

A REPORT ON
MEDICAL HEALTH CARE SYSTEM
(USING MACHINE LEARNING)

**Submitted in partial fulfilment of the requirement for the award of the degree
of**

MASTER OF COMPUTER APPLICATIONS

Submitted by:

Student Name: Vivek Mehra

Roll no.: 1102924

Under the Guidance of

Dr. Sushil Dimri

Professor



Department of Computer Applications
Graphic Era (Deemed to be University)
Dehradun, Uttarakhand



Graphic Era UNIVERSITY

(Declared as Deemed-to-be-University under section 3 of UGC Act, 1956)

566/6, Bell Road, Clement Town, Dehradun, Uttarakhand, Website: www.geu.ac.in

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the Dissertation entitled "MEDICAL HEALTH CARE SYSTEM(USING MACHINE LEARNING)" in partial fulfillment of the requirements for the award of the degree of Master's of Computer Application, submitted in the Department of Computer Application of the Graphic Era University, Dehradun, is an authentic record of my own work carried out during a period from 2023-2024, under the supervision of Dr.Sushil Dimri, Professor, Department of Computer Application, Graphic Era University, Dehradun (Uttarakhand).

The matter presented in this report has not been submitted by me for the award of any other degree of this or any other Institute/University.

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Supervisor(s) Name & Signature

Signature Head of the Department

The Viva-Voce examination of.....has been held on

External Examiner

Table of Content

	<i>Page No.</i>
<i>Acknowledgement</i>	<i>i</i>
<i>Abstract</i>	<i>ii</i>
Chapter 1. Introduction	6
1.1. Project Introduction	6
1.2. Overview	11
1.3. Motivation Objective	12
Chapter 2. Problem Statement and Technology used	13
2.1. Problem Statement	13
2.2. Requirements specifications	13
2.3. Introduction to Data Science And Machine Learning Technologies	16
Chapter 3. Methodology and Project Management	19
3.1. Possible Approach	19
3.2. Planning and scheduling	20
Chapter 4. System Design And System Analysis	22
4.1. Algorithm	22
4.2. Dataflow Diagram	24

Chapter 5. Input Design	25
5.1. Code Snippets	
5.2. Web based Flask App code snippet	
 Chapter 6. Output Design	 32
6.1. Main Terminal run	32
6.2. Web Interface	33
 Chapter 7. Conclusions and Future Work	 34
7.1. Conclusion	34
7.2. Scope for Future Work	35
 Bibliography	 37
References	38



Graphic Era UNIVERSITY

(Declared as Deemed-to-be-University under section 3 of UGC Act, 1956)

566/6, Bell Road, Clement Town, Dehradun, Uttarakhand, Website: www.geu.ac.in

Acknowledgement

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to **Dr. Sushil Dimri(Professor)** for his guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I would like to express my gratitude towards my parents & member of Graphic Era University for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to industry persons for giving me such attention and time.

Abstract

This project introduces a sophisticated Flask-based web application aimed at revolutionizing how individuals approach preliminary health diagnostics and management. At its core, the application leverages a Support Vector Classifier (SVC) model to predict potential diseases based on user-provided symptoms. This model, trained on extensive medical data, enables the application to deliver reliable and precise predictions, acting as a valuable preliminary diagnostic tool.

Upon entering the platform, users are greeted with an intuitive interface where they can input their symptoms. The user-friendly design ensures that individuals, regardless of their technical proficiency, can effortlessly navigate the application. This ease of use is crucial, as it encourages more users to engage with the tool, potentially identifying health concerns early and seeking timely medical intervention.

The process begins when users enter a list of symptoms into the application. The symptoms are then processed by the SVC model, which has been meticulously trained on a diverse set of medical cases. This model is capable of identifying patterns and correlations between symptoms and diseases, thus predicting the most likely disease the user might be experiencing. The prediction mechanism is highly sophisticated, relying on a large database of symptoms and their associated conditions to ensure accuracy.

One of the standout features of this application is the comprehensive health report generated for each user. Once a disease is predicted, the application retrieves detailed information about it from multiple datasets. These datasets encompass a wide range of medical information, including symptom descriptions, precautionary measures, medications, diets, and workout routines. This holistic approach ensures that users receive not just a diagnosis, but a complete guide on managing and understanding their condition.

For instance, if a user's symptoms suggest they might have diabetes, the application will provide a detailed description of diabetes, including its causes, symptoms, and potential complications. Additionally, it will offer practical advice on how to manage the condition through lifestyle changes, such as specific dietary recommendations and workout plans tailored to their needs. This level of detail is invaluable for users, as it transforms a simple diagnostic tool into a comprehensive health management system.

The application's backend functionality is equally impressive. It utilizes several helper functions to retrieve and format the necessary data for the health report. These functions extract relevant information from the datasets and present it in a user-friendly format. For example, the helper functions can fetch descriptions of diseases, lists of precautions to take, recommended medications,

and suitable dietary plans. This ensures that the information provided is both accurate and easily understandable.

Beyond its diagnostic and advisory capabilities, the application also includes several additional pages to enhance user engagement and information accessibility. The 'About' page offers insights into the application's purpose and the team behind it, fostering trust and transparency. The 'Contact' page provides users with a direct line of communication for further inquiries or support. The 'Developer' page highlights the technical aspects of the project, appealing to those interested in the technology behind the application. Lastly, the 'Blog' section serves as an educational resource, featuring articles on various health topics, thus contributing to the user's overall health literacy.

In summary, this Flask-based web application is a powerful tool designed to bridge the gap between initial symptom recognition and professional medical diagnosis. By offering accurate disease predictions and comprehensive health management advice, it empowers users to take control of their health proactively. The integration of multiple datasets ensures that the information is thorough and reliable, while the user-friendly interface makes the application accessible to a broad audience. Ultimately, this project aims to provide individuals with the tools they need to manage their health effectively, promoting early detection and intervention for better health outcomes.

Chapter 1

Introduction

1.1 Project Introduction

In the contemporary digital age, healthcare technology is rapidly evolving to provide innovative solutions for patient care and self-management. This project exemplifies such innovation by developing a sophisticated Flask-based web application designed to assist users in identifying potential diseases based on their symptoms and offering comprehensive health management advice. The core functionality of this application is powered by a Support Vector Classifier (SVC) model, a machine learning algorithm that has been meticulously trained on extensive medical datasets to accurately predict diseases from a list of user-provided symptoms.

The motivation behind this project stems from the need for accessible, reliable preliminary diagnostic tools that can bridge the gap between initial symptom recognition and professional medical consultation. Often, individuals experience symptoms but may not have immediate access to healthcare providers. This application aims to fill that void by providing users with a preliminary diagnosis and detailed health management recommendations, thereby promoting early detection and intervention.

User Interaction and Experience

Upon visiting the application, users are greeted with an intuitive and user-friendly interface designed to ensure ease of navigation. The primary interaction involves the user inputting their symptoms into a simple, straightforward form. The interface is crafted to accommodate users of varying technical proficiencies, ensuring that anyone can use the application without requiring specialized knowledge.

The process begins with the user entering a list of symptoms. These symptoms are then processed

by the SVC model, which has been trained on a diverse set of medical cases. The model employs sophisticated algorithms to identify patterns and correlations between symptoms and diseases, predicting the most likely disease the user might be experiencing. This predictive capability is the result of extensive training and validation using a robust dataset that includes numerous symptom-disease pairs.

Disease Prediction and Health Reports

One of the standout features of this application is its ability to generate a comprehensive health report for each user. Once a disease is predicted, the application retrieves detailed information about it from multiple integrated datasets. These datasets encompass a wide range of medical information, including symptom descriptions, precautionary measures, medications, diets, and workout routines. This holistic approach ensures that users receive not only a preliminary diagnosis but also a complete guide on managing and understanding their condition.

For example, if a user's symptoms suggest they might have diabetes, the application will provide a detailed description of diabetes, including its causes, symptoms, and potential complications. Additionally, it will offer practical advice on how to manage the condition through lifestyle changes, such as specific dietary recommendations and workout plans tailored to the user's needs. This level of detail is invaluable, transforming the application from a simple diagnostic tool into a comprehensive health management system.

Backend Functionality and Data Integration

The backend of the application is built with several helper functions that retrieve and format the necessary data for the health report. These functions extract relevant information from the datasets and present it in a user-friendly format. For instance, the helper functions can fetch descriptions of diseases, lists of precautions to take, recommended medications, and suitable dietary plans. This ensures that the information provided is both accurate and easily understandable.

The application utilizes several key datasets, including:

Symptom Descriptions: Detailed explanations of various symptoms to help users understand their conditions better.

Precautions: Recommended precautionary measures for different diseases to help prevent worsening or spreading of the condition.

Medications: Suggested medications that are commonly prescribed for the predicted disease.

Diets: Dietary recommendations tailored to the needs of patients with specific conditions.

Workouts: Exercise plans designed to support health and recovery for various diseases.

Enhancing User Engagement

Beyond its diagnostic and advisory capabilities, the application includes several additional pages to enhance user engagement and provide comprehensive support. These include:

About Page: This section offers insights into the application's purpose, the underlying technology, and the team behind it. It fosters trust and transparency, helping users understand the credibility and goals of the project.

Contact Page: Providing users with a direct line of communication for further inquiries, support, or feedback. This feature is crucial for addressing user concerns and improving the application based on user input.

Developer Page: Highlighting the technical aspects of the project, this section appeals to those interested in the technology and development process behind the application. It can also serve as a resource for other developers looking to learn from or contribute to the project.

Blog Section: Serving as an educational resource, the blog features articles on various health topics, contributing to the user's overall health literacy. This section is designed to keep users informed about health trends, preventive measures, and new insights in medical science.

1.2 Overview

This project introduces a Flask-based web application designed to aid users in identifying potential diseases based on their symptoms and providing comprehensive health management advice. At the heart of this application lies a sophisticated Support Vector Classifier (SVC) model, meticulously trained on extensive medical datasets to deliver accurate disease predictions. The user experience is prioritized through an intuitive interface, where users can easily input their symptoms with minimal effort. Behind the scenes, the SVC model processes these symptoms, utilizing advanced algorithms to analyze patterns and correlations, ultimately predicting the most likely disease the user may be experiencing. This predictive capability is the culmination of rigorous training and validation using diverse and robust datasets, ensuring the reliability and accuracy of the predictions.

Once a disease is predicted, the application goes further by generating a detailed health report tailored to the user's specific condition. This report serves as a comprehensive guide, offering insights into the predicted disease, including its causes, symptoms, and potential complications. Additionally, the report provides a wealth of actionable information, such as recommended precautionary measures to prevent the condition from worsening or spreading, as well as suggested medications commonly prescribed for the disease. Furthermore, the application offers dietary recommendations customized to the needs of patients with the specific condition, as well as exercise plans designed to support health and recovery.

By providing users with access to such comprehensive health management advice, the application empowers individuals to take proactive steps towards managing their health effectively. The combination of accurate disease predictions and actionable recommendations serves to bridge the gap between initial symptom recognition and professional medical consultation. Moreover, the user-friendly interface ensures accessibility for individuals of varying technical proficiencies, promoting widespread adoption and utilization of the application.

Through its user-centric design and robust predictive capabilities, the application aims to promote early detection and intervention, ultimately improving health outcomes for its users. By empowering individuals with the tools and information they need to make informed decisions about their health, the application seeks to foster a culture of proactive health management and preventive care. In doing so, it contributes to the broader goal of enhancing public health and well-being.

1.3 Motivation objectives

The motivation behind this project lies in addressing the challenges individuals face in accessing healthcare services due to factors such as geographical limitations, financial constraints, and busy lifestyles. By providing a user-friendly web application accessible from any internet-enabled device, the project aims to overcome these barriers, enabling individuals to receive preliminary health assessments conveniently. Early detection of health conditions is crucial for effective treatment and management, yet individuals may delay seeking professional medical advice due to various reasons. This project seeks to facilitate early detection and intervention by offering a tool that can accurately predict potential diseases based on symptoms, potentially improving health outcomes. Additionally, the project aims to enhance health literacy by providing users with detailed health reports containing descriptions of predicted diseases, precautionary measures, recommended medications, dietary advice, and exercise recommendations. Through these objectives, the project aims to empower individuals with knowledge about their health conditions, enabling informed decision-making and promoting proactive health management. Ultimately, the project contributes to the broader goal of promoting proactive health management and improving health outcomes for individuals worldwide.

Chapter 2

Problem Statement And Technology Used

2.1 Problem Statement

The problem this project aims to address is the limited accessibility to timely healthcare services and the lack of reliable preliminary diagnostic tools for individuals experiencing symptoms of potential medical conditions. Many people face obstacles such as geographical constraints, financial limitations, and busy schedules, which can hinder their ability to seek professional medical consultation promptly. As a result, there is a need for a user-friendly web application that can accurately predict potential diseases based on symptoms and provide comprehensive health management advice.

Additionally, there is a lack of health literacy among individuals, leading to misunderstandings about symptoms, conditions, and appropriate health management strategies. This lack of understanding can contribute to delays in seeking medical attention and ineffective self-management practices. Therefore, there is a need for a solution that not only provides accurate disease predictions but also educates users about their health conditions, empowering them to make informed decisions and take proactive steps towards managing their health effectively.

Furthermore, existing diagnostic tools may not be easily accessible to individuals with limited technical proficiency or those facing language barriers or disabilities. Therefore, the solution should be inclusive and accessible to users of varying backgrounds and abilities.

In summary, the problem statement revolves around the need for a user-friendly, accurate, and inclusive web application that can predict potential diseases based on symptoms, provide comprehensive health management advice, and promote health literacy among users.

2.2 Requirements Specifications

This section outlines the hardware and software requirements necessary for the successful completion of the Medical Health Care System project. These requirements ensure that the data processing, analysis, model training, and evaluation are conducted efficiently and effectively.

Hardware Requirements

Processor:

Minimum: Intel Core i5 or equivalent

Recommended: Intel Core i7 or equivalent

RAM:

Minimum: 8 GB

Recommended: 16 GB or higher

Storage:

Minimum: 256 GB SSD

Recommended: 512 GB SSD or higher for faster data access and storage

Graphics Processing Unit (GPU):

Optional: NVIDIA GPU with CUDA support for accelerated machine learning training (e.g., NVIDIA GTX 1060 or higher)

Internet Connection:

Stable internet connection for accessing online datasets, libraries, and tools.

Software Requirements

To successfully develop, deploy, and run the described Flask-based web application for disease prediction and health management, the following software components are required:

Python: Python is the primary programming language used for developing the backend of the application, implementing the machine learning model, and handling data preprocessing tasks. The recommended version is Python 3.7 or higher.

Flask: Flask is the web framework used to create the web application. It provides essential tools for routing, request handling, and template rendering.

Pandas: Pandas is used for data manipulation and analysis. It is crucial for handling the medical datasets, cleaning data, and transforming it into a suitable format for model training and prediction.

Scikit-learn: Scikit-learn is the machine learning library used to implement the Support Vector Classifier (SVC) and other machine learning functionalities. It supports data preprocessing, model training, and evaluation.

NumPy: NumPy is essential for numerical computations and handling multi-dimensional arrays, which are integral to data preprocessing and manipulation tasks in the project.

SQLite3: SQLite3 is used for database management. It is lightweight and suitable for storing user data, symptom inputs, and generated health reports.

Pickle: Pickle is used for serializing and deserializing the trained machine learning model, allowing the model to be saved to a file and later loaded for making predictions.

HTML/CSS: These are the core technologies for frontend development. HTML is used for structuring web pages, CSS for styling and layout.

Bootstrap: Bootstrap is a frontend framework that facilitates the creation of responsive and visually appealing user interfaces. It provides pre-designed components and styles, speeding up the development process.

Jinja2: Jinja2 is the templating engine used with Flask to render dynamic web pages. It allows the integration of Python code with HTML to create interactive and data-driven user interfaces.

Git: Git is a version control system used for tracking changes in the source code and facilitating collaborative development. It is essential for maintaining the project's codebase and managing version history.

Virtualenv: Virtualenv is a tool for creating isolated Python environments. It ensures that the project's dependencies are managed separately from the system's Python environment, preventing version conflicts and dependency issues.

IDE/Text Editor: An Integrated Development Environment (IDE) like PyCharm, VSCode, or a text editor like Sublime Text is essential for writing and managing the project's code. These tools provide features like syntax highlighting, debugging, and project management capabilities.

Documentation:

Jupyter Notebook: For creating and sharing documents that contain live code, equations, visualizations, and narrative text.

Sphinx or MkDocs: For creating project documentation

Data Sources: Public datasets from repositories such as Kaggle, UCI Machine Learning Repository, or government health databases.

2.3 Introduction to Data Science and Machine Learning Technology

Data science and machine learning are transformative technologies that leverage data to drive insights, predictions, and decision-making across various industries. These fields have become crucial in the healthcare sector, where they enable the analysis of vast amounts of data to improve patient care, optimize operations, and manage costs. This project, focused on predicting medical insurance costs, harnesses the power of data science and machine learning to provide accurate and actionable forecasts, benefiting individuals, insurance companies, and healthcare providers alike.

Key Features of Data Science and Machine Learning:

Data Handling and Preprocessing:

Data handling and preprocessing are foundational steps in data science that ensure the data used for analysis is clean, consistent, and ready for modeling. This involves:

- **Handling Missing Values:** Techniques such as imputation or removal of missing data to ensure completeness.
- **Encoding Categorical Variables:** Converting categorical data into numerical format using methods like one-hot encoding or label encoding.
- **Feature Scaling:** Normalizing or standardizing numerical features to ensure they contribute equally to the model's performance.

Exploratory Data Analysis (EDA):

EDA involves using statistical summaries and visualizations to understand the data's structure, distribution, and relationships between variables. This step helps in identifying trends, patterns, and anomalies, guiding the selection of appropriate models and features.

- **Visualization Tools:** Tools like Matplotlib and Seaborn are used for creating plots and charts to visualize data distributions and relationships.

Machine Learning Algorithms:

Machine learning algorithms are the core of predictive modeling. For this project, various regression algorithms are utilized to predict medical insurance costs:

- **Linear Regression:** A simple and interpretable model that establishes a linear relationship between independent variables and the target variable.
- **Random Forest Regression:** An ensemble method that improves predictive accuracy by combining multiple decision trees.
- **Gradient Boosting Regression:** Another ensemble technique that builds models sequentially to correct errors of previous models.
- **Support Vector Machine (SVM) Regression:** Uses kernel functions to capture complex relationships in the data.
- **K-Nearest Neighbors (KNN) Regression:** A non-parametric method that predicts values based on the closest data points in the feature space.

Model Evaluation:

Evaluating the performance of machine learning models is crucial to ensure their accuracy and reliability. Common evaluation metrics include:

- **R² Score:** Measures the proportion of variance in the target variable explained by the model.
- **Mean Absolute Error (MAE):** The average of absolute differences between predicted and actual values.
- **Mean Squared Error (MSE):** The average of squared differences between predicted and actual values.
- **Root Mean Squared Error (RMSE):** The square root of the MSE, providing an error metric in the same units as the target variable.

Benefits of Data Science and Machine Learning:

Predictive Power: Enables accurate predictions of future events, such as medical insurance costs, which aids in financial planning and risk management.

Scalability: Capable of handling large datasets and complex problems, essential for processing extensive healthcare data.

Automation: Reduces the need for manual intervention, increasing efficiency and consistency in decision-making processes.

Insight Generation: Extracts actionable insights from data, helping stakeholders understand trends and make informed decisions.

Introduction to Specific Technologies Used:

Python Programming Language:

Python is a versatile and widely-used programming language in data science and machine learning due to its readability, extensive libraries, and supportive community. It facilitates data manipulation, analysis, and modeling with ease.

Pandas: A powerful library for data manipulation and analysis, providing data structures like DataFrames for handling structured data.

NumPy: A fundamental package for numerical computing in Python, offering support for arrays and mathematical functions.

Matplotlib and Seaborn: Libraries for data visualization, enabling the creation of static, animated, and interactive plots.

Scikit-Learn: Scikit-learn is a robust machine learning library in Python that provides simple and efficient tools for data mining and data analysis. It is built on NumPy, SciPy, and Matplotlib.

Regression Models: Includes implementations of various regression models such as Linear Regression, Random Forest, Gradient Boosting, SVM, and KNN.

Model Selection: Tools for model selection and evaluation, including cross-validation and hyperparameter tuning.

Benefits of Python and Scikit-Learn:

Ease of Use: Python's syntax is simple and readable, making it accessible to beginners and experts alike.

Extensive Libraries: A vast ecosystem of libraries and frameworks supports various aspects of data science and machine learning.

Community Support: A large and active community contributes to a wealth of resources, tutorials, and tools.

Cross-Platform Compatibility: Python runs on various operating systems, ensuring broad compatibility.

3.1 Possible Approach

1. Problem Definition and Objectives
2. Data Collection and Preparation
3. Model Training and Evaluation
4. Application Development
5. Helper Functions Development
6. Integration and Testing
7. Deployment
8. Maintenance and Updates

Detailed Approach

1. Problem Definition and Objectives

Problem: Users often find it difficult to identify potential diseases based on their symptoms due to the overlap in symptoms across various conditions.

Objectives: Develop a web application that allows users to input their symptoms and receive potential diagnoses. Provide additional medical information such as disease descriptions, precautions, medications, diets, and workout recommendations.

2. Data Collection and Preparation

Objectives: Gather and preprocess the necessary datasets to ensure the machine learning model has accurate and consistent data.

Steps:

Collect Data: Gather datasets containing symptoms, diseases, precautions, medications, diets, and workout recommendations.

Data Cleaning: Remove any inconsistencies, handle missing values, and ensure the data is in a usable format.

Data Transformation: Convert categorical data into numerical form if necessary and normalize or standardize the data.

3. Model Training and Evaluation

Objectives: Train a machine learning model to predict diseases based on symptoms and evaluate its performance.

Steps:

Feature Selection: Identify relevant features (symptoms) for disease prediction.

Model Selection: Choose an appropriate machine learning algorithm (e.g., Support Vector Classifier).

Training: Train the model using the prepared dataset.

Evaluation: Assess the model's performance using metrics such as accuracy, precision, recall, and F1-score.

Model Tuning: Fine-tune the model parameters to improve performance.

Model Saving: Save the trained model using pickle for later use in the application.

4. Application Development

Objectives: Develop the web application using Flask, ensuring it is user-friendly and functional.

Steps:

Setup Flask Project: Create a new Flask project and set up the directory structure.

Define Routes and Views: Implement routes for the home page, prediction page, and other

informational pages.

Design User Interface: Create HTML templates for the web pages and style them using CSS and JavaScript.

Forms Implementation: Add forms for users to input their symptoms.

5. Helper Functions Development

Objectives: Develop utility functions to support the application, such as retrieving disease details based on predictions.

Steps:

Helper Functions: Implement functions to fetch disease descriptions, precautions, medications, diets, and workout recommendations.

Data Integration: Ensure the helper functions can access and return the required data from the datasets.

6. Integration and Testing

Objectives: Integrate all components of the application and perform thorough testing to ensure functionality and reliability.

Steps:

Component Integration: Integrate the model, routes, views, and helper functions.

Unit Testing: Test individual components to ensure they work as expected.

Integration Testing: Test the interaction between different components of the application.

User Acceptance Testing: Gather feedback from potential users and make necessary adjustments.

7. Deployment

Objectives: Deploy the application to a production environment where it can be accessed by users.

Steps:

Choose a Platform: Select a deployment platform (e.g., Heroku, AWS, Google Cloud).

Prepare for Deployment: Configure necessary files (e.g., Procfile, requirements.txt) and ensure the application is ready for deployment.

Deploy: Deploy the application and perform final testing in the production environment.

8. Maintenance and Updates

Objectives: Maintain the application post-deployment and update it as needed to ensure it remains functional and accurate.

Steps:

Monitoring: Monitor the application for any issues or bugs.

Dataset Updates: Regularly update the datasets to reflect the latest medical information.

Model Retraining: Retrain the model periodically to maintain or improve prediction accuracy.

Feature Enhancements: Add new features based on user feedback and evolving requirements.

3.2 Planning and Scheduling

Many individuals struggle with mental health concerns but may hesitate to seek professional help due to stigma, accessibility, or cost. We aim to develop a user-friendly chatbot that provides confidential support, resources, and basic mental health screenings.

Phase 1: Discovery and Definition

Goal: Define the project scope, target audience, and desired functionalities of the chatbot.

Activities:

User Research: Conduct surveys and interviews to understand the needs and preferences of individuals struggling with mental health concerns. This could include preferred communication styles, desired functionalities, and privacy considerations.

Competitive Analysis: Research existing mental health chatbots, identify their strengths and

weaknesses, and look for opportunities to differentiate.

Requirement Gathering: Document user stories and functional requirements for the chatbot, focusing on areas such as conversation flow, mental health resources, and screening capabilities.

Technical Feasibility Assessment: Evaluate the feasibility of implementing functionalities like natural language processing (NLP) for understanding user input and sentiment analysis for tailoring responses.

Ethical Considerations: Develop a framework for responsible chatbot development, focusing on user privacy, data security, and potential limitations of a chatbot for diagnosis or treatment.

Deliverables:

Project charter outlining goals, scope, target audience, and ethical considerations.

User stories and functional requirement document.

Feasibility report outlining technical considerations and limitations.

Phase 2: Design and Development

Goal: Develop the core functionalities of the mental health chatbot.

Activities:

Conversation Design: Design conversation flows that are empathetic, informative, and engaging. Consider potential user scenarios and emotional states.

Content Development: Develop a comprehensive database of mental health information, resources (e.g., helplines, support groups), and basic mental health screening questions.

NLP Integration: Integrate NLP tools to enable the chatbot to understand user input and respond appropriately.

Chatbot Development: Build the chatbot based on the chosen platform (e.g., messaging app integration, standalone app).

Internal Testing: Conduct internal testing to ensure the functionality, accuracy of information, and user-friendliness of the chatbot.

Phase 3: Testing and Refinement

Goal: Refine the chatbot based on user feedback and address any identified issues.

Activities:

User Testing: Conduct user testing with the target audience to gather feedback on the conversation flow, content accuracy, and overall user experience.

Bug fixing: Address any bugs or usability issues identified during internal and user testing.

Content Refinement: Refine the chatbot's content based on user feedback and ensure the information is up-to-date and reliable.

Phase 4: Deployment and Maintenance

Goal: Deploy the chatbot and establish a maintenance plan.

Activities:

Deployment: Deploy the chatbot on the chosen platform.

Monitoring: Set up monitoring tools to track user engagement, identify any issues, and gather data to inform future improvements.

Maintenance Plan: Develop a plan for ongoing maintenance, bug fixes, and potential feature updates based on user feedback and data analysis.

4.1 Algorithm

1. Import Necessary Libraries and Modules:

- 1.1. Import Flask, request, render_template, and jsonify from flask.
- 1.2. Import numpy and pandas for data manipulation.
- 1.3. Import pickle for loading the machine learning model.

2. Initialize Flask App:

- 2.1. Create a Flask application instance.

3. Load Datasets:

- 3.1. Read CSV files containing symptom descriptions, precautions, workouts, descriptions, medications, and diets into pandas DataFrames.

4. Load Pre-trained Model:

- 4.1. Load a pre-trained model using pickle.

5. Helper Function to Retrieve Disease Details:

- 5.1. Define a helper function that takes a disease name and retrieves the corresponding description, precautions, medications, diets, and workouts from the respective DataFrames.

6. Define Symptom and Disease Dictionaries:

- 6.1. Create a dictionary mapping symptoms to indices.
- 6.2. Create a dictionary mapping disease index to disease names.

7. Model Prediction Function:

- 7.1. Define a function get_predicted_value that takes a list of patient symptoms, converts it to a vector and predicts the disease using the pre-trained model.

8. Define Routes and Views:

- 8.1. Define the root route / to render the home page.

9. Define the /predict route to handle prediction logic:

- 9.1. If the request method is POST, retrieve symptoms from the form.
- 9.2. Validate the input symptoms.
- 9.3. Predict the disease using the model.

9.4. Retrieve disease details using the helper function.

9.5. Prepare a report and render the report.html template with the prediction results.

9.6. Define additional routes for about, contact, developer, blog, and report pages.

10. Run the Flask Application:

10.1. Set the debug mode to True and run the application.

4.2 Dataflow Diagram

A dataflow diagram (DFD) is a graphical representation of the flow of data through a system. In your case, the system includes users interacting with your web application to input their symptoms and receive a potential diagnosis.

Dataflow diagram (DFD) for symptom-based disease diagnosis web application:

1. External Entities:

Users: These are the people interacting with your application. They input their symptoms into the system and receive potential diagnoses.

2. Processes:

Web Application Interface: This process represents the backend of your web application, built using Flask. It receives input from users, handles requests, and sends responses. It serves as the intermediary between users and the decision-making process for diagnosis.

Decision Tree Classifier Model: This process embodies the core functionality of your application. It's where the magic happens. This is where the symptom data provided by users is processed using the Decision Tree Classifier model to generate potential diagnoses.

3. Data Stores:

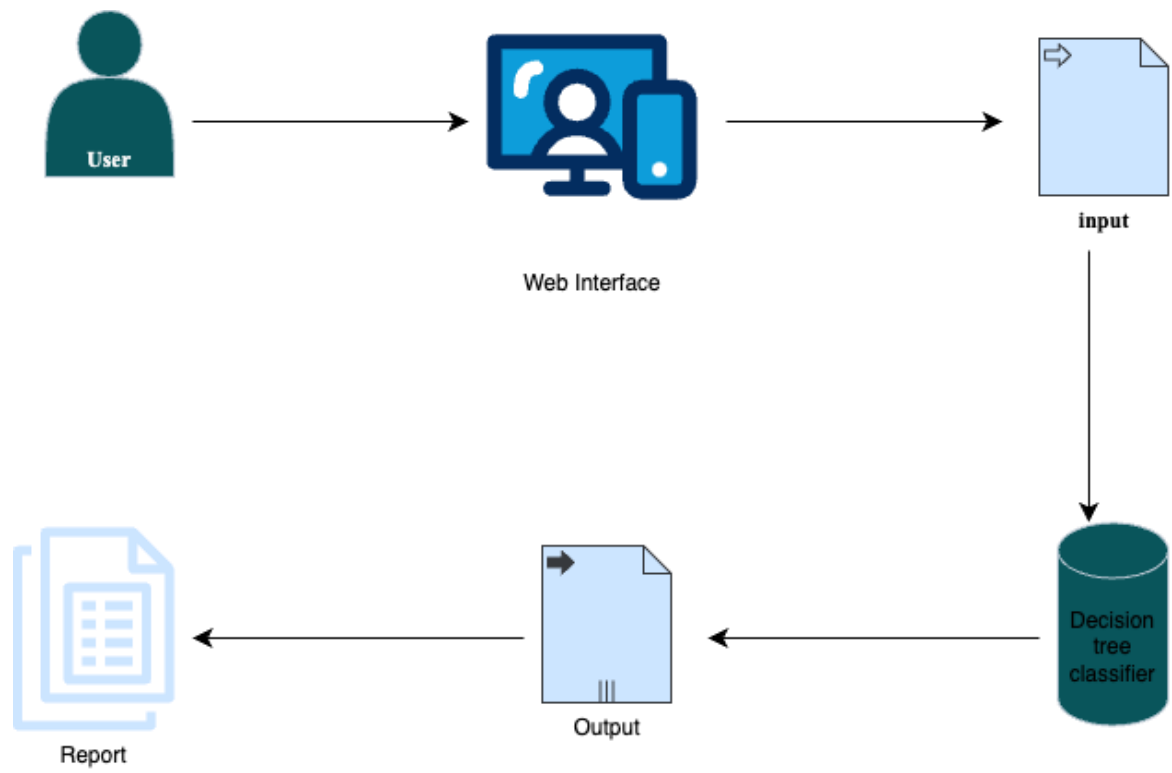
User Symptom Input: This is where the symptoms reported by users are stored temporarily before being processed by the Decision Tree Classifier model. It holds the data until it's ready to be fed into the model.

Diagnosis Output: This data store holds the potential diagnoses generated by the Decision Tree Classifier model. Once the model processes the symptoms, it produces potential illnesses based on its learned patterns.

4. Data Flows:

User Input Data Flow: This represents the flow of data from the user interface (via the Flask backend) to the Decision Tree Classifier model. It's the transmission of the symptoms reported by users to the process responsible for generating diagnoses.

Diagnosis Data Flow: This flow represents the transmission of the potential diagnosis from the Decision Tree Classifier model back to the user interface. Once the model has processed the symptoms, it sends back the potential illnesses for display to the user.



Dataflow Diagram

Chapter 5

Input Design

Load datasets and tools:

load dataset & tools

```
[2]: import pandas as pd
[3]: dataset = pd.read_csv("Training.csv")
[4]: dataset.head()
```

	itching	skin_rash	nodal_skin_eruptions	continuous_sneezing	shivering	chills	joint_pain	stomach_pain	acidity	ulcers_on_tongue	...	blackheads	scarring	skin_peeling	
0	1	1		1	0	0	0	0	0	0	0 ...	0	0	0	0
1	0	1		1	0	0	0	0	0	0	0 ...	0	0	0	0
2	1	0		1	0	0	0	0	0	0	0 ...	0	0	0	0
3	1	1		0	0	0	0	0	0	0	0 ...	0	0	0	0
4	1	1		1	0	0	0	0	0	0	0 ...	0	0	0	0

5 rows x 133 columns

Train test Split:

train test split

```
[9]: from sklearn.model_selection import train_test_split
     from sklearn.preprocessing import LabelEncoder

[10]: X = dataset.drop('prognosis', axis=1) #dropping column wise
     y = dataset['prognosis']

     # encoding prognonsis
     le = LabelEncoder()
     le.fit(y)
     Y = le.transform(y) #storing in Y

     X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3, random_state=20)

[11]: X_train.shape, X_test.shape, y_train.shape, y_test.shape

[11]: ((3444, 132), (1476, 132), (3444,), (1476,))
```

Training Top models

Importing different libraries to train the top models for various classification models on the datasets.

```
[12]: from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC #svm=Support vector machine svc=support vector classifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, confusion_matrix
import numpy as np

# Create a dictionary to store models
models = {
    'SVC': SVC(kernel='linear'),
    'RandomForest': RandomForestClassifier(n_estimators=100, random_state=42),
    'GradientBoosting': GradientBoostingClassifier(n_estimators=100, random_state=42),
    'KNeighbors': KNeighborsClassifier(n_neighbors=5),
    'MultinomialNB': MultinomialNB()
}

# Loop through the models, train, test, and print results
#this loop will take support vector classifier for 1st time and so on
for model_name, model in models.items(): #items will give both key and value
    # Train the model
    model.fit(X_train, y_train)

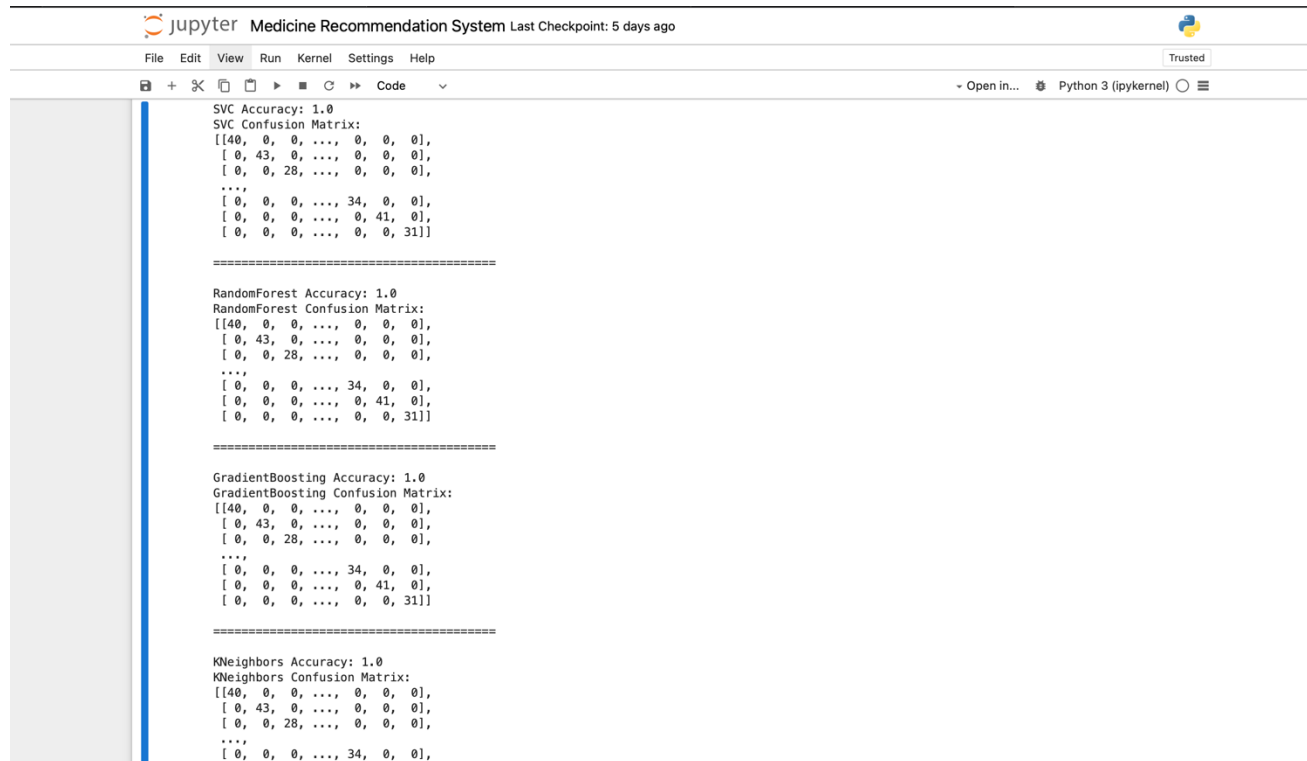
    # Test the model
    predictions = model.predict(X_test)

    # 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=', ')) #convert array to string using array2string of numpy library

    print("\n" + "="*40 + "\n")
```

Output of the trained top models:



```

SVC Accuracy: 1.0
SVC Confusion Matrix:
[[40, 0, 0, ..., 0, 0, 0],
 [ 0, 43, 0, ..., 0, 0, 0],
 [ 0, 0, 28, ..., 0, 0, 0],
 ...,
 [ 0, 0, 0, ..., 34, 0, 0],
 [ 0, 0, 0, ..., 0, 41, 0],
 [ 0, 0, 0, ..., 0, 0, 31]]

=====

RandomForest Accuracy: 1.0
RandomForest Confusion Matrix:
[[40, 0, 0, ..., 0, 0, 0],
 [ 0, 43, 0, ..., 0, 0, 0],
 [ 0, 0, 28, ..., 0, 0, 0],
 ...,
 [ 0, 0, 0, ..., 34, 0, 0],
 [ 0, 0, 0, ..., 0, 41, 0],
 [ 0, 0, 0, ..., 0, 0, 31]]

=====

GradientBoosting Accuracy: 1.0
GradientBoosting Confusion Matrix:
[[40, 0, 0, ..., 0, 0, 0],
 [ 0, 43, 0, ..., 0, 0, 0],
 [ 0, 0, 28, ..., 0, 0, 0],
 ...,
 [ 0, 0, 0, ..., 34, 0, 0],
 [ 0, 0, 0, ..., 0, 41, 0],
 [ 0, 0, 0, ..., 0, 0, 31]]

=====

KNeighbors Accuracy: 1.0
KNeighbors Confusion Matrix:
[[40, 0, 0, ..., 0, 0, 0],
 [ 0, 43, 0, ..., 0, 0, 0],
 [ 0, 0, 28, ..., 0, 0, 0],
 ...,
 [ 0, 0, 0, ..., 34, 0, 0],
 [ 0, 0, 0, ..., 0, 41, 0],
 [ 0, 0, 0, ..., 0, 0, 31]]

```

```
KNeighbors Accuracy: 1.0
KNeighbors Confusion Matrix:
[[40, 0, 0, ..., 0, 0, 0],
 [ 0, 43, 0, ..., 0, 0, 0],
 [ 0, 0, 28, ..., 0, 0, 0],
 ...,
 [ 0, 0, 0, ..., 34, 0, 0],
 [ 0, 0, 0, ..., 0, 41, 0],
 [ 0, 0, 0, ..., 0, 0, 31]]
```

```
=====
MultinomialNB Accuracy: 1.0
MultinomialNB Confusion Matrix:
[[40, 0, 0, ..., 0, 0, 0],
 [ 0, 43, 0, ..., 0, 0, 0],
 [ 0, 0, 28, ..., 0, 0, 0],
 ...,
 [ 0, 0, 0, ..., 34, 0, 0],
 [ 0, 0, 0, ..., 0, 41, 0],
 [ 0, 0, 0, ..., 0, 0, 31]]
=====
```

Single Prediction:

single prediction

```
[13]: # selecting svc
svc = SVC(kernel='linear')
svc.fit(X_train,y_train)
ypred = svc.predict(X_test)
accuracy_score(y_test,ypred)
```

```
[13]: 1.0
```

Saving And loading SVC model:

```
[17]: # save svc model
import pickle
pickle.dump(svc,open('models/svc.pkl','wb')) #wb for binary
```

```
[18]: # load model
svc = pickle.load(open('models/svc.pkl','rb')) #rd is read binary form
```

Test 1:

```
[16]: # test 1:
print("predicted disease :",svc.predict(X_test.iloc[0].values.reshape(1,-1)))
print("Actual Disease :", y_test[0]) #actual disease
```

```
predicted disease : [40]
Actual Disease : 40
```

```
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages/sklearn/base.py:493: UserWarning: X does not have valid feature names, but SVC was fitted with feature names
warnings.warn(
```

Test 2:

```
[16]: # test 2:
print("predicted disease :",svc.predict(X_test.iloc[100].values.reshape(1,-1)))
print("Actual Disease :", y_test[100])
```

```
predicted disease : [39]
Actual Disease : 39
```

```
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages/sklearn/base.py:493: UserWarning: X does not have valid feature names, but SVC was fitted with feature names
warnings.warn(
```

Loading databases:

```
[25]: sym_des = pd.read_csv("syptoms_df.csv")
precautions = pd.read_csv("precautions_df.csv")
workout = pd.read_csv("workout_df.csv")
description = pd.read_csv("description.csv")
medications = pd.read_csv('medications.csv')
diets = pd.read_csv("diets.csv")
```

Using logic for recommendation:

```
[18]: #####
# custome and helping functions
#####helper funtions#####
def helper(dis):
    desc = description[description['Disease'] == predicted_disease]['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']

    return desc,pre,med,die,wrkout

symptoms_dict = {'itching': 0, 'skin_rash': 1, 'nodal_skin_eruptions': 2, 'continuous_sneezing': 3, 'shivering': 4, 'chills': 5, 'joint_pain': 6, 'stomach_pain': 7, 'muscle_pain': 8, 'joint_bone_pain': 9, 'stomach_bloating': 10, 'constipation': 11, 'acid_heartburn': 12, 'diarrhoea': 13, 'mucous_stool': 14, 'yellowish_stool': 15, 'dark_urine': 16, 'body_ache': 17, 'swelling': 18, 'red_sore_eyes': 19, 'headache': 20, 'sore_throat': 21, 'itchy_blisters': 22, 'skin_itch': 23, 'rash': 24, 'blister': 25, 'burning': 26, 'stomach_pain': 27, 'joint_pain': 28, 'muscle_pain': 29, 'joint_bone_pain': 30, 'stomach_bloating': 31, 'constipation': 32, 'acid_heartburn': 33, 'diarrhoea': 34, 'mucous_stool': 35, 'yellowish_stool': 36, 'dark_urine': 37, 'body_ache': 38, 'swelling': 39, 'red_sore_eyes': 40, 'headache': 41, 'sore_throat': 42, 'itchy_blisters': 43, 'skin_itch': 44, 'rash': 45, 'blister': 46, 'burning': 47, 'stomach_pain': 48, 'joint_pain': 49, 'muscle_pain': 50, 'joint_bone_pain': 51, 'stomach_bloating': 52, 'constipation': 53, 'acid_heartburn': 54, 'diarrhoea': 55, 'mucous_stool': 56, 'yellowish_stool': 57, 'dark_urine': 58, 'body_ache': 59, 'swelling': 60, 'red_sore_eyes': 61, 'headache': 62, 'sore_throat': 63, 'itchy_blisters': 64, 'skin_itch': 65, 'rash': 66, 'blister': 67, 'burning': 68, 'stomach_pain': 69, 'joint_pain': 70, 'muscle_pain': 71, 'joint_bone_pain': 72, 'stomach_bloating': 73, 'constipation': 74, 'acid_heartburn': 75, 'diarrhoea': 76, 'mucous_stool': 77, 'yellowish_stool': 78, 'dark_urine': 79, 'body_ache': 80, 'swelling': 81, 'red_sore_eyes': 82, 'headache': 83, 'sore_throat': 84, 'itchy_blisters': 85, 'skin_itch': 86, 'rash': 87, 'blister': 88, 'burning': 89, 'stomach_pain': 90, 'joint_pain': 91, 'muscle_pain': 92, 'joint_bone_pain': 93, 'stomach_bloating': 94, 'constipation': 95, 'acid_heartburn': 96, 'diarrhoea': 97, 'mucous_stool': 98, 'yellowish_stool': 99, 'dark_urine': 100, 'body_ache': 101, 'swelling': 102, 'red_sore_eyes': 103, 'headache': 104, 'sore_throat': 105, 'itchy_blisters': 106, 'skin_itch': 107, 'rash': 108, 'blister': 109, 'burning': 110, 'stomach_pain': 111, 'joint_pain': 112, 'muscle_pain': 113, 'joint_bone_pain': 114, 'stomach_bloating': 115, 'constipation': 116, 'acid_heartburn': 117, 'diarrhoea': 118, 'mucous_stool': 119, 'yellowish_stool': 120, 'dark_urine': 121, 'body_ache': 122, 'swelling': 123, 'red_sore_eyes': 124, 'headache': 125, 'sore_throat': 126, 'itchy_blisters': 127, 'skin_itch': 128, 'rash': 129, 'blister': 130, 'burning': 131, 'stomach_pain': 132, 'joint_pain': 133, 'muscle_pain': 134, 'joint_bone_pain': 135, 'stomach_bloating': 136, 'constipation': 137, 'acid_heartburn': 138, 'diarrhoea': 139, 'mucous_stool': 140, 'yellowish_stool': 141, 'dark_urine': 142, 'body_ache': 143, 'swelling': 144, 'red_sore_eyes': 145, 'headache': 146, 'sore_throat': 147, 'itchy_blisters': 148, 'skin_itch': 149, 'rash': 150, 'blister': 151, 'burning': 152, 'stomach_pain': 153, 'joint_pain': 154, 'muscle_pain': 155, 'joint_bone_pain': 156, 'stomach_bloating': 157, 'constipation': 158, 'acid_heartburn': 159, 'diarrhoea': 160, 'mucous_stool': 161, 'yellowish_stool': 162, 'dark_urine': 163, 'body_ache': 164, 'swelling': 165, 'red_sore_eyes': 166, 'headache': 167, 'sore_throat': 168, 'itchy_blisters': 169, 'skin_itch': 170, 'rash': 171, 'blister': 172, 'burning': 173, 'stomach_pain': 174, 'joint_pain': 175, 'muscle_pain': 176, 'joint_bone_pain': 177, 'stomach_bloating': 178, 'constipation': 179, 'acid_heartburn': 180, 'diarrhoea': 181, 'mucous_stool': 182, 'yellowish_stool': 183, 'dark_urine': 184, 'body_ache': 185, 'swelling': 186, 'red_sore_eyes': 187, 'headache': 188, 'sore_throat': 189, 'itchy_blisters': 190, 'skin_itch': 191, 'rash': 192, 'blister': 193, 'burning': 194, 'stomach_pain': 195, 'joint_pain': 196, 'muscle_pain': 197, 'joint_bone_pain': 198, 'stomach_bloating': 199, 'constipation': 200, 'acid_heartburn': 201, 'diarrhoea': 202, 'mucous_stool': 203, 'yellowish_stool': 204, 'dark_urine': 205, 'body_ache': 206, 'swelling': 207, 'red_sore_eyes': 208, 'headache': 209, 'sore_throat': 210, 'itchy_blisters': 211, 'skin_itch': 212, 'rash': 213, 'blister': 214, 'burning': 215, 'stomach_pain': 216, 'joint_pain': 217, 'muscle_pain': 218, 'joint_bone_pain': 219, 'stomach_bloating': 220, 'constipation': 221, 'acid_heartburn': 222, 'diarrhoea': 223, 'mucous_stool': 224, 'yellowish_stool': 225, 'dark_urine': 226, 'body_ache': 227, 'swelling': 228, 'red_sore_eyes': 229, 'headache': 230, 'sore_throat': 231, 'itchy_blisters': 232, 'skin_itch': 233, 'rash': 234, 'blister': 235, 'burning': 236, 'stomach_pain': 237, 'joint_pain': 238, 'muscle_pain': 239, 'joint_bone_pain': 240, 'stomach_bloating': 241, 'constipation': 242, 'acid_heartburn': 243, 'diarrhoea': 244, 'mucous_stool': 245, 'yellowish_stool': 246, 'dark_urine': 247, 'body_ache': 248, 'swelling': 249, 'red_sore_eyes': 250, 'headache': 251, 'sore_throat': 252, 'itchy_blisters': 253, 'skin_itch': 254, 'rash': 255, 'blister': 256, 'burning': 257, 'stomach_pain': 258, 'joint_pain': 259, 'muscle_pain': 260, 'joint_bone_pain': 261, 'stomach_bloating': 262, 'constipation': 263, 'acid_heartburn': 264, 'diarrhoea': 265, 'mucous_stool': 266, 'yellowish_stool': 267, 'dark_urine': 268, 'body_ache': 269, 'swelling': 270, 'red_sore_eyes': 271, 'headache': 272, 'sore_throat': 273, 'itchy_blisters': 274, 'skin_itch': 275, 'rash': 276, 'blister': 277, 'burning': 278, 'stomach_pain': 279, 'joint_pain': 280, 'muscle_pain': 281, 'joint_bone_pain': 282, 'stomach_bloating': 283, 'constipation': 284, 'acid_heartburn': 285, 'diarrhoea': 286, 'mucous_stool': 287, 'yellowish_stool': 288, 'dark_urine': 289, 'body_ache': 290, 'swelling': 291, 'red_sore_eyes': 292, 'headache': 293, 'sore_throat': 294, 'itchy_blisters': 295, 'skin_itch': 296, 'rash': 297, 'blister': 298, 'burning': 299, 'stomach_pain': 300, 'joint_pain': 301, 'muscle_pain': 302, 'joint_bone_pain': 303, 'stomach_bloating': 304, 'constipation': 305, 'acid_heartburn': 306, 'diarrhoea': 307, 'mucous_stool': 308, 'yellowish_stool': 309, 'dark_urine': 310, 'body_ache': 311, 'swelling': 312, 'red_sore_eyes': 313, 'headache': 314, 'sore_throat': 315, 'itchy_blisters': 316, 'skin_itch': 317, 'rash': 318, 'blister': 319, 'burning': 320, 'stomach_pain': 321, 'joint_pain': 322, 'muscle_pain': 323, 'joint_bone_pain': 324, 'stomach_bloating': 325, 'constipation': 326, 'acid_heartburn': 327, 'diarrhoea': 328, 'mucous_stool': 329, 'yellowish_stool': 330, 'dark_urine': 331, 'body_ache': 332, 'swelling': 333, 'red_sore_eyes': 334, 'headache': 335, 'sore_throat': 336, 'itchy_blisters': 337, 'skin_itch': 338, 'rash': 339, 'blister': 340, 'burning': 341, 'stomach_pain': 342, 'joint_pain': 343, 'muscle_pain': 344, 'joint_bone_pain': 345, 'stomach_bloating': 346, 'constipation': 347, 'acid_heartburn': 348, 'diarrhoea': 349, 'mucous_stool': 350, 'yellowish_stool': 351, 'dark_urine': 352, 'body_ache': 353, 'swelling': 354, 'red_sore_eyes': 355, 'headache': 356, 'sore_throat': 357, 'itchy_blisters': 358, 'skin_itch': 359, 'rash': 360, 'blister': 361, 'burning': 362, 'stomach_pain': 363, 'joint_pain': 364, 'muscle_pain': 365, 'joint_bone_pain': 366, 'stomach_bloating': 367, 'constipation': 368, 'acid_heartburn': 369, 'diarrhoea': 370, 'mucous_stool': 371, 'yellowish_stool': 372, 'dark_urine': 373, 'body_ache': 374, 'swelling': 375, 'red_sore_eyes': 376, 'headache': 377, 'sore_throat': 378, 'itchy_blisters': 379, 'skin_itch': 380, 'rash': 381, 'blister': 382, 'burning': 383, 'stomach_pain': 384, 'joint_pain': 385, 'muscle_pain': 386, 'joint_bone_pain': 387, 'stomach_bloating': 388, 'constipation': 389, 'acid_heartburn': 390, 'diarrhoea': 391, 'mucous_stool': 392, 'yellowish_stool': 393, 'dark_urine': 394, 'body_ache': 395, 'swelling': 396, 'red_sore_eyes': 397, 'headache': 398, 'sore_throat': 399, 'itchy_blisters': 400, 'skin_itch': 401, 'rash': 402, 'blister': 403, 'burning': 404, 'stomach_pain': 405, 'joint_pain': 406, 'muscle_pain': 407, 'joint_bone_pain': 408, 'stomach_bloating': 409, 'constipation': 410, 'acid_heartburn': 411, 'diarrhoea': 412, 'mucous_stool': 413, 'yellowish_stool': 414, 'dark_urine': 415, 'body_ache': 416, 'swelling': 417, 'red_sore_eyes': 418, 'headache': 419, 'sore_throat': 420, 'itchy_blisters': 421, 'skin_itch': 422, 'rash': 423, 'blister': 424, 'burning': 425, 'stomach_pain': 426, 'joint_pain': 427, 'muscle_pain': 428, 'joint_bone_pain': 429, 'stomach_bloating': 430, 'constipation': 431, 'acid_heartburn': 432, 'diarrhoea': 433, 'mucous_stool': 434, 'yellowish_stool': 435, 'dark_urine': 436, 'body_ache': 437, 'swelling': 438, 'red_sore_eyes': 439, 'headache': 440, 'sore_throat': 441, 'itchy_blisters': 442, 'skin_itch': 443, 'rash': 444, 'blister': 445, 'burning': 446, 'stomach_pain': 447, 'joint_pain': 448, 'muscle_pain': 449, 'joint_bone_pain': 450, 'stomach_bloating': 451, 'constipation': 452, 'acid_heartburn': 453, 'diarrhoea': 454, 'mucous_stool': 455, 'yellowish_stool': 456, 'dark_urine': 457, 'body_ache': 458, 'swelling': 459, 'red_sore_eyes': 460, 'headache': 461, 'sore_throat': 462, 'itchy_blisters': 463, 'skin_itch': 464, 'rash': 465, 'blister': 466, 'burning': 467, 'stomach_pain': 468, 'joint_pain': 469, 'muscle_pain': 470, 'joint_bone_pain': 471, 'stomach_bloating': 472, 'constipation': 473, 'acid_heartburn': 474, 'diarrhoea': 475, 'mucous_stool': 476, 'yellowish_stool': 477, 'dark_urine': 478, 'body_ache': 479, 'swelling': 480, 'red_sore_eyes': 481, 'headache': 482, 'sore_throat': 483, 'itchy_blisters': 484, 'skin_itch': 485, 'rash': 486, 'blister': 487, 'burning': 488, 'stomach_pain': 489, 'joint_pain': 490, 'muscle_pain': 491, 'joint_bone_pain': 492, 'stomach_bloating': 493, 'constipation': 494, 'acid_heartburn': 495, 'diarrhoea': 496, 'mucous_stool': 497, 'yellowish_stool': 498, 'dark_urine': 499, 'body_ache': 500, 'swelling': 501, 'red_sore_eyes': 502, 'headache': 503, 'sore_throat': 504, 'itchy_blisters': 505, 'skin_itch': 506, 'rash': 507, 'blister': 508, 'burning': 509, 'stomach_pain': 510, 'joint_pain': 511, 'muscle_pain': 512, 'joint_bone_pain': 513, 'stomach_bloating': 514, 'constipation': 515, 'acid_heartburn': 516, 'diarrhoea': 517, 'mucous_stool': 518, 'yellowish_stool': 519, 'dark_urine': 520, 'body_ache': 521, 'swelling': 522, 'red_sore_eyes': 523, 'headache': 524, 'sore_throat': 525, 'itchy_blisters': 526, 'skin_itch': 527, 'rash': 528, 'blister': 529, 'burning': 530, 'stomach_pain': 531, 'joint_pain': 532, 'muscle_pain': 533, 'joint_bone_pain': 534, 'stomach_bloating': 535, 'constipation': 536, 'acid_heartburn': 537, 'diarrhoea': 538, 'mucous_stool': 539, 'yellowish_stool': 540, 'dark_urine': 541, 'body_ache': 542, 'swelling': 543, 'red_sore_eyes': 544, 'headache': 545, 'sore_throat': 546, 'itchy_blisters': 547, 'skin_itch': 548, 'rash': 549, 'blister': 550, 'burning': 551, 'stomach_pain': 552, 'joint_pain': 553, 'muscle_pain': 554, 'joint_bone_pain': 555, 'stomach_bloating': 556, 'constipation': 557, 'acid_heartburn': 558, 'diarrhoea': 559, 'mucous_stool': 560, 'yellowish_stool': 561, 'dark_urine': 562, 'body_ache': 563, 'swelling': 564, 'red_sore_eyes': 565, 'headache': 566, 'sore_throat': 567, 'itchy_blisters': 568, 'skin_itch': 569, 'rash': 570, 'blister': 571, 'burning': 572, 'stomach_pain': 573, 'joint_pain': 574, 'muscle_pain': 575, 'joint_bone_pain': 576, 'stomach_bloating': 577, 'constipation': 578, 'acid_heartburn': 579, 'diarrhoea': 580, 'mucous_stool': 581, 'yellowish_stool': 582, 'dark_urine': 583, 'body_ache': 584, 'swelling': 585, 'red_sore_eyes': 586, 'headache': 587, 'sore_throat': 588, 'itchy_blisters': 589, 'skin_itch': 590, 'rash': 591, 'blister': 592, 'burning': 593, 'stomach_pain': 594, 'joint_pain': 595, 'muscle_pain': 596, 'joint_bone_pain': 597, 'stomach_bloating': 598, 'constipation': 599, 'acid_heartburn': 600, 'diarrhoea': 601, 'mucous_stool': 602, 'yellowish_stool': 603, 'dark_urine': 604, 'body_ache': 605, 'swelling': 606, 'red_sore_eyes': 607, 'headache': 608, 'sore_throat': 609, 'itchy_blisters': 610, 'skin_itch': 611, 'rash': 612, 'blister': 613, 'burning': 614, 'stomach_pain': 615, 'joint_pain': 616, 'muscle_pain': 617, 'joint_bone_pain': 618, 'stomach_bloating': 619, 'constipation': 620, 'acid_heartburn': 621, 'diarrhoea': 622, 'mucous_stool': 623, 'yellowish_stool': 624, 'dark_urine': 625, 'body_ache': 626, 'swelling': 627, 'red_sore_eyes': 628, 'headache': 629, 'sore_throat': 630, 'itchy_blisters': 631, 'skin_itch': 632, 'rash': 633, 'blister': 634, 'burning': 635, 'stomach_pain': 636, 'joint_pain': 637, 'muscle_pain': 638, 'joint_bone_pain': 639, 'stomach_bloating': 640, 'constipation': 641, 'acid_heartburn': 642, 'diarrhoea': 643, 'mucous_stool': 644, 'yellowish_stool': 645, 'dark_urine': 646, 'body_ache': 647, 'swelling': 648, 'red_sore_eyes': 649, 'headache': 650, 'sore_throat': 651, 'itchy_blisters': 652, 'skin_itch': 653, 'rash': 654, 'blister': 655, 'burning': 656, 'stomach_pain': 657, 'joint_pain': 658, 'muscle_pain': 659, 'joint_bone_pain': 660, 'stomach_bloating': 661, 'constipation': 662, 'acid_heartburn': 663, 'diarrhoea': 664, 'mucous_stool': 665, 'yellowish_stool': 666, 'dark_urine': 667, 'body_ache': 668, 'swelling': 669, 'red_sore_eyes': 670, 'headache': 671, 'sore_throat': 672, 'itchy_blisters': 673, 'skin_itch': 674, 'rash': 675, 'blister': 676, 'burning': 677, 'stomach_pain': 678, 'joint_pain': 679, 'muscle_pain': 680, 'joint_bone_pain': 681, 'stomach_bloating': 682, 'constipation': 683, 'acid_heartburn': 684, 'diarrhoea': 685, 'mucous_stool': 686, 'yellowish_stool': 687, 'dark_urine': 688, 'body_ache': 689, 'swelling': 690, 'red_sore_eyes': 691, 'headache': 692, 'sore_throat': 693, 'itchy_blisters': 694, 'skin_itch': 695, 'rash': 696, 'blister': 697, 'burning': 698, 'stomach_pain': 699, 'joint_pain': 700, 'muscle_pain': 701, 'joint_bone_pain': 702, 'stomach_bloating': 703, 'constipation': 704, 'acid_heartburn': 705, 'diarrhoea': 706, 'mucous_stool': 707, 'yellowish_stool': 708, 'dark_urine': 709, 'body_ache': 710, 'swelling': 711, 'red_sore_eyes': 712, 'headache': 713, 'sore_throat': 714, 'itchy_blisters': 715, 'skin_itch': 716, 'rash': 717, 'blister': 718, 'burning': 719, 'stomach_pain': 720, 'joint_pain': 721, 'muscle_pain': 722, 'joint_bone_pain': 723, 'stomach_bloating': 724, 'constipation': 725, 'acid_heartburn': 726, 'diarrhoea': 727, 'mucous_stool': 728, 'yellowish_stool': 729, 'dark_urine': 730, 'body_ache': 731, 'swelling': 732, 'red_sore_eyes': 733, 'headache': 734, 'sore_throat': 735, 'itchy_blisters': 736, 'skin_itch': 737, 'rash': 738, 'blister': 739, 'burning': 740, 'stomach_pain': 741, 'joint_pain': 742, 'muscle_pain': 743, 'joint_bone_pain': 744, 'stomach_bloating': 745, 'constipation': 746, 'acid_heartburn': 747, 'diarrhoea': 748, 'mucous_stool': 749, 'yellowish_stool': 750, 'dark_urine': 751, 'body_ache': 752, 'swelling': 753, 'red_sore_eyes': 754, 'headache': 755, 'sore_throat': 756, 'itchy_blisters': 757, 'skin_itch': 758, 'rash': 759, 'blister': 760, 'burning': 761, 'stomach_pain': 762, 'joint_pain': 763, 'muscle_pain': 764, 'joint_bone_pain': 765, 'stomach_bloating': 766, 'constipation': 767, 'acid_heartburn': 768, 'diarrhoea': 769, 'mucous_stool': 770, 'yellowish_stool': 771, 'dark_urine': 772, 'body_ache': 773, 'swelling': 774, 'red_sore_eyes': 775, 'headache': 776, 'sore_throat': 777, 'itchy_blisters': 778, 'skin_itch': 779, 'rash': 780, 'blister': 781, 'burning': 782, 'stomach_pain': 783, 'joint_pain': 784, 'muscle_pain': 785, 'joint_bone_pain': 786, 'stomach_bloating': 787, 'constipation': 788, 'acid_heartburn': 789, 'diarrhoea': 790, 'mucous_stool': 791, 'yellowish_stool': 792, 'dark_urine': 793, 'body_ache': 794, 'swelling': 795, 'red_sore_eyes': 796, 'headache': 797, 'sore_throat': 798, 'itchy_blisters': 799, 'skin_itch': 800, 'rash': 801, 'blister': 802, 'burning': 803, 'stomach_pain': 804, 'joint_pain': 805, 'muscle_pain': 806, 'joint_bone_pain': 807, 'stomach_bloating': 808, 'constipation': 809, 'acid_heartburn': 810, 'diarrhoea': 811, 'mucous_stool': 812, 'yellowish_stool': 813, 'dark_urine': 814, 'body_ache': 815, 'swelling': 816, 'red_sore_eyes': 817, 'headache': 818, 'sore_throat': 819, 'itchy_blisters': 820, 'skin_itch': 821, 'rash': 822, 'blister': 823, 'burning': 824, 'stomach_pain': 825, 'joint_pain': 826, 'muscle_pain': 827, 'joint_bone_pain': 828, 'stomach_bloating': 829, 'constipation': 830, 'acid_heartburn': 831, 'diarrhoea': 832, 'mucous_stool': 833, 'yellowish_stool': 834, 'dark_urine': 835, 'body_ache': 836, 'swelling': 837, 'red_sore_eyes': 838, 'headache': 839, 'sore_throat': 840, 'itchy_blisters': 841, 'skin_itch': 842, 'rash': 843, 'blister': 844, 'burning': 845, 'stomach_pain': 846, 'joint_pain': 847, 'muscle_pain': 848, 'joint_bone_pain': 849, 'stomach_bloating': 850, 'constipation': 851, 'acid_heartburn': 852, 'diarrhoea': 853, 'mucous_stool': 854, 'yellowish_stool': 855, 'dark_urine': 856, 'body_ache': 857, 'swelling': 858, 'red_sore_eyes': 859, 'headache': 860, 'sore_throat': 861, 'itchy_blisters': 862, 'skin_itch': 863, 'rash': 864, 'blister': 865, 'burning': 866, 'stomach_pain': 867, 'joint_pain': 868, 'muscle_pain': 869, 'joint_bone_pain': 870, 'stomach_bloating': 871, 'constipation': 872, 'acid_heartburn': 873, 'diarrhoea': 874, 'mucous_stool': 875, 'yellowish_stool': 876, 'dark_urine': 877, 'body_ache': 878, 'swelling': 879, 'red_sore_eyes': 880, 'headache': 881, 'sore_throat': 882, 'itchy_blisters': 883, 'skin_itch': 884, 'rash': 885, 'blister': 886, 'burning': 887, 'stomach_pain': 888, 'joint_pain': 889, 'muscle_pain': 890, 'joint_bone_pain': 891, 'stomach_bloating': 892, 'constipation': 893, 'acid_heartburn': 894, 'diarrhoea': 895, 'mucous_stool': 896, 'yellowish_stool': 897, 'dark_urine': 898, 'body_ache': 899, 'swelling': 900, 'red_sore_eyes': 901, 'headache': 902, 'sore_throat': 903, 'itchy_blisters': 904, 'skin_itch': 905, 'rash': 906, 'blister': 907, 'burning': 908, 'stomach_pain': 909, 'joint_pain': 910, 'muscle_pain': 911, 'joint_bone_pain': 912, 'stomach_bloating': 913, 'constipation': 914, 'acid_heartburn': 915, 'diarrhoea': 916, 'mucous_stool': 917, 'yellowish_stool': 918, 'dark_urine': 919, 'body_ache': 920, 'swelling': 921, 'red_sore_eyes': 922, 'headache': 923, 'sore_throat': 924, 'itchy_blisters': 925, 'skin_itch': 926, 'rash': 927, 'blister': 928, 'burning': 929, 'stomach_pain': 930, 'joint_pain': 931, 'muscle_pain': 932, 'joint_bone_pain': 933, 'stomach_bloating': 934, 'constipation': 935, 'acid_heartburn': 936, 'diarrhoea': 937, 'mucous_stool': 938, 'yellowish_stool': 939, 'dark_urine': 940, 'body_ache': 941, 'swelling': 942, 'red_sore_eyes': 943, 'headache': 944, 'sore_throat': 945, 'itchy_blisters': 946, 'skin_itch': 947, 'rash': 948, 'blister': 949, 'burning': 950, 'stomach_pain': 951, 'joint_pain': 952, 'muscle_pain': 953, 'joint_bone_pain': 954, 'stomach_bloating': 955, 'constipation': 956, 'acid_heartburn': 957, 'diarrhoea': 958, 'mucous_stool': 959, 'yellowish_stool': 960, 'dark_urine': 961, 'body_ache': 962, 'swelling': 963, 'red_sore_eyes': 964, 'headache': 965, 'sore_throat': 966, 'itchy_blisters': 967, 'skin_itch': 968, 'rash': 969, 'blister': 970, 'burning': 971, 'stomach_pain': 972, 'joint_pain': 973, 'muscle_pain': 974, 'joint_bone_pain': 975, 'stomach_bloating': 976, 'constipation': 977, 'acid_heartburn': 978, 'diarrhoea': 979, 'mucous_stool': 980, 'yellowish_stool': 981, 'dark_urine': 982, 'body_ache': 983, 'swelling': 984, 'red_sore_eyes': 985, 'headache': 986, 'sore_throat': 987, 'itchy_blisters': 988, 'skin_itch': 989, 'rash': 990, 'blister': 991, 'burning': 992, 'stomach_pain': 993, 'joint_pain': 994, 'muscle_pain': 995, 'joint_bone_pain': 996, 'stomach_bloating': 997, 'constipation': 998, 'acid_heartburn': 999, 'diarrhoea': 1000, 'mucous_stool': 1001, 'yellowish_stool': 1002, 'dark_urine': 1003, 'body_ache': 1004, 'swelling': 1005, 'red_sore_eyes': 1006, 'headache': 1007, 'sore_throat': 1008, 'itchy_blisters': 1009, 'skin_itch': 1010, 'rash': 1011, 'blister': 1012, 'burning': 1013, 'stomach_pain': 1014, 'joint_pain': 1015, 'muscle_pain': 1016, 'joint_bone_pain': 1017, 'stomach_bloating': 1018, 'constipation': 1019, 'acid_heartburn': 1020, 'diarrhoea': 1021, 'mucous_stool': 1022, 'yellowish_stool': 1023, 'dark_urine': 1024, 'body_ache': 1025, 'swelling': 1026, 'red_sore_eyes': 1027, 'headache': 1028, 'sore_throat': 1029, 'itchy_blisters': 1030, 'skin_itch': 1031, 'rash': 1032, 'blister': 1033, 'burning': 1034, 'stomach_pain': 1035, 'joint_pain': 1036, 'muscle_pain': 1037, 'joint_bone_pain': 1038, 'stomach_bloating': 1039, 'constipation': 1040, 'acid_heartburn': 1041, 'diarrhoea': 1042, 'mucous_stool': 1043, 'yellowish_stool': 1044, 'dark_urine': 1045, 'body_ache': 1046, 'swelling': 1047, 'red_sore_eyes': 1048, 'headache': 1049, 'sore_throat': 1050, 'itchy_blisters': 1051, 'skin_itch': 1052, 'rash': 1053, 'blister': 1054, 'burning': 1055, 'stomach_pain': 1056, 'joint_pain': 1057, 'muscle_pain': 1058, 'joint_bone_pain': 1059, 'stomach_bloating': 1060, 'constipation': 1061, 'acid_heartburn': 1062, 'diarrhoea': 1063, 'mucous_stool': 1064, 'yellowish_stool': 1065, 'dark_urine': 1066, 'body_ache': 1067, 'swelling': 1068, 'red_sore_eyes': 1069, 'headache': 1070, 'sore_throat': 1071, 'itchy_blisters': 1072, 'skin_itch': 1073, 'rash': 1074, 'blister': 1075, 'burning': 1076, 'stomach_pain': 1077, 'joint_pain': 1078, 'muscle_pain': 1079, 'joint_bone_pain': 1080, 'stomach_bloating': 1081, 'constipation': 1082, 'acid_heartburn': 1083, 'diarrhoea': 1084, 'mucous_stool': 1085, 'yellowish_stool': 1086, 'dark_urine': 1087, 'body_ache': 1088, 'swelling': 1089, 'red_sore_eyes': 1090, 'headache': 1091, 'sore_throat': 1092, 'itchy_blisters': 1093, 'skin_itch': 1094, 'rash': 1095, 'blister': 1096, 'burning': 1097, 'stomach_pain': 1098, 'joint_pain': 1099, 'muscle_pain': 1100, 'joint_bone_pain': 1101, 'stomach_bloating': 1102, 'constipation': 1103, 'acid_heartburn': 1104, 'diarrhoea': 1105, 'mucous_stool': 1106, 'yellowish_stool': 1107, 'dark_urine': 1108, 'body_ache': 1109, 'swelling': 1110, 'red_sore_eyes': 1111, 'headache': 1112, 'sore_throat': 1113, 'itchy_blisters': 1114, 'skin_itch': 1115, 'rash': 1116, 'blister': 1117, 'burning': 1118, 'stomach_pain': 1119, 'joint_pain': 1120, 'muscle_pain': 1121, 'joint_bone_pain': 1122, 'stomach_bloating': 1123, 'constipation': 1124, 'acid_heartburn': 1125, 'diarrhoea': 1126, 'mucous_stool': 1127, 'yellowish_stool': 1128, 'dark_urine': 1129, 'body_ache': 1130, 'swelling': 1131, 'red_sore_eyes': 1132, 'headache': 1133, 'sore_throat': 1134, 'itchy_blisters': 1135, 'skin_itch': 1136, 'rash': 1137, 'blister': 1138, 'burning': 11
```

```

Enter your symptoms..... itching
=====predicted disease=====
Fungal infection
=====description=====
Fungal infection is a common skin condition caused by fungi.
=====precautions=====
1 : bath twice
2 : use detol or neem in bathing water
3 : keep infected area dry
4 : use clean cloths
=====medications=====
5 : ['Antifungal Cream', 'Fluconazole', 'Terbinafine', 'Clotrimazole', 'Ketoconazole']
=====workout=====
6 : Avoid sugary foods
7 : Consume probiotics
8 : Increase intake of garlic
9 : Include yogurt in diet
10 : Limit processed foods
11 : Stay hydrated
12 : Consume green tea
13 : Eat foods rich in zinc
14 : Include turmeric in diet
15 : Eat fruits and vegetables
=====diets=====
16 : ['Antifungal Diet', 'Probiotics', 'Garlic', 'Coconut oil', 'Turmeric']
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages/sklearn/base.py:493: UserWarning: X does not have valid feature names, but SVC
feature names
warnings.warn(

[23]: description[description['Disease'] == predicted_disease]['Description']

[23]: 0    Fungal infection is a common skin condition ca...
      Name: Description, dtype: object

```

Test 2nd case:

```

•[24]: # Test 2
# Split the user's input into a list of symptoms (assuming they are comma-separated) # yellow_crust_ooze,red_sore_around_nose,small_dents_in_nails,inflammatory
symptoms = input("Enter your symptoms:")
user_symptoms = [s.strip() for s in symptoms.split(',')]
# Remove any extra characters, if any
user_symptoms = [symptom.strip("[]' ") for symptom in user_symptoms]
predicted_disease = get_predicted_value(user_symptoms)

desc, pre, med, die, wrkout = helper(predicted_disease)

print("=====predicted disease=====")
print(predicted_disease)
print("=====description=====")
print(desc)
print("=====precautions=====")
i = 1
for p_i in pre[0]:
    print(i, ": ", p_i)
    i += 1

print("=====medications=====")
for m_i in med:
    print(i, ": ", m_i)
    i += 1

print("=====workout=====")
for w_i in wrkout:
    print(i, ": ", w_i)
    i += 1

print("=====diets=====")
for d_i in die:
    print(i, ": ", d_i)
    i += 1

```

Output of the 2nd case:

```

Enter your symptoms..... cough
=====predicted disease=====
Urinary tract infection
=====description=====
Urinary tract infection is an infection in any part of the urinary system.
=====precautions=====
1 : drink plenty of water
2 : increase vitamin c intake
3 : drink cranberry juice
4 : take probiotics
=====medications=====
5 : ['Antibiotics', 'Urinary analgesics', 'Phenazopyridine', 'Antispasmodics', 'Probiotics']
=====workout=====
6 : Stay hydrated
7 : Consume cranberry products
8 : Include vitamin C-rich foods
9 : Limit caffeine and alcohol
10 : Consume probiotics
11 : Avoid spicy and acidic foods
12 : Consult a healthcare professional
13 : Follow medical recommendations
14 : Maintain good hygiene
15 : Limit sugary foods and beverages
=====diets=====
16 : ['UTI Diet', 'Hydration', 'Cranberry juice', 'Probiotics', 'Vitamin C-rich foods']
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages/sklearn/base.py:493: UserWarning: X does not have valid feature names, but SVC
feature names
warnings.warn(

```

```
Medicine-Recommendation-System-Personalized-Medical-Recommendation-System-with-Machine-Learning-main | main.py
Project
main.py x
1 from flask import Flask, request, render_template, jsonify # Import jsonify
2 import numpy as np
3 import pandas as pd
4 import pickle
5
6
7 # flask app
8 app = Flask(__name__)
9
10
11
12 # load database dataset=====
13 sym_des = pd.read_csv("symtoms_df.csv")
14 precautions = pd.read_csv("precautions_df.csv")
15 workout = pd.read_csv("workout_df.csv")
16 description = pd.read_csv("description.csv")
17 medications = pd.read_csv("medications.csv")
18 diets = pd.read_csv("diets.csv")
19
20
21 # load model=====
22 svc = pickle.load(open('models/svc.pkl','rb'))
23
24
25 #=====
26 # custom and helping functions
27 #=====helper functions=====
28 1 usage
29 def helper(dis):
30     desc = description[description['Disease'] == dis]['Description']
31     desc = " ".join([w for w in desc])
32
33     pre = precautions[precautions['Disease'] == dis][['Precaution_1', 'Precaution_2', 'Precaution_3', 'Precaution_4']]
34     pre = [col for col in pre.values]
35
36     med = medications[medications['Disease'] == dis]['Medication']
37     med = [med for med in med.values]
```

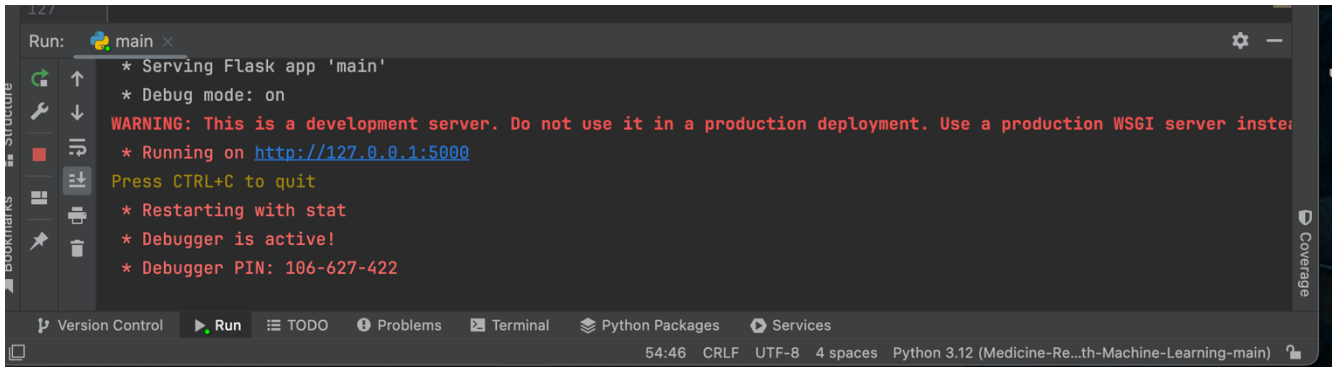
30

```
Medicine-Recommendation-System-Personalized-Medical-Recommendation-System-with-Machine-Learning-main main.py
main.py
69 def home():
70     """
71     """
72     return render_template(template_name_or_list='report.html', report=report)
73
74
75
76
77
78
79
80
81
82
83
84 # about view function and path
85 @app.route('/about')
86 def about():
87     return render_template('about.html')
88
89 # contact view function and path
90 @app.route('/contact')
91 def contact():
92     return render_template("contact.html")
93
94
95
96
97
98
99 # developer view function and path
100 @app.route('/developer')
101 def developer():
102     return render_template("developer.html")
103
104
105
106
107
108
109 # blog view function and path
110 @app.route('/blog')
111 def blog():
112     return render_template("blog.html")
113
114
115
116
117
118
119 #report
120 @app.route('/report')
121 def report():
122     return render_template("report.html")
123
124
125
126
127
128
129 #python main
130 if __name__ == '__main__':
131
132     app.run(debug=True)
```

Chapter 6

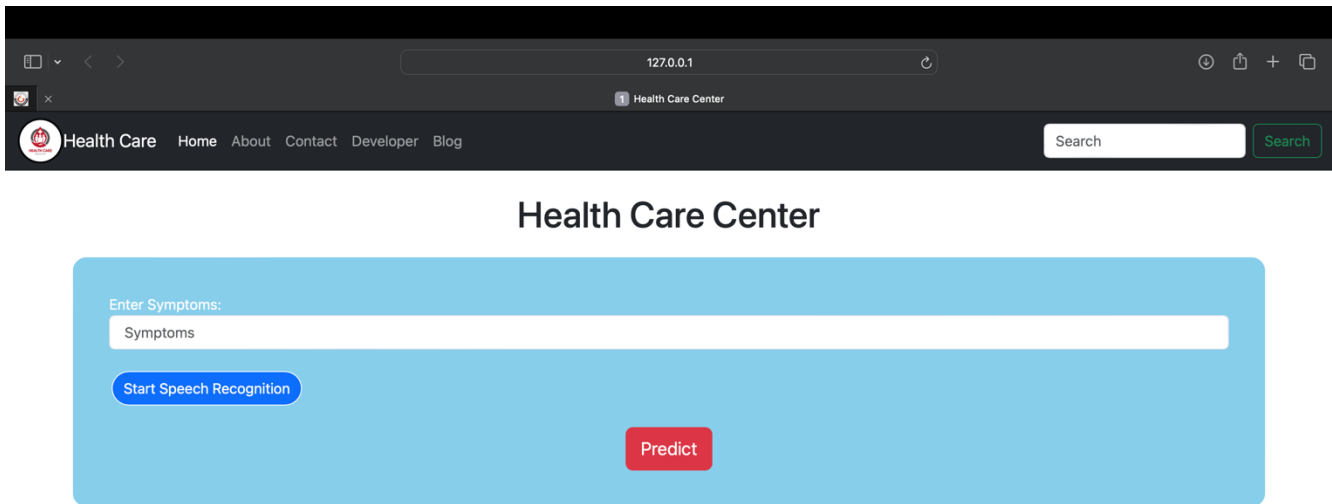
Output Design

Run the main from the terminal:



```
Run: main x
* Serving Flask app 'main'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 106-627-422
```

Open the URL provided by the code:



User will enter Symptoms on the Symptom field:

Health Care Center

Enter Symptoms:

itching, skin_rash, chills

Start Speech Recognition

Predict

Output will be in the form of medical report which is downloadable:

Health Report

Symptoms Entered

itching
skin_rash
chills

Predicted Disease: Fungal infection

Fungal infection is a common skin condition caused by fungi.

Precautions

bath twice
use detol or neem in bathing water
keep infected area dry
use clean cloths

Medications

[Antifungal Cream, 'Fluconazole', 'Terbinafine', 'Clotrimazole', 'Ketoconazole']

Recommended Diet

[Antifungal Diet, 'Probiotics', 'Garlic', 'Coconut oil', 'Turmeric']

Workout Recommendations

Go Back Download Report

Chapter 7

Conclusions And Future Work

Our Symptom-Based Disease Diagnosis Web Application represents a significant advancement in the integration of machine learning and healthcare. By leveraging the power of technology, we have created a tool that offers users immediate access to healthcare information, allowing them to gain insights into potential health conditions quickly and conveniently. This web app is more than just a technological innovation; it is a bridge that connects individuals to vital health information, thereby fostering a more informed and health-conscious society.

At the heart of our application lies a Decision Tree Classifier model, a machine learning tool that processes user-reported symptoms to predict potential illnesses. This model, integrated seamlessly with a Flask-based backend, ensures that users receive accurate and timely information based on their inputs. The user interface is designed to be intuitive, ensuring that individuals, regardless of their technical proficiency, can navigate the application with ease and confidence.

The importance of this project cannot be overstated in the current era of healthcare. With the rise of remote healthcare solutions, driven by technological advancements and the necessity to minimize physical contact due to global health crises, our application provides a critical resource. It offers an alternative to traditional healthcare consultations, allowing users to assess their symptoms from the safety and comfort of their homes. This is particularly beneficial for individuals in remote or underserved areas where access to healthcare professionals may be limited.

Furthermore, our application promotes proactive health management. By providing users with potential diagnoses based on their symptoms, it encourages them to seek medical advice for conditions that might otherwise go unnoticed. Early detection and intervention are crucial in managing many health conditions, and our tool aims to facilitate this by empowering users with information. This proactive approach can lead to better health outcomes and a reduction in the burden on healthcare systems.

The accessibility of our web app is a cornerstone of its design. We have ensured that the application is accessible to a wide range of users, including those with limited technological skills. The user interface is straightforward, and the process of inputting symptoms and receiving a diagnosis is streamlined and user-friendly. This inclusivity ensures that more individuals can benefit from the application, thus extending its positive impact.

Moreover, the reliability of the information provided by our Decision Tree Classifier model is grounded in extensive data and training. We have utilized comprehensive datasets to train our model, ensuring that it can accurately identify a wide range of conditions based on user inputs. While the application is not a substitute for professional medical advice, it serves as a valuable preliminary tool that can guide users in seeking appropriate medical attention.

In conclusion, our Symptom-Based Disease Diagnosis Web Application is a testament to the potential of combining machine learning with healthcare to create meaningful, user-centric solutions. It is designed to make healthcare information more accessible, reliable, and actionable. In doing so, we hope to contribute to a more informed and health-aware public, ultimately improving the well-being of individuals worldwide. As we continue to refine and expand our application, we are committed to maintaining its relevance and effectiveness in an ever-evolving healthcare landscape. This project is a step forward in making healthcare more accessible and empowering individuals to take charge of their health with the aid of modern technology.

Scope for Future Work

The Symptom-Based Disease Diagnosis Web Application represents a robust foundation in the fusion of machine learning and healthcare. However, there is significant scope for future work to enhance its functionality, accuracy, user experience, and overall impact. Below are some key areas for potential future development:

1. Enhanced Model Accuracy and Breadth

Improved Data Quality: Integrating more comprehensive and diverse datasets to train the Decision Tree Classifier can improve the model's accuracy. Including data from different demographics, geographical regions, and a wider range of symptoms and conditions can enhance the model's predictive capabilities.

Advanced Algorithms: Exploring more sophisticated machine learning algorithms, such as ensemble methods or neural networks, could provide better accuracy and more nuanced predictions compared to the current Decision Tree Classifier.

2. Personalized User Experience

User Profiles: Implementing user profiles that store historical symptom data and diagnoses can personalize the experience. This would enable the model to consider past health information, potentially improving prediction accuracy.

Adaptive Learning: Incorporating mechanisms for the model to learn from user feedback can enhance its reliability. Users can provide feedback on the accuracy of the diagnosis, allowing the model to refine its predictions over time.

3. Expanded Symptom Database

More Symptoms and Conditions: Continuously updating the symptom and condition database to include newly discovered diseases and emerging health conditions ensures the app remains relevant and comprehensive.

Localization and Language Support: Adding support for multiple languages and localized health data can make the application more accessible to non-English speaking

users and those from various cultural backgrounds.

4. Integration with Wearable Devices

Real-Time Data Collection: Integrating the application with wearable health devices can provide real-time data on vital signs, activity levels, and other health metrics. This real-time data can enhance the model's predictive accuracy and provide more timely health insights.

Preventive Health Monitoring: Using data from wearables, the application could offer proactive health monitoring, alerting users to potential health issues before they become symptomatic.

5. Telemedicine Integration

Direct Communication with Healthcare Providers: Adding features that allow users to directly connect with healthcare professionals for consultations can bridge the gap between preliminary self-diagnosis and professional medical advice.

Electronic Health Records (EHR) Integration: Integrating with EHR systems can enable seamless sharing of symptom data with healthcare providers, ensuring that users receive a more coordinated and informed care experience.

6. Regulatory Compliance and Security

HIPAA and GDPR Compliance: Ensuring the application complies with healthcare data privacy regulations such as HIPAA (Health Insurance Portability and Accountability Act) and GDPR (General Data Protection Regulation) is critical. This includes implementing robust data encryption, secure storage, and user consent protocols.

Security Enhancements: Regularly updating the application's security measures to protect against data breaches and cyber threats is essential to maintain user trust and safeguard sensitive health information.

7. User Education and Engagement

Health Education Content: Providing users with educational content about common symptoms and conditions can empower them to make informed health decisions. This could include articles, videos, and interactive content.

Community Features: Implementing community features such as forums or support groups can offer users a platform to share experiences and support each other, fostering a sense of community and engagement.

8. Artificial Intelligence and Natural Language Processing

NLP for Symptom Input: Utilizing natural language processing (NLP) to allow users to input symptoms in a conversational manner can enhance usability and make the application more intuitive.

Chatbot Integration: Developing a chatbot that can guide users through the symptom input process, answer common health-related questions, and provide immediate feedback can improve the user experience.

9. Continuous Improvement and Feedback Loops

User Feedback Mechanisms: Implementing comprehensive feedback mechanisms where users can report their experiences and suggest improvements can help continuously refine the application.

Regular Updates: Maintaining a regular schedule of updates and improvements based on user feedback and advancements in medical knowledge ensures the application remains current and effective.

Bibliography

1. Scikit-Learn Documentation: The official documentation for Scikit-Learn provides detailed information, examples, and API references for using machine learning algorithms, model evaluation techniques, and data preprocessing methods.
2. Pandas Documentation: The Pandas documentation offers extensive guides, tutorials, and examples for data manipulation, exploration, and analysis using Pandas data structures such as DataFrames and Series.
3. NumPy Documentation: NumPy's official documentation provides documentation for numerical computing, array manipulation, mathematical functions, and linear algebra operations essential for machine learning tasks.
4. Matplotlib Documentation: Matplotlib's documentation includes tutorials, examples, and API references for creating various types of plots, charts, and visualizations to analyze and present data effectively.
5. Seaborn Documentation: Seaborn's documentation offers guidance on statistical data visualization, including examples and explanations of Seaborn's high-level interface for creating informative and attractive plots.
6. Python Documentation: The official Python documentation provides a comprehensive reference for the Python programming language, covering syntax, built-in functions, standard libraries, and more.
7. Machine Learning Mastery: Machine Learning Mastery is a valuable resource for machine learning practitioners, offering tutorials, articles, and courses on machine learning algorithms, techniques, and best practices.
8. Towards Data Science: Towards Data Science is a popular platform for data science and machine learning enthusiasts, featuring articles, tutorials, and insights from industry experts and practitioners. Towards
9. <https://www.kaggle.com/datasets/noorsaeed/medicine-recommendation-system-dataset>

References

1. Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: Data mining, inference, and prediction (2nd ed.). Springer Science & Business Media.
2. Raschka, S., & Mirjalili, V. (2019). Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow (3rd ed.). Packt Publishing.
3. Brownlee, J. (2020). Machine learning mastery with Python: Understand your data, create accurate models, and work projects end-to-end. Machine Learning Mastery.
4. Géron, A. (2019). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems (2nd ed.). O'Reilly Media.
5. VanderPlas, J. (2016). Python data science handbook: Essential tools for working with data. O'Reilly Media.
6. McKinney, W. (2017). Python for data analysis: Data wrangling with Pandas, NumPy, and IPython (2nd ed.). O'Reilly Media.