

# **Drug Recommendation System using GPT**

*Submitted in partial fulfillment of the requirements for the degree of*

## **Bachelor of Technology** in **Computer Science Engineering with Specialization in Bioinformatics**

*by*

**Edwin Joshua Samraj**

**20BCB0055**

**Under the guidance of  
Prof. Swarnalatha P**

**School of Computer Science and Engineering,  
VIT, Vellore.**



May, 2024

## **DECLARATION**

I hereby declare that the thesis entitled “**Drug Recommendation System using GPT**” submitted by me, for the award of the degree of *Bachelor of Technology in Computer Science Engineering with Specialization in Bioinformatics* to VIT is a record of bonafide work carried out by me under the supervision of Prof. Swarnalatha P.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place : Vellore

Date : 8-5-24

**Signature of the Candidate**

## **CERTIFICATE**

This is to certify that the thesis entitled “**Drug Recommendation System using GPT**” submitted by **Edwin Joshua Samraj & 20BCB0055**, **School of Computer Science and Engineering**, VIT, for the award of the degree of *Bachelor of Technology in Computer Science Engineering with Specialization in Bioinformatics*, is a record of bonafide work carried out by him / her under my supervision during the period, 01. 12. 2023 to 30.04.2024, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place : Vellore

Date : 8-5-24

**Signature of the Guide**

**Internal Examiner**

**External Examiner**

**Dr. Rajkumar S**  
**HoD - B.Tech. CSE (Bioinformatics) And B.Tech. CSE & Business Systems**  
**Department of Analytics**  
**School of Computer Science and Engineering (SCOPE)**

## **ACKNOWLEDGEMENTS**

We extend our sincere appreciation to all individuals and organizations who contributed to the realization of this research endeavor.

First and foremost, we express our deepest gratitude to Prof. Swarnalatha P, for their invaluable guidance, encouragement, and support throughout the duration of this project. Their expertise and insightful feedback have been instrumental in shaping the direction and outcomes of our research.

Our appreciation extends to the anonymous reviewers whose constructive feedback and suggestions helped strengthen the rigor and clarity of our research paper.

Furthermore, we acknowledge Vellore Institute of Technology for providing the necessary resources and infrastructure that facilitated the execution of this research.

Lastly, we would like to express our heartfelt gratitude to our families and friends for their unwavering support and understanding throughout this journey.

This research would not have been possible without the collective efforts and contributions of all individuals involved. Thank you.

**Edwin Joshua Samraj**

## **Executive Summary**

The paper introduces a novel drug recommendation system leveraging advanced NLP models like GPT and BioBERT integrated with Flask. Motivated by the complexity of medical data and the need for precise recommendations, the system aims to aid healthcare professionals in decision-making. By addressing gaps in existing systems through enhanced data processing and user-friendly interfaces, the project bridges the disparity between healthcare needs and available technology. Technical specifications ensure scalability, reliability, and security, with a feasibility study covering technical, economic, and social aspects. Design details include a layered architecture and class diagrams, highlighting system components and interactions. Constraints, alternatives, and trade-offs are discussed, offering solutions for potential challenges. A detailed schedule outlines tasks and milestones, ensuring systematic development and testing. Results showcase model effectiveness through evaluation metrics, paving the way for future enhancements. The executive summary encapsulates the project's significance, emphasizing its contribution to personalized healthcare management and informed decision-making.

<b>CONTENTS</b>		<b>Page No.</b>
	<b>Acknowledgement</b>	<b>i.</b>
	<b>Executive Summary</b>	<b>ii.</b>
	<b>Table of Contents</b>	<b>iii.</b>
	<b>List of Figures</b>	<b>iv.</b>
	<b>List of Tables</b>	<b>v.</b>
	<b>Abbreviations</b>	<b>vi.</b>
	<b>Symbols and Notations</b>	<b>vii.</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 Objectives	1
	1.2 Motivation	2
	1.3 Background	3
<b>2</b>	<b>PROJECT DESCRIPTION AND GOALS</b>	<b>4</b>
	2.1 Survey on Existing System	4
	2.2 Research Gap	5
	2.3 Problem Statement	6
<b>3</b>	<b>TECHNICAL SPECIFICATION</b>	<b>7</b>
	3.1 Requirements 3.1.1 Functional 3.1.2 Non-Functional	
	3.2 Feasibility Study 3.2.1 Technical Feasibility 3.2.2 Economic Feasibility 3.2.3 Social Feasibility	
	3.3 System Specification 3.3.1 Hardware Specification 3.3.2 Software Specification 3.3.3 Standards and Policies	
<b>4</b>	<b>DESIGN APPROACH AND DETAILS</b>	
	4.1 System Architecture	

	4.2 Design 4.2.1 Class Diagram	
	4.3 Constraints, Alternatives and Tradeoffs	
<b>5</b>	<b>SCHEDULE, TASKS AND MILESTONES</b>	
	5.1 Gantt Chart	
	5.2 Module Description 5.2.1 Module -GPT and BioBERT Implementation 5.2.2 Module - Flask API Development 5.2.3 Module - Integration and Testing	
	5.3 Testing 5.3.1 Unit Testing 5.3.2 Integration Testing	
<b>6</b>	<b>PROJECT DEMONSTRATION</b>	
<b>7</b>	<b>RESULT &amp; DISCUSSION</b>	
<b>8</b>	<b>SUMMARY</b>	
<b>9</b>	<b>REFERENCES</b>	
	<b>APPENDIX A – SAMPLE CODE</b>	

## **List of Figures**

<b>Figure No.</b>	<b>Captions</b>	<b>Page No.</b>
1.1		15
1.2	Part-1 of Class diagram	16
1.3	Part-2 of Class diagram	17
1.4	Part-3 of Class diagram	17
2.1		19
2.2	Unit Testing done using 'pytest'	20
2.3	Integration Testing done on -	21
3.1	Jupyter Notebook	21
3.2		22
3.3		23
3.4		24



## **List of Tables**

<b>Table No.</b>	<b>Title</b>	<b>Page No.</b>
1.1	Gantt Chart	19

## List of Abbreviations

GPT	Generative Pre-trained Transformer
BioBERT	Bidirectional Encoder Representations from Transformers
NLP	Natural Language Processing

# 1. INTRODUCTION

## 1.1. OBJECTIVE

The objective of this paper is to develop a robust and efficient drug recommendation system leveraging state-of-the-art natural language processing (NLP) models such as GPT (Generative Pre-trained Transformer) and BioBERT (Bidirectional Encoder Representations from Transformers). By integrating these models into a web application using Flask, we aim to provide healthcare professionals with a user-friendly platform for generating personalized drug recommendations based on patient information and medical histories.

## 1.2 MOTIVATION

With the exponential growth of medical literature and the increasing complexity of healthcare data, there is a pressing need for intelligent systems that can assist healthcare providers in making informed decisions about drug prescriptions. Traditional drug recommendation methods often rely on manual literature review or simplistic rule-based algorithms, which may lack accuracy and scalability. By harnessing the power of advanced NLP models trained on biomedical text data, we seek to enhance the precision and efficiency of drug recommendation processes, ultimately improving patient outcomes and reducing healthcare costs.

## 1.3 BACKGROUND

Drug recommendation plays a pivotal role in modern healthcare by helping clinicians select the most appropriate medications for individual patients based on factors such as medical history, symptoms, genetic predispositions, and drug interactions. However, the sheer volume and complexity of biomedical literature make it challenging for healthcare professionals to stay up-to-date with the latest research and guidelines. Moreover, the one-size-fits-all approach of many existing drug recommendation systems may overlook subtle nuances in patient profiles, leading to suboptimal treatment outcomes. By leveraging advanced NLP techniques and web-based technologies, we aim to address these challenges and pave the way for more personalized and data-driven drug recommendations.

## **2. PROJECT DESCRIPTION AND GOALS**

### **2.1. SURVEY OF EXISTING SYSTEMS**

In previous research, the significance of health-related user-generated content has been widely recognized. Pioneering work by Jane Sarasohn-Kahn highlighted the importance of individuals seeking narratives from “patients like them” online, a resource often scarce within their immediate social circles [1]. Over the past decade, scholars have delved into the emotional impact of user experiences and the severity of adverse drug reactions through sentiment and semantic analysis [2]. However, despite these efforts, extensive exploration within the medical and health domain has been hindered by concerns regarding the trustworthiness and quality of user-expressed medical language.

Consequently, there exists a notable gap in research pertaining to the extraction of insights into patients’ health conditions, including symptom analysis, drug recommendations, and drug side effects. This project endeavors to address these limitations by conducting a thorough survey of existing systems, aiming to provide enhanced understanding and actionable insights into patient health profiles.

### **2.2 RESEARCH GAP**

Despite the progress made in drug recommendation research, there remains a significant gap between the capabilities of existing systems and the evolving needs of healthcare providers. Many current approaches rely on structured data sources and simplistic algorithms, overlooking the rich contextual information available in unstructured biomedical text. Additionally, the lack of user-friendly interfaces hinders the adoption of advanced recommendation models in real-world clinical settings. Our project aims to bridge this gap by developing a novel drug recommendation system that harnesses the power of advanced NLP models while providing an intuitive web interface for healthcare professionals.

### **2.3 PROBLEM STATEMENT**

The primary challenge addressed by this project is the need for a more accurate, scalable, and user-friendly drug recommendation system in healthcare. Existing systems often struggle to incorporate unstructured textual data from sources such as electronic health records and medical literature, limiting their ability to generate personalized recommendations based on nuanced patient profiles. Moreover, the complexity of biomedical text presents unique

challenges for NLP models, requiring specialized architectures and domain-specific knowledge. Our goal is to design and implement a system that overcomes these challenges, empowering healthcare providers with timely and personalized drug recommendations to improve patient care outcomes.

### 3. TECHNICAL SPECIFICATIONS

#### 3.1 REQUIREMENTS

##### 3.1.1 FUNCTIONAL

- Integration of GPT, BioBERT, and Flask frameworks to enable seamless drug recommendation functionality.
- Development of an intuitive user interface for inputting symptoms and receiving drug recommendations.
- Implementation of a recommendation algorithm capable of processing user inputs and generating personalized drug suggestions.
- Incorporation of a database to store and retrieve drug information as well as user interactions.

##### 3.1.2 NON-FUNCTIONAL

- **Scalability:** The system should accommodate a growing user base and an expanding database of drugs and symptoms.
- **Reliability:** Ensuring consistent and accurate drug recommendations, minimizing downtime, and handling errors gracefully.
- **Security:** Implementing measures to protect user data privacy and prevent unauthorized access to sensitive information.
- **Performance:** Optimizing system response time and resource utilization to provide a seamless user experience.
- **Usability:** Designing an intuitive and user-friendly interface accessible to individuals with varying levels of technical expertise.

#### 3.2 FEASIBILITY STUDY

##### 3.2.1 TECHNICAL FEASIBILITY

- Assessment of the technical resources required for implementing and maintaining the system, including hardware, software, and development expertise.
- Evaluation of the compatibility and interoperability of GPT, BioBERT, and Flask

frameworks to ensure seamless integration.

- Identification of potential technical challenges and mitigation strategies to address them effectively.

### 3.2.2 ECONOMIC FEASIBILITY

- Estimation of the development and operational costs associated with building and maintaining the drug recommendation system.
- Analysis of the potential return on investment (ROI) and cost-benefit ratio to assess the economic viability of the project.
- Exploration of alternative funding sources and revenue models to sustain the system in the long term.

### 3.2.3 SOCIAL FEASIBILITY

- Examination of the societal impact of the drug recommendation system, including its potential to improve healthcare accessibility and outcomes.
- Consideration of ethical implications related to data privacy, algorithm transparency, and fairness in drug recommendations.
- Engagement with stakeholders, including healthcare professionals and patient advocacy groups, to gather feedback and address concerns regarding the system's implementation and use.

## 22.1 SYSTEM SPECIFICATION

### 3.3.1 HARDWARE SPECIFICATION

- Deployment of servers or cloud infrastructure capable of hosting the drug recommendation system and supporting its computational requirements.
- Provisioning of adequate storage capacity to accommodate the system's database of drugs, symptoms, and user interactions.
- Selection of hardware components based on performance, reliability, and scalability considerations.

### 3.3.2 SOFTWARE SPECIFICATION

- Utilization of programming languages such as Python for backend development and JavaScript for frontend implementation.
- Integration of GPT and BioBERT models using relevant libraries and frameworks (e.g., Hugging Face Transformers).
- Deployment of Flask as the web framework for building the user interface and handling HTTP requests.

- Implementation of database management systems like PostgreSQL or MongoDB to store and retrieve relevant data.

### 3.3.3 STANDARDS AND POLICIES

- Adherence to relevant industry standards and regulations governing healthcare data privacy and security (e.g., HIPAA compliance).
- Establishment of policies and procedures for user consent, data protection, and responsible use of the drug recommendation system.
- Documentation of system architecture, algorithms, and APIs to facilitate transparency, reproducibility, and collaboration among stakeholders.

## 4. DESIGN APPROACH AND DETAILS

### 4.1 SYSTEM ARCHITECTURE

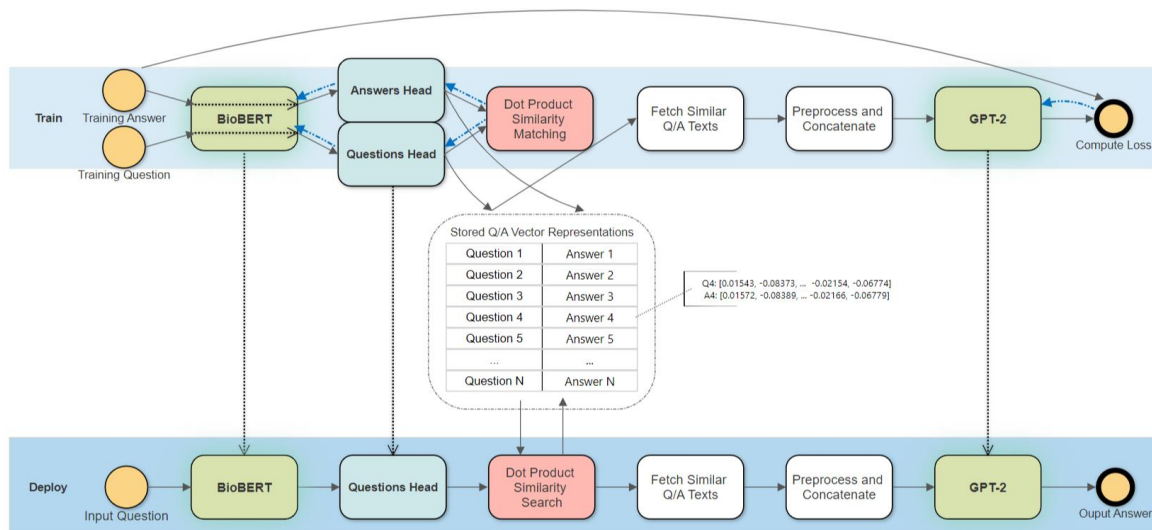


Figure 1.1

The system architecture for the medical chatbot leveraging TensorFlow, Flask, and Hugging Face Transformers is designed to facilitate efficient processing of user queries, generation of drug recommendations, and seamless interaction with the end user. The architecture comprises several components, including natural language processing models, tokenizer libraries, and database management systems.

- **Presentation Layer:** The presentation layer is implemented using Flask, a user-friendly, micro web framework for Python. Flask provides an intuitive interface for users to input medical symptoms and receive responses from the chatbot. The user interface enhances user experience by offering real-time interaction and visual feedback.

- **Application Layer:** At the core of the architecture lies the application layer, responsible for processing user queries, extracting relevant information, and generating responses. TensorFlow is utilized for deep learning tasks, such as question embedding extraction using BioBERT and text generation using GPT-2. The application layer orchestrates the integration of different models and libraries to ensure accurate and timely responses to user queries.
- **Data Layer:** The data layer encompasses the management of structured and unstructured data used by the system. This includes biomedical question-answer pairs stored in a pandas DataFrame and pre-trained embeddings of questions and answers. The data layer ensures efficient access, retrieval, and manipulation of data to support various functionalities of the medical chatbot.

## 4.2 DESIGN

### 4.2.1 CLASS DIAGRAM

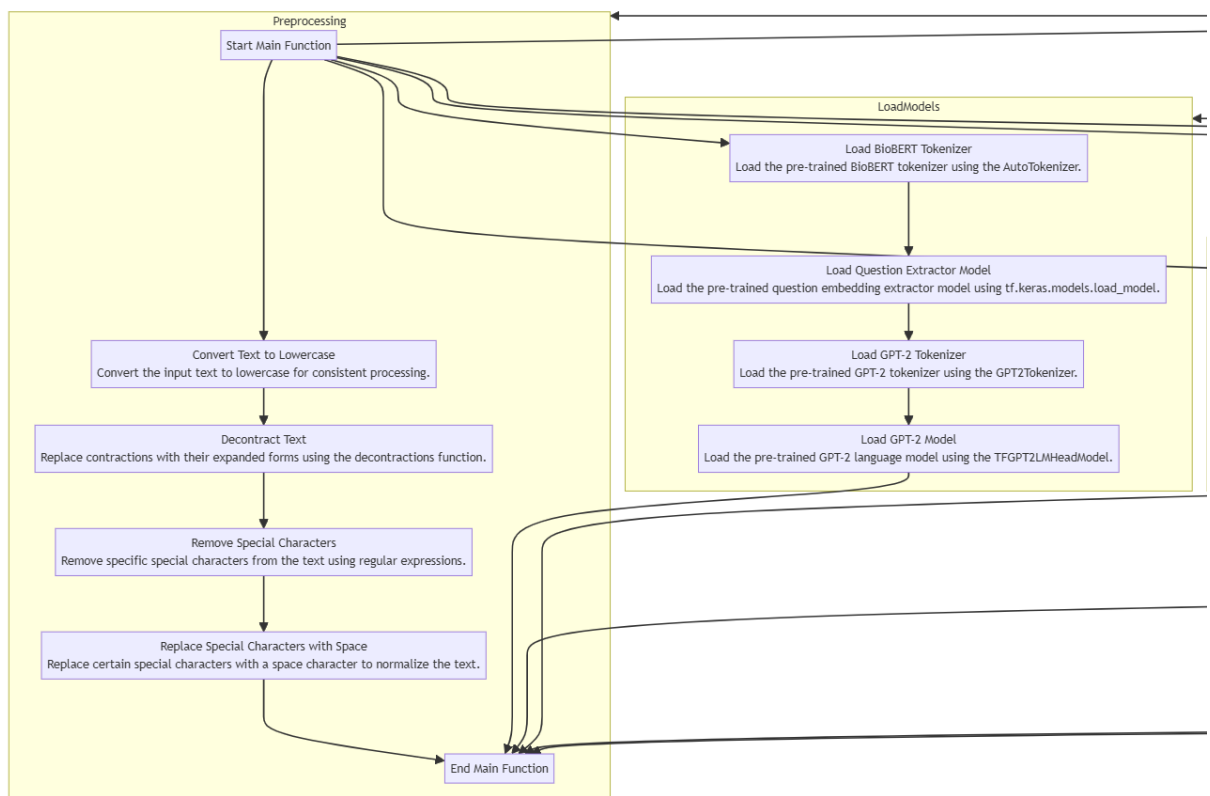


Figure 1.2 – Part-1 of Class diagram



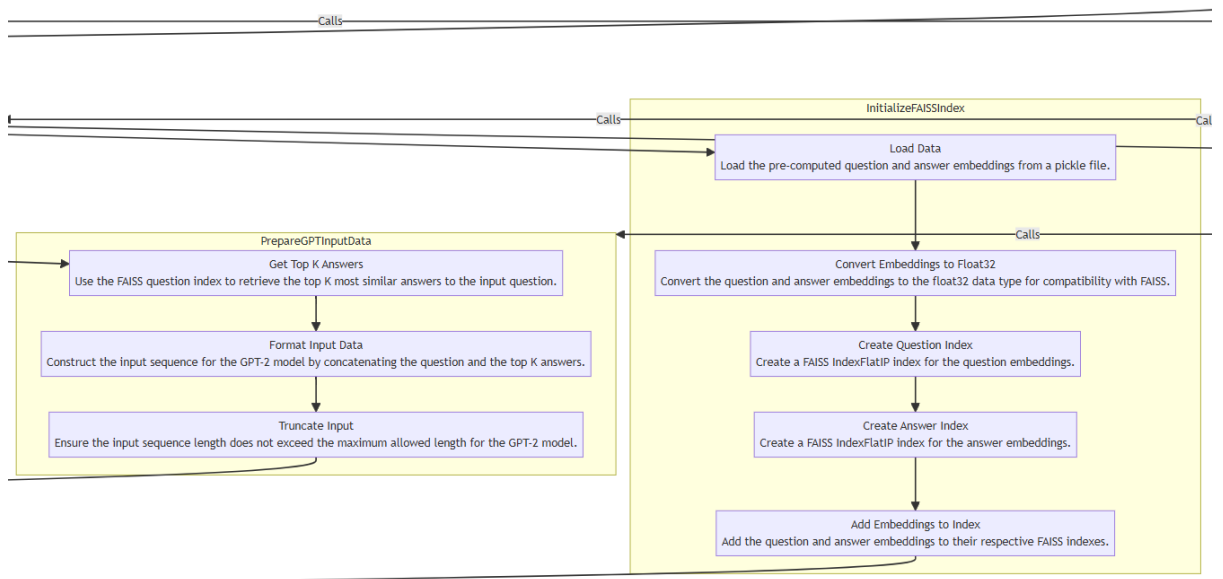


Figure 1.3 – Part-2 of Class diagram

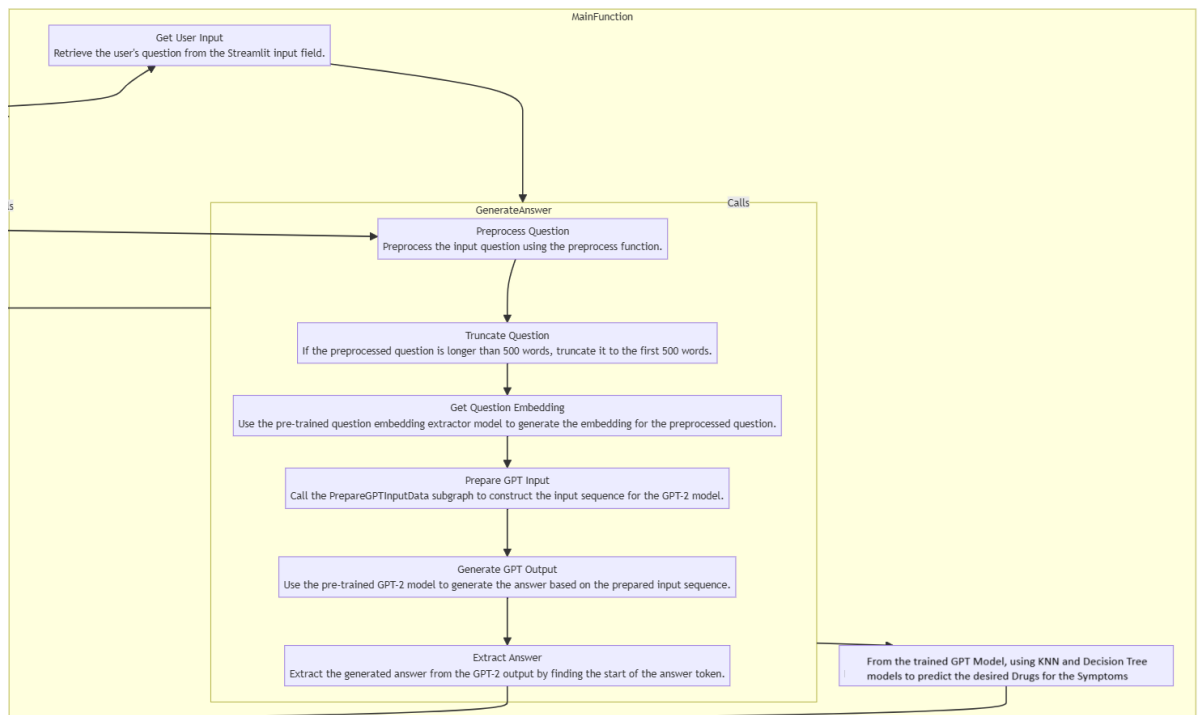


Figure 1.4 – Part-3 of Class diagram

### 4.3 CONSTRAINTS, ALTERNATIVES AND TRADEOFFS

#### Constraints:

1. **Hardware Constraints:** The code doesn't have explicit hardware constraints mentioned, but it heavily relies on the computational capabilities of the system, especially for running TensorFlow and Hugging Face's transformers models. Depending on the size of the models and the data being processed, the hardware requirements could vary. Particularly, loading and running large language models like GPT-2 and BioBERT may require significant memory (RAM) and processing power (CPU/GPU).
2. **Software Dependencies:** The code has several software dependencies such as TensorFlow, Hugging Face's transformers library, Flask, NLTK, Faiss, Pandas, and others. Ensuring compatibility and availability of these libraries across different platforms is essential. Additionally, compatibility with different versions of these libraries could be a concern.
3. **Data Availability:** The code relies on various data files (train\_gpt\_data.pkl, CSV files) and pre-trained models (question\_extractor\_model\_2\_11, tf\_gpt2\_model\_2\_150\_10000, BioBERT, GPT-2). Any changes in the data location, format, or the absence of these files could lead to errors during execution.

#### Alternatives:

1. **Hardware Scaling:** If hardware limitations are encountered, one alternative could be to migrate the code to a more powerful computing environment, such as a cloud-based solution (e.g., AWS, Google Cloud Platform) with scalable computing resources.
2. **Model Optimization:** Depending on the specific requirements and performance bottlenecks, optimizing the machine learning models (e.g., reducing model size, quantization, pruning) could improve efficiency and reduce resource consumption.
3. **Data Management:** Ensuring proper data management practices, such as version control, backup, and efficient data loading techniques, can help mitigate issues related to data availability and compatibility.

#### Trade-offs:

1. **Model Complexity vs. Inference Speed:** Using large language models like GPT-2 and BioBERT enables more accurate responses and better language understanding but may come at the cost of increased inference time, especially on resource-constrained devices.
2. **Accuracy vs. Efficiency:** There's often a trade-off between model accuracy and computational efficiency. Employing more complex models and extensive preprocessing techniques may improve accuracy but could also require more computational resources and

time.

3. **Dependency Management:** While using external libraries and pre-trained models can expedite development, it introduces dependencies that need to be managed. Depending too heavily on external dependencies can lead to version conflicts, compatibility issues, and maintenance challenges over time.

## 5. SCHEDULE, TASKS AND MILESTONES

### 5.1 GANTT CHART

Table 1.1 – Gantt Chart

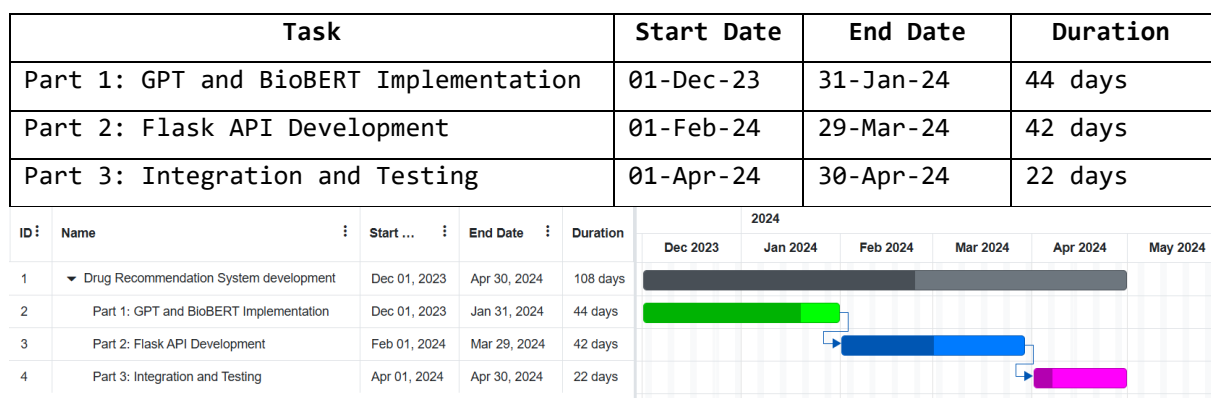


Figure 2.1

### 5.2 MODULE DESCRIPTION

#### 5.2.1 MODULE – GPT and BioBERT Implementation

- **Objective:** Implement GPT and BioBERT models for drug recommendation.
- **Tasks:**
  1. Research and select appropriate pre-trained models.
  2. Fine-tune GPT and BioBERT models on drug-related datasets.
  3. Develop algorithms for drug recommendation based on model outputs.
  4. Test and validate the implemented models.

#### 5.2.2 MODULE – Flask API Development

- **Objective:** Develop a Flask API for integrating the recommendation models with the frontend.
- **Tasks:**
  1. Design the API endpoints for receiving input data and returning recommendations.
  2. Implement Flask routes and controllers for handling API requests.

3. Integrate the GPT and BioBERT models with the Flask API.
4. Implement error handling and input validation.

### 5.2.3 MODULE – *Integration and Testing*

- **Objective:** Integrate all components and perform testing to ensure functionality and reliability.
- **Tasks:**
  1. Integrate the Flask API with the frontend interface.
  2. Conduct unit tests on individual modules (GPT, BioBERT, Flask).
  3. Perform integration testing to verify interactions between modules.
  4. Debug and resolve any issues identified during testing.

## 5.3 TESTING

### 5.3.1 UNIT TESTING

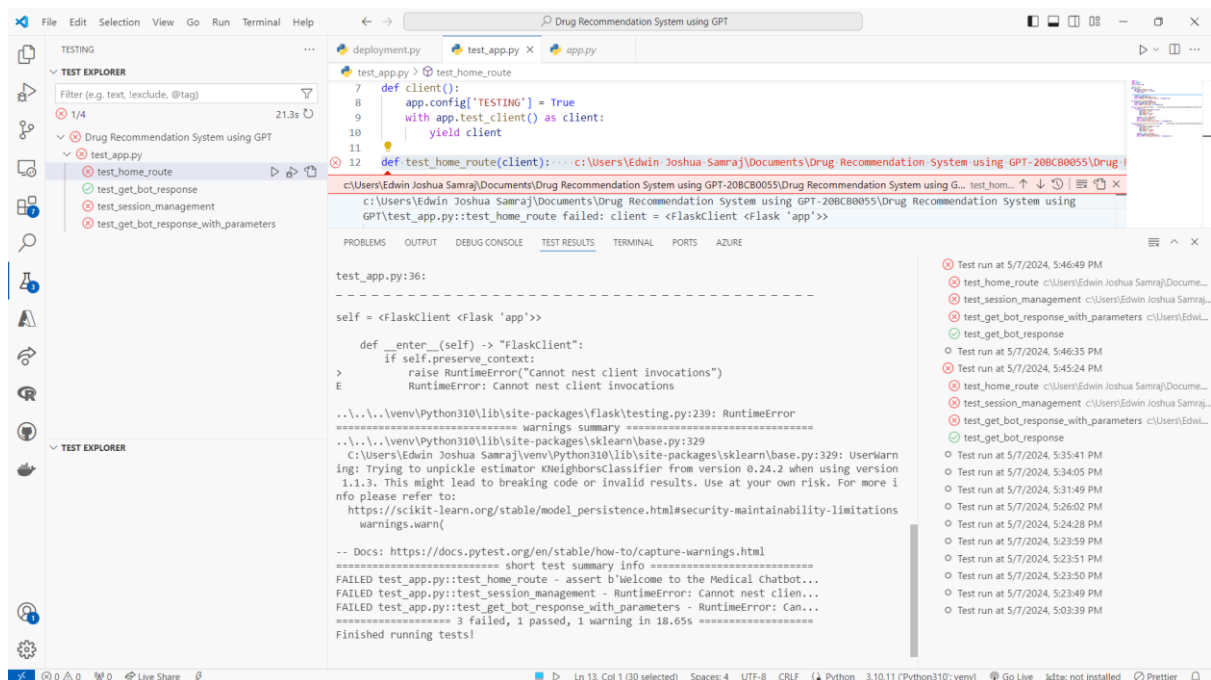
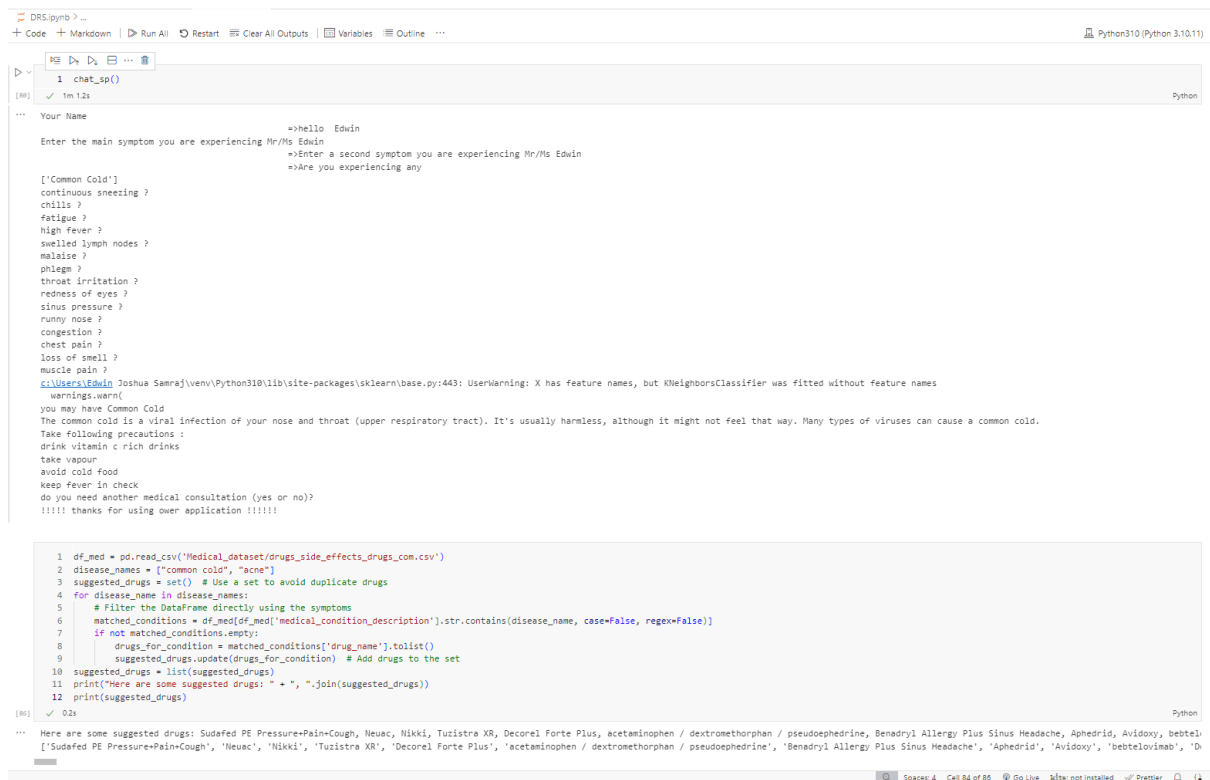


Figure 2.2 – Unit Testing done using 'pytest'

## 5.3.2 INTEGRATION TESTING



```
1 chat_sp()
[00] ✓ 1m 12s Python

... Your Name
Enter the main symptom you are experiencing Mr/Ms Edwin
[Common Cold]
continuous sneezing ?
chills ?
fatigue ?
high fever ?
swelled lymph nodes ?
malaise ?
phlegm ?
throat irritation ?
redness of eyes ?
sinus pressure ?
runny nose ?
congestion ?
chest pain ?
loss of smell ?
muscle pain ?
C:\Users\Edwin Joshua Samraj\venv\Python310\lib\site-packages\sklearn\base.py:443: UserWarning: X has feature names, but KNeighborsClassifier was fitted without feature names
warnings.warn(
you may have Common Cold
The common cold is a viral infection of your nose and throat (upper respiratory tract). It's usually harmless, although it might not feel that way. Many types of viruses can cause a common cold.
Take following precautions :
drink vitamin c rich drinks
take vapour
avoid cold food
keep fever in check
do you need another medical consultation (yes or no)?
!!!!!! thanks for using over application !!!!!

1 df_med = pd.read_csv('Medical_dataset/drugs_side_effects_drugs_com.csv')
2 disease_names = ["common cold", "acne"]
3 suggested_drugs = set() # Use a set to avoid duplicate drugs
4 for disease_name in disease_names:
5     # Filter the DataFrame directly using the symptoms
6     matched_conditions = df_med[df_med['medical_condition_description'].str.contains(disease_name, case=False, regex=False)]
7     if not matched_conditions.empty:
8         drugs_for_condition = matched_conditions['drug_name'].tolist()
9         suggested_drugs.update(drugs_for_condition) # Add drugs to the set
10 suggested_drugs = list(suggested_drugs)
11 print("Here are some suggested drugs: " + " ".join(suggested_drugs))
12 print(suggested_drugs)
[00] ✓ 02s Python

... Here are some suggested drugs: Sudafed PE PressurePainCough, Neuac, Nikki, Tuzistra XR, Decorel Forte Plus, acetaminophen / dextromethorphan / pseudoephedrine, Benadryl Allergy Plus Sinus Headache, Aphedrid, Avidoxy, bebtel
['Sudafed PE PressurePainCough', 'Neuac', 'Nikki', 'Tuzistra XR', 'Decorel Forte Plus', 'acetaminophen / dextromethorphan / pseudoephedrine', 'Benadryl Allergy Plus Sinus Headache', 'Aphedrid', 'Avidoxy', 'bebtelovimab', 'D
```

Figure 2.3 – Integration Testing done on Jupyter Notebook

## 6. PROJECT DEMONSTRATION



```
PS C:\Users\Edwin Joshua Samraj\Documents\Drug Recommendation System using GPT-20BCB0055\Drug Recommendation System using GPT> python app.py
C:\Users\Edwin Joshua Samraj\venv\Python310\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator KNeighborsClassifier from
version 0.24.2 when using version 1.1.3. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
warnings.warn(
* Serving Flask app 'app'
* Debug mode: off
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
```

Figure 3.1

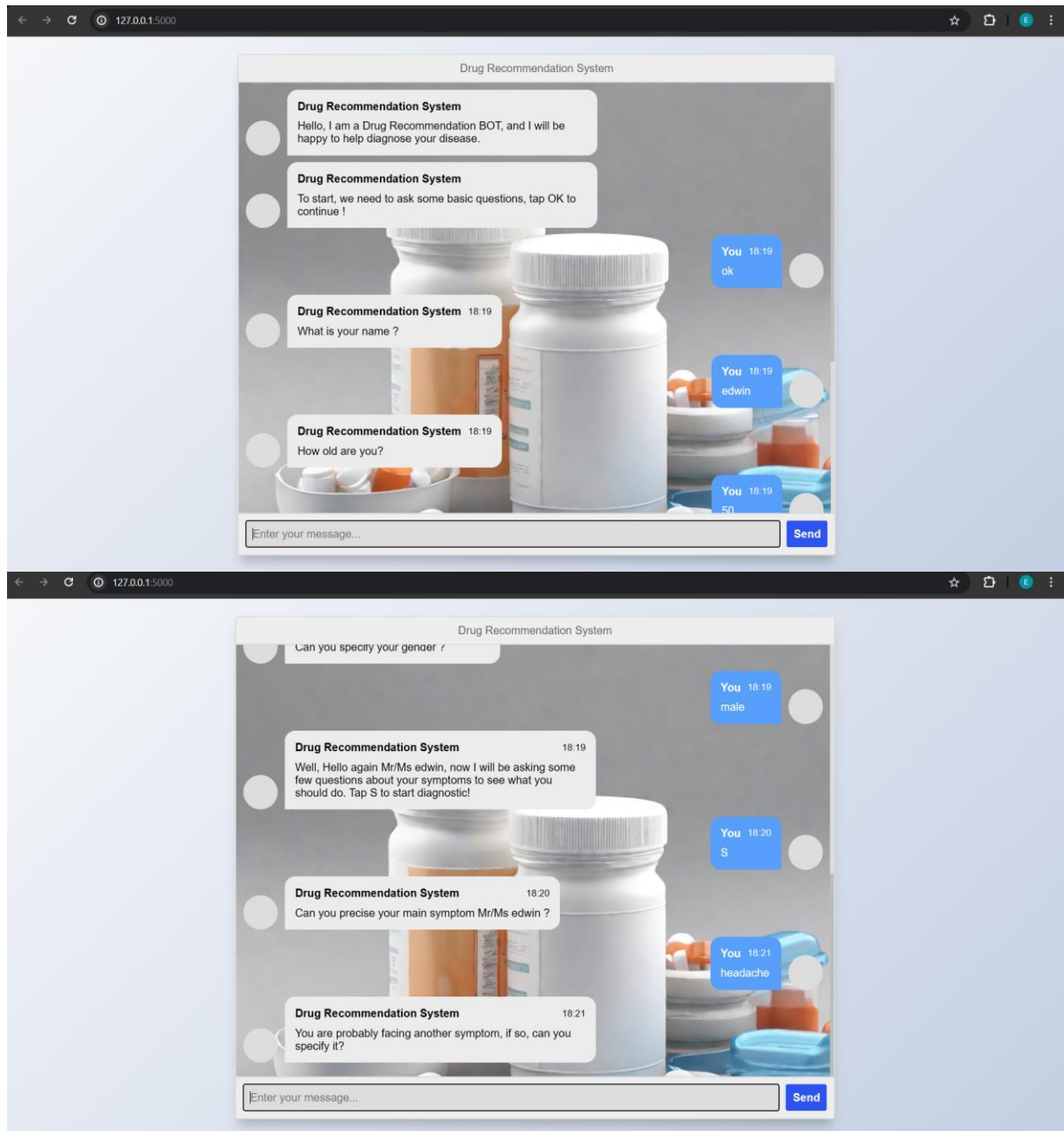


Figure 3.2

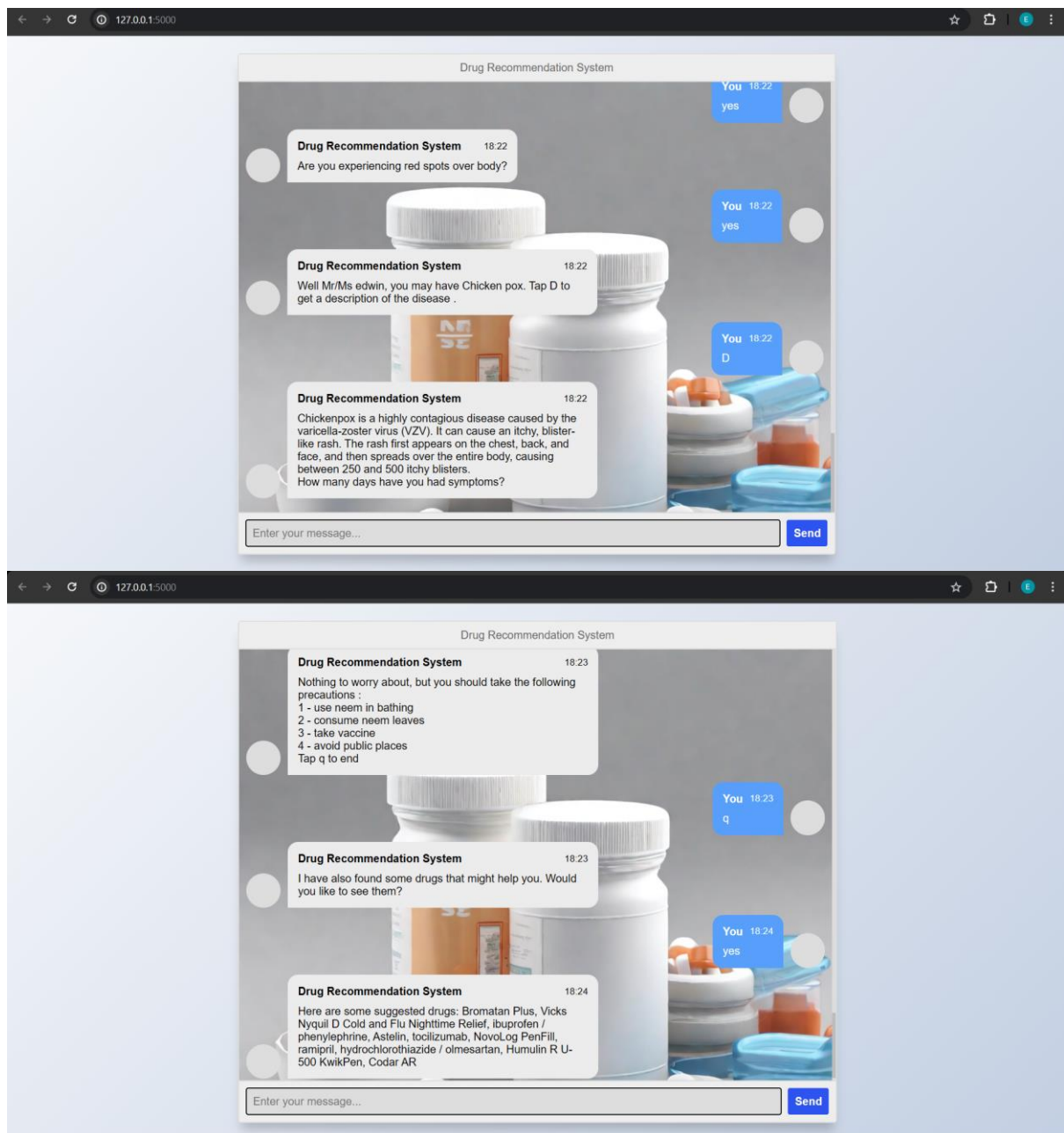


Figure 3.3



```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TEST RESULTS TERMINAL PORTS AZURE JUPYTER
blems - Total 2 Problems Joshua Samraj\Documents\Drug Recommendation System using GPT-20BCB0055\Drug Recommendation System using GPT> python app.py
C:\Users\Edwin Joshua Samraj\venv\Python310\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator KNeighborsClassifier from
version 0.24.2 when using version 1.1.3. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
warnings.warn(
* Serving Flask app 'app'
* Debug mode: off
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
127.0.0.1 - - [07/May/2024 18:19:24] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:19:25] "GET /static/styles/style.css HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:19:26] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:19:26] "GET /static/styles/test.png HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:19:26] "GET /static/styles/style.css HTTP/1.1" 304 -
127.0.0.1 - - [07/May/2024 18:19:26] "GET /static/styles/test.png HTTP/1.1" 304 -
127.0.0.1 - - [07/May/2024 18:19:39] "GET /get?msg=ok HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:19:42] "GET /get?msg=edwin HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:19:45] "GET /get?msg=50 HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:19:51] "GET /get?msg=male HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:20:55] "GET /get?msg=S HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:21:07] "GET /get?msg=headache HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:22:00] "GET /get?msg=fever HTTP/1.1" 200 -
hey2
['headache', 'high fever']
['Chicken pox', 'Dengue', 'Malaria', 'Typhoid', 'Common Cold']
127.0.0.1 - - [07/May/2024 18:22:03] "GET /get?msg=0 HTTP/1.1" 200 -
['Chicken pox', 'Dengue']
127.0.0.1 - - [07/May/2024 18:22:07] "GET /get?msg=yes HTTP/1.1" 200 -
['Chicken pox', 'Dengue']
127.0.0.1 - - [07/May/2024 18:22:14] "GET /get?msg=yes HTTP/1.1" 200 -
['Chicken pox']
HANAAA
['Chicken pox']
127.0.0.1 - - [07/May/2024 18:22:18] "GET /get?msg=yes HTTP/1.1" 200 -
127.0.0.1 - - [07/May/2024 18:22:23] "GET /get?msg=no HTTP/1.1" 200 -
['Chicken pox']
127.0.0.1 - - [07/May/2024 18:22:30] "GET /get?msg=yes HTTP/1.1" 200 -
['Chicken pox']
127.0.0.1 - - [07/May/2024 18:22:33] "GET /get?msg=yes HTTP/1.1" 200 -
['Chicken pox']
127.0.0.1 - - [07/May/2024 18:22:35] "GET /get?msg=yes HTTP/1.1" 200 -
['Chicken pox']
127.0.0.1 - - [07/May/2024 18:23:52] "GET /get?msg=3 HTTP/1.1" 200 -
Here are some suggested drugs: Bromatan Plus, Vicks Nyquil D Cold and Flu Nighttime Relief, ibuprofen / phenylephrine, Astelin, toclizumab, Novolog Pe
nFill, ramipril, hydrochlorothiazide / olmesartan, Humulin R U-500 KwikPen, Codar AR, sitagliptin, Alavert, azelastine, Cleocin Phosphate, Zithranol-RR
, Fabior, Riaz, Toujeo Max SoloStar, prazosin, Aphedrin, risankizumab, Rukobia, Dixaphedrine, Scot-Tussin Allergy, levamlopidine, dextromethorphan / di
phenhydramine / phenylephrine, Dristan Cold Multi Symptom Formula, Pepcid, Afeditab CR, Zetonna, dapagliflozin, Clindets, Corzide 40/5, leflunomide, No
rel AD, brompheniramine / codeine / phenylephrine, camphor / eucalyptus / menthol, Akurza, Trexall, Hydralisic, Chlorex-A 12, Ipratropium Inhalation Ae
rosol, Medotac, Tri-Previfem, Dulcolax Stool Softener, aluminum hydroxide, Histex-PE, Migergot, Almacone, Analpram-HC, SASTid, chondroitin / glucosamin
e / methylsulfonylmethane, Arthricream, acetaminophen / dextromethorphan, azilsartan medoxomil, Azuphen MB, benzoic acid / hyoscyamine / methenamine /
methylene blue / phenyl salicylate, Zumandimine, Onset Forte, Dimetapp Children's Cold & Allergy, Steglatro, NovoLog Mix 70/30, trandolapril / verapami
l, tipranavir, bromocriptine, Ocella, Accupril, Lo-Zumandimine, Voltaren, Clinacort, Mylanta Coat & Cool, Ubrelyv, Daypro, methylprednisolone, hydrochl
orothiazide / quinapril, cefiderocol, Phosphasal, Omnaris, Cleanse & Treat, Heartburn Relief, Myoflex, LoHist-D, flunisolide, Methergine, corticoid HBP
Nighttime Multi-Symptom Cold, Prandin, Dicel, immune globulin intravenous, Absorica LD, Synjardy XR, Moderna COVID-19 Vaccine, Viramune, Abatuss DMX,
Diclostroom, prednisolone, Virtussin DAC, Excedrin Tension Headache, Genpril, ChlorTan, Humalog, adapalene / benzoyl peroxide, Humulin N Pen, Evoclin,
acetaminophen / aspirin / caffeine, Jentaduo XR, sars-cov-2 (covid-19) ad26 vaccine, recombinant, Uta, Konsyl, Fenovar, Acid Reducer Maximum Strength
, Dymista, penicillin g benzathine / procaine penicillin, insulin aspart/insulin aspart protamine, Tenoretic 50, Maxipime, dapsone, Claravis, Nalfon, A
llergy Relief D, Tirostint-Sol, Equate Sleep Aid, Ziac, Diuril Sodium, Emsin Clear, doxycycline, Benadryl Allergy Plus Congestion, Zegerid with Magnesi
um Hydroxide, Ryclora, Binora, Zyrtec, Vicks Nyquil Cold & Flu Nighttime Relief (Alcohol Free), Ipratropium Inhalation Solution, hydrochlorothiazide / l
osartan, MG217 Medicated Tar, chlorpheniramine / phenylephrine / phenyltoloxamine, Glucoten, Lopressor HCT, wal-finate, NoHist LQ, NovoLog FlexPen, Imi
trex Statdose, Novolog, liraglutide, Corzide, saquinavir, acetaminophen / chlorpheniramine / dextromethorphan, Pepcid AC, insulin inhalation, rapid act
ing, Edarbyclor, sars-cov-2 (covid-19) mrna-1273 vaccine, Lagevrio, acetaminophen / aspirin, chromium picolinate, magnesium hydroxide, Capron DM, Biotu
ssin DAC, Codditussin DAC, chlorpheniramine / ibuprofen / phenylephrine, Adoxa CK, nevirapine, Imitrex, Minolira, Winlevi, Aleve, pegloticase, Acromel,
dolutegravir / rilpivirine, lisinopril, Tybost, Zostrix Sports, Lantus, valacyclovir, Capzasin-P, Diovan, trandolapril, Zestril, Novacort, hydrochlorot
hiazide / irbesartan, Dutoprol, Bydureon BCise, nisoldipine, Chlor-Mal, Kineret, Nuzyra, Edarbi, Amoclan, Balnetar, Zithranol, Cardura, ragweed pollen
allergen extract, Tylenol Cold Multi-Symptom Severe, azelastine / fluticasone, Dytan-D, Allergy Relief 24 Hour, Children's Triacting Night Time, canagl
iflozin, Epiduo Forte, aliskiren, Betatar Gel, Schiff Move Free, Farxiga, emtricitabine / nelfinavir / tenofovir, ceftriaxone, Pfizer-BioNTech COVID-19
(12y+) Bivalent Booster Vaccine PF, dextromethorphan / promethazine, acetaminophen / dextromethorphan / doxylamine, Microzide, Bydureon, amoxicillin,
Duac, Tannic-12 S, cefprozil, GaviLAX, Citrate of Magnesia, Phenagil, Yasmin, Aprodine, dextlansoprazole, Fototar, efavirenz / emtricitabine / tenofovir
, Septra, Prezobix, acetaminophen / dextromethorphan / guaifenesin / pseudoephedrine, Clarinex-D 12 Hour, Achromycin V, trimethoprim, Flonase Allergy
Relief, hyoscyamine / methenamine / methylene blue / phenyl salicylate / sodium biphosphate, triamcinolone, rilpivirine, Accuretic, mixed grass pollens
allergen extract, Gianvi, QNASL, Mylanta Maximum Strength, plazomicin, Aloprim, Gelusil, dapagliflozin / saxagliptin, Acnevair, BP 10-Wash, Vilamit MB,
REGEN-COV, Exforge HCT, Invokamet, A-Phedrin, nateglinide, amlodipine / perindopril, dextromethorphan / pyrilamine, benzoyl peroxide / hydrocortisone,
Olumiant, lamivudine / nevirapine / zidovudine, Tinameed Plantar, aliskiren / hydrochlorothiazide, guselkumab, ertugliflozin, Ery-Tab, Xopenex Concentr
ate, Excedrin Quick Tab, emtricitabine, Neuac, Conal, Lotrel, isotretinoin, captopril / hydrochlorothiazide, Cafergot, Fiber Lax, clemastine, Zyluprim,
cilastatin / imipenem, CeraVe SA Renewing, Mucinex Nightshift Cold & Flu, Clarifoam EF, glycerin, Sudafed PE PressurePain+Cough, Altreno, Morgidox, L
evovoxyl, Paxlovid, Urimar-T, darunavir, Levemir, Fiasp, Glumetaz, Xigduo XR, perindopril, cabotegravir / rilpivirine, Fleet Bisacodyl, metformin, Advil
Allergy Sinus, Suprax, esomeprazole / naproxen, Fostex Medicated, doravirine, Abreva, sorbitol, Humira, Selzentry, Tylenol Cough & Sore Throat Nighttime
, captopril, Milantex, Sudafed PE Severe Cold, Hytrin, Dolobid, M-End PE, Liqumat Medium, Conjupri, Tygacil, Sulfatrim Pediatric, Aller-Ease, Leader A
llergy Relief D-24, ceftaroline, Vancocin HCL, Catapres-TTS, Choline Magnesium Trisiliclyate, Frova, certolizumab, Azor, Sorilux, fluticasone / vilante
rol, Prilosec, Lofena, tildrakizumab, Atrovent HFA, Exocaine Plus, naproxen, Erythrocin Lactobionate, Clinoril, hydrochlorothiazide / telmisartan, Alav
ert D-12 Hour Allergy and Sinus, Hyophen, bictegravir / emtricitabine / tenofovir alafenamide, Valu-Dryl, Loryna, SymlinPen 120, acetaminophen / dextro
methorphan / diphenhydramine, Micardis HCT, dicloxacillin, Flagyl IV, candesartan / hydrochlorothiazide, empagliflozin / linagliptin, Vaseretic, UroAv-
81, Diabetic Tussin Night Time Formula, Starlix, Adoxa, Dextilant, caffeine, Ed Chlor Ped Jr., Cyltezo, allopurinol, Ozempic, Seglurumet, raltegravir, c
hlorpheniramine / phenylephrine, dextromethorphan / pseudoephedrine, clascoterone, Sani-Supp, octreotide, Ancef, ergotamine, Westhroid, Codimal DM, g
```

Figure 3.4



## 7. RESULT AND DISCUSSION

### Finetuning BioBERT Model:

```
1 #displaying biobert model summary
2 biobert_model.summary()
```

Model: "tf\_bert\_model\_1"

Layer (type)	Output Shape	Param #
bert (TFBertMainLayer)	multiple	108310272

Total params: 108,310,272

Trainable params: 108,310,272

Non-trainable params: 0

	questions	answers	tags
0	how do i stop smoking now	stopping smoking is about will power and being...	['addiction', 'stop smoking']
1	i had a tubaligation 4 years ago and also have...	hello this sounds quite unfamiliar that due to...	['pregnancy', 'diet', 'endometriosis']
2	could extra caffeine consumption be a cause of...	extra caffeine can cause gastric discomfort th...	['breast cancer', 'cancer', 'breasts']
3	hello- i am a 24 year old female 5"4 & 115 lb ...	hello thanks for submitting your question here...	['hair loss', 'diet', 'acne', 'ovulation and o...']
4	i was wanting to know if you could tell me if ...	i am glad to help you out. this is not possibl...	['am i pregnant', 'pregnant', 'urine pregnancy...']
...	...	...	...
29747	how can accidental of acetaminophen overdose b...	to avoid unintentional overdoses among adults ...	['drug overdose', 'acetaminophen']
29748	what should i do if i take an overdose of maxalt?	if you take more medication than you have been...	['drug overdose']
29749	what do i do in case of an overdose of relpax?	call your doctor or poison control center or g...	['drug overdose']
29750	is overdose with acetaminophen usually acciden...	in the u. s. suicide attempts account for over...	['drug overdose', 'acetaminophen']
29751	how does an overdose of acetaminophen cause li...	the answer is that liver damage from acetamino...	['drug overdose', 'injury', 'liver', 'acetamin...

	questions	answers	tags
0	how do i stop smoking now	stopping smoking is about will power and being...	[addiction, stop smoking]
1	i had a tubaligation 4 years ago and also have...	hello this sounds quite unfamiliar that due to...	[pregnancy, diet, endometriosis]
2	could extra caffeine consumption be a cause of...	extra caffeine can cause gastric discomfort th...	[breast cancer, cancer, breasts]
3	hello- i am a 24 year old female 5"4 & 115 lb ...	hello thanks for submitting your question here...	[hair loss, diet, acne, ovulation and ovaries]
4	i was wanting to know if you could tell me if ...	i am glad to help you out. this is not possibl...	[am i pregnant, pregnant, urine pregnancy test...]
23801/23801	[=====] - 7862s 327ms/step - loss: 0.6119 - custom_metric_acc: 0.7737 - val_loss: 0.5745 - val_custom_metric_acc: 0.8136		
Epoch 2/5			
23801/23801	[=====] - 7781s 327ms/step - loss: 0.5128 - custom_metric_acc: 0.8611 - val_loss: 0.5595 - val_custom_metric_acc: 0.8380		
Epoch 3/5			
23801/23801	[=====] - 7784s 327ms/step - loss: 0.4518 - custom_metric_acc: 0.9020 - val_loss: 0.5779 - val_custom_metric_acc: 0.8239		
Epoch 4/5			
23801/23801	[=====] - 7785s 327ms/step - loss: 0.4042 - custom_metric_acc: 0.9204 - val_loss: 0.5885 - val_custom_metric_acc: 0.8187		
Epoch 5/5			
3040/23801	[==>.....] - ETA: 1:44:55 - loss: 0.3494 - custom_metric_acc: 0.9488		

```

1 #find best threshold and accuracy of the model with best train accuracy on validation data
2 best_threshold_acc(predicted_labels)

```

Could not render content for 'application/vnd.jupyter.widget-view+json'

{"model\_id":"31b3a6a8b7ab4525b7392494e1933b3e","version\_minor":0,"version\_major":2}

```

accuracy for threshold -1.0 is 0.4999579867238047
accuracy for threshold -0.9 is 0.4999579867238047
accuracy for threshold -0.8 is 0.5001260398285858
accuracy for threshold -0.7000000000000001 is 0.5029829426098648
accuracy for threshold -0.6000000000000001 is 0.514074447525418
accuracy for threshold -0.5 is 0.542979581547769
accuracy for threshold -0.4 is 0.5984371061255357
accuracy for threshold -0.30000000000000004 is 0.65960843626586
accuracy for threshold -0.2 is 0.7234686160826821
accuracy for threshold -0.1 is 0.7790101672128392
accuracy for threshold 0.0 is 0.8185866733887909
accuracy for threshold 0.1 is 0.845391143601378
accuracy for threshold 0.2 is 0.8606839761364591
accuracy for threshold 0.30000000000000004 is 0.8583312326695236
accuracy for threshold 0.4 is 0.8383329132005714
accuracy for threshold 0.5 is 0.8011931770439459
accuracy for threshold 0.6000000000000001 is 0.7006974203848416
accuracy for threshold 0.7000000000000001 is 0.5449962188051424
accuracy for threshold 0.8 is 0.5055037391815814
accuracy for threshold 0.9 is 0.5002100663809764
accuracy for threshold 1.0 is 0.5000420132761952

```

best accuracy is 0.8606839761364591 at threshold 0.2

```

1 #displaying the pdf of correctly predicted positive and negative points
2 import seaborn as sns
3 _,correct_pred_pos,correct_pred_neg=acc_threshold(predicted_labels,0.2)
4 sns.distplot(correct_pred_pos)
5 sns.distplot(correct_pred_neg)

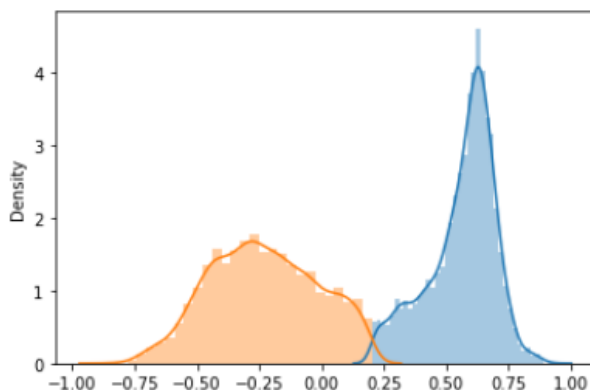
```

```

accuracy for threshold 0.2 is 0.8606839761364591
/usr/local/lib/python3.6/dist-packages/seaborn/distributions.py:2557: FutureWarning
  warnings.warn(msg, FutureWarning)
/usr/local/lib/python3.6/dist-packages/seaborn/distributions.py:2557: FutureWarning
  warnings.warn(msg, FutureWarning)

```

<matplotlib.axes.\_subplots.AxesSubplot at 0x7fcf2613c240>



```

1 #finding the best accuracy and the corresponding threshold of the model of best validation accuracy
2 best_threshold_acc(predicted_labels)

```

Could not render content for 'application/vnd.jupyter.widget-view+json'

```

{"model_id":"eae8ad25aa964a8f97b79b795108a547","version_minor":0,"version_major":2}

```

```

accuracy for threshold -1.0 is 0.4999579867238047
accuracy for threshold -0.9 is 0.4999579867238047
accuracy for threshold -0.8 is 0.4999579867238047
accuracy for threshold -0.7000000000000001 is 0.5008822788001008
accuracy for threshold -0.6000000000000001 is 0.5060919250483152
accuracy for threshold -0.5 is 0.5318040500798252
accuracy for threshold -0.4 is 0.5820519284093774
accuracy for threshold -0.30000000000000004 is 0.6506175951600706
accuracy for threshold -0.2 is 0.7279220233593816
accuracy for threshold -0.1 is 0.7923703890429376
accuracy for threshold 0.0 is 0.8386690194101336
accuracy for threshold 0.1 is 0.8675741534324847
accuracy for threshold 0.2 is 0.8825308797580035
accuracy for threshold 0.30000000000000004 is 0.8852197294345013
accuracy for threshold 0.4 is 0.8705150827661541
accuracy for threshold 0.5 is 0.8350558776573397
accuracy for threshold 0.6000000000000001 is 0.7401899000084027
accuracy for threshold 0.7000000000000001 is 0.5780186538946307
accuracy for threshold 0.8 is 0.5128140492395598
accuracy for threshold 0.9 is 0.5008822788001008
accuracy for threshold 1.0 is 0.5000420132761952

```

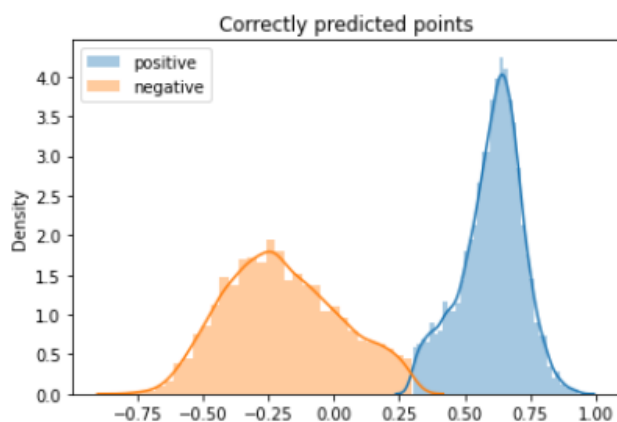
best accuracy is 0.8852197294345013 at threshold 0.30000000000000004

```

1 #plotting the pdf of correctly predicted positive and negative points
2 import matplotlib.pyplot as plt
3 import seaborn as sns
4 _,correct_pred_pos,correct_pred_neg=acc_threshold(predicted_labels,0.3)
5 sns.distplot(correct_pred_pos,label='positive')
6 sns.distplot(correct_pred_neg,label='negative')
7 plt.title('Correctly predicted points')
8 plt.legend()
9 plt.show()
10

```

accuracy for threshold 0.3 is 0.8852197294345013



```

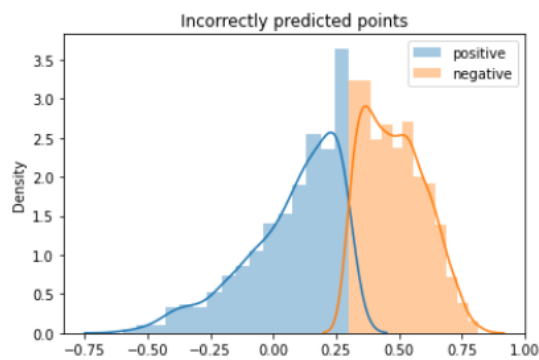
1 #finding incorrectly predicted positive and negative points
2 incorrect_pos=[]
3 incorrect_neg=[]
4 for i in range(len(val_labels)):
5     if val_labels[i]==1 and predicted_labels[i]<0.3:
6         incorrect_pos.append(predicted_labels[i])
7     elif val_labels[i]==-1 and predicted_labels[i]>=0.3:
8         incorrect_neg.append(predicted_labels[i])

```

```

1 #plotting the pdf of model predicted similarities for incorrectly predicted positive and negative points
2 sns.distplot(incorrect_pos,label='positive')
3 sns.distplot(incorrect_neg,label='negative')
4 plt.title('Incorrectly predicted points')
5 plt.legend()
6 plt.show()

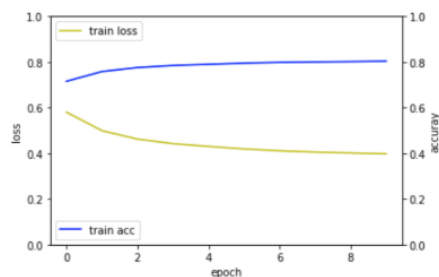
```



```

Epoch 1/10
142075/142075 [=====] - 39s 272us/step - loss: 0.5802 - acc: 0.7152
Epoch 2/10
142075/142075 [=====] - 43s 302us/step - loss: 0.4990 - acc: 0.7582
Epoch 3/10
142075/142075 [=====] - 35s 246us/step - loss: 0.4621 - acc: 0.7758
Epoch 4/10
142075/142075 [=====] - 42s 299us/step - loss: 0.4422 - acc: 0.7848
Epoch 5/10
142075/142075 [=====] - 43s 301us/step - loss: 0.4300 - acc: 0.7899
Epoch 6/10
142075/142075 [=====] - 43s 300us/step - loss: 0.4189 - acc: 0.7948
Epoch 7/10
142075/142075 [=====] - 43s 301us/step - loss: 0.4109 - acc: 0.7983
Epoch 8/10
142075/142075 [=====] - 43s 301us/step - loss: 0.4053 - acc: 0.8000
Epoch 9/10
142075/142075 [=====] - 43s 305us/step - loss: 0.4013 - acc: 0.8015
Epoch 10/10
142075/142075 [=====] - 43s 302us/step - loss: 0.3976 - acc: 0.8036

```



```

69978/69978 [=====] - 12s 165us/step
loss_and_metrics : [1.0738699619418919, 0.6469318928772125]

```

## Finetuning GPT-2 Model:

```
1 #Cleaning the gpt train data
2 train_gpt_data=train_embeds[['short_question','short_answer','question_embeddings_norm','answer_embeddings_norm']].copy()
3 train_gpt_data.columns=['question','answer','Q_FFNN_embeddings','A_FFNN_embeddings']
4 train_gpt_data.head(3)
```

Python

	question	answer	Q_FFNN_embeddings	A_FFNN_embeddings
0	can an antibiotic through an iv give you a ras...	yes it can even after you have finished the pr...	[-0.0015947532801336905, 0.028313202930346674...	[-0.033252108881325626, 0.036488885218617664, ...
1	can you test positive from having the hep b va...	test positive for what if you had a hep b vacc...	[0.02388940038582553, 0.03578648615547307, -0...	[0.00845440863602465, 0.0381747505749067, 0.01...
2	what are the dietary restrictions for celiac d...	omitting gluten from the diet is the key to co...	[0.01185235921221315, 0.03739971001082176, -0...	[0.0030481943703320473, 0.024438320236403705, ...

```
1 #saving gpt training data to disk
2 train_gpt_data.to_pickle("./train_gpt_data.pkl")
```

Python

```
1 #Cleaning gpt validation data
2 validation_gpt_data=validation_embeds[['short_question','short_answer','question_embeddings_norm','answer_embeddings_norm']].copy()
3 validation_gpt_data.columns=['question','answer','Q_FFNN_embeddings','A_FFNN_embeddings']
4 validation_gpt_data.head(3)
```

Python

	question	answer	Q_FFNN_embeddings	A_FFNN_embeddings
0	do i have a yeast infection	hi this can be a vaginal fungal infection whic...	[-0.012255767552121263, 0.011278647301171585, ...	[-0.038504731105611124, 0.005560366067635298, ...
1	i need to buy health insurance asap what do i ...	go to healthcare gov call my husband 407 222 9...	[0.08883379130035676, 0.039120400705552394, -0...	[0.0843680653010217, 0.04661907941883923, 0.00...
2	i had an acute ebv antibody test done and my r...	hi yes a chronic ebv infection also can be rul...	[-0.016364305356089962, 0.001889975145291855, ...	[-0.022903376308870023, -0.0005906026628375108...

epoch 0

HBox(children=(FloatProgress(value=0.0, max=23783.0), HTML(value='')))

```
tf.Tensor(371.42813, shape=(), dtype=float32)
tf.Tensor(388.3936, shape=(), dtype=float32)
tf.Tensor(331.53702, shape=(), dtype=float32)
tf.Tensor(394.851, shape=(), dtype=float32)
tf.Tensor(317.90687, shape=(), dtype=float32)
tf.Tensor(335.27008, shape=(), dtype=float32)
tf.Tensor(402.66028, shape=(), dtype=float32)
tf.Tensor(405.32172, shape=(), dtype=float32)
tf.Tensor(338.03772, shape=(), dtype=float32)
tf.Tensor(484.97726, shape=(), dtype=float32)
tf.Tensor(334.72992, shape=(), dtype=float32)
tf.Tensor(321.94858, shape=(), dtype=float32)
tf.Tensor(356.25452, shape=(), dtype=float32)
tf.Tensor(390.04855, shape=(), dtype=float32)
tf.Tensor(327.19766, shape=(), dtype=float32)
tf.Tensor(363.35968, shape=(), dtype=float32)
tf.Tensor(290.69012, shape=(), dtype=float32)
tf.Tensor(385.61945, shape=(), dtype=float32)
tf.Tensor(289.3142, shape=(), dtype=float32)
tf.Tensor(311.32895, shape=(), dtype=float32)
tf.Tensor(364.6267, shape=(), dtype=float32)
tf.Tensor(312.8274, shape=(), dtype=float32)
tf.Tensor(384.81598, shape=(), dtype=float32)
tf.Tensor(344.28122, shape=(), dtype=float32)
tf.Tensor(333.40027, shape=(), dtype=float32)
tf.Tensor(331.32706, shape=(), dtype=float32)
tf.Tensor(417.59045, shape=(), dtype=float32)
```

epoch\_loss by 100 tf.Tensor(375.68814, shape=(), dtype=float32)  
epoch 1

HBox(children=(FloatProgress(value=0.0, max=23783.0), HTML(value='')))

```
tf.Tensor(381.5399, shape=(), dtype=float32)
tf.Tensor(305.80383, shape=(), dtype=float32)
tf.Tensor(318.666, shape=(), dtype=float32)
tf.Tensor(370.933, shape=(), dtype=float32)
tf.Tensor(359.79773, shape=(), dtype=float32)
tf.Tensor(338.52667, shape=(), dtype=float32)
tf.Tensor(307.0343, shape=(), dtype=float32)
tf.Tensor(302.97275, shape=(), dtype=float32)
tf.Tensor(330.74194, shape=(), dtype=float32)
tf.Tensor(345.58212, shape=(), dtype=float32)
tf.Tensor(273.55933, shape=(), dtype=float32)
tf.Tensor(332.16684, shape=(), dtype=float32)
tf.Tensor(320.96143, shape=(), dtype=float32)
tf.Tensor(381.3083, shape=(), dtype=float32)
tf.Tensor(394.91925, shape=(), dtype=float32)
tf.Tensor(279.74667, shape=(), dtype=float32)
tf.Tensor(397.94757, shape=(), dtype=float32)
tf.Tensor(359.10754, shape=(), dtype=float32)
tf.Tensor(380.33856, shape=(), dtype=float32)
tf.Tensor(289.72632, shape=(), dtype=float32)
tf.Tensor(289.57968, shape=(), dtype=float32)
tf.Tensor(392.56567, shape=(), dtype=float32)
tf.Tensor(371.2386, shape=(), dtype=float32)
tf.Tensor(298.49344, shape=(), dtype=float32)
tf.Tensor(375.69684, shape=(), dtype=float32)
tf.Tensor(323.0045, shape=(), dtype=float32)
tf.Tensor(385.38028, shape=(), dtype=float32)
```

epoch\_loss by 100 tf.Tensor(348.9519, shape=(), dtype=float32)  
epoch 5

```

1 #displaying sample original context, expected answer and generated answer for train dataset
2 for i in range(5):
3     v,exp,fin=return_sample_output_predicted()
4     print(tokenizer.decode(v))
5     print(exp)
6     print(fin)
7     print('-----')

```

Python

can tell you it took a long time for the injury to heal because we use our hands for everything but with time i am able to still use my hand and the pain is gone splint the finger  
: i think its gonna be better if you add another finger with a plastic surgery instead of the amputated one but to get the other fingers closer its gonna cause you troubles with  
: hi this is a very common problem that can be due to a joint injury or a joint injury that may be due to a joint injury or a joint injury that r  
-----  
ered myself and i felt something unusual i went all the way in with my entire index finger around 3 inches deep i felt a hard kind of pointy bump i am not sure if this is my pro  
: hi just a myth the only way cancer spreads quickly is if it is diagnosed late or that is very vigorous do hope this helps good luck  
: hi if you have prostate cancer then you should be fine if you have a prostate cancer then you should be fine if you have a prostate  
-----  
iacin and fish oil my hdl is now and has been for several years 40 47 my ldl and triglycerides are below 100 and are around 80 90 my internal doctor always says to just continue  
: congratulations it sounds like you are doing an excellent job with your lifestyle changes diet and exercise are the cornerstones of treating both heart disease and diabetes but  
: i am sorry to hear about your situation but i am sorry to hear about your situation but i am not sure what you are going through i am not sure v  
-----  
to be continued `ANSWER: ear ringing and fatigue i started amoxicillin the next day and the sinus pressure went away but the rest of my symptoms remained and i was absolutely  
: most probably it is an ear canal infection you need to get antibiotic course also to get examined by ent do you have spinning or dizziness discharges or itching  
: hi this is due to a bacterial infection that is causing ear pain and inflammation in the ear it is not a bacterial infection but it is a bacterial  
-----  
her eyes a few moments later hi there thank you for answering my question no when i instill them she is lying on her side and she feels them go to her eye from inside her ear no  
: hi this sounds like a joke but i do not think it is true it is true that you can not chew gum or gum gum is not a sign of tooth decay and it  
-----

```

1 #displaying sample original context, expected answer and generated answer for validation dataset
2 for i in range(5):
3     v,exp,fin=return_validation_sample_output_predicted()
4     print(tokenizer.decode(v))
5     print(exp)
6     print(fin)
7     print('-----')

```

Python

to healthcare gov and begin the process `QUESTION: i have medicare now my husband has none how does that work with the new health insurance policies will my medicare change  
: you can shop for coverage before january 1 2014 but insurance will not take effect before then  
: you can not get insurance under the ac a however you can still get insurance under the medic  
-----  
ples with fullid `QUESTION: constant masturbation cause bumps on penis i damaged the skin area where the bumps seem to have formed so i noticed a group of bumps on the shaft of  
: this could be a condition called hidradenitis suppurativa which could occur at any time it is a scarring condition of recurrent boils in the groin under the bra line and even i  
: i am sorry to hear of your child is health problems but i do not know why you are having boils or boils but i do not know why you are having br  
-----  
warts so he is safe to pet them or get licked as much as he wants my question my little dog has a wart like breakout to of head back leg with around her buttocks i just started  
: that would be highly unlikely since those tools a vaginal speculum would need to be unwashed unsterilized and still wet from a previous patient that had genital herpes most vag  
: hi this is a very common form of herpes simple x herpes simple x herpes simple x herpes simple x herpes simple x herpes simple x herpes simple x herpes  
-----  
the zaps and dizziness now tll after i take the celexa in the morning time insomnia and light headed just dont know if the celexa is causing a allergic reaction or if its the pi  
: it is not safe to take 5400 mg of ibuprofen per day so you would be wise to see a rheumatologist for a better and safer way to control your osteoarthritis in your hip i am sorr  
: hi this is a very common problem with cort is one and it can be due to a combination of medications and medications that are used to treat oste o ar  
-----  
available equivalent montelukast i have noticed an increase of sweetening and tiredness since switching over to the generic i am going to stop taking the med for a few days to see  
: during the meals this medication should be taken with food in your stomach  
: i do not know of any over the counter drugs that would be  
-----

## Final Output:

```
1 print(classification_report(y_test,knn_clf.predict(X_test)))
```

✓ 0.1s

	precision	recall	f1-score	support
(vertigo) Paroymsal				
Positional Vertigo	1.00	1.00	1.00	1
AIDS	1.00	1.00	1.00	1
Acne	1.00	1.00	1.00	1
Alcoholic hepatitis	1.00	1.00	1.00	1
Allergy	1.00	1.00	1.00	1
Arthritis	1.00	1.00	1.00	1
Bronchial Asthma	1.00	1.00	1.00	1
Cervical spondylosis	1.00	1.00	1.00	1
Chicken pox	1.00	1.00	1.00	1
Chronic cholestasis	1.00	1.00	1.00	1
Common Cold	1.00	1.00	1.00	1
Dengue	1.00	1.00	1.00	1
Diabetes	1.00	1.00	1.00	1
Dimorphic hemmorhoids(piles)	1.00	1.00	1.00	1
Drug Reaction	1.00	1.00	1.00	1
Fungal infection	1.00	1.00	1.00	1
GERD	1.00	1.00	1.00	1
Gastroenteritis	1.00	1.00	1.00	1
Heart attack	1.00	1.00	1.00	1
Hepatitis B	1.00	1.00	1.00	1
Hepatitis C	1.00	1.00	1.00	1
Hepatitis D	1.00	1.00	1.00	1
Hepatitis E	1.00	1.00	1.00	1
Hypertension	1.00	1.00	1.00	1
Hyperthyroidism	1.00	1.00	1.00	1
Hypoglycemia	1.00	1.00	1.00	1
Hypothyroidism	1.00	1.00	1.00	1
Impetigo	1.00	1.00	1.00	1

```
1 print(classification_report(y_test,dt_clf.predict(X_test)))
```

✓ 0.0s

		precision	recall	f1-score	support
(vertigo) Paroymsal	Positional Vertigo	1.00	1.00	1.00	1
	AIDS	1.00	1.00	1.00	1
	Acne	1.00	1.00	1.00	1
	Alcoholic hepatitis	1.00	1.00	1.00	1
	Allergy	1.00	1.00	1.00	1
	Arthritis	1.00	1.00	1.00	1
	Bronchial Asthma	1.00	1.00	1.00	1
	Cervical spondylosis	1.00	1.00	1.00	1
	Chicken pox	1.00	1.00	1.00	1
	Chronic cholestasis	1.00	1.00	1.00	1
	Common Cold	1.00	1.00	1.00	1
	Dengue	1.00	1.00	1.00	1
	Diabetes	1.00	1.00	1.00	1
	Dimorphic hemmorhoids(piles)	1.00	1.00	1.00	1
	Drug Reaction	1.00	1.00	1.00	1
	Fungal infection	1.00	1.00	1.00	1
	GERD	1.00	1.00	1.00	1
	Gastroenteritis	1.00	1.00	1.00	1
	Heart attack	1.00	1.00	1.00	1
	Hepatitis B	1.00	1.00	1.00	1
	Hepatitis C	1.00	1.00	1.00	1
	Hepatitis D	1.00	1.00	1.00	1
	Hepatitis E	1.00	1.00	1.00	1
...					
	accuracy			1.00	41
	macro avg	1.00	1.00	1.00	41
	weighted avg	1.00	1.00	1.00	41

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#)...

```
1 severityDictionary
```

✓ 0.0s

```
{'itching': 1,
'skin_rash': 3,
'nodal_skin_eruptions': 4,
'continuous_sneezing': 4,
'shivering': 5,
'chills': 3,
'joint_pain': 3,
'stomach_pain': 5,
'acidity': 3,
'ulcers_on_tongue': 4,
'muscle_wasting': 3,
'vomiting': 5,
'burning_micturition': 6,
'spotting_urination': 6,
'fatigue': 4,
'weight_gain': 3,
'anxiety': 4,
'cold_hands_and_feets': 5,
'mood_swings': 3,
'weight_loss': 3,
'restlessness': 5,
'lethargy': 2,
'patches_in_throat': 6,
'irregular_sugar_level': 5,
'cough': 4,
...
'small_dents_in_nails': 2,
'inflammatory_nails': 2,
'blister': 4,
'red_sore_around_nose': 2,
'yellow_crust_ooze': 3}
```

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#)...

```
1 chat_sp()
✓ 1m 12s

Your Name                                =>hello Edwin
Enter the main symptom you are experiencing Mr/Ms Edwin
=>Enter a second symptom you are experiencing Mr/Ms Edwin
=>Are you experiencing any

['Common Cold']
continuous sneezing ?
chills ?
fatigue ?
high fever ?
swelled lymph nodes ?
#false ?
phlegm ?
throat irritation ?
redness of eyes ?
sinus pressure ?
runny nose ?
congestion ?
chest pain ?
loss of smell ?
muscle pain ?
c:\Users\Edwin Joshua Samraj\venv\Python310\lib\site-packages\sklearn\base.py:443: UserWarning: X has feature names, but KNeighborsClassifier was fitted without feature names
  warnings.warn(
you may have Common Cold
The common cold is a viral infection of your nose and throat (upper respiratory tract). It's usually harmless, although it might not feel that way. Many types of viruses can cause a common cold.
Take following precautions :
drink vitamin c rich drinks
take vapour
avoid cold food
keep fever in check
do you need another medical consultation (yes or no)?
!!!! thanks for using over application !!!!!

1 df_med = pd.read_csv('Medical_dataset/drugs_side_effects_drugs_com.csv')
2 disease_names = ["common cold", "acne"]
3 suggested_drugs = set() # Use a set to avoid duplicate drugs
4 for disease_name in disease_names:
5     # Filter the DataFrame directly using the symptoms
6     matched_conditions = df_med[df_med['medical_condition_description'].str.contains(disease_name, case=False, regex=False)]
7     if not matched_conditions.empty:
8         drugs_for_condition = matched_conditions['drug_name'].tolist()
9         suggested_drugs.update(drugs_for_condition) # Add drugs to the set
10 suggested_drugs = list(suggested_drugs)
11 print("Here are some suggested drugs: " + " ".join(suggested_drugs))
12 print(suggested_drugs)
✓ 02s

Here are some suggested drugs: Sudafed PE Pressure+Pain+Cough, Neuac, Nikki, Tuzistra XR, Decorel Forte Plus, acetaminophen / dextromethorphan / pseudoephedrine, Benadryl Allergy Plus Sinus Head
['Sudafed PE Pressure+Pain+Cough', 'Neuac', 'Nikki', 'Tuzistra XR', 'Decorel Forte Plus', 'acetaminophen / dextromethorphan / pseudoephedrine', 'Benadryl Allergy Plus Sinus Headache', 'Aphedrid',
```

The paper presents experimental results demonstrating the effectiveness and accuracy of the drug recommendation system. Evaluation metrics such as precision, recall, and F1-score are used to assess the performance of the system.

## Future Directions:

The paper discusses potential future directions for research, including improving the scalability and robustness of the system, integrating additional data sources, and exploring advanced NLP techniques for further enhancement of drug recommendation capabilities. And improving the front-end User-Interface in a more user-friendly manner.



## 8. SUMMARY

In this paper, we present a novel approach for drug recommendation utilizing the synergistic capabilities of state-of-the-art natural language processing (NLP) models, specifically GPT (Generative Pre-trained Transformer) and BioBERT, integrated within a Flask framework. The objective of our study is to develop an effective and user-friendly system for personalized drug recommendation, catering to the diverse needs of medical practitioners and patients.

Our methodology begins with comprehensive data collection, gathering relevant medical information such as symptoms, diseases, drug side effects, and drug names from various reliable sources. Subsequently, the collected data undergoes preprocessing to ensure cleanliness and structural coherence, laying the foundation for accurate analysis.

The core of our approach lies in the integration of advanced NLP models. GPT, renowned for its generative capabilities, is employed to generate personalized drug recommendations based on user-provided symptoms or medical conditions. Furthermore, we leverage BioBERT, a specialized variant of BERT tailored for biomedical text, to enhance drug recommendation accuracy within the medical domain.

To provide seamless interaction and accessibility, we implement the drug recommendation system within a Flask framework, facilitating an intuitive and user-friendly web interface. This interface allows users to input symptoms or medical conditions and receive personalized drug recommendations in a clear and understandable format.

Our experimental results demonstrate the effectiveness and accuracy of the proposed drug recommendation system. Through evaluation metrics such as precision, recall, and F1-score, we validate the system's performance and robustness, affirming its suitability for practical use in medical settings.

In conclusion, our study contributes to the field of medical informatics by offering a sophisticated yet accessible solution for drug recommendation. By harnessing the power of advanced NLP models and web technologies, our system provides a valuable resource for medical practitioners and patients, facilitating informed decision-making and personalized healthcare management.

## 9. REFERENCES

- [1] Sarasohn-Kahn, Jane. "The wisdom of patients: Health care meets online social media." (2008).
- [2] Gopalakrishnan, Vinodhini, and Chandrasekaran Ramaswamy. "Patient opinion mining to analyze drugs satisfaction using supervised learning." *Journal of applied research and technology* 15.4 (2017): 311-319.
- [3] Josue Uwimana, "Intelligent Cognitive Clinician", 2023 International Conference on the Cognitive Computing and Complex Data (ICCD), pp.185-190, 2023.
- [4] C. Silpa, B. Sravani, D. Vinay, C. Mounika and K. Poorvitha, "Drug Recommendation System in Medical Emergencies using Machine Learning," 2023 International Conference on Innovative Data Communication Technologies and Application (ICIDCA), Uttarakhand, India, 2023, pp. 107-112, doi: 10.1109/ICIDCA56705.2023.10099607.
- [5] <https://github.com/yashajoshi/Drug-Recommendation-using-Review-Mining>
- [6] [9] Doulaverakis, C., Nikolaidis, G., Kleontas, A. et al. GalenOWL: Ontology-based drug recommendations discovery. *J Biomed Semant* 3, 14 (2012). <https://doi.org/10.1186/2041-1480-3-14> [10]
- [7] Leilei Sun, Chuanren Liu, Chonghui Guo, Hui Xiong, and Yanming Xie. 2016. Data-driven Automatic Treatment Regimen Development and Recommendation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1865–1874. DOI:<https://doi.org/10.1145/2939672.2939866> [11]
- [8] V. Goel, A. K. Gupta and N. Kumar, "Sentiment Analysis of Multilingual Twitter Data using Natural Language Processing," 2018 8th International Conference on Communication Systems and Network Technologies (CSNT), Bhopal,
- [9] <https://medium.com/@marshettyruthvik/drug-recommendation-system-1b32d1cda680>
- [10] Ranschaert ER, Morozov S, Algra PR. *Artificial Intelligence in Medical Imaging: Opportunities, Applications and Risks*. Springer; 2019.
- [11] Obermeyer Z, Powers B, Vogeli C, Mullainathan S. Dissecting racial bias in an algorithm used to manage the health of populations. *Science*. 2019;366: 447–453.
- [12] Wen A, Fu S, Moon S, El Wazir M, Rosenbaum A, Kaggal VC, et al. Desiderata for delivering NLP to accelerate healthcare AI advancement and a Mayo Clinic NLP-as-a-

- service implementation. NPJ Digit Med. 2019;2: 130.
- [13] Fadhil A. Beyond Patient Monitoring: Conversational Agents Role in Telemedicine & Healthcare Support For Home-Living Elderly Individuals. arXiv [cs.CY]. 2018. Available: <http://arxiv.org/abs/1803.06000>
  - [14] Nicolas Bayerque G. A short history of chatbots and artificial intelligence. In: VentureBeat [Internet]. VentureBeat; 15 Aug 2016 [cited 1 Jul 2020]. Available: <https://venturebeat.com/2016/08/15/a-short-history-of-chatbots-and-artificial-intelligence/>
  - [15] Epstein J, Klinkenberg WD. From Eliza to Internet: a brief history of computerized assessment. Comput Human Behav. 2001;17: 295–314.
  - [16] Weizenbaum J. ELIZA — a computer program for the study of natural language communication between man and machine. Commun ACM. 1966;9: 36–45.
  - [17] Yao M. 6 Technical Approaches For Building Conversational AI. In: TOPBOTS [Internet]. 11 Sep 2018 [cited 26 Jun 2020]. Available: <https://www.topbots.com/building-conversational-ai/>
  - [18] Wolf T. How to build a State-of-the-Art Conversational AI with Transfer Learning. In: HuggingFace [Internet]. 9 May 2019 [cited 21 Aug 2020]. Available: <https://medium.com/huggingface/how-to-build-a-state-of-the-art-conversational-ai-with-transfer-learning-2d818ac26313>
  - [19] Sanh V. 🐼 Smaller, faster, cheaper, lighter: Introducing DistilBERT, a distilled version of BERT. In: HuggingFace [Internet]. 28 Aug 2019 [cited 21 Aug 2020]. Available: <https://medium.com/huggingface/distilbert-8cf3380435b5>

## APPENDIX A – SAMPLE CODE

```
import pandas as pd
import numpy as np
from nltk.corpus import wordnet
import csv
import json
import itertools
from spacy.lang.en.stop_words import STOP_WORDS
import spacy
import joblib
import zlib
from flask import Flask, render_template, request, session
# import deployment

app = Flask(__name__)

nlp = spacy.load('en_core_web_sm')

# save data
data = {"users": []}
with open('DATA.json', 'w') as outfile:
    json.dump(data, outfile)

def write_json(new_data, filename='DATA.json'):
    with open(filename, 'r+') as file:
        # First we load existing data into a dict.
        file_data = json.load(file)
        # Join new_data with file_data inside emp_details
        file_data["users"].append(new_data)
        # Sets file's current position at offset.
        file.seek(0)
        # convert back to json.
        json.dump(file_data, file, indent=4)

with open('Medical_dataset/symptom_Description.csv') as csv_file:
    csv_reader = csv.reader(csv_file, delimiter=',')
    disease_names = [row[0] for row in csv_reader]

df_tr = pd.read_csv('Medical_dataset/Training.csv')
df_tt = pd.read_csv('Medical_dataset/Testing.csv')

df_med = pd.read_csv('Medical_dataset/drugs_side_effects_drugs_com.csv')

symp = []
disease = []
for i in range(len(df_tr)):
```

```

symp.append(df_tr.columns[df_tr.iloc[i] == 1].to_list())
disease.append(df_tr.iloc[i, -1])

# # I- GET ALL SYMPTOMS

all_symp_col = list(df_tr.columns[:-1])

def clean_symp(sym):
    return sym.replace('_', ' ').replace('.1', '').replace('(typhos)',
    '').replace('yellowish', 'yellow').replace(
        'yellowing', 'yellow')

all_symp = [clean_symp(sym) for sym in (all_symp_col)]

def preprocess(doc):
    nlp_doc = nlp(doc)
    d = []
    for token in nlp_doc:
        if (not token.text.lower() in STOP_WORDS and token.text.isalpha()):
            d.append(token.lemma_.lower())
    return ' '.join(d)

all_symp_pr = [preprocess(sym) for sym in all_symp]

# associate each processed symp with column name
col_dict = dict(zip(all_symp_pr, all_symp_col))

# II- Syntactic Similarity

# Returns all the subsets of a set. This is a generator.
# {1,2,3}->[{}, {1}, {2}, {3}, {1,3}, {1,2}, ...]
def powerset(seq):
    if len(seq) <= 1:
        yield seq
        yield []
    else:
        for item in powerset(seq[1:]):
            yield [seq[0]] + item
            yield item

# Sort list based on length
def sort(a):
    for i in range(len(a)):

```

```

        for j in range(i + 1, len(a)):
            if len(a[j]) > len(a[i]):
                a[i], a[j] = a[j], a[i]
a.pop()
return a

# find all permutations of a list
def permutations(s):
    permutations = list(itertools.permutations(s))
    return ([' '.join(permutation) for permutation in permutations])

# check if a txt and all different combination if it exists in processed symp
list
def DoesExist(txt):
    txt = txt.split(' ')
    combinations = [x for x in powerset(txt)]
    sort(combinations)
    for comb in combinations:
        # print(permutations(comb))
        for sym in permutations(comb):
            if sym in all_symp_pr:
                # print(sym)
                return sym
    return False

# Jaccard similarity 2docs
def jaccard_set(str1, str2):
    list1 = str1.split(' ')
    list2 = str2.split(' ')
    intersection = len(list(set(list1).intersection(list2)))
    union = (len(list1) + len(list2)) - intersection
    return float(intersection) / union

# apply vanilla jaccard to symp with all corpus
def syntactic_similarity(symp_t, corpus):
    most_sim = []
    poss_sym = []
    for symp in corpus:
        d = jaccard_set(symp_t, symp)
        most_sim.append(d)
    order = np.argsort(most_sim)[::-1].tolist()
    for i in order:
        if DoesExist(symp_t):
            return 1, [corpus[i]]
        if corpus[i] not in poss_sym and most_sim[i] != 0:

```

```

        poss_sym.append(corpus[i])
    if len(poss_sym):
        return 1, poss_sym
    else:
        return 0, None

# check a pattern if it exists in processed symp list
def check_pattern(inp, dis_list):
    import re
    pred_list = []
    ptr = 0
    patt = "^" + inp + "$"
    regexp = re.compile(inp)
    for item in dis_list:
        if regexp.search(item):
            pred_list.append(item)
    if (len(pred_list) > 0):
        return 1, pred_list
    else:
        return ptr, None

```

# III- Semantic Similarity

```

from nltk.wsd import lesk
from nltk.tokenize import word_tokenize

```

```

def WSD(word, context):
    sens = lesk(context, word)
    return sens

```

```

# semantic similarity 2docs
def semanticD(doc1, doc2):
    doc1_p = preprocess(doc1).split(' ')
    doc2_p = preprocess(doc2).split(' ')
    score = 0
    for tock1 in doc1_p:
        for tock2 in doc2_p:
            syn1 = WSD(tock1, doc1)
            syn2 = WSD(tock2, doc2)
            if syn1 is not None and syn2 is not None:
                x = syn1.wup_similarity(syn2)
                # x=syn1.path_similarity((syn2))
                if x is not None and x > 0.25:
                    score += x

```

```

    return score / (len(doc1_p) * len(doc2_p))

# apply semantic similarity to symp with all corpus
def semantic_similarity(symp_t, corpus):
    max_sim = 0
    most_sim = None
    for symp in corpus:
        d = semanticD(symp_t, symp)
        if d > max_sim:
            most_sim = symp
            max_sim = d
    return max_sim, most_sim

# given a symp suggest possible synonyms
def suggest_syn(sym):
    symp = []
    synonyms = wordnet.synsets(sym)
    lemmas = [word.lemma_names() for word in synonyms]
    lemmas = list(set(itertools.chain(*lemmas)))
    for e in lemmas:
        res, sym1 = semantic_similarity(e, all_symp_pr)
        if res != 0:
            symp.append(sym1)
    return list(set(symp))

# One-Hot-Vector dataframe
def OHV(cl_sym, all_sym):
    l = np.zeros([1, len(all_sym)])
    for sym in cl_sym:
        l[0, all_sym.index(sym)] = 1
    return pd.DataFrame(l, columns=all_sym)

def contains(small, big):
    a = True
    for i in small:
        if i not in big:
            a = False
    return a

# list of symptoms --> possible diseases
def possible_diseases(l):
    poss_dis = []
    for dis in set(disease):
        if contains(l, symVONdisease(df_tr, dis)):

```



```

        poss_dis.append(dis)
    return poss_dis

# disease --> all symptoms
def symVONdisease(df, disease):
    ddf = df[df.prognosis == disease]
    m2 = (ddf == 1).any()
    return m2.index[m2].tolist()

## disease --> list of recommended drugs
def suggest_drugs(symptoms):
    suggested_drugs = set() # Use a set to avoid duplicate drugs
    for disease_name in disease_names:
        # Filter the DataFrame directly using the symptoms
        matched_conditions =
df_med[df_med['medical_condition_description'].str.contains(disease_name,
case=False, regex=False)]
        if not matched_conditions.empty:
            drugs_for_condition = matched_conditions['drug_name'].tolist()
            suggested_drugs.update(drugs_for_condition) # Add drugs to the
set
    suggested_drugs = list(suggested_drugs)
    print("Here are some suggested drugs: " + ", ".join(suggested_drugs))
    del suggested_drugs[10:]
    return suggested_drugs # Convert set back to list for consistent output

# IV- Prediction Model (KNN)
# load model
knn_clf = joblib.load('model/knn.pkl')

# ## VI- SEVERITY / DESCRIPTION / PRECAUTION
# get dictionaries for severity-description-precaution for all diseases

severityDictionary = dict()
description_list = dict()
precautionDictionary = dict()

def getDescription():
    global description_list
    with open('Medical_dataset/symptom_Description.csv') as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            _description = {row[0]: row[1]}
            description_list.update(_description)

```

```

def getSeverityDict():
    global severityDictionary
    with open('Medical_dataset/symptom_severity.csv') as csv_file:

        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        try:
            for row in csv_reader:
                _diction = {row[0]: int(row[1])}
                severityDictionary.update(_diction)
        except:
            pass

def getprecautionDict():
    global precautionDictionary
    with open('Medical_dataset/symptom_precaution.csv') as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            _prec = {row[0]: [row[1], row[2], row[3], row[4]]}
            precautionDictionary.update(_prec)

# load dictionaries
getSeverityDict()
getprecautionDict()
getDescription()

# calcul patient condition
def calc_condition(exp, days):
    sum = 0
    for item in exp:
        if item in severityDictionary.keys():
            sum = sum + severityDictionary[item]
    if ((sum * days) / (len(exp)) > 13):
        return 1
        print("You should take the consultation from doctor. ")
    else:
        return 0
        print("It might not be that bad but you should take precautions.")

# print possible symptoms
def related_sym(psym1):
    s = "could you be more specific, <br>"

```

```

i = len(s)
for num, it in enumerate(psym1):
    s += str(num) + ") " + clean_symp(it) + "<br>"
if num != 0:
    s += "Select the one you meant."
    return s
else:
    return 0

@app.route("/")
def home():
    return render_template("home.html")

@app.route("/get")
def get_bot_response():
    s = request.args.get('msg')
    if "step" in session:
        if session["step"] == "Q_C":
            name = session["name"]
            age = session["age"]
            gender = session["gender"]
            session.clear()
            if s == "q":
                "Thank you for using ower web site Mr/Ms " + name
            else:
                session["step"] = "FS"
                session["name"] = name
                session["age"] = age
                session["gender"] = gender
    if s.upper() == "OK":
        return "What is your name ?"
    if 'name' not in session and 'step' not in session:
        session['name'] = s
        session['step'] = "age"
        return "How old are you? "
    if session["step"] == "age":
        session["age"] = int(s)
        session["step"] = "gender"
        return "Can you specify your gender ?"
    if session["step"] == "gender":
        session["gender"] = s
        session["step"] = "Depart"
    if session['step'] == "Depart":
        session['step'] = "BFS"
        return "Well, Hello again Mr/Ms " + session[
            "name"] + ", now I will be asking some few questions about your
            symptoms to see what you should do. Tap S to start diagnostic!"

```

```

if session['step'] == "BFS":
    session['step'] = "FS" # first symp
    return "Can you precise your main symptom Mr/Ms " + session["name"] +
" ?"
if session['step'] == "FS":
    sym1 = s
    sym1 = preprocess(sym1)
    sim1, psym1 = syntactic_similarity(sym1, all_symp_pr)
    temp = [sym1, sim1, psym1]
    session['FSY'] = temp # info du 1er symptome
    session['step'] = "SS" # second symptomee
    if sim1 == 1:
        session['step'] = "RS1" # related_sym1
        s = related_sym(psym1)
        if s != 0:
            return s
        else:
            return "You are probably facing another symptom, if so, can you
specify it?"
    if session['step'] == "RS1":
        temp = session['FSY']
        psym1 = temp[2]
        psym1 = psym1[int(s)]
        temp[2] = psym1
        session['FSY'] = temp
        session['step'] = 'SS'
        return "You are probably facing another symptom, if so, can you
specify it?"
    if session['step'] == "SS":
        sym2 = s
        sym2 = preprocess(sym2)
        sim2 = 0
        psym2 = []
        if len(sym2) != 0:
            sim2, psym2 = syntactic_similarity(sym2, all_symp_pr)
        temp = [sym2, sim2, psym2]
        session['SSY'] = temp # info du 2eME symptome(sym,sim,psym)
        session['step'] = "semantic" # face semantic
        if sim2 == 1:
            session['step'] = "RS2" # related sym2
            s = related_sym(psym2)
            if s != 0:
                return s
    if session['step'] == "RS2":
        temp = session['SSY']
        psym2 = temp[2]
        psym2 = psym2[int(s)]
        temp[2] = psym2

```

```

    session['SSY'] = temp
    session['step'] = "semantic"
if session['step'] == "semantic":
    temp = session["FSY"] # recuperer info du premier
    sym1 = temp[0]
    sim1 = temp[1]
    temp = session["SSY"] # recuperer info du 2 eme symptome
    sym2 = temp[0]
    sim2 = temp[1]
    if sim1 == 0 or sim2 == 0:
        session['step'] = "BFsim1=0"
    else:
        session['step'] = 'PD' # to possible_diseases
if session['step'] == "BFsim1=0":
    if sim1 == 0 and len(sym1) != 0:
        sim1, psym1 = semantic_similarity(sym1, all_symp_pr)
        temp = []
        temp.append(sym1)
        temp.append(sim1)
        temp.append(psym1)
        session['FSY'] = temp
        session['step'] = "sim1=0" # process of semantic similarity=1 for
first sympt.
    else:
        session['step'] = "BFsim2=0"
if session['step'] == "sim1=0": # semantic no => suggestion
    temp = session["FSY"]
    sym1 = temp[0]
    sim1 = temp[1]
    if sim1 == 0:
        if "suggested" in session:
            sugg = session["suggested"]
            if s == "yes":
                psym1 = sugg[0]
                sim1 = 1
                temp = session["FSY"]
                temp[1] = sim1
                temp[2] = psym1
                session["FSY"] = temp
                sugg = []
            else:
                del sugg[0]
        if "suggested" not in session:
            session["suggested"] = suggest_syn(sym1)
            sugg = session["suggested"]
        if len(sugg) > 0:
            msg = "are you experiencing any " + sugg[0] + "?"
            return msg

```

```

    if "suggested" in session:
        del session["suggested"]
    session['step'] = "BFsim2=0"
if session['step'] == "BFsim2=0":
    temp = session["SSY"] # recuperer info du 2 eme symptome
    sym2 = temp[0]
    sim2 = temp[1]
    if sim2 == 0 and len(sym2) != 0:
        sim2, psym2 = semantic_similarity(sym2, all_symp_pr)
        temp = []
        temp.append(sym2)
        temp.append(sim2)
        temp.append(psym2)
        session['SSY'] = temp
        session['step'] = "sim2=0"
    else:
        session['step'] = "TEST"
if session['step'] == "sim2=0":
    temp = session["SSY"]
    sym2 = temp[0]
    sim2 = temp[1]
    if sim2 == 0:
        if "suggested_2" in session:
            sugg = session["suggested_2"]
            if s == "yes":
                psym2 = sugg[0]
                sim2 = 1
                temp = session["SSY"]
                temp[1] = sim2
                temp[2] = psym2
                session["SSY"] = temp
                sugg = []
            else:
                del sugg[0]
        if "suggested_2" not in session:
            session["suggested_2"] = suggest_syn(sym2)
            sugg = session["suggested_2"]
        if len(sugg) > 0:
            msg = "Are you experiencing " + sugg[0] + "?"
            session["suggested_2"] = sugg
            return msg
    if "suggested_2" in session:
        del session["suggested_2"]
    session['step'] = "TEST" # test if semantic and syntactic and
suggestion not found
if session['step'] == "TEST":
    temp = session["FSY"]
    sim1 = temp[1]

```

```

psym1 = temp[2]
temp = session["SSY"]
sim2 = temp[1]
psym2 = temp[2]
if sim1 == 0 and sim2 == 0:
    # GO TO THE END
    result = None
    session['step'] = "END"
else:
    if sim1 == 0:
        psym1 = psym2
        temp = session["FSY"]
        temp[2] = psym2
        session["FSY"] = temp
    if sim2 == 0:
        psym2 = psym1
        temp = session["SSY"]
        temp[2] = psym1
        session["SSY"] = temp
    session['step'] = 'PD' # to possible_diseases
if session['step'] == 'PD':
    # MAYBE THE LAST STEP
    # create patient symp list
    temp = session["FSY"]
    sim1 = temp[1]
    psym1 = temp[2]
    temp = session["SSY"]
    sim2 = temp[1]
    psym2 = temp[2]
    print("hey2")
    if "all" not in session:
        session["asked"] = []
        session["all"] = [col_dict[psym1], col_dict[psym2]]
        print(session["all"])
    session["diseases"] = possible_diseases(session["all"])
    print(session["diseases"])
    all_sym = session["all"]
    diseases = session["diseases"]
    dis = diseases[0]
    session["dis"] = dis
    session['step'] = "for_dis"
if session['step'] == "DIS":
    if "symv" in session:
        if len(s) > 0 and len(session["symv"]) > 0:
            symts = session["symv"]
            all_sym = session["all"]
            if s == "yes":
                all_sym.append(symts[0])

```

```

        session["all"] = all_sym
        print(possible_diseases(session["all"]))
        del symts[0]
        session["symv"] = symts
    if "symv" not in session:
        session["symv"] = symVONdisease(df_tr, session["dis"])
    if len(session["symv"]) > 0:
        if symts[0] not in session["all"] and symts[0] not in
session["asked"]:
            asked = session["asked"]
            asked.append(symts[0])
            session["asked"] = asked
            symts = session["symv"]
            msg = "Are you experiencing " + clean_symp(symts[0]) + "?"
            return msg
        else:
            del symts[0]
            session["symv"] = symts
            s = ""
            print("HANAAA")
            return get_bot_response()
    else:
        PD = possible_diseases(session["all"])
        diseases = session["diseases"]
        if diseases[0] in PD:
            session["testpred"] = diseases[0]
            PD.remove(diseases[0])
            #         diseases=session["diseases"]
            #         del diseases[0]
            session["diseases"] = PD
            session['step'] = "for_dis"
    if session['step'] == "for_dis":
        diseases = session["diseases"]
        if len(diseases) <= 0:
            session['step'] = 'PREDICT'
        else:
            session["dis"] = diseases[0]
            session['step'] = "DIS"
            session["symv"] = symVONdisease(df_tr, session["dis"])
            return get_bot_response() # turn around sympt of dis
            # predict possible diseases
    if session['step'] == "PREDICT":
        result = knn_clf.predict(OHV(session["all"], all_symp_col))
        session['step'] = "END"
    if session['step'] == "END":
        if result is not None:
            if result[0] != session["testpred"]:
                session['step'] = "Q_C"

```



```

        return "as you provide me with few symptoms, I am sorry to
announce that I cannot predict your " \
        "disease for the moment!!! <br> Can you specify more
about what you are feeling or Tap q to " \
        "stop the conversation "
    session['step'] = "Description"
    session["disease"] = result[0]
    return "Well Mr/Ms " + session["name"] + ", you may have " +
result[
        0] + ". Tap D to get a description of the disease ."
    else:
        session['step'] = "Q_C" # test if user want to continue the
conversation or not
        return "can you specify more what you feel or Tap q to stop the
conversation"
    if session['step'] == "Description":
        y = {"Name": session["name"], "Age": session["age"], "Gender":
session["gender"], "Disease": session["disease"],
            "Sympts": session["all"]}
        write_json(y)
        session['step'] = "Severity"
        if session["disease"] in description_list.keys():
            return description_list[session["disease"]] + " \n <br> How many
days have you had symptoms?"
        else:
            if " " in session["disease"]:
                session["disease"] = session["disease"].replace(" ", "_")
            return "please visit <a href='" + "https://en.wikipedia.org/wiki/"
+ session["disease"] + "'> here </a>"
        if session['step'] == "Severity":
            session['step'] = 'FINAL'
            if calc_condition(session["all"], int(s)) == 1:
                return "you should take the consultation from doctor <br> Tap q to
check for any suggested drugs"
            else:
                msg = 'Nothing to worry about, but you should take the following
precautions :<br> '
                i = 1
                for e in precautionDictionary[session["disease"]]:
                    msg += '\n ' + str(i) + ' - ' + e + '<br>'
                    i += 1
                msg += ' Tap q to end'
                return msg

    if session['step'] == "FINAL":
        session['step'] = "SUGGEST_DRUGS"
        suggested_drugs = suggest_drugs(session["disease"]) # Get suggested
drugs based on symptoms

```

```

        session["suggested_drugs"] = suggested_drugs
    if suggested_drugs:
        return "I have also found some drugs that might help you. Would you like to see them?"

    elif session['step'] == "SUGGEST_DRUGS":
        session['step'] = "FINISH"
        if s.lower() == "yes":
            suggested_drugs = session["suggested_drugs"]
            if suggested_drugs:
                return "Here are some suggested drugs: " + ", ".join(suggested_drugs)
            else:
                return "I'm sorry, I couldn't find any suggested drugs for your symptoms."
        else:
            return "Okay, let me know if you need anything else."

    if session['step'] == "FINISH":
        session['step'] = "BYE"
        return "Your diagnosis was perfectly completed. Do you need another medical consultation (yes or no)? "
    if session['step'] == "BYE":
        name = session["name"]
        age = session["age"]
        gender = session["gender"]
        session.clear()
        if s.lower() == "yes":
            session["gender"] = gender
            session["name"] = name
            session["age"] = age
            session['step'] = "FS"
            return "HELLO again Mr/Ms " + session["name"] + " Please tell me your main symptom. "
        else:
            return "THANKS Mr/Ms " + name + " for using me for more information please contact <b> +91-9840503206</b>"

if __name__ == "__main__":
    import random # define the random module
    import string
    S = 10 # number of characters in the string.
    # call random.choices() string module to find the string in Uppercase + numeric data.
    ran = ''.join(random.choices(string.ascii_uppercase + string.digits, k=S))
    # chat_sp()
    app.secret_key = str(ran)
    app.run()

```