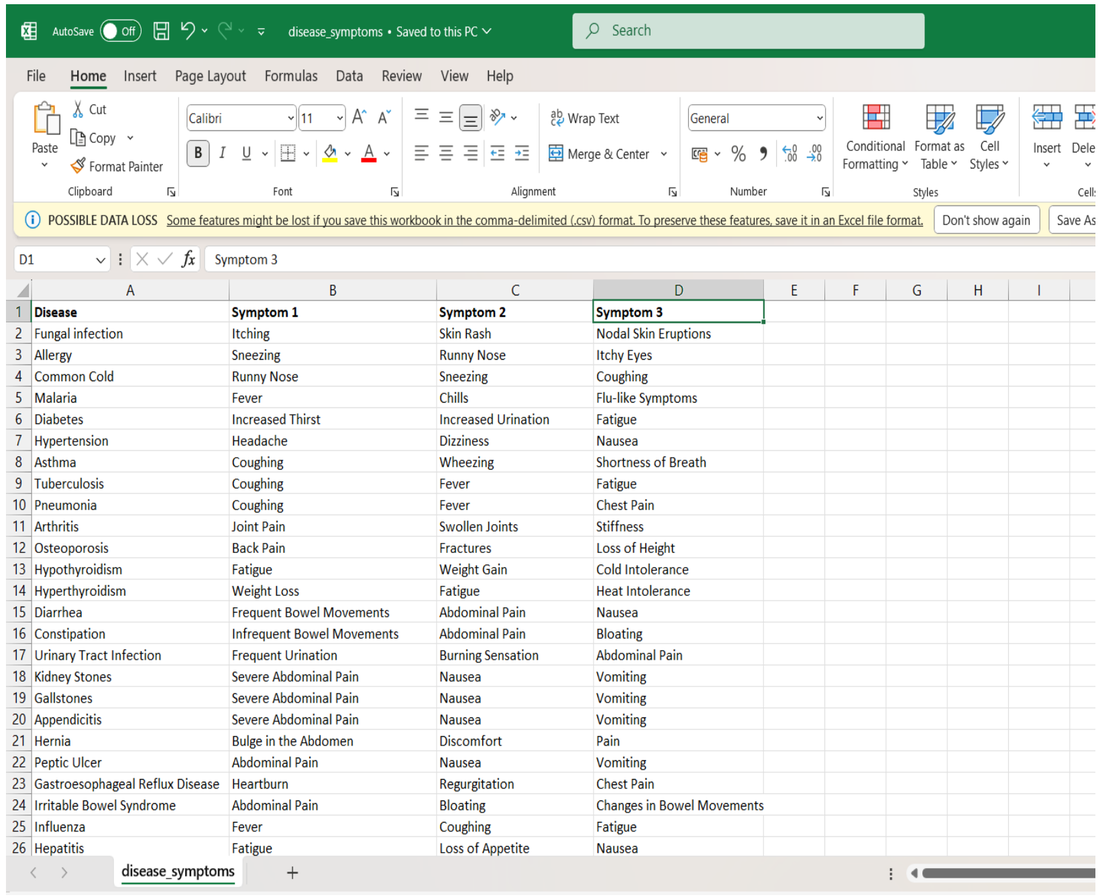
Compatibility Testing :

Objective: Ensure the system works across different devices, browsers, and operating systems

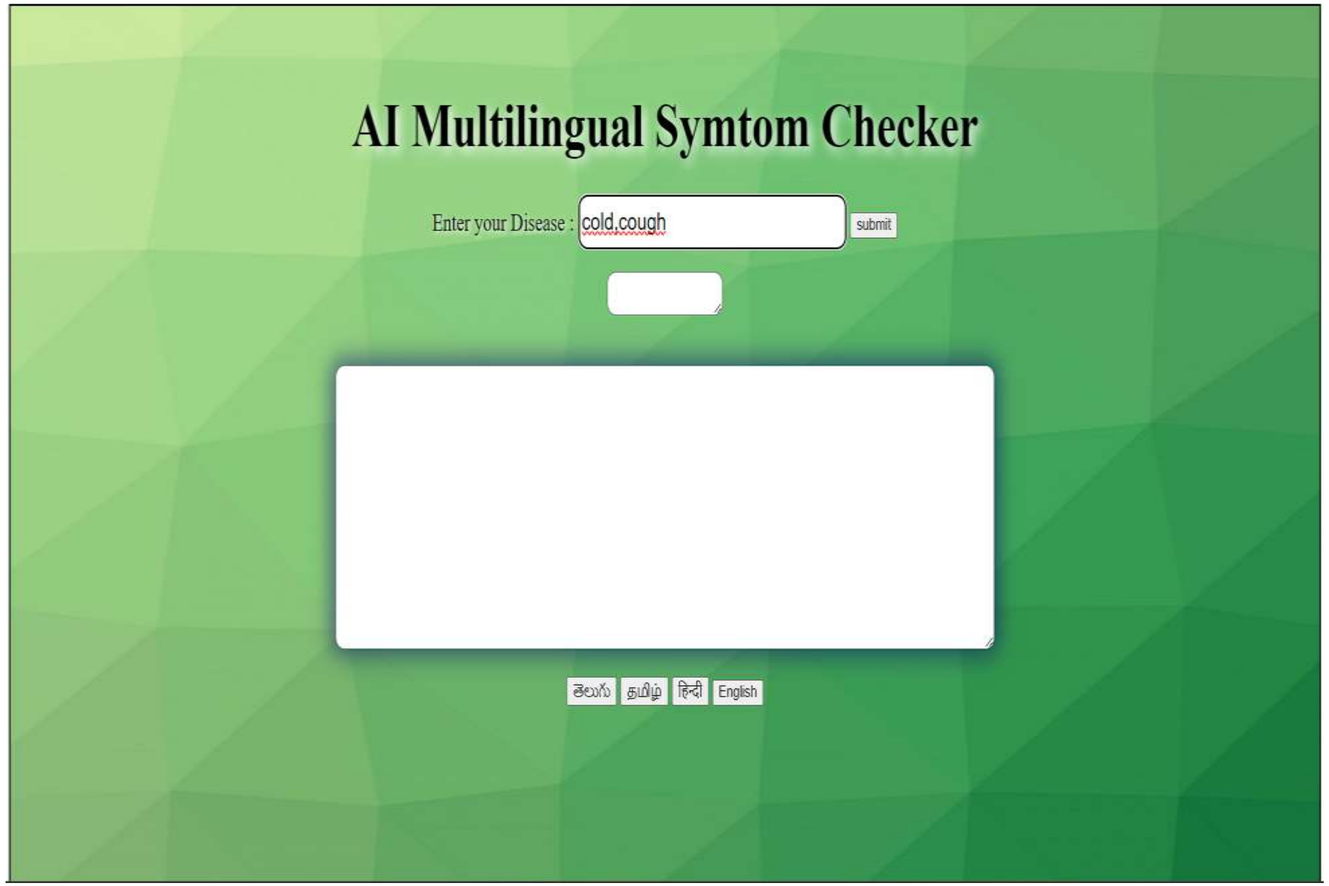
Multilingual Testing :

Objective: Ensure the system provides accurate disease information in different languages

INPUT AND OUTPUT



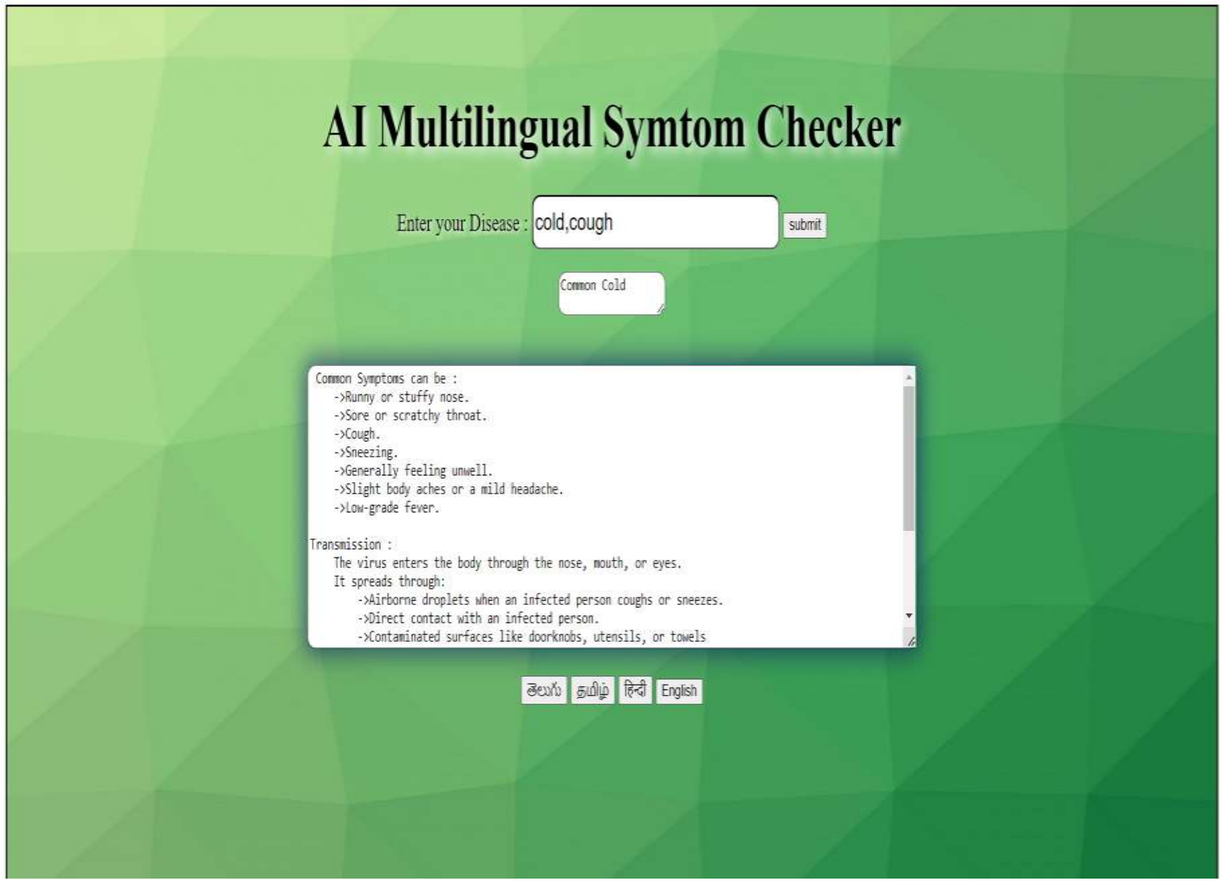
Disease	Symptom 1	Symptom 2	Symptom 3
Fungal infection	Itching	Skin Rash	Nodal Skin Eruptions
Allergy	Sneezing	Runny Nose	Itchy Eyes
Common Cold	Runny Nose	Sneezing	Coughing
Malaria	Fever	Chills	Flu-like Symptoms
Diabetes	Increased Thirst	Increased Urination	Fatigue
Hypertension	Headache	Dizziness	Nausea
Asthma	Coughing	Wheezing	Shortness of Breath
Tuberculosis	Coughing	Fever	Fatigue
Pneumonia	Coughing	Fever	Chest Pain
Arthritis	Joint Pain	Swollen Joints	Stiffness
Osteoporosis	Back Pain	Fractures	Loss of Height
Hypothyroidism	Fatigue	Weight Gain	Cold Intolerance
Hyperthyroidism	Weight Loss	Fatigue	Heat Intolerance
Diarrhea	Frequent Bowel Movements	Abdominal Pain	Nausea
Constipation	Infrequent Bowel Movements	Abdominal Pain	Bloating
Urinary Tract Infection	Frequent Urination	Burning Sensation	Abdominal Pain
Kidney Stones	Severe Abdominal Pain	Nausea	Vomiting
Gallstones	Severe Abdominal Pain	Nausea	Vomiting
Appendicitis	Severe Abdominal Pain	Nausea	Vomiting
Hernia	Bulge in the Abdomen	Discomfort	Pain
Peptic Ulcer	Abdominal Pain	Nausea	Vomiting
Gastroesophageal Reflux Disease	Heartburn	Regurgitation	Chest Pain
Irritable Bowel Syndrome	Abdominal Pain	Bloating	Changes in Bowel Movements
Influenza	Fever	Coughing	Fatigue
Hepatitis	Fatigue	Loss of Appetite	Nausea

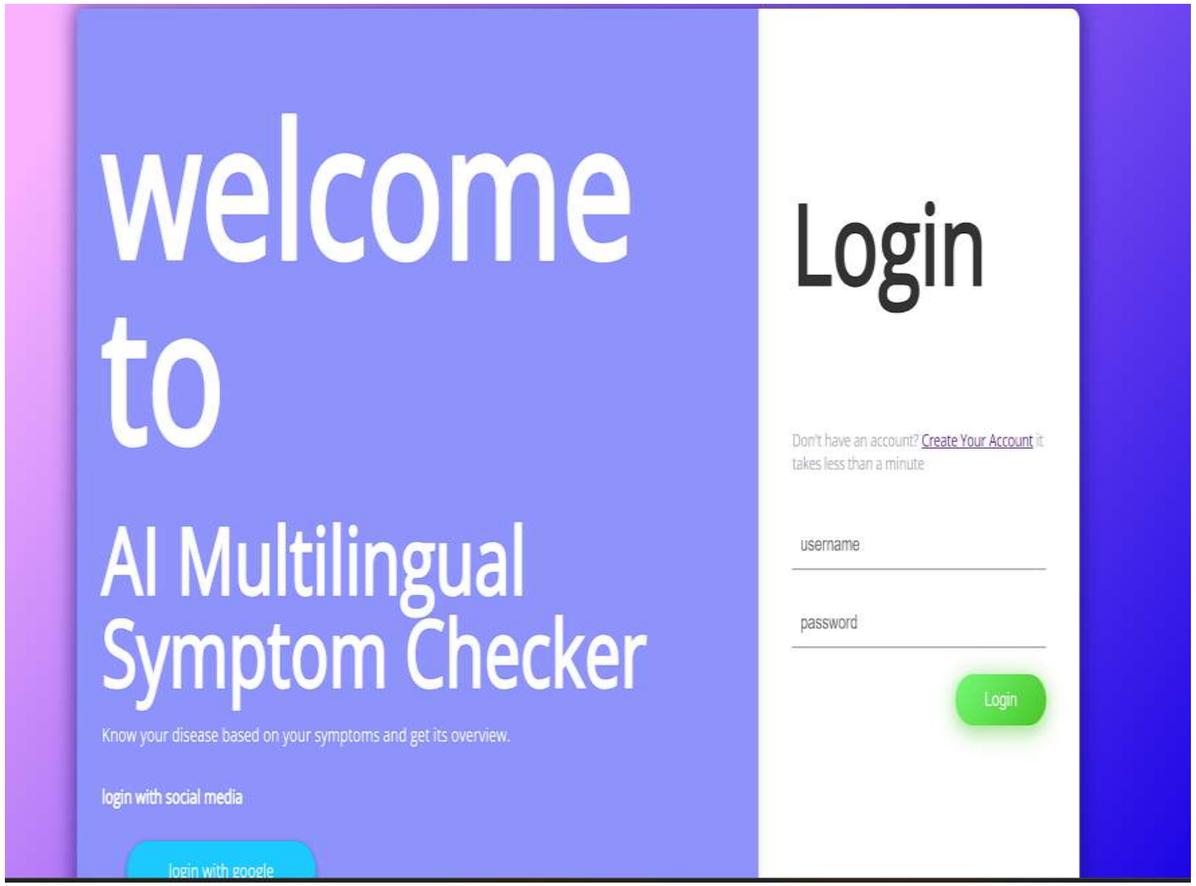


SOURCE CODE

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"  content="width=device-width, initial-scale=1.0">
    <meta lang="UTF-8">

    <title>Document</title>
    <style>
        .container{
            /* text-align: center;
            display: flex; */
            justify-content: center;
            align-items: center;
             justify-content: center;
            align-items: center;
            text-align: center;
        background-image:url('https://toppng.com/uploads/preview/polygon-gradient-convex-triangles-polygons-green-11570313404ux75cbssfx.jpg');
            background-size:100% 100%;
        border:2px solid black;
            display: flex;
            flex-direction: column;
            position: relative;
            height: 100vh;
        }
        .content{
            position: absolute;
            top:50%;
            left:50%;
            transform: translate(-50%,-50%);
            padding:50px;
        }
        .container h1{
            position: absolute;
            top:5%;
            color:darkblack;
        text-shadow: 4px 4px 10px white;
        font-size: 50px;
            text-transform: capitalize;
            text-align: center;
            justify-content: center;
            align-items: center;
        }
        .content input{
            width: 300px;
            height: 40px;
            font-size: 20px;
        }
        .content label{
        color: darkblack;
        font-size: 20px;
         text-shadow: 2px 2px 4px white;
        }
        #telugu{
             box-shadow: 0 0 20px 6px #090b6f85;
            height: 40px;
            font-size: 20px;
        }
        .content label{
        color: darkblack;
        font-size: 20px;
         text-shadow: 2px 2px 4px white;
        }
        #telugu{
             box-shadow: 0 0 20px 6px #090b6f85;
             border-radius: 10px;
        }

    }
    </style>

</head>
<body>
    <div class="container">
        <h1>AI Multilingual Symtom Checker</h1>
        <div class="content">

            <label>Enter your Disease : </label>
            <input type="text" placeholder="Enter Disease Symptoms" style="border-radius: 10px">
        <button onclick="showText()">submit</button><br><br>
         <textarea cols="15" rows="2" style="border-radius: 10px"></textarea><br><br><br>
            <textarea cols="100" rows="15" id="telugu">
</textarea><br><br>
            <div class="buttons">
                <button onclick="telugu()">తెలుగు</button>
                <button>தமிழ்</button>
                <button>हिन्दी</button>
                <button>English</button>
            </div>
        </div>
    </div>

</body>
</html>
```

OUTPUT


AI Multilingual Symtom Checker
Enter your Disease : cold,cough
submit
Common Cold
Common Symptoms can be :
->Runny or stuffy nose.
->Sore or scratchy throat.
->Cough.
->Sneezing.
->Generally feeling unwell.
->Slight body aches or a mild headache.
->Low-grade fever.
Transmission :
The virus enters the body through the nose, mouth, or eyes.
It spreads through:
->Airborne droplets when an infected person coughs or sneezes.
->Direct contact with an infected person.
->Contaminated surfaces like doorknobs, utensils, or towels
తెలుగు
தமிழ்
हिंदी
English


welcome to
AI Multilingual Symptom Checker
Know your disease based on your symptoms and get its overview.
login with social media
login with google
Login
Don't have an account? Create Your Account it takes less than a minute
username
password
Login

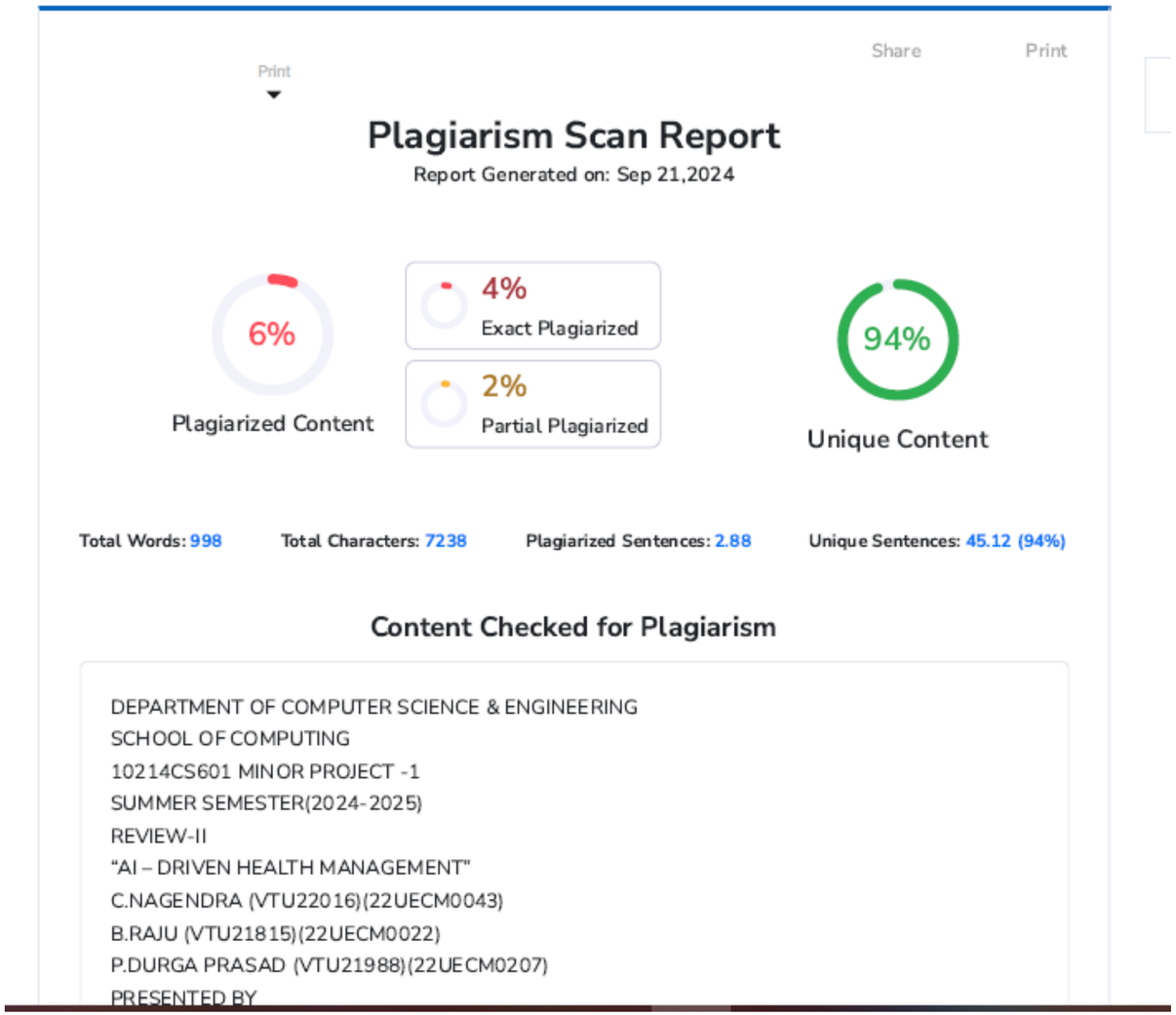
CONCLUSION

The Symptom Checker project successfully implemented a basic rule-based system capable of analysing user-inputted symptoms and providing potential diagnoses. By integrating a symptom-disease dataset and a multilingual disease information database, the system proved effective in offering users healthcare information in their preferred language, enhancing accessibility and user experience.

The project demonstrated that rule-based systems can effectively manage common symptoms and provide accurate disease predictions. However, the system's limitations, such as handling ambiguous inputs or rare diseases, highlighted the potential for further improvements. Future enhancements, including the incorporation of Natural Language Processing (NLP) and machine learning models, could significantly improve the accuracy and flexibility of the system. Additionally, expanding the dataset and improving translation accuracy will make the tool more comprehensive and useful for a wider audience.

In conclusion, the project achieved its goal of building a functional, multilingual symptom checker while laying the groundwork for future enhancements that could transform it into a more sophisticated healthcare solution.

Plagiarism Report of PPT



Share Print

Print

Plagiarism Scan Report

Report Generated on: Sep 21,2024

6%
Plagiarized Content

4%
Exact Plagiarized

2%
Partial Plagiarized

94%
Unique Content

Total Words: 998 Total Characters: 7238 Plagiarized Sentences: 2.88 Unique Sentences: 45.12 (94%)

Content Checked for Plagiarism

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING
10214CS601 MINOR PROJECT -1
SUMMER SEMESTER(2024-2025)
REVIEW-II
"AI – DRIVEN HEALTH MANAGEMENT"
C.NAGENDRA (VTU22016)(22UECM0043)
B.RAJU (VTU21815)(22UECM0022)
P.DURGA PRASAD (VTU21988)(22UECM0207)
PRESENTED BY

Web references/video links

1. https://youtube.com/playlist?list=PLfFghEzKVmjuhQwKhYXvdU94GSU-6Jcjr&si=0aPFas9u3dshjous
2. https://youtu.be/GtyBiNMZtQo?si=FRp8GHjJvK8B-6zn
3. https://youtu.be/Lq6ik1WHRtQ?si=M8iVIOQ2SpyucbR7

REFERENCES

[1] A Smart Research Framework for Local Undergraduate Colleges' Literacy Survey Based on Distributed Data Collection System Zili Niu 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS)

[2] Exploring the Role of the Internet, Quality and Communication in Shaping Mental Health: Analysis of the Information National Trends Survey Avishek Choudhury;Onur Asan;Turki Alelyani IEEE Journal of Biomedical and Informatics

[3] Measuring Digital Literacy in Hungary: A Scoping Review Mariann Békésy 2022 IEEE 20th Jubilee International Symposium on Intelligent Systems and Informatics (SISY)

[4] Artificial Intelligence-Based Education and Management System for Patients with Chronic Diseases Lingyu Wang;Xi Jiang;Tuaner Zhang;Xiaozhu Wang;K S Shashidhar 2022 International Conference on Knowledge Engineering and Communication Systems (ICKES)

THANK YOU

Vel Tech
Rangarajan Dr. Sagunthala
R&D Institute of Science and Technology
(Deemed to be University Estd. u/s 3 of UGC Act, 1956)