A Mini-Project Report
on

MEDOPT

Submitted for partial fulfillment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

BY

Mr. Mohammed Abdul Muqtadeer (2451-20-733-070)
Mr. Ahmed Abdul Hamees Sajid (2451-20-733-072)
Mr. Mohammed Abdul Muqsith (2451-20-733-084)

Under the guidance of

I.Navakanth
Assistant Professor, MVSREC

Department of Computer Science and Engineering
Maturi Venkata Subba Rao(MVSR) Engineering College
(An Autonomous Institution)
(Affiliated to Osmania University & Recognized by AICTE)
Nadergul(V), Balapur(M), RR Dist. Hyderabad – 501 510

2022-23.

Maturi Venkata Subba Rao (MVSR) Engineering College
(An Autonomous Institution)

(Affiliated to Osmania University, Hyderabad)
Nadergul(V), Hyderabad-501510

Certificate

This is to certify that the mini-project work entitled "MEDOPT: Recommendation System**"** is a bonafide work carried out by **Mr. Mohammed Abdul Muqtadeer (2451-20-733-070), Mr. Ahmed Abdul Hamees Sajid (2451-20-733-072)**, **Mr. Mohammed Abdul Muqsith (2451-20-733-084)** in partial fulfillment of the requirements for the award of degree of **BACHELOR OF ENGINEERING IN COMPUTER SCIENCE AND ENGINEERING** from Maturi Venkata Subba Rao (MVSR) Engineering College, affiliated to OSMANIA UNIVERSITY, Hyderabad, under our guidance and supervision.

The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

Internal Guide
I.Navakanth
Asisstant Professor

Project Coordinators

I.Navakanth, Assistant Professor

External Examiner

DECLARATION

This is to certify that the work reported in the present mini-project entitled "**MEDOPT"** is a record of bonafide work done by us in the Department of Computer Science and Engineering, Maturi Venkata Subba Rao (MVSR) Engineering College, Osmania University. The reports are based on the mini-project work done entirely by us and not copied from any other source.

The results embodied in this mini-project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

Mohammed Abdul Muqtadeer	Ahmed Abdul Hamees Sajid	Mohammed Abdul Muqsith
2451-20-733-070	2451-20-733-072	2451-20-733-084

ACKNOWLEDGEMENTS

We would like to express our sincere gratitude and indebtedness to my mini-project guide **I.Navakanth** for his valuable suggestions and interest throughout the course of this mini-project.

We are also thankful to our principal **Dr. G Kanaka Durga** and **Mr. Prasanna Kumar**, Professor and Head, Department of Computer Science and Engineering, Maturi Venkata Subba Rao (MVSR) Engineering College, Hyderabad for providing excellent infrastructure for completing this mini-project successfully as a part of our B.E. Degree (CSE). We would like to thank our mini-project coordinators **M.Anupama, I.Navakanth, K.Murali Krishna** for their constant monitoring, guidance and support.

We convey our heartfelt thanks to the lab staff for allowing us to use the required equipment whenever needed.

Finally, we would like to take this opportunity to thank our families for their support through the work. We sincerely acknowledge and thank all those who gave directly or indirectly their support in completion of this work.

Mohammed Abdul Muqtadeer (2451-20-733-070)
Ahmed Abdul Hameed Sajid (2451-20-733-072)
Mohammed Abdul Muqsith (2451-20-733-084)

ABSTRACT

The rapid advancements in technology and the increasing availability of numerous drugs have paved the way for innovative solutions in healthcare services. In this project, we propose a Medicine Recommendation System **MEDOPT** that leverages machine learning techniques to provide medication recommendations for patients.

The primary goal of the **MEDOPT** is to assist healthcare professionals in making informed decisions when prescribing medications and for patients to view alternatives.

The **Medopt** utilizes a comprehensive database of medications, including information on drug interactions and side effects, to ensure the safety and efficacy of recommended meds.

The system continuously learns from new data and updates its knowledge base to ensure up-to-date and reliable recommendations.

TABLE OF CONTENT

CONTENTS

LIST OF TABLES

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 Problem Statement

Prescribing the most suitable medication for a patient can be a challenging task for healthcare professionals due to the vast array of available medications and the varying responses of individuals to different drugs. Furthermore, patients may have specific contraindications, allergies, or previous adverse reactions that need to be taken into consideration. Inadequate medication choices can result in suboptimal treatment outcomes, increased healthcare costs, and potential harm to patients.

Existing medication recommendation systems often focus on drug interactions or generic guidelines, providing limited personalized recommendations tailored to individual patient characteristics. Consequently, healthcare professionals may face difficulties in identifying suitable alternative medications when the initial prescription is not optimal or contraindicated.

Therefore, there is a need for an advanced Medicine Alternative Recommendation System that leverages patient-specific information, medical knowledge, and advanced algorithms to suggest alternative medications when the initial prescription is not suitable or feasible. Such a system would assist healthcare professionals in making informed decisions and enhance patient safety and treatment efficacy.

The primary goal of the **MEDOPT** is to provide healthcare professionals with accurate and personalized alternative medication options that align with the individual patient's needs. By integrating a vast database of medications, patient information, and clinical guidelines, the **MEDOPT** aims to improve treatment outcomes, minimize adverse drug reactions, and optimize patient care.

1.2 Existing System

Medical recommendation websites have gained significant popularity as platforms that provide users with information and suggestions related to medical conditions, treatments, and healthcare providers. These websites aim to assist users in making informed decisions about their health and medical needs.

However, it's important to note that the information provided on such websites should not replace professional medical advice and consultation with healthcare professionals.

1mg is an online healthcare platform in India that provides a range of services related to medicines, healthcare products, and diagnostic tests. Some of the key functions and features of the 1mg website include:

Medicine Ordering: Users can search for and order a wide range of medicines, including prescription drugs, over-the-counter medications, and healthcare products, directly from the website.

Medicine Information: 1mg provides comprehensive information about medicines, including their uses, dosage instructions, side effects, and precautions.

Digital Prescriptions: The platform allows users to upload their prescriptions digitally and order the prescribed medicines online.

Healthcare Product Shopping: In addition to medicines, 1mg offers a wide range of healthcare products such as wellness and nutrition products, personal care items, medical devices, and more. Users can explore various categories, read product descriptions, and make purchases conveniently.

1.3 Proposed System

Medopt is a website that focuses on advising alternatives for drugs prescribed. The website would also give an aid for customer to finder cheaper medicines with same chemical formula. Each medicine will display its various users and adverse side-effects.

The proposed Medical Alternative Recommendation System (MEDOPT) aims to address the challenge of providing personalized and informed alternative medication options for patients when the initial prescription is not suitable or feasible.

The system leverages patient-specific data, medical knowledge, and advanced algorithms to generate accurate and tailored alternative medication recommendations, enhancing patient safety and treatment efficacy.

Key Features of the MEDOPT:

Medication Database: MEDOPT integrates a comprehensive and up-to-date database of medications, including alternative options, contraindications, drug interactions, and side effects. This database serves as a reliable source of information for generating alternative medication recommendations.

Advanced Algorithmic Approaches: It employs advanced machine learning algorithms, such as COSINE SIMILARITY to analyze patient profiles and medication data. These algorithms identify patterns, correlations, and similarities to predict alternative medications that align with input drug characteristics.

Treatment Effectiveness Evaluation: It evaluates the effectiveness of alternative medications by leveraging clinical data, research studies, and real-world evidence. This evaluation helps ensure that the recommended alternatives have demonstrated efficacy and safety profiles.

User-Friendly Interface: The MEDOPT provides a user-friendly interface for healthcare professionals to input patient data and receive alternative medication recommendations. The interface is intuitive, easy to navigate, and designed to seamlessly integrate into clinical workflows, minimizing any disruption to existing practices.

Real-Time Updates: MEDOPT continuously updates its knowledge base, incorporating new research findings, clinical guidelines, and emerging medications

1.4 Scope of MEDOPT

Some healthcare organizations and professional associations have developed guidelines and recommendations for integrating specific alternative therapies into conventional medical practice. These guidelines help healthcare providers make informed decisions about incorporating alternative approaches based on the available evidence and patient needs.

Patient interest and demand for alternative therapies influence the extent to which they are integrated into medical practice. As patients seek alternative approaches, healthcare providers may be more open to incorporating them into treatment plans or referring patients to qualified practitioners of alternative medicine.

Alternative therapies can carry risks, especially when used without proper medical supervision or in combination with conventional treatments. For example, herbal remedies may interact with prescription medications, leading to adverse reactions. It is crucial to consult with healthcare professionals who have expertise in both conventional and alternative medicine to minimize these risks.

Alternative systems and conventional medicine often operate separately, resulting in a lack of collaboration and integration. Integrative medicine, which combines evidence-based conventional medicine with alternative approaches, is an emerging field that seeks to bridge this gap. However, achieving effective integration remains a challenge in many healthcare systems.

Efforts are being made to conduct scientific research and clinical trials to evaluate the safety, efficacy, and effectiveness of various alternative therapies. This helps in establishing evidence-based practices and integrating them into mainstream medicine where appropriate. However, it is important to note that not all alternative therapies have sufficient scientific evidence supporting their use.

CHAPTER II

2. TOOLS AND TECHNOLOGIES

A Medicine Recommendation System (MEDOPT) utilizes a combination of tools and technologies to effectively analyze patient data, medication information, and generate accurate recommendations. Here are some commonly used tools and technologies in the development of an MEDOPT:

Programming Languages: Commonly used languages include Python was used which offer a wide range of libraries, frameworks, and tools for data analysis, machine learning, and web development.

Machine Learning and Data Analysis Libraries: Libraries such as scikit-learn, provide powerful machine learning algorithms and tools for data preprocessing, feature selection, model training, and evaluation.

Natural Language Processing (NLP): NLP techniques are essential for extracting meaningful information from medical literature, clinical guidelines, and patient data. Tools such as NLTK (Natural Language Toolkit) assist in text processing, entity recognition, and sentiment analysis.

Web Development Frameworks: Web development frameworks such as Django are used to build user interfaces and interactive web applications for healthcare professionals and patients to interact with the MEDOPT. This frameworks simplify the development process and enable seamless integration with backend systems.

Data Visualization Libraries: Data visualization libraries like Matplotlib assist in creating interactive and visually appealing visualizations of patient data, medication trends, and treatment outcomes.

2.1 Literature Survey

Literature Survey for Medicine Alternative Recommendation Systems:

"A Review of Decision Support Systems for Medicine Substitution" by F. Takeda et al. (2019). This paper provides an overview of decision support systems (DSS) for medicine substitution. It discusses different approaches and techniques used in existing systems, including rule-based systems, knowledge-based systems, and machine learning-based systems.

"Personalized Medicine Recommender Systems: A Systematic Review" by V. Peska et al. (2018). This systematic review explores personalized medicine recommender systems and their applications. It discusses various approaches, such as content-based filtering, collaborative filtering, and hybrid methods.

"A Survey on Medical Expert Systems: Literature Review and Challenges" by S. Abedin et al. (2018). This survey provides an overview of medical expert systems, including medicine recommendation systems. It discusses the different types of expert systems. The review highlights the potential benefits of expert systems in healthcare decision-making and identifies challenges.

"Pharmacogenomics-Based Personalized Medicine: A Literature Review" by P. Lam et al. (2019). This literature review focuses on pharmacogenomics-based personalized medicine, which involves tailoring medication choices based on individual genetic characteristics. The paper discusses the potential of pharmacogenomics to improve medication selection and minimize adverse drug reactions.

"Medicine Recommendation System: A Review" by S. Banerjee et al. (2019)
This review paper provides an overview of medicine recommendation systems and their potential in healthcare.

These literature sources provide valuable insights into the different approaches, challenges, and potential benefits of medicine alternative recommendation systems. They highlight the importance of personalized medicine recommendations, integration with clinical decision-making processes, and the need for accurate and up-to-date medical knowledge.

2.2 Hardware Requirements

The hardware requirements for a Medicine Recommendation System (MRS) can vary depending on factors such as the scale of the system, the complexity of algorithms, the amount of data to be processed, and the expected user load. Here are some key hardware components we required:

Server Infrastructure: MEDOPT typically requires a robust server infrastructure to handle the computational demands of processing large datasets, running complex algorithms, and serving multiple concurrent users.

Processing Power: The MRS may benefit from high-performance processors (e.g., multi-core CPUs or GPUs) to handle computationally intensive tasks such as machine learning algorithms, data analysis, and real-time recommendations.

Memory (RAM): Sufficient RAM is crucial for efficiently processing and storing large datasets. The size of the dataset, the complexity of algorithms, and the number of concurrent users accessing the system will determine the required amount of RAM. Adequate memory helps in faster data retrieval, caching, and running resource-intensive algorithms.

Storage: The MRS needs sufficient storage capacity to store and manage the medication database, patient data, and other relevant information.

Security Measures: Healthcare systems handle sensitive patient data, and security is of utmost importance. The hardware setup should include appropriate security measures, such as firewalls.

2.3 Software Requirements

Visual Studio Code (VS Code) is a popular and powerful code editor that offered several benefits for our project.
VS Code provides numerous features that aid in writing code efficiently. It offers intelligent code completion, syntax highlighting, code formatting, and built-in Git integration, among others.

Jupyter Notebook in our project offered several advantages, especially as we were working with data analysis, exploration
Jupyter Notebook supports various data analysis libraries like NumPy, Pandas, Matplotlib, and Seaborn. These libraries offer powerful capabilities for data manipulation, analysis, and visualization, made it easier to process and understand our data.

Figma is a powerful design and prototyping tool that can be beneficial for various aspects of our project, especially if it involves user interface (UI) and user experience (UX) design.
Figma is a cloud-based design tool that allows multiple team members to work simultaneously on the same project. It enables real-time collaboration, making it easier for designers, developers, and stakeholders to collaborate.

The **Django** web framework is a free, open source framework that can speed up development of a web application being built in the Python programming language.
Starting with the Django web framework is more efficient way to build a web app than starting from scratch, which requires building the backend, APIs, javascript and sitemaps.

CHAPTER III

SYSTEM DESIGN

System design for a MEDOPT involves creating a software solution that can suggest alternative medicines to users based on the chemical formula.
Here are some key design components of our system:

User requirements: Understand the needs of the users, such as patients, doctors, or pharmacists, who will interact with the recommendation system.

Data collection and integration: Gather relevant data from various sources, including medical databases, research papers, user feedback, and expert knowledge. This data will serve as the foundation for the recommendation algorithm.

Recommendation algorithm: Develop an algorithm that can analyze user inputs, such as symptoms, medical history, and current medications, to suggest suitable alternatives. Consider factors such as effectiveness, side effects, drug interactions, cost, and availability. Machine learning techniques, such as collaborative filtering or content-based filtering, can be employed to improve recommendation accuracy.

Medical knowledge base: Create a comprehensive database of medicines, including information about their composition, indications, contraindications, dosage, and potential interactions.

User interface: an intuitive and user-friendly interface for users to interact with the recommendation system. Consider the specific needs of different user groups, such as patients or healthcare professionals, and provide appropriate features, such as advanced search options, filters, and medication comparisons.

Continuous improvement and feedback: Incorporate feedback mechanisms to collect user ratings, reviews, and feedback on the recommended alternatives. Use this feedback to improve the recommendation system over time, making it more personalized and accurate.

Testing and evaluation: Thoroughly test the recommendation system to ensure its accuracy, reliability, and performance. Conduct usability testing with users to gather feedback and refine the user experience.

3.1 Flow Charts

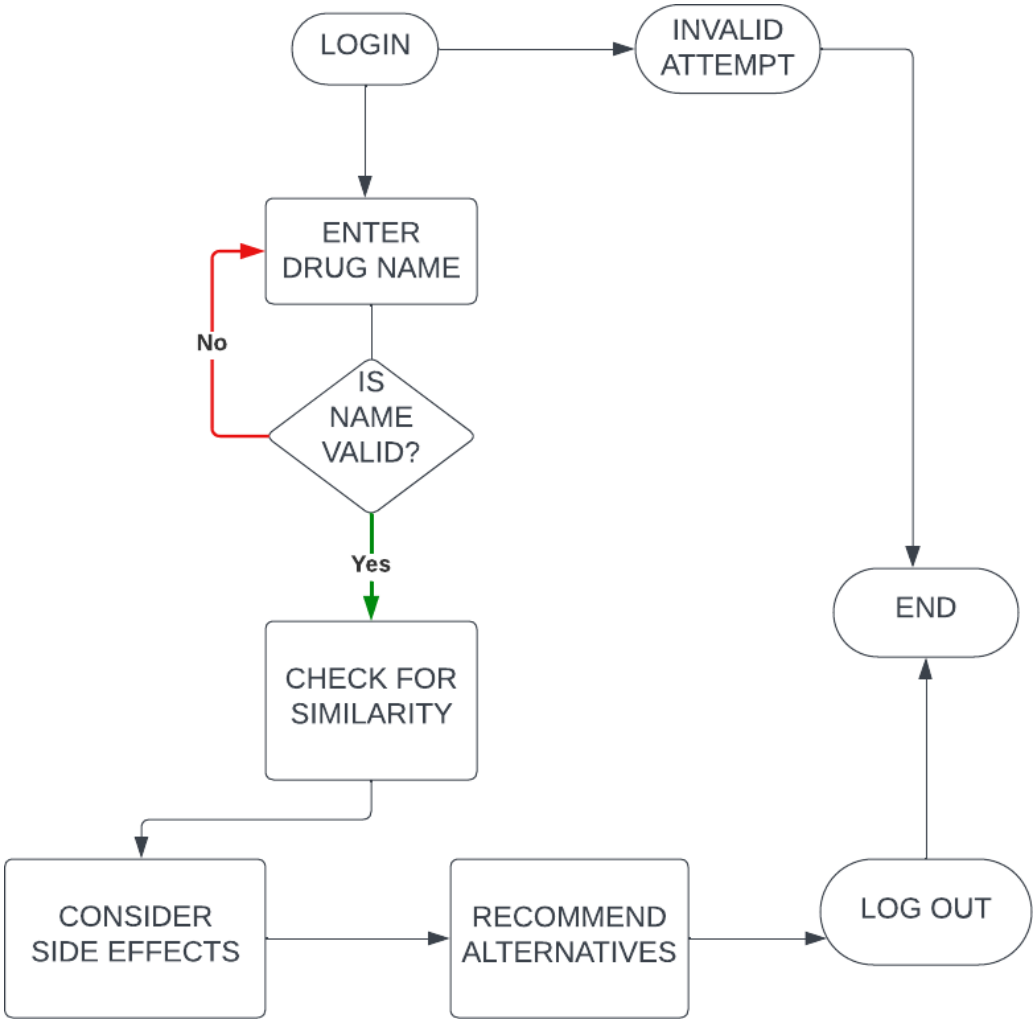

Figure 3.1.1

STEPS:

1. **Start**: The system begins by log in.
2. **Enter Drug name**: The user enters the medicines they want alternatives for.
3. Perform medicine Analysis: The system analyzes the drugs entered to identify possible alternatives.
4. **Identify possible alternatives**: Based on the name, the system identifies potential medical.
5. **Filter Suitable Medications**: Using the identified conditions the system filters out medications that are suitable for the patient's input.

6. **Consider Contraindications**: The system cross-references the suitable medications with the patient's input to ensure there are no contraindications or potential adverse reactions.
7. **Recommend Medications**: Based on the previous steps, the system generates a list of recommended medications that are suitable for the patient's condition input.
8. **End**: The system completes the recommendation process with log out.

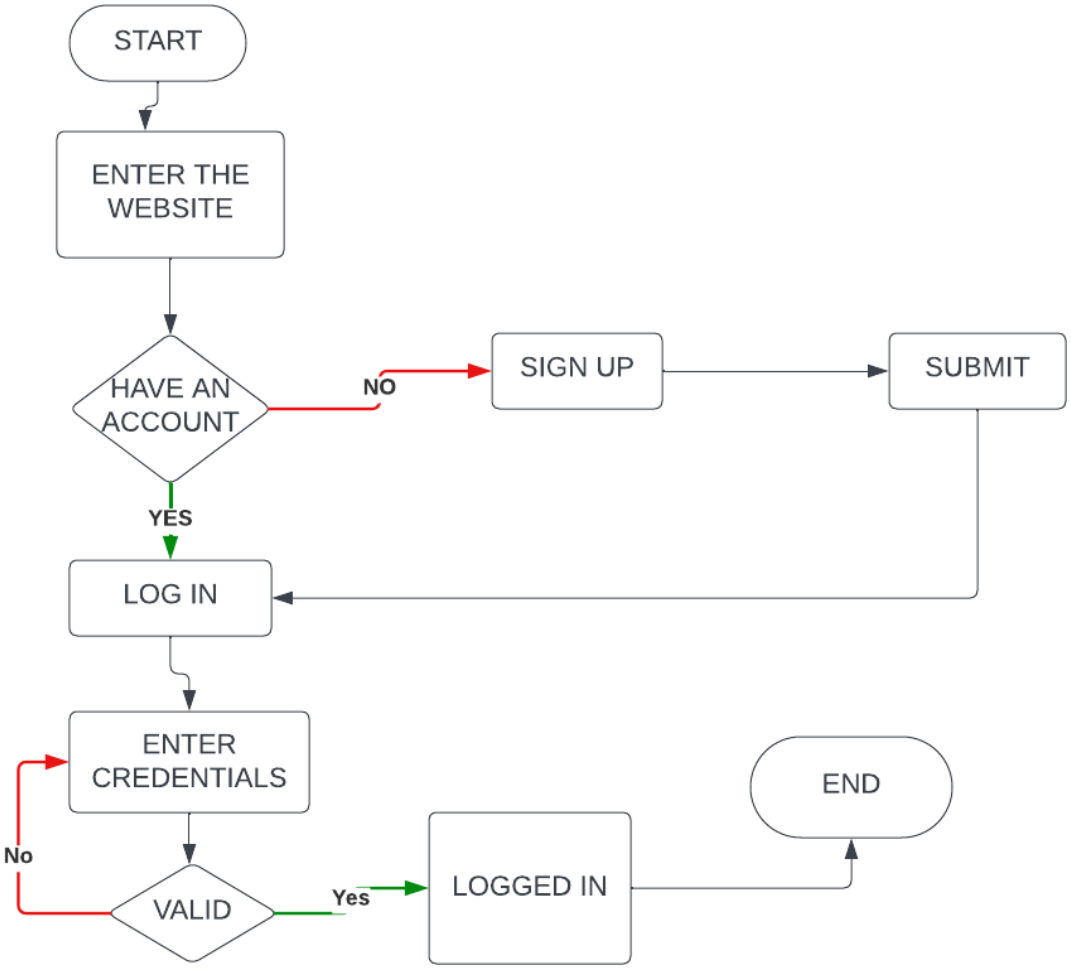

Figure 3.1.2

STEPS

1. **Show Registration Form:** Display the registration form to the user, prompting them to enter their registration details (e.g., name, email, password, etc.).
2. **User Fills in Form:** The user enters their registration information in the form.
3. **Validate Form Data:** Validate the entered form data to ensure it meets the required criteria (e.g., valid email format, password length, etc.).

4. **Create User Account:** If the form data is valid, create a user account using the provided information.
5. **Registration Success/Failure:** Based on the result of creating the user account, proceed accordingly.
    - If the account creation is successful, show a registration success message to the user.
    - If the account creation fails, show an error message and allow the user to resubmit the form.
6. **Show Registration Success Message:** Display a success message to the user after their registration is successful.
7. **Show Login Form:** Display the login form to the user.
8. **User Enters Credentials:** The user enters their login credentials (e.g., email/username and password) in the login form.
9. **Validate Credentials:** Validate the entered credentials to ensure they match an existing user account.
10. **Credential Validation:** Based on the validation of credentials.
    - If the credentials are valid, redirect the user to their dashboard or home page.
    - If the credentials are invalid, show an error message and allow the user to resubmit the login form.
11. **Redirect to User Dashboard/Home Page:** Once the user's credentials are validated, redirect them to home page.

3.2 System Architecture

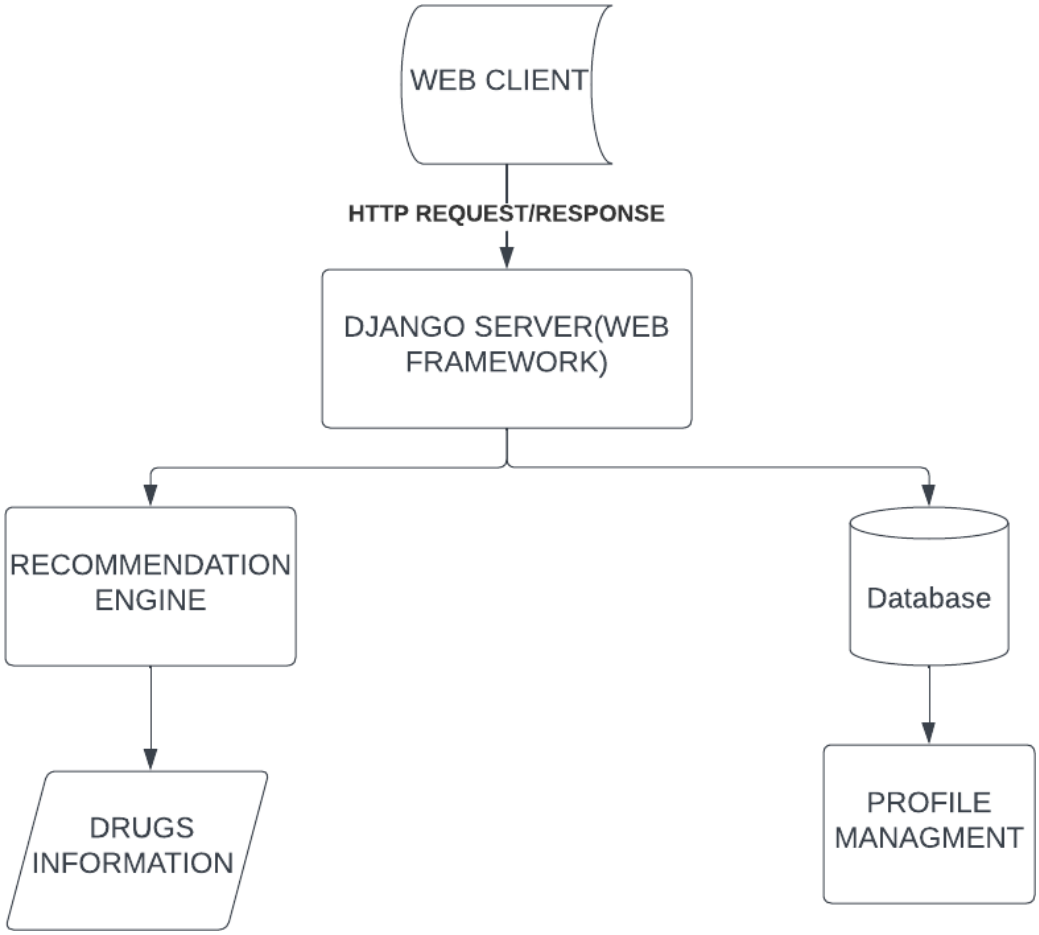

Figure 3.2

Recommendation Engine: Build the recommendation engine that generates personalized recommendations based on user data. This could involve machine learning algorithms, collaborative filtering, or other techniques to analyze user preferences and suggest relevant medicines.

Database: Choose a suitable database management system (e.g., PostgreSQL, MySQL) to store your data. Configure the database settings in Django's settings file.

Django Framework: In the Django framework, the server is responsible for running your web application and handling HTTP requests from clients. Django provides a built-in development server for testing and debugging purposes, and for production deployments.

Web Client: The web client refers to the user-facing part of the application that runs in a web browser. The web client is responsible for presenting the user interface, handling user interactions, and making requests to the server for data or actions.

Profile management: Profile management is an essential component of many system architectures, allowing users to create and manage their accounts and personalize their experience within the system.

Username	**Email Address**	**First Name**	**Last Name**	**Staff Status**
Abdul	245120@gmail.com	Abdul	Muqsith	
Saad	234566@gmail.com	Saad	Muqtadeer	

Table 3.2.1

Name	Description	Active Ingredient	Alcohol Warning	Breastfeeding Warning	Manufacturer	Combined text	Clean Combined Text
Gibocer	100 mg	Metformine	Yes	Yes	Retra Life		

Table 3.2.2

CHAPTER IV

4. System Implementation and Methodologies

To implement a medicine recommendation system, follow a system architecture that includes various components for data management, recommendation generation, and user interface.

1. Data Collection and Storage:

- Gather relevant data: Collect data about medicines, their attributes (e.g., active ingredients, dosage, indications, side effects), and any available alternatives.

- Create a database: Set up a database to store the collected data. Choose a suitable database management system (e.g., PostgreSQL, MySQL) and design a schema that can accommodate the medicine information, alternative relationships, and any additional relevant data.

2. Recommendation Engine:

- Data preprocessing: Clean and preprocess the collected data to ensure consistency and reliability. Normalize medicine attributes, handle missing data, and establish data relationships (e.g., active ingredient similarity, therapeutic category).

- Recommendation algorithms: Implement recommendation algorithms to generate alternative medicine suggestions. This can involve techniques such as content-based filtering, collaborative filtering, or hybrid approaches. These algorithms should consider factors like medicine attributes, user preferences, and effectiveness.

3. User interface: Develop API endpoints using a framework like Django to handle user requests and provide responses. These endpoints should facilitate actions such as searching for medicines, retrieving alternatives, and saving user preferences.

   Recommendation generation: Implement the logic in the application layer that interacts with the recommendation engine to generate alternative medicine recommendations based on user inputs and preferences.

Methodologies

When developing a medicine alternatives system, several methodologies and techniques can be employed to generate accurate and relevant alternative medicine suggestions. Here are some commonly used methodologies in medicine alternatives systems:

1. Similarity Analysis: Similarity analysis is a technique used to find medicines that share similar properties or characteristics with the medicine in question. This can involve analysing attributes such as active ingredients, therapeutic categories, indications, dosage forms, and side effects. By identifying medicines with similar attributes, the system can suggest them as potential alternatives.

2. Knowledge-Based Systems: Knowledge-based systems utilize medical domain knowledge, clinical guidelines, and drug databases to recommend alternative medicines. These systems incorporate expert knowledge, drug interactions, contraindications, and patient-specific characteristics to provide informed suggestions. By considering factors like patient allergies, medical conditions, and potential drug interactions, the system can recommend alternatives that are safer and more suitable.

3. Evidence-Based Medicine: Evidence-based medicine involves analysing clinical research, studies, and evidence to suggest alternative medicines. The system can consider published research on the effectiveness and safety of medicines for specific conditions or compare the efficacy of different treatment options. By considering the available evidence, the system can recommend alternative medicines that have been shown to be effective.

4. Natural Language Processing (NLP) and Text Mining: NLP and text mining techniques can be employed to extract information from medical literature, research papers, and patient reviews. By analysing this textual data, the system can identify mentions or discussions of alternative medicines and their effectiveness for specific conditions. NLP techniques can help understand patient experiences and opinions regarding alternative treatments.

5. Collaborative Filtering: Collaborative filtering techniques can be used to recommend alternative medicines based on the experiences and preferences of other users. By analysing the behaviour and feedback of similar users, the system can suggest alternative medicines that have been well-received by individuals with similar medical conditions or preferences.

6. Hybrid Approaches: Hybrid approaches combine multiple methodologies to provide more accurate and diverse alternative medicine recommendations. For example, a system may leverage similarity analysis, knowledge-based rules, and collaborative filtering to generate recommendations that consider both medicine attributes and user preferences.

7. Continuous Learning and Feedback Loop: It is essential to continuously learn from user feedback and update the system's recommendations accordingly. Feedback from users, including ratings, reviews, and patient outcomes, can be collected to evaluate the effectiveness and relevance of the suggested alternative medicines. This feedback loop helps refine and improve the recommendation algorithms over time.

It is important to note that the recommendations provided by a medicine alternatives system should be used as a reference and should not replace professional medical advice.

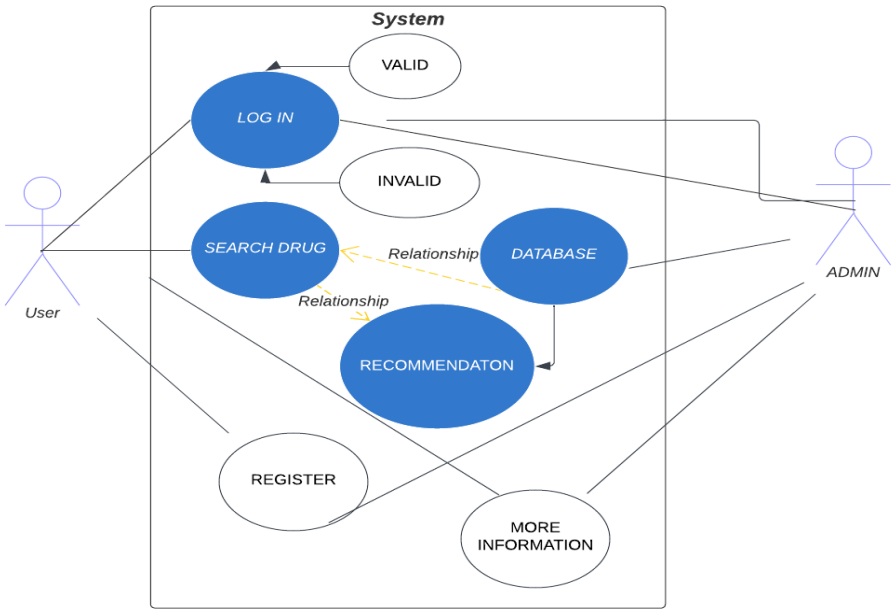

Figure 4.1

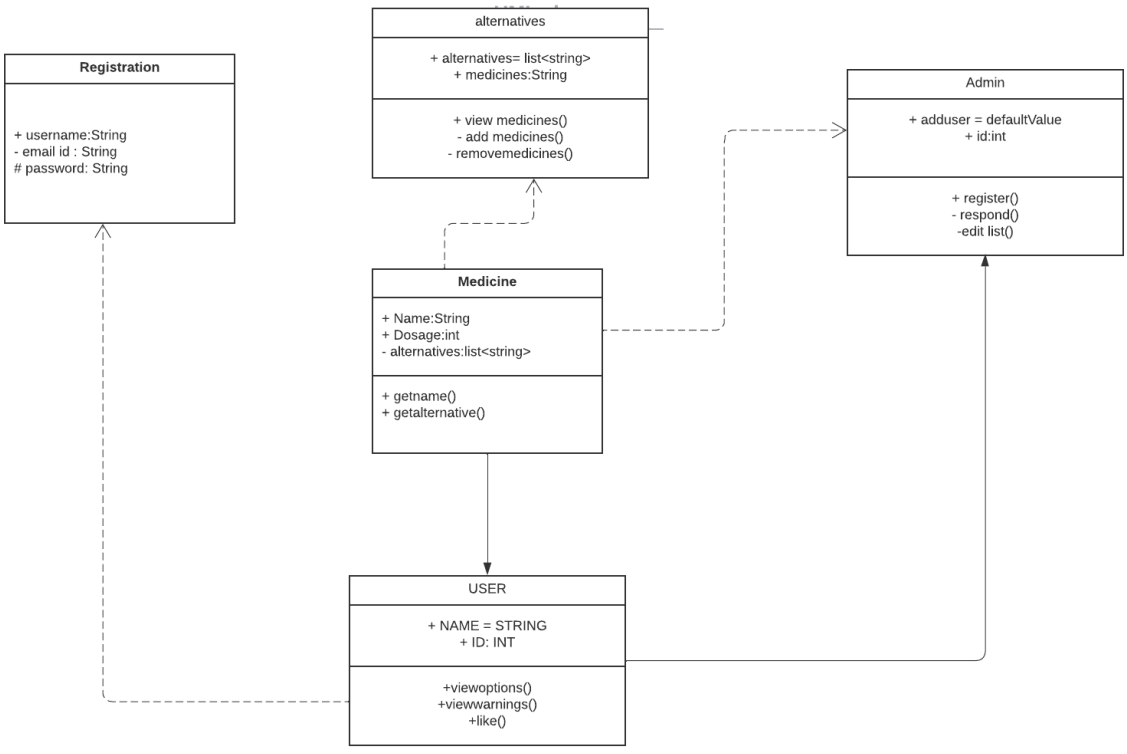

Figure 4.2

4.1 Algorithms

Several algorithms can be used for generating medicine alternatives in a medicine recommendation system.

Content-based filtering algorithms analyze the attributes and characteristics of medicines to recommend alternatives with similar properties. The algorithms consider features such as active ingredients, therapeutic categories, indications, dosage forms, and side effects. The system matches these attributes to find medicines that have similar profiles and suggest them as alternatives.

Cosine similarity is a commonly used metric for measuring the similarity between two vectors, and it can be applied in medicine alternative systems as well. Here's how cosine similarity can be used in such a system:

1. Data Representation:

Represent medicines and their attributes as vectors. Each attribute of a medicine can be considered as a dimension in the vector space. For example, you can represent medicines based on attributes such as active ingredients, therapeutic categories, indications, dosage forms, and side effects.

2.Vector Representation:

Create a vector representation for each medicine by assigning values to the dimensions based on the presence or absence of specific attributes. This can be binary (0 or 1) if an attribute is present or not, or you can use more sophisticated representations, such as TF-IDF (Term Frequency-Inverse Document Frequency) for attribute weighting.

3.Cosine Similarity Calculation:

To determine the similarity between two medicines, calculate the cosine similarity between their vector representations. The cosine similarity is the cosine of the angle between the two vectors and ranges from -1 to 1. A value close to 1 indicates high similarity, while a value close to -1 indicates dissimilarity.

4. Medicine Alternatives:

To recommend alternative medicines, identify medicines with high cosine similarity to the medicine in question. The medicines that have the highest cosine similarity scores are considered the most similar or alternative medicines.

4.2 Sample Code

FRONTEND

HTML code for Login page

```
<!DOCTYPE html>
<html lang="en">
<head>
<!-- Design by foolishdeveloper.com -->
<title>Login Page| Medopt-MedicineRecommendationSystem</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link rel="preconnect" href="https://fonts.gstatic.com">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">
<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;500;600&display=swap" rel="stylesheet">
<!--Stylesheet-->

</head>
<body>
<div class="background">
<div class="shape"></div>
<div class="shape"></div>
</div>
<form method="post">
<h3>Login Here</h3>
{% csrf_token %}
<label for="username">Username</label>
<input type="text" placeholder="Enter Username" id="username" name="username">

<label for="password">Password</label>
<input type="password" placeholder="Password" id="password" name="pass">

<button type="submit">Log In</button>
<!-- <input type="button" value=""> -->
<!-- <div class="social">
<div class="go"><i class="fab fa-google"></i> Google</div>
<div class="fb"><i class="fab fa-facebook"></i> Facebook</div>
</div> -->
<a href="{% url 'signup' %}" >Create a account</a>
</form>
</body>
</html>
```

HTML code signup page connected to backend

```
<!DOCTYPE html>
<html lang="en">
<head>
<!-- Design by foolishdeveloper.com -->
<title>Signup| Medopt-MedicineRecommendationSystem</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link rel="preconnect" href="https://fonts.gstatic.com">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">
<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;500;600&display=swap" rel="stylesheet">
<!--Stylesheet-->

</head>
<body>
<div class="background">
<div class="shape"></div>
<div class="shape"></div>
</div>
<form action="" method="post">
{% csrf_token %}
<h3>Medopt Signup</h3>

<label for="username">Username</label>
<input type="text" placeholder="Username" name="username" id="username">

<label for="email">Email</label>
<input type="email" placeholder="Email or Phone" name="email" id="email">

<label for="password1">Password</label>
<input type="password" placeholder="Password" id="password1" name="password1">


<label for="password2">Confrim Password</label>
<input type="password" placeholder="Confrom Password" id="password2" name="password2">
<button type="submit">Signup</button>

<a href="/login">i have already account</a>
</form>
</body>
</html>
```

Contact us page with html code which is connected to Django by default database(SQLite) records user feedback

```
<div class="container-fluid px-0">
<img src="/static/images/8.jpeg" class="d-block w-100 mx-9" alt="...">
</div>
<div class = "container my-4">
<h1 class="text-center">Contact Us</h1>
<form method="post" action="/contact">
{% csrf_token %}
<div class="form-group">
<label for="name">Name</label>
<input type="text" class="form-control" id="name" name="name" placeholder="Enter your
Name">
</div>
<div class="form-group">
<label for="email">Email address</label>
<input type="email" class="form-control" id="email" name="email" placeholder="Enter
Your Email">
</div>
<div class="form-group">
<label for="phone">Phone Number</label>
<input type="phone" class="form-control" id="phone" name="phone" placeholder="Enter
Your Phone Number">
</div>
<div class="form-group">
<label for="desc">Tell me about what you want to contact me for...</label>
<textarea class="form-control" id="desc" rows="3" name="desc"></textarea>
</div>
<button type="submit" class="my-4 btn btn-primary">Submit</button>
</form>
</div>
```

BACKEND

Python code for preprocessing of data using Sklearn, NLTK, pandas... libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import re
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.metrics.pairwise import linear_kernel, cosine_similarity
from sklearn.cluster import KMeans
from nltk.corpus import stopwords
from nltk.tokenize import RegexpTokenizer
df = pd.read_csv(r"C:\Users\hamee\Downloads\1mg.csv\1mg.csv")
df = df.drop_duplicates()
df.info()
manufacturer_count = df['manufacturer'].value_counts()
manufacturer_count = manufacturer_count[:10,]
manufacturer_count
activeIngredient_count = df['activeIngredient'].value_counts()
activeIngredient_count =activeIngredient_count[:10,]

activeIngredient_count
activeIngredient_count = df['activeIngredient'].value_counts()
activeIngredient_count =activeIngredient_count[:10,]

activeIngredient_count
df['combined_text'] = df['desc'] + ' ' + df['activeIngredient'] + ' ' + df['name']
df['combined_text'] = df['combined_text'].astype(str)

df['cleaned_combined_text'] = df['combined_text'].apply(_removeNonAscii)
df['cleaned_combined_text'] = df['combined_text'].apply(func = make_lower_case)
df['cleaned_combined_text'] = df['combined_text'].apply(func=remove_punctuation)
df['cleaned_combined_text'] = df['combined_text'].apply(func=remove_html)
df = df.reset_index()
tfidfvectorizer = TfidfVectorizer(analyzer='word',ngram_range=(1, 2),min_df=0,
stop_words='english')
tfidfmatrix = tfidfvectorizer.fit_transform(df['cleaned_combined_text'])
cosine_sim = linear_kernel(tfidfmatrix, tfidfmatrix)
def getrecommendation(medicinename,medicine_idx, similarity,products, topn):
idx = medicine_idx[medicinename]
sim_scores = list(enumerate(similarity[idx]))
sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
sim_scores = sim_scores[1:5*topn]
result_indices = [i[0] for i in sim_scores]
```

```
# extract id's from main data frame.
idx_range = products.iloc[result_indices]
# return all idx from data frame.
recommendation = df[df['name'].isin(idx_range.values)]
# return only the names of the medicine.
return recommendation[['name']].head(topn)
products = pd.Series(df.index, index=df['name'])
medicine_idx = df['name']
medicinesample1 = 'Glibocer M 500mg/0.3mg Tablet'
res = getrecommendation(
medicinesample1,
products,
cosine_sim,
medicine_idx,
10)
res[['name']]
```

Cosine similarity backend

```
from django.shortcuts import render, HttpResponse, redirect
from datetime import datetime
from home.models import Contact
from django.contrib import messages
from django.contrib.auth.models import User
from django.contrib.auth import authenticate
from django.contrib.auth import login, logout
import csv
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import TfidfVectorizer
# Create your views here.

def index(request):
return render(request, 'index.html')

def about(request):
return render(request, 'about.html')

def services(request):
return render(request, 'services.html')

def profile(request):
return render(request, 'profile.html')

def logout_view(request):
logout(request)
return redirect('home')
```

```
def contact(request):
if request.method == "POST":
name = request.POST.get('name')
email = request.POST.get('email')
phone = request.POST.get('phone')
desc = request.POST.get('desc')
contact = Contact (name=name, email=email, phone=phone, desc=desc, date = datetime.today())
contact.save()
messages.success(request, 'Profile has been Updated!!')
return render(request, 'contact.html')

def signup(request):
if request.method== 'POST':
uname=request.POST.get('username')
email=request.POST.get('email')
pass1=request.POST.get('password1')
pass2=request.POST.get('password2')
if pass1!=pass2:
return HttpResponse("Your password and confrom password are same!!")
else:
my_user=User.objects.create_user(uname, email, pass1)
my_user.save()
return redirect('login')
return render(request, 'signup.html')

def log(request):
if request.method== 'POST':
username=request.POST.get('username')
pass1=request.POST.get('pass')
user=authenticate(request, username=username, password=pass1)
if user is not None:
login(request, user)
return redirect('home')
else:
error_message = "Invalid username or password"
return render(request, 'login.html', {'error_message': error_message})
# return HttpResponse ("Username or Password is incorrect!!!")
return render(request, 'login.html')

def search_view(request):
if request.method == 'GET':
search_query = request.GET.get('search_input')
search_results = []
```

```
if search_query:
csv_path = 'datasets/1mgadded.csv' # Path to your CSV file
items = []
names = []
with open(csv_path, 'r', encoding='utf-8') as csvfile:
reader = csv.DictReader(csvfile)
for row in reader:
items.append(row)
names.append(row['cleaned_combined_text'])
# if search_query.lower() in row['Drug_Name'].lower():
#     search_results.append(row)
vectorizer = TfidfVectorizer()
vectors = vectorizer.fit_transform(names + [search_query])
query_vector = vectors[-1] # Vector representation of the search query
item_vectors = vectors[:-1] # Vector representations of the items
similarities = cosine_similarity(query_vector, item_vectors).flatten()

# Sort items based on similarity
sorted_indices = similarities.argsort()[::-1][:13]
for index in sorted_indices:
search_results.append(items[index])

return render(request, 'search.html', {'search_results': search_results, 'search_query':
search_query})
return render(request, 'search.html')

def details(request):
item_id = request.GET.get('item_id')
name = request.GET.get('name')
desc = request.GET.get('desc')
manufacturer = request.GET.get('manufacturer')
active_ingredient = request.GET.get('activeIngredient')
alcoholWarning = request.GET.get('alcoholWarning')
breastfeedingWarning = request.GET.get('breastfeedingWarning')
pregnancyWarning = request.GET.get('pregnancyWarning')

context = {
'item_id': item_id,
'name': name,
'desc': desc,
'manufacturer': manufacturer,
'active_ingredient': active_ingredient,
'alcoholWarning':alcoholWarning,
'breastfeedingWarning':breastfeedingWarning,
'pregnancyWarning':pregnancyWarning
}
```

```
return render(request, 'details.html', context)

# autocomplete_app/views.py
from django.http import JsonResponse
from .utils import get_names_from_csv

def autocomplete_app(request):
search_term = request.GET.get('term', '')
csv_file_path = 'C:/Users/saad/OneDrive/Desktop/Medopt/Hello/datasets/1mgadded.csv' #
Replace with the actual path to your CSV file
names = get_names_from_csv(csv_file_path)
matched_names = [name for name in names if search_term.lower() in name.lower()]
return JsonResponse(matched_names, safe=False)
```

CHAPTER V

TESTING / RESULTS

Sign up Page

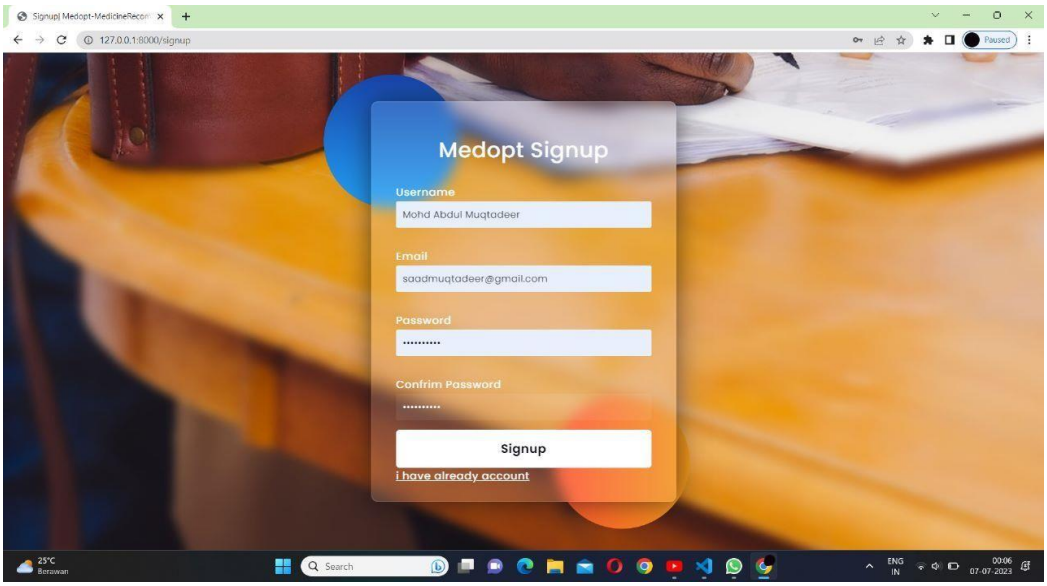

Figure 5.1

Log in Page

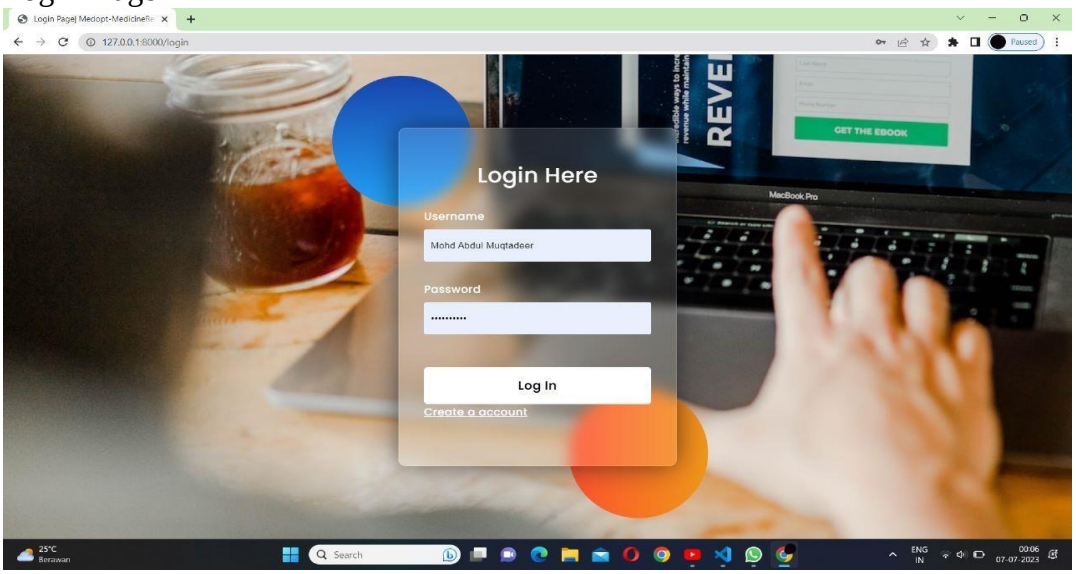

Figure 5.2

Home Page

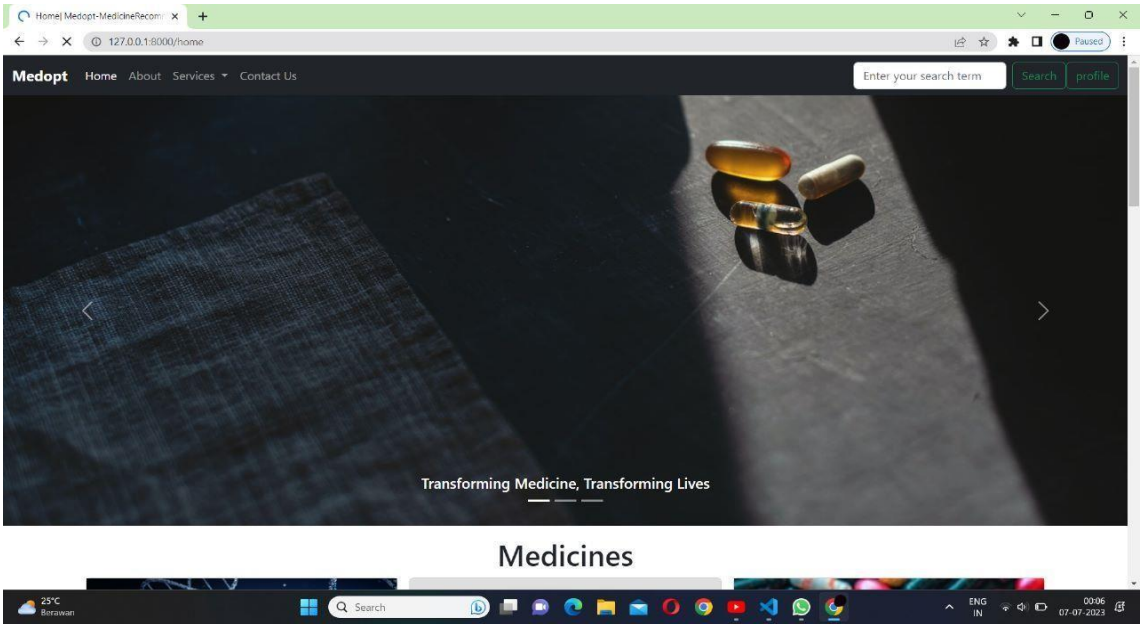

Figure 5.3

Contact Us Page

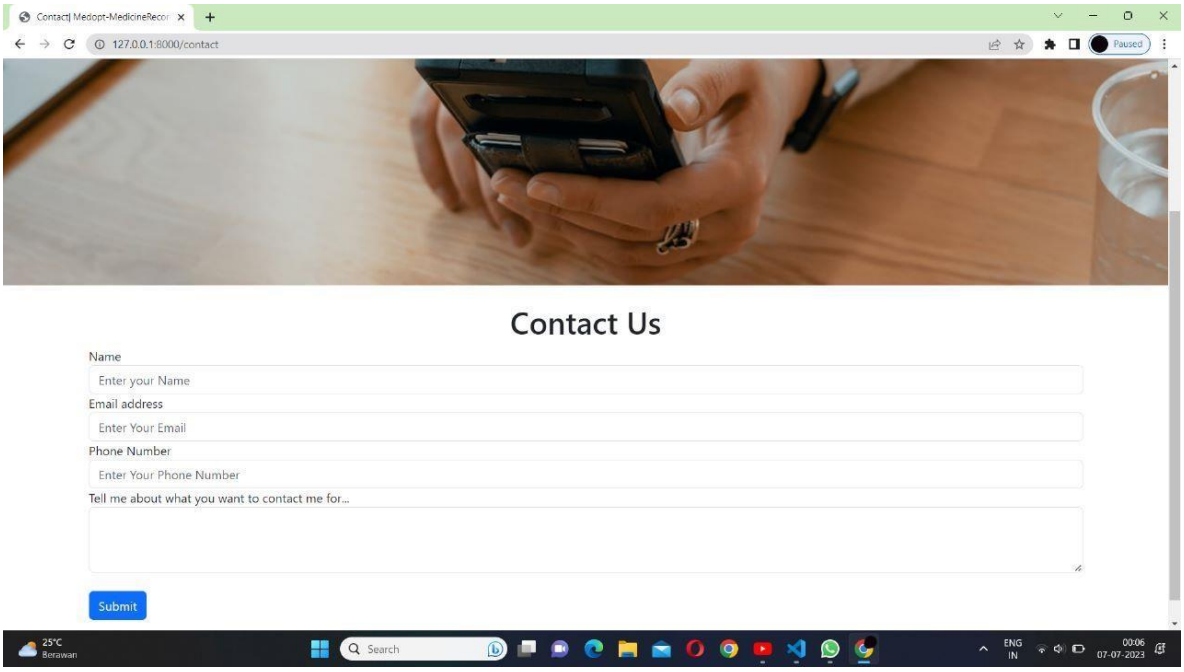

Figure 5.4

About Us Page

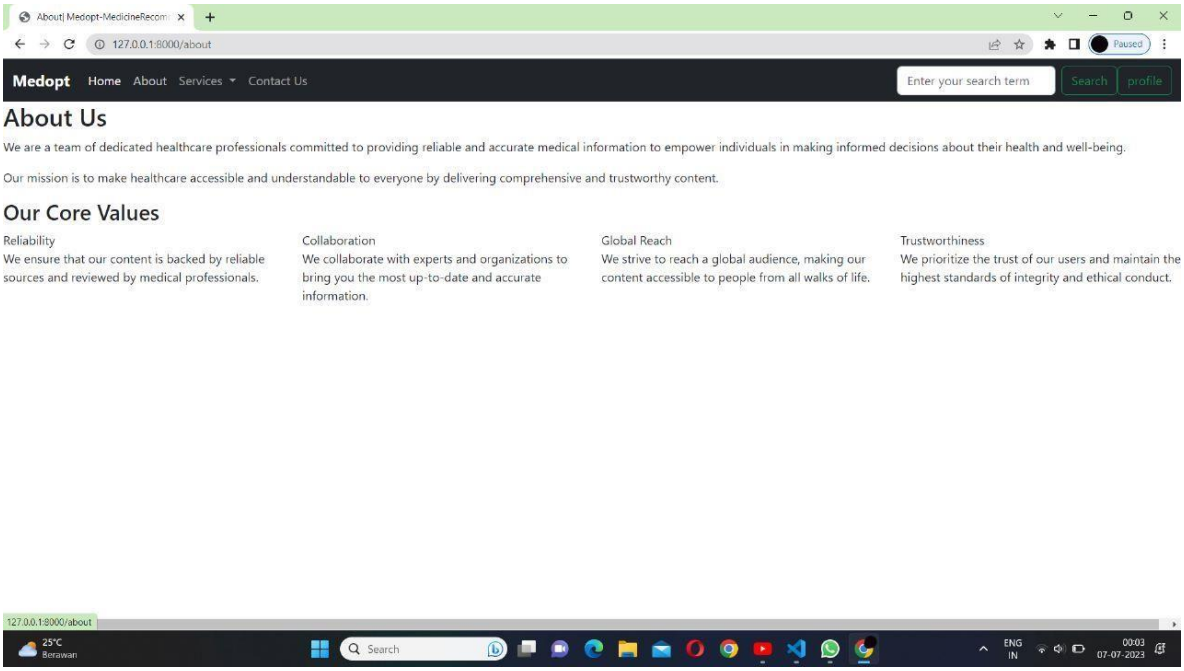

Figure 5.5

Test Case 1

Input: Gibocer M 500mg/0.3 mg tablet

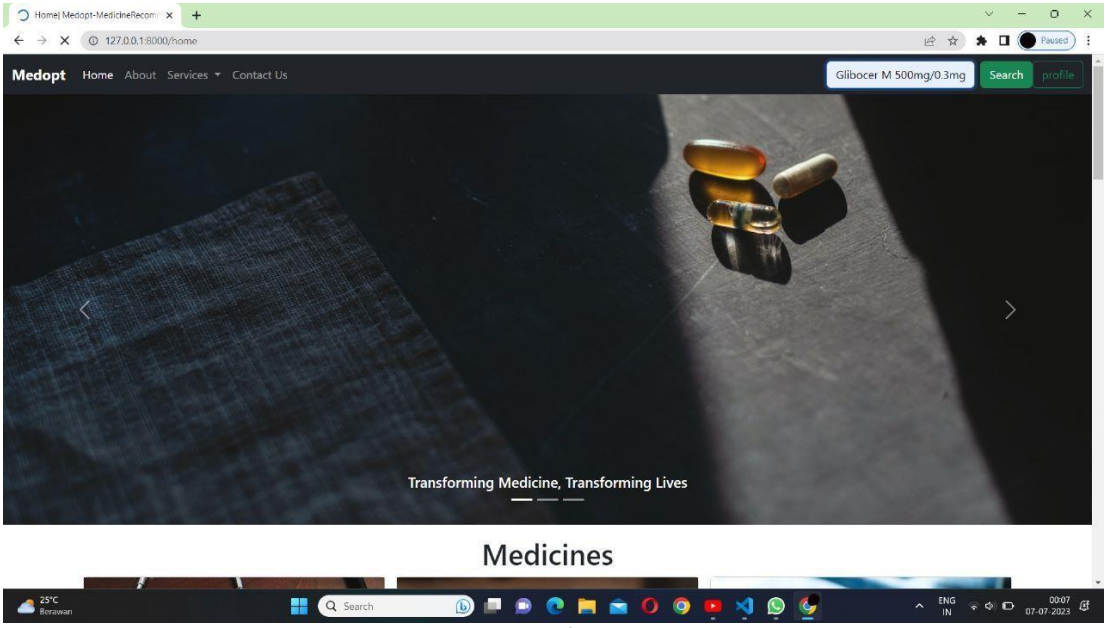

Figure 5.6

Output

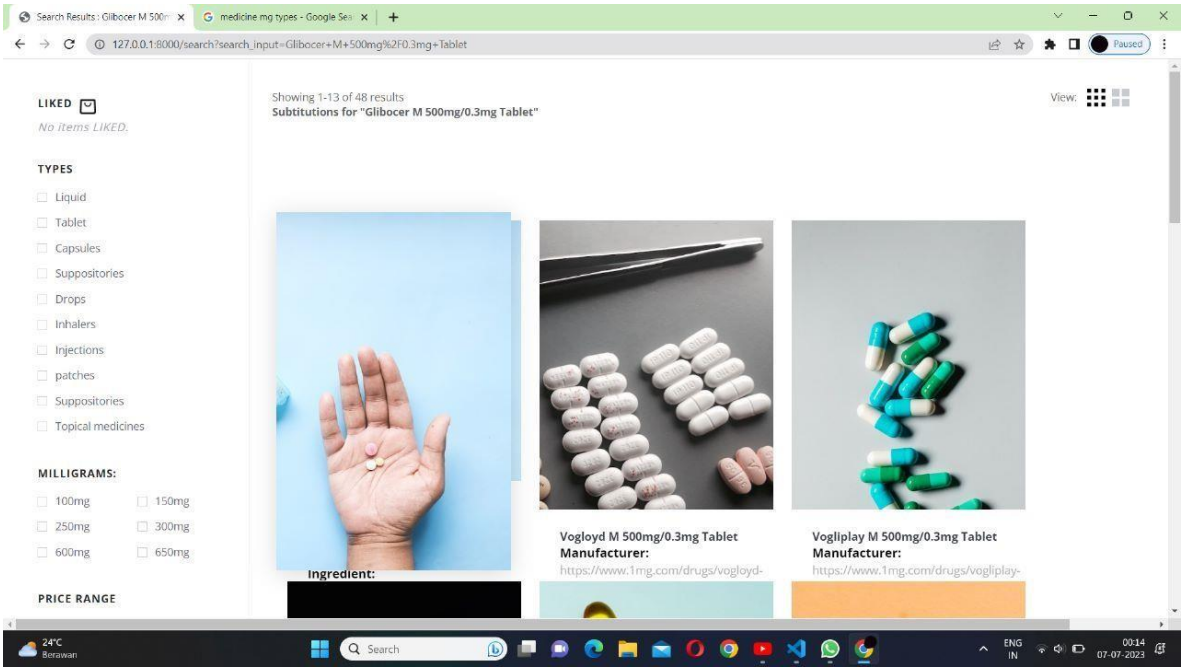

Figure 5.7

View Warnings

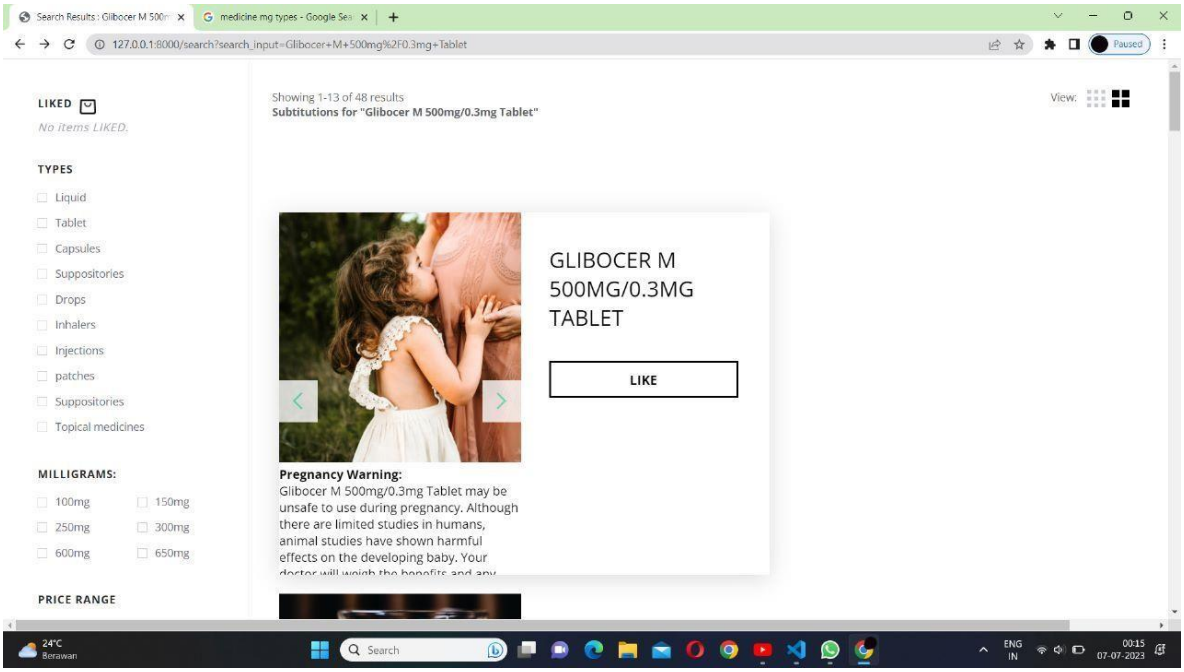

Figure 5.8

3D view and Like Drugs

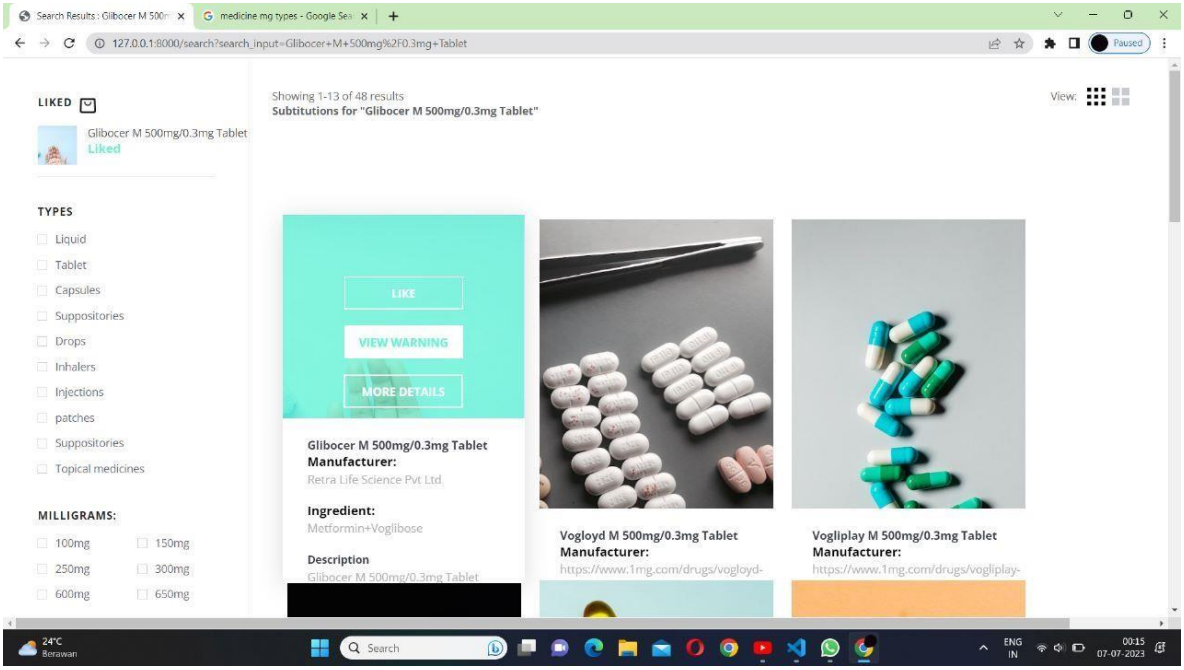

Figure 5.9

More Details

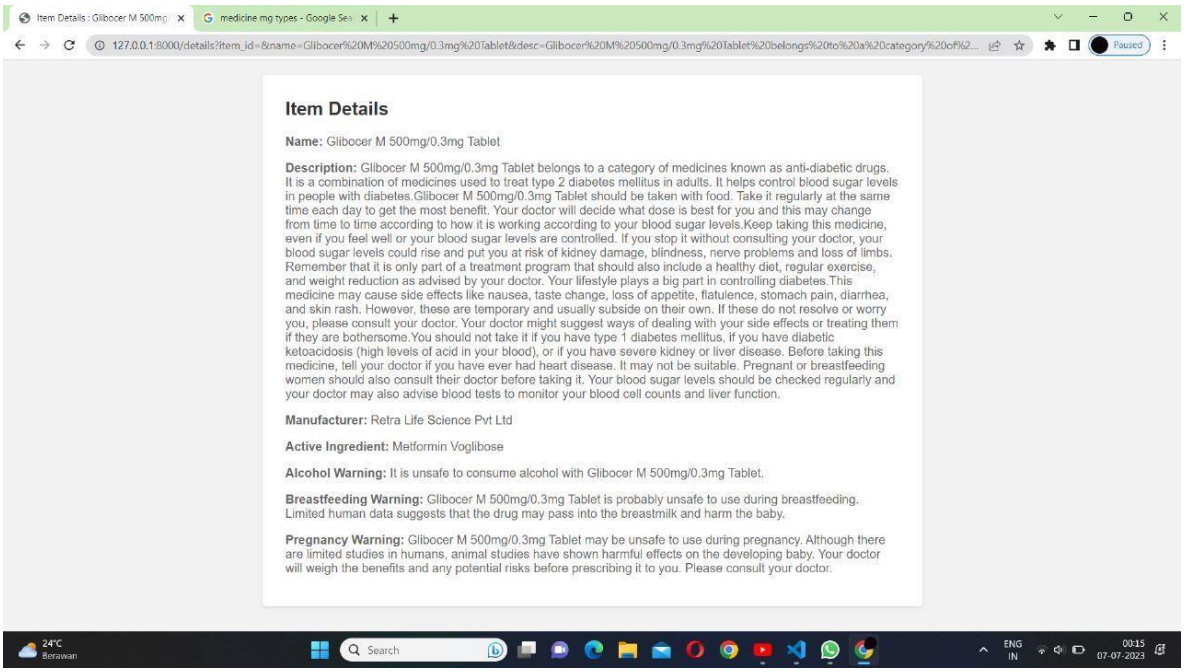

Figure 5.10

CHAPTER VI

CONCLUSION & FUTURE ENHANCEMENTS

In conclusion, a medicine alternative system can provide valuable recommendations to users, helping them explore alternative treatment options and make informed decisions about their healthcare. By leveraging various methodologies, such as cosine similarity, content-based filtering, collaborative filtering, and knowledge-based systems, the system can generate relevant and personalized alternative medicine suggestions.
Here are some potential future enhancements and considerations for improving a medicine alternative system:

Enhanced Data Collection:

Continuously gather and update data from reliable sources, including medical databases, research publications, and user feedback. Incorporate real-world evidence, clinical trials, and emerging treatments to ensure the system is up-to-date with the latest information.

Integration of Patient Profiles:

Consider incorporating individual patient profiles and medical history into the recommendation process. By analyzing patient-specific factors such as age, gender, medical conditions, allergies, and treatment responses, the system can provide more personalized and tailored alternative medicine recommendations.

Long-term Treatment Monitoring:

Develop capabilities to monitor the effectiveness and outcomes of alternative treatments over time. By tracking patient experiences, treatment responses, and health outcomes, the system can learn and adapt recommendations based on real-world evidence.

Integration with Electronic Health Records (EHRs):

Integrate the medicine alternative system with electronic health record systems to access comprehensive patient information.

Contextual Factors:

Consider incorporating contextual factors such as geographic location, local healthcare guidelines, and cultural preferences. These factors can influence the availability and suitability of alternative medicines in different regions and populations.

Collaboration with Healthcare Professionals:

Collaborate with healthcare professionals, physicians, pharmacists, and other domain experts to validate and enhance the system's recommendations. Their expertise can help validate the accuracy, safety, and appropriateness of alternative medicine suggestions.

Overall, continuous improvement, user feedback, integration of real-world evidence, and collaboration with healthcare professionals are key factors for the future enhancement and success of a medicine alternative system.

REFERENCES/ BIBLIOGRAPHY

Overview
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5417584/

Example
https://f1000research.com/articles/11-526

More Information
https://www.sciencedirect.com/topics/neuroscience/alternative-medical-systems

https://www.slideshare.net/anjalatchi/alternative-system-medicine-231269306

Implementation
https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada

https://wisdomml.in/drug-recommendation-system-using-nlp-and-machine-learning- approach-in-python/

Algorithms

https://builtin.com/machine-learning/cosine-similarity

Framework
https://www.ibm.com/topics/django

APPENDIX (SOURCE CODE)

javascript backend for 3D views

```
$(document).ready(function(){

$(".largeGrid").click(function(){
$(this).find('a').addClass('active');
$('.smallGrid a').removeClass('active');

$('.product').addClass('large').each(function(){
});
setTimeout(function(){
$('.info-large').show();
}, 200);
setTimeout(function(){

$('.view_gallery').trigger("click");
}, 400);

return false;
});

$(".smallGrid").click(function(){
$(this).find('a').addClass('active');
$('.largeGrid a').removeClass('active');

$('div.product').removeClass('large');
$(".make3D").removeClass('animate');
$('.info-large').fadeOut("fast");
setTimeout(function(){
$('div.flip-back').trigger("click");
}, 400);
return false;
});

$(".smallGrid").click(function(){
$('.product').removeClass('large');
return false;
});

$('.colors-large a').click(function(){return false;});


$('.product').each(function(i, el){
```

```
// Lift card and show stats on Mouseover
$(el).find('.make3D').hover(function(){
$(this).parent().css('z-index', "20");
$(this).addClass('animate');
$(this).find('div.carouselNext, div.carouselPrev').addClass('visible');
}, function(){
$(this).removeClass('animate');
$(this).parent().css('z-index', "1");
$(this).find('div.carouselNext, div.carouselPrev').removeClass('visible');
});

// Flip card to the back side
$(el).find('.view_gallery').click(function(){

$(el).find('div.carouselNext, div.carouselPrev').removeClass('visible');
$(el).find('.make3D').addClass('flip-10');
setTimeout(function(){
$(el).find('.make3D').removeClass('flip-
10').addClass('flip90').find('div.shadow').show().fadeTo( 80 , 1, function(){
$(el).find('.product-front, .product-front div.shadow').hide();
});
}, 50);

setTimeout(function(){
$(el).find('.make3D').removeClass('flip90').addClass('flip190');
$(el).find('.product-back').show().find('div.shadow').show().fadeTo( 90 , 0);
setTimeout(function(){
$(el).find('.make3D').removeClass('flip190').addClass('flip180').find('div.shadow').hide();
setTimeout(function(){
$(el).find('.make3D').css('transition', '100ms ease-out');
$(el).find('.cx, .cy').addClass('s1');
setTimeout(function(){$(el).find('.cx, .cy').addClass('s2');}, 100);
setTimeout(function(){$(el).find('.cx, .cy').addClass('s3');}, 200);
$(el).find('div.carouselNext, div.carouselPrev').addClass('visible');
}, 100);
}, 100);
}, 150);
});

// Flip card back to the front side
$(el).find('.flip-back').click(function(){

$(el).find('.make3D').removeClass('flip180').addClass('flip190');
setTimeout(function(){
$(el).find('.make3D').removeClass('flip190').addClass('flip90');
```

```
$(el).find('.product-back div.shadow').css('opacity', 0).fadeTo( 100 , 1, function(){
$(el).find('.product-back, .product-back div.shadow').hide();
$(el).find('.product-front, .product-front div.shadow').show();
});
}, 50);
setTimeout(function(){
$(el).find('.make3D').removeClass('flip90').addClass('flip-10');
$(el).find('.product-front div.shadow').show().fadeTo( 100 , 0);
setTimeout(function(){
$(el).find('.product-front div.shadow').hide();
$(el).find('.make3D').removeClass('flip-10').css('transition', '100ms ease-out');
$(el).find('.cx, .cy').removeClass('s1 s2 s3');
}, 100);
}, 150);

});

makeCarousel(el);
});

$('.add-cart-large').each(function(i, el){
$(el).click(function(){
var carousel = $(this).parent().parent().find(".carousel-container");
var img = carousel.find('img').eq(carousel.attr("rel"))[0];
var position = $(img).offset();

var productName = $(this).parent().find('h4').get(0).innerHTML;

$("body").append('<div class="floating-cart"></div>');
var cart = $('div.floating-cart');
$("<img src='"+img.src+"' class='floating-image-large' />").appendTo(cart);

$(cart).css({'top' : position.top + 'px', "left" : position.left + 'px'}).fadeIn("slow").addClass('moveToCart');
setTimeout(function(){$("body").addClass("MakeFloatingCart");}, 800);

setTimeout(function(){
$('div.floating-cart').remove();
$("body").removeClass("MakeFloatingCart");


var cartItem = "<div class='cart-item'><div class='img-wrap'><img src='"+img.src+"' alt=" /></div><span>"+productName+"</span><strong>liked</strong><div class='cart-item-border'></div><div class='delete-item'></div></div>";
```

```
$("#cart .empty").hide();
$("#cart").append(cartItem);
$("#checkout").fadeIn(500);

$("#cart .cart-item").last()
.addClass("flash")
.find(".delete-item").click(function(){
$(this).parent().fadeOut(300, function(){
$(this).remove();
if($("#cart .cart-item").size() == 0){
$("#cart .empty").fadeIn(500);
$("#checkout").fadeOut(500);
}
})
});
setTimeout(function(){
$("#cart .cart-item").last().removeClass("flash");
}, 10 );

}, 1000);


});
})

/* ----  Image Gallery Carousel------ */
function makeCarousel(el){


var carousel = $(el).find('.carousel ul');
var carouselSlideWidth = 315;
var carouselWidth = 0;
var isAnimating = false;
var currSlide = 0;
$(carousel).attr('rel', currSlide);

// building the width of the casousel
$(carousel).find('li').each(function(){
carouselWidth += carouselSlideWidth;
});
$(carousel).css('width', carouselWidth);

// Load Next Image
$(el).find('div.carouselNext').on('click', function(){
var currentLeft = Math.abs(parseInt($(carousel).css("left")));
var newLeft = currentLeft + carouselSlideWidth;
```

```
if(newLeft == carouselWidth || isAnimating === true){return;}
$(carousel).css({'left': "-" + newLeft + "px",
"transition": "300ms ease-out"
});
isAnimating = true;
currSlide++;
$(carousel).attr('rel', currSlide);
setTimeout(function(){isAnimating = false;}, 300);
});

// Load Previous Image
$(el).find('div.carouselPrev').on('click', function(){
var currentLeft = Math.abs(parseInt($(carousel).css("left")));
var newLeft = currentLeft - carouselSlideWidth;
if(newLeft < 0 || isAnimating === true){return;}
$(carousel).css({'left': "-" + newLeft + "px",
"transition": "300ms ease-out"
});
isAnimating = true;
currSlide--;
$(carousel).attr('rel', currSlide);
setTimeout(function(){isAnimating = false;}, 300);
});
}

$('.sizes a span, .categories a span').each(function(i, el){
$(el).append('<span class="x"></span><span class="y"></span>');

$(el).parent().on('click', function(){
if($(this).hasClass('checked')){
$(el).find('.y').removeClass('animate');
setTimeout(function(){
$(el).find('.x').removeClass('animate');
}, 50);
$(this).removeClass('checked');
return false;
}

$(el).find('.x').addClass('animate');
setTimeout(function(){
$(el).find('.y').addClass('animate');
}, 100);
$(this).addClass('checked');
return false;
});
});
```

```
$('.add_to_cart').click(function(){
var productCard = $(this).parent();
var position = productCard.offset();
var productImage = $(productCard).find('img').get(0).src;
var productName = $(productCard).find('.product_name').get(0).innerHTML;

$("body").append('<div class="floating-cart"></div>');
var cart = $('div.floating-cart');
productCard.clone().appendTo(cart);
$(cart).css({'top' : position.top + 'px', "left" : position.left + 'px'}).fadeIn("slow").addClass('moveToCart');
setTimeout(function(){$("body").addClass("MakeFloatingCart");}, 800);
setTimeout(function(){
$('div.floating-cart').remove();
$("body").removeClass("MakeFloatingCart");


var cartItem = "<div class='cart-item'><div class='img-wrap'><img src='"+productImage+"' alt=" /></div><span>"+productName+"</span><strong>Liked</strong><div class='cart-item-border'></div><div class='delete-item'></div></div>";

$("#cart .empty").hide();
$("#cart").append(cartItem);
$("#checkout").fadeIn(500);

$("#cart .cart-item").last()
.addClass("flash")
.find(".delete-item").click(function(){
$(this).parent().fadeOut(300, function(){
$(this).remove();
if($("#cart .cart-item").size() == 0){
$("#cart .empty").fadeIn(500);
$("#checkout").fadeOut(500);
}
})
});
setTimeout(function(){
$("#cart .cart-item").last().removeClass("flash");
}, 10 );

}, 1000);
});


});
```