Capstone project proposal

Intelligent Medical Diagnosis and Drug Recommendation System Using Patient Statement

# Cover letter

Capstone Project Group A

**Professor Meysam Effati**

**Lambton College, Mississauga**

Dear Professor Effati,

I hope this letter finds you well. I am writing to submit my project proposal for your consideration in the upcoming course on Advanced Machine Learning Applications. The project aims to develop an integrated machine learning application that can assist in the diagnosis and treatment of patients by utilizing natural language processing and classification models.

The proposed application comprises three models that work in tandem to enhance the efficiency and accuracy of medical diagnosis and prescription processes:

1. Symptom Extraction Model: Leveraging an open-source large language model from OpenAI, this model will process patient-provided illness descriptions to extract relevant symptoms. For example, given a statement such as "I have neck pain, at the same time I have body aches and I am having cough," the model will identify "neck pain," "body aches," and "cough" as symptoms.

2. Disease Prediction Model: Using the dataset available at [Kaggle](https://www.kaggle.com/datasets/kaushil268/disease-prediction-using-machine-learning?select=Training.csv), this model will classify the extracted symptoms to predict potential diseases. The dataset provides a comprehensive set of symptom-disease pairs that will enable the model to learn and predict with high accuracy.

3. Drug Recommendation Model: For the final step, the application will suggest appropriate drugs based on the predicted disease using the dataset from [Kaggle](https://www.kaggle.com/datasets/jithinanievarghese/drugs-side-effects-and-medical-condition). This model aims to improve treatment recommendations by considering various factors such as side effects and medical conditions associated with each drug.

The entire system will be deployed within a Docker container to ensure portability and scalability. We will use Flask API to create a user-friendly interface for interaction with the models.

This project not only offers a significant learning opportunity but also holds the potential to provide a practical solution in the healthcare domain, where timely and accurate diagnosis is critical. I am confident that with your guidance, this project will be a valuable contribution to our understanding and application of machine learning techniques.

Thank you for considering my proposal. I am looking forward to your feedback and am excited about the possibility of developing this project under your mentorship.

Sincerely,

Capstone Project Group A

## Project Overview

**Project Title**: Intelligent Medical Diagnosis and Drug Recommendation System Using Patient Statement

Objective

The primary objective of this project is to develop an integrated machine learning application that assists in the diagnosis and treatment of patients by utilizing advanced natural language processing (NLP) and classification models. The system will automate the extraction of symptoms from patient descriptions, predict possible diseases, and recommend appropriate medications, thereby enhancing the efficiency and accuracy of medical diagnostics and treatment planning.

Scope

The project encompasses the development and deployment of a three-stage machine learning pipeline:

1. Symptom Extraction Model:

- Input: Patient's natural language description of their symptoms.

- Output: A list of identified symptoms.

- Technology: An open-source large language model from OpenAI will be employed to perform NLP tasks and extract relevant symptoms from patient statements.

2. Disease Prediction Model:

- Input: Symptoms identified by the Symptom Extraction Model.

- Output: Predicted disease(s).

- Dataset: The disease prediction will be based on the dataset available on Kaggle (https://www.kaggle.com/datasets/kaushil268/disease-prediction-using-machine-learning?select=Training.csv), which includes comprehensive symptom-disease pairs for training the model.

3. Drug Recommendation Model:

- Input: Predicted disease(s).

- Output: Suggested drugs and medications.

- Dataset: Drug recommendation will leverage the dataset from Kaggle (https://www.kaggle.com/datasets/jithinanievarghese/drugs-side-effects-and-medical-condition), which provides information on drugs, their side effects, and associated medical conditions.

**Deployment**:

The entire system will be packaged and deployed using Docker to ensure consistent and scalable performance across different environments. A Flask API will be developed to provide a user-friendly interface for inputting patient data and retrieving diagnosis and drug recommendations.

**Significance**:

This project aims to streamline the medical diagnosis process by integrating cutting-edge machine learning models, potentially reducing the time and effort required by healthcare professionals in diagnosing illnesses and prescribing treatments. The system's automated approach can assist in early detection and timely treatment, ultimately improving patient outcomes.

**Expected Outcomes:**

- A functional prototype of the integrated machine learning system.

- Enhanced accuracy and efficiency in symptom extraction, disease prediction, and drug recommendation.

- A scalable and user-friendly application that can be utilized in real-world healthcare settings.

This project not only serves as an educational endeavor in applying machine learning techniques but also as a step towards practical, impactful solutions in the healthcare industry.

## Approach and Methodology

To develop an integrated machine learning application for symptom extraction, disease prediction, and drug recommendation, addressing the project's requirements through a systematic and data-driven approach.

1. **Symptom Extraction Model**

Approach:

Utilize an open-source large language model from OpenAI for natural language processing (NLP).

Preprocess patient input data to clean and normalize text, removing noise and irrelevant information.

Methodology:

Data Preprocessing: Tokenization, lemmatization, and stop-word removal to prepare text data for analysis.

Model Selection: Fine-tune a pre-trained large language model (e.g., GPT-3) to recognize and extract symptoms from patient descriptions.

Training: Use annotated medical texts to train the model, focusing on identifying medical symptoms and related terminology.

Validation: Perform cross-validation with a separate set of annotated texts to ensure accuracy and reliability of symptom extraction.

2. **Disease Prediction Model**

Approach:

Leverage a supervised machine learning model to classify diseases based on extracted symptoms.

Use the dataset from Kaggle (https://www.kaggle.com/datasets/kaushil268/disease-prediction-using-machine-learning?select=Training.csv) for training and validation.

Methodology:

Data Exploration and Preprocessing: Analyze and clean the dataset to handle missing values and normalize symptom data.

Feature Engineering: Transform extracted symptoms into feature vectors suitable for model input.

Model Selection: Experiment with various classification algorithms (e.g., Decision Trees, Random Forests, Support Vector Machines) to identify the best-performing model.

Training and Tuning: Train the model on the training dataset and optimize hyperparameters to improve performance.

Evaluation: Validate the model using a holdout validation set and metrics such as accuracy, precision, recall, and F1-score to assess predictive capability.

3**. Drug Recommendation Model**

Approach:

Implement a recommendation system to suggest appropriate drugs based on the predicted disease.

Utilize the dataset from Kaggle (https://www.kaggle.com/datasets/jithinanievarghese/drugs-side-effects-and-medical-condition) for drug information, side effects, and medical conditions.

Methodology:

Data Exploration and Preprocessing: Clean and preprocess the drug dataset to ensure compatibility with the predicted disease output.

Feature Engineering: Develop a mapping between diseases and potential drug treatments, considering factors like side effects and contraindications.

Model Selection: Use collaborative filtering or content-based filtering techniques to build the recommendation system.

Training and Tuning: Train the recommendation model using the preprocessed dataset and fine-tune parameters to enhance recommendation accuracy.

Evaluation: Test the recommendation system with known disease-drug pairs and assess performance based on precision, recall, and user satisfaction metrics.

3. **Deployment**

Approach:

Deploy the integrated application in a containerized environment using Docker.

Develop a Flask API to facilitate interaction between users and the models.

Methodology:

Containerization: Package all components (models, preprocessing scripts, Flask API) into Docker containers to ensure consistency across different environments.

API Development: Implement RESTful endpoints in Flask for symptom extraction, disease prediction, and drug recommendation.

Integration Testing: Perform end-to-end testing of the application to verify the seamless operation of all components.

Scalability and Maintenance: Ensure the application can scale as needed and establish protocols for regular updates and maintenance.

## Timeline and Deliverables

**Project Duration: May 20th - August 13th**

**Phase 1**: Project Initiation and Data Preparation (May 20th - June 2nd)

* May 20th - May 22nd: Project Kickoff
  + Define project objectives, scope, and methodology.
  + Set up project management tools and repository (GitHub).
* May 23rd - May 29th: Data Collection and Exploration
  + Collect datasets from Kaggle.
  + Perform initial data exploration to understand the structure and content.
* May 30th - June 2nd: Data Cleaning and Preprocessing
  + Clean and preprocess datasets to handle missing values and normalize data.
  + Prepare text data for symptom extraction (tokenization, lemmatization, etc.).

**Deliverables:**

Initial data exploration report

Cleaned and preprocessed datasets

**Milestone Report 1**

**Phase 2**: Model Development (June 3rd - July 7th)

* June 3rd - June 12th: Symptom Extraction Model
  + Fine-tune the OpenAI large language model for symptom extraction.
  + Train and validate the model using annotated medical texts.
  + Evaluate model performance and refine as necessary.

**Deliverables**:

Trained Symptom Extraction Model

**Milestone Report 2 (June 12th)**

* June 13th - June 22nd: Disease Prediction Model
  + Perform feature engineering on symptoms data.
  + Train multiple classification algorithms and select the best model.
  + Validate the model using the disease prediction dataset.
  + Optimize model hyperparameters.

**Deliverables**:

Trained Disease Prediction Model

**Milestone Report 3 (June 22nd)**

* June 23rd - July 2nd: Drug Recommendation Model
* Develop a mapping between diseases and drug treatments.
* Train a recommendation system using collaborative/content-based filtering.
* Validate the recommendation system with known disease-drug pairs.

**Deliverables**:

Trained Drug Recommendation Model

**Milestone Report 4 (July 2nd)**

**Phase 3: Integration and Testing (July 8th - July 27th)**

* July 8th - July 15th: Model Integration
* Integrate all three models into a cohesive system.
* Develop a Flask API to enable interaction with the models.

**Deliverables**:

Integrated application with API

**Milestone Report 5 (July 15th)**

* July 16th - July 20th: Containerization
  + Package the application using Docker to ensure consistent deployment.
* July 21st - July 27th: End-to-End Testing
* Conduct thorough testing to ensure seamless operation of all components.
* Perform scalability testing and resolve any issues.

**Deliverables**:

Docker container setup

End-to-end testing report

**Phase 4: Analysis, Refinement, and Documentation (July 28th - August 7th)**

* July 28th - August 1st: Result Analysis
* Analyze the performance metrics of each model (accuracy, precision, recall, F1-score).
* Evaluate the strengths and weaknesses of each model.
* August 2nd - August 7th: Documentation
* Prepare the final report documenting the methodology, results, and analysis.
* Compile interpretation of results and insights gained.

**Deliverables**:

Result metrics and analysis report

Final Project Report (August 7th)

**Phase 5: Final Presentation Preparation (August 8th - August 13th)**

* August 8th - August 10th: Presentation Development
* Create presentation slides summarizing the project objectives, methodology, results, and insights.
* Develop a demo to showcase the application's functionality.
* August 11th - August 12th: Presentation Rehearsal
* Practice the presentation and demo to ensure smooth delivery.
* August 13th: Final Presentation and Demo
* Present the project and demonstrate the application to the course instructor and peers.

**Deliverables**:

Final presentation slides

Application demo

GitHub repository with all project code and documentation

## Budget and Pricing

Given that our project relies on free and open-source resources, the direct financial costs associated with the development and deployment of the Intelligent Medical Diagnosis and Drug Recommendation System are minimal. However, it is important to outline the elements we will be using and justify the cost-efficiency of our approach.

**Resources and Cost Efficiency**

1. OpenAI Large Language Model:

Cost: Free (using OpenAI's freely available models and research access)

Justification: Leveraging OpenAI's state-of-the-art language models for symptom extraction ensures high accuracy without incurring licensing fees.

1. Kaggle Datasets:

Disease Prediction Dataset: Kaggle Disease Prediction Dataset

Drug Recommendation Dataset: Kaggle Drug Dataset

Cost: Free

Justification: Using publicly available datasets from Kaggle provides a rich source of training data for both disease prediction and drug recommendation models without any associated costs.

1. Development Tools:

Programming Languages: Python (free and open-source)

Frameworks and Libraries: TensorFlow, PyTorch, Scikit-learn, NLTK, SpaCy (all free and open-source)

Integrated Development Environment (IDE): Visual Studio Code, Jupyter Notebook (both free and open-source)

1. Deployment:

Containerization: Docker (free version)

Web Framework: Flask (free and open-source)

1. Project Management and Collaboration:

Version Control: Git and GitHub (free for public repositories)

Communication and Collaboration: Slack, Zoom (using free versions or educational licenses)

**Indirect Costs**

While there are no direct financial expenses, the project does entail indirect costs related to time and effort:

1. Time Investment:

Estimated at approximately 12 weeks of development, testing, and documentation.

1. Manpower:

Team Members: 2-4 members dedicating part-time effort throughout the project duration.

1. Infrastructure:

Hardware: Personal computers/laptops (assuming availability, no additional cost).

Internet: Reliable internet connection (assumed available, no additional cost).

**Summary**

Total Direct Costs: $0 (utilizing free and open-source resources)

Indirect Costs:

* Time and manpower invested by team members
* Utilization of existing hardware and internet infrastructure

## Qualifications and Experience

Our team comprises a diverse group of professionals with a blend of expertise in software engineering, economics, and mathematics. This multidisciplinary background uniquely positions us to tackle the complex challenges involved in developing an intelligent medical diagnosis and drug recommendation system. Here is an overview of our qualifications and relevant experience:

**Team Composition and Expertise**

Software Engineering:

Qualifications: Bachelor's and Master's degrees in Computer Science, Software Engineering, and related fields.

Experience:

* Extensive experience in full-stack development, including backend (Python, Flask) and frontend (JavaScript, React).
* Proficient in containerization and deployment using Docker and Kubernetes.
* Hands-on experience with version control systems like Git and collaboration platforms such as GitHub.
* Developed and deployed multiple web applications and APIs, ensuring scalability and security.

Economics:

Qualifications: Bachelor's and Master's degrees in Economics.

Experience:

* Strong analytical skills and experience in data analysis, economic modeling, and statistical analysis.
* Expertise in econometrics and predictive modeling, which are crucial for understanding and interpreting large datasets.
* Experience with software tools and libraries for data analysis, including R and Python (Pandas, NumPy).

Mathematics:

Qualifications: Bachelor's and Master's degrees in Mathematics.

Experience:

* In-depth knowledge of mathematical concepts and theories, which are fundamental to machine learning algorithms.
* Experience in mathematical modeling and optimization techniques.
* Proficiency in programming languages such as Python and MATLAB for implementing complex algorithms.
* Relevant Experience in Machine Learning

Natural Language Processing (NLP):

* Developed NLP models for text analysis and sentiment analysis using libraries such as NLTK, SpaCy, and transformers from Hugging Face.
* Implemented text preprocessing techniques and trained models for entity recognition and text classification tasks.

Classification and Prediction Models:

* Experience with supervised learning algorithms (e.g., Decision Trees, Random Forests, Support Vector Machines) for classification tasks.
* Conducted feature engineering and hyperparameter tuning to optimize model performance.
* Hands-on experience with Scikit-learn, TensorFlow, and PyTorch for building and training machine learning models.

Recommendation Systems:

* Developed collaborative filtering and content-based filtering recommendation systems.
* Used machine learning algorithms to predict user preferences and recommend products/services.
* Experience with handling large datasets and implementing efficient recommendation algorithms.

Previous Projects

* Healthcare Analytics: Worked on a project to predict patient readmission rates using logistic regression and random forests, significantly improving hospital resource allocation.
* Economic Forecasting: Developed models to forecast economic indicators such as GDP growth and inflation rates, providing valuable insights for policy-making.
* E-commerce Personalization: Implemented a recommendation system for an e-commerce platform, enhancing user experience and increasing sales conversion rates.

## Team Members

Mahmood Hossain : Mahmood has experience in DevOps, MLOps, Big Data and AI. He has developed and deployed various projects for on premises and for cloud deployment. As a graduate in Economics, Mahmood started working in advertising agencies, media and then in machine learning domain.

Member

Member

Member

Member

## Resources

Below are some past projects and case studies that showcase our team's successful work in software engineering, machine learning, and data analysis. These examples demonstrate our ability to deliver high-quality, impactful solutions across various domains.

1. **Healthcare Analytics Project**

Project Title: Patient Readmission Rate Prediction

Description:

Developed a predictive model to forecast patient readmission rates within 30 days of discharge.

Utilized logistic regression and random forest algorithms to analyze patient data, including medical history, treatment plans, and demographic information.

Implemented feature engineering to identify key predictors of readmission.

Outcome:

Achieved a significant improvement in prediction accuracy, allowing hospitals to better allocate resources and reduce readmission rates.

Provided actionable insights that helped healthcare providers enhance patient care and follow-up procedures.

2. **Economic Forecasting Model**

Project Title: GDP Growth and Inflation Rate Forecasting

Description:

Created predictive models to forecast key economic indicators such as GDP growth and inflation rates.

Used time series analysis and econometric modeling techniques to analyze historical economic data.

Applied machine learning algorithms to enhance the accuracy of forecasts.

Outcome:

Delivered accurate and reliable forecasts that informed policy-making and economic planning.

The models were used by governmental agencies to anticipate economic trends and make data-driven decisions.

3. **E-commerce Personalization System**

Project Title: Personalized Product Recommendation Engine

Description:

Developed a recommendation system for an e-commerce platform to enhance user experience by suggesting relevant products.

Implemented collaborative filtering and content-based filtering techniques to predict user preferences.

Integrated the recommendation engine with the platform’s existing infrastructure.

Outcome:

Improved user engagement and satisfaction by providing personalized product suggestions.

Increased sales conversion rates and average order value, contributing to higher revenue for the e-commerce platform.

4. **Sentiment Analysis Tool**

Project Title: Social Media Sentiment Analysis

Description:

Designed and implemented an NLP model to analyze sentiment from social media posts.

Utilized pre-trained transformers (e.g., BERT) to classify posts into positive, negative, and neutral sentiment categories.

Developed a dashboard for real-time sentiment tracking and visualization.

Outcome:

Enabled businesses to monitor and respond to customer sentiment effectively.

Provided valuable insights into customer opinions and brand perception, driving informed marketing strategies.

5. **Financial Risk Assessment**

Project Title: Credit Risk Prediction Model

Description:

Created a machine learning model to assess the credit risk of loan applicants.

Employed algorithms such as logistic regression, decision trees, and gradient boosting to predict default probabilities.

Conducted extensive data preprocessing and feature selection to enhance model performance.

Outcome:

Achieved high prediction accuracy, reducing the risk of defaults and improving the decision-making process for loan approvals.

Helped financial institutions manage risk more effectively and maintain a healthier loan portfolio.

## Risk Assessment and Mitigation

**1. Data Quality and Availability**

Risk: The datasets from Kaggle may have missing values, inconsistencies, or may not be comprehensive enough for training robust models.

Mitigation:

Data Cleaning: Implement thorough data cleaning processes to handle missing values and correct inconsistencies.

Data Augmentation: Use data augmentation techniques to enhance the datasets, such as generating synthetic data or sourcing additional datasets.

Continuous Monitoring: Regularly monitor the data quality throughout the project to identify and address any emerging issues promptly.

2**. Model Performance and Accuracy**

Risk: The machine learning models may not achieve the desired accuracy or performance levels, impacting the reliability of the symptom extraction, disease prediction, and drug recommendation.

Mitigation:

Iterative Model Development: Use an iterative approach to model development, continually refining and improving models based on performance metrics.

Hyperparameter Tuning: Employ techniques such as grid search and randomized search to optimize model hyperparameters.

Cross-validation: Use cross-validation to ensure models generalize well to unseen data.

3. **Integration Challenges**

Risk: Integrating multiple models (symptom extraction, disease prediction, and drug recommendation) into a cohesive system may present technical challenges.

Mitigation:

Modular Development: Develop models as independent modules with well-defined interfaces to facilitate easier integration.

Regular Integration Testing: Conduct integration testing at each development phase to identify and resolve issues early.

Clear Documentation: Maintain detailed documentation for each model and the integration process to ensure clarity and consistency.

**4. Computational Resources**

Risk: The computational requirements for training and deploying machine learning models may exceed available resources, leading to delays.

Mitigation:

Resource Planning: Estimate computational requirements at the project outset and plan resource allocation accordingly.

Cloud Services: Utilize cloud-based services such as AWS, Google Cloud, or Azure for scalable compute resources.

Efficient Algorithms: Optimize code and use efficient algorithms to reduce computational overhead.

**5. Security and Privacy**

Risk: Handling medical data and patient information entails risks related to data security and privacy compliance.

Mitigation:

Data Anonymization: Ensure all patient data is anonymized to protect privacy.

Security Measures: Implement robust security measures, including encryption and secure access controls.

Compliance: Adhere to relevant data protection regulations such as GDPR and HIPAA.

**6. User Adoption and Usability**

Risk: The end-users (e.g., healthcare providers) may face challenges in adopting the system if it is not user-friendly.

Mitigation:

User-Centered Design: Involve end-users in the design process to ensure the system meets their needs and is intuitive to use.

Training and Support: Provide comprehensive training and support materials to facilitate user adoption.

Feedback Loop: Establish a feedback mechanism to gather user input and continuously improve the system.

**7. Project Timeline and Deliverables**

Risk: The project may face delays due to unforeseen challenges, impacting the delivery timeline.

Mitigation:

Realistic Scheduling: Develop a realistic project schedule with buffer time for unexpected delays.

Regular Updates: Conduct regular progress reviews and update the schedule as needed.

Milestone Reports: Prepare milestone reports to track progress and address any issues promptly.

## Terms and Conditions

**1. Scope of Work**

Project Deliverables: The deliverables for this project include milestone reports, a final report, a GitHub repository with all project code, result metrics, an analysis of the strengths and weaknesses of the models, interpretation of the results, and a final presentation with a demo.

Project Phases: The project will be completed in five phases: Project Initiation and Data Preparation, Model Development, Integration and Testing, Analysis, Refinement, and Documentation, and Final Presentation Preparation.

**2. Payment Terms**

Cost: As the project utilizes free and open-source tools and datasets, there are no direct costs associated with the development and deployment of the system.

Compensation for Time and Effort: Any agreed-upon compensation for the team's time and effort will be outlined in a separate agreement, if applicable.

**3. Intellectual Property**

Ownership: All intellectual property developed during this project, including source code, models, and documentation, will be owned by the proposing team, unless otherwise agreed upon in writing.

Licensing: Any use of the project's outputs will be subject to the licensing terms agreed upon by both parties.

**4. Confidentiality**

Data Privacy: All data used in this project will be handled with the utmost confidentiality. Patient data will be anonymized to protect privacy.

Non-Disclosure: Both parties agree to keep all project-related information confidential and not to disclose any proprietary information to third parties without prior written consent.

**5. Liability**

Limitation of Liability: The proposing team shall not be held liable for any indirect, incidental, or consequential damages arising from the use of the project deliverables.

Warranty Disclaimer: The project deliverables are provided "as is" without any warranties, express or implied.

**6. Amendments**

Modifications: Any amendments or modifications to this agreement must be made in writing and signed by both parties.

**7. Governing Law**

Jurisdiction: This agreement shall be governed by and construed in accordance with the laws of the jurisdiction in which the proposing team is based.

**8. Dispute Resolution**

Mediation and Arbitration: Any disputes arising out of or in connection with this agreement shall first be attempted to be resolved through mediation. If mediation fails, the dispute shall be settled by binding arbitration.

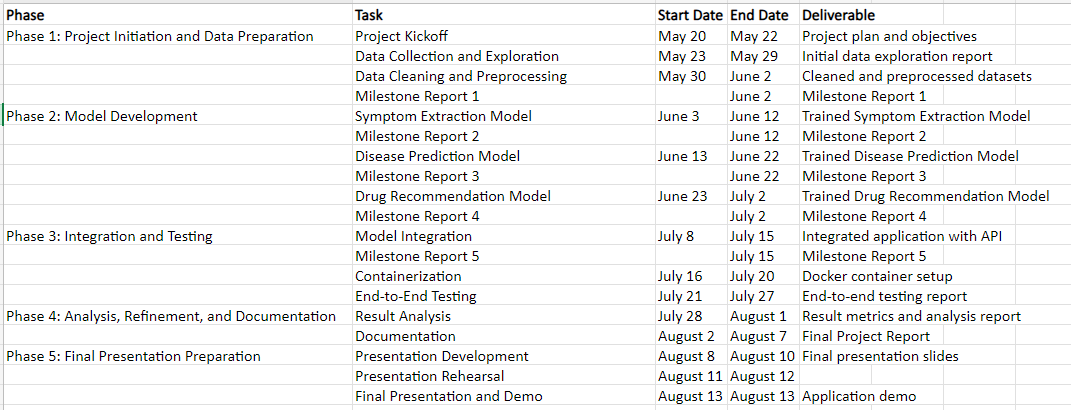
**9. Force Majeure**

Unforeseen Events: Neither party shall be liable for any delay or failure to perform its obligations under this agreement due to causes beyond its reasonable control, including but not limited to acts of God, war, riots, natural disasters, and governmental actions.

## Appendices

**Appendix A: Project Timeline and Gantt Chart**

**Project Duration: May 20th - August 13th**



**Appendix B: System Architecture Diagram**

**Description:**

Input Layer: Patient illness description.

Symptom Extraction Model: Utilizes OpenAI’s large language model to extract symptoms from the patient description.

Disease Prediction Model: Classifies the extracted symptoms to predict the disease using the Kaggle disease prediction dataset.

Drug Recommendation Model: Suggests appropriate drugs for the diagnosed disease using the Kaggle drug dataset.

API Layer: A Flask API to integrate all models and provide a user-friendly interface for healthcare providers.

Containerization: Docker is used to ensure consistent deployment across different environments.

**Appendix C: Data Flow Diagram**

Description:

Step 1: Patient inputs illness description.

Step 2: The Symptom Extraction Model processes the text and extracts symptoms.

Step 3: Extracted symptoms are fed into the Disease Prediction Model.

Step 4: The Disease Prediction Model outputs a probable disease.

Step 5: The predicted disease is input into the Drug Recommendation Model.

Step 6: The Drug Recommendation Model suggests appropriate drugs.

Step 7: Results are returned to the user through the Flask API.

**Appendix D: Glossary of Terms**

API (Application Programming Interface): A set of protocols for building and interacting with software applications.

Docker: A platform for developing, shipping, and running applications in containers.

Flask: A lightweight web framework for Python.

Hyperparameter Tuning: The process of adjusting model parameters to improve performance.

NLP (Natural Language Processing): A field of AI that focuses on the interaction between computers and human language.

Scikit-learn: A machine learning library for Python.

TensorFlow and PyTorch: Open-source deep learning frameworks.