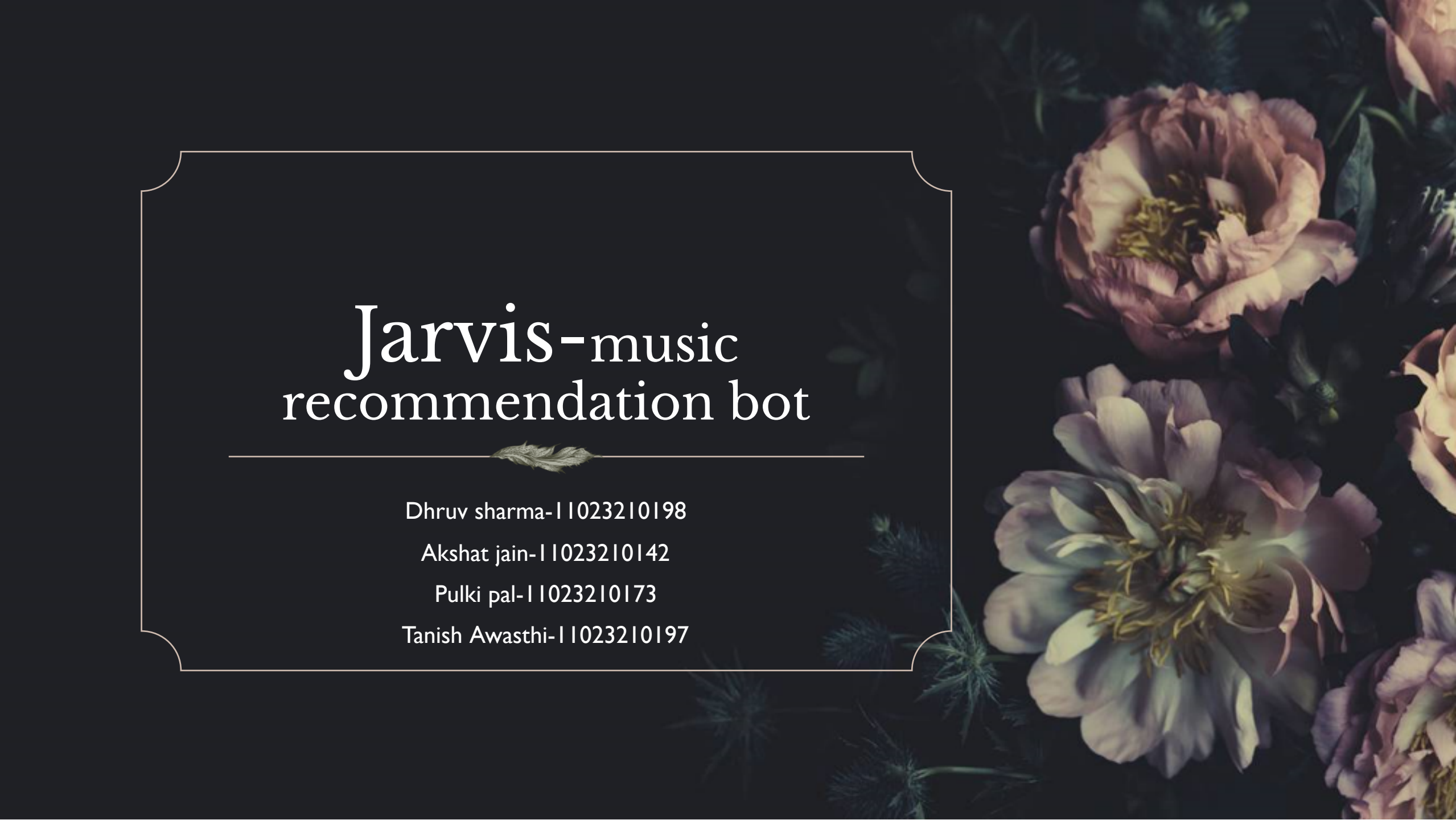
Jarvis-music recommendation bot

Dhruv sharma-11023210198

Akshat jain-11023210142

Pulki pal-11023210173

Tanish Awasthi-11023210197



Index

The power of Jarvis

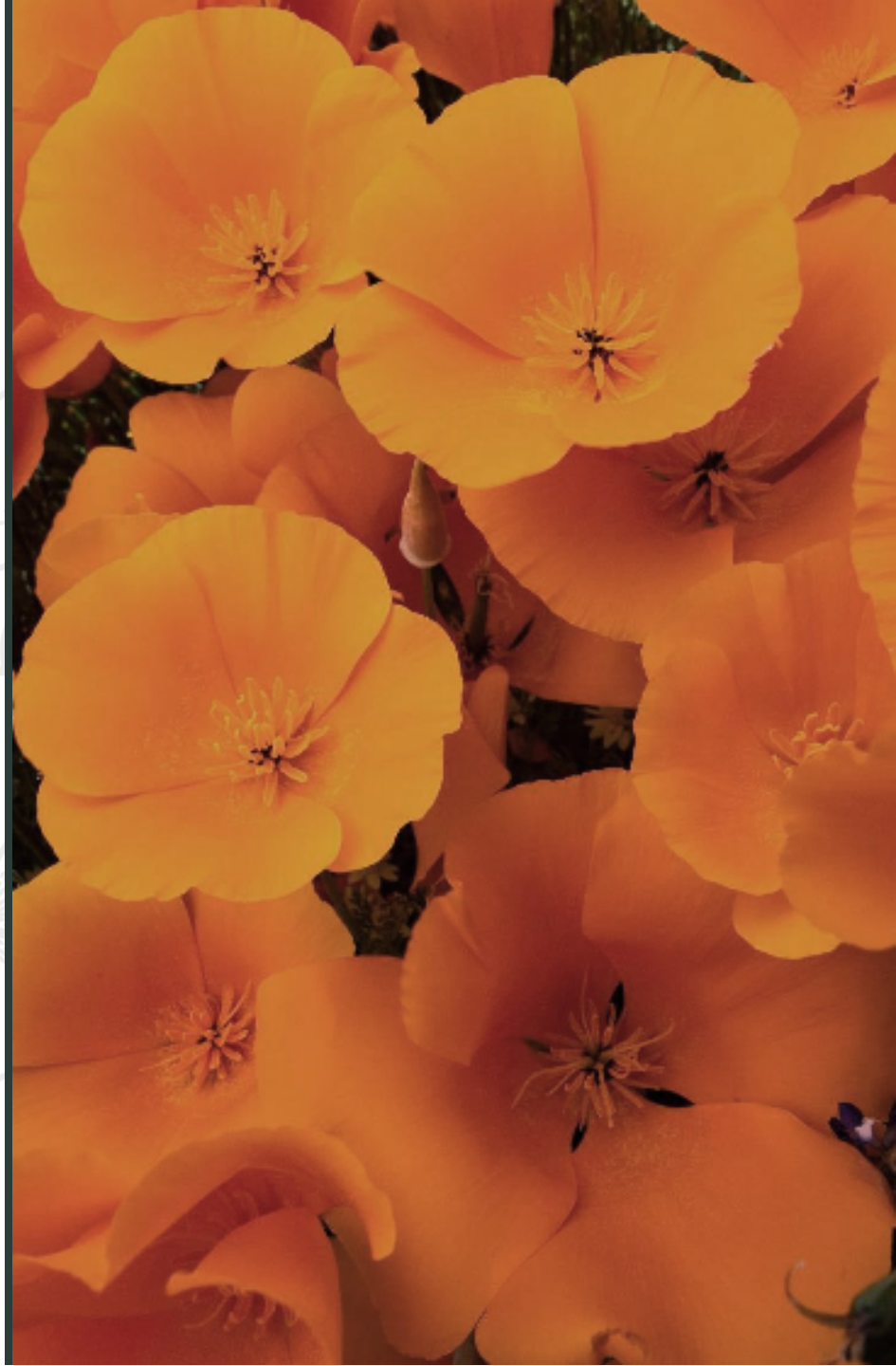
Jarvis is an AI-powered music recommendation bot designed to provide personalized music suggestions based on users' preferences.

1.Natural Language Processing (NLP): Jarvis understands text input.

2. Music Database: Access to millions of songs from various genres and artists.

3. Personalization: Learns users' preferences and adapts recommendations.

4. Discovery Feature: Introduces users to new artists and genres.



Libraries:

The Requests library is the de facto standard for making HTTP requests in Python. It abstracts the complexities of making requests behind a beautiful, simple API so that you can focus on interacting with services and consuming data in your application.

Textbob

1. TextBlob is a Python library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more. TextBlob stands on the giant shoulders of NLTK and pattern, and plays nicely with both.

Pytube

Pytube is a lightweight, Python library used for downloading videos from YouTube. It provides a simple way to interact with YouTube's video streaming platform and enables downloading videos, audio, and metadata like video titles and descriptions directly from YouTube.

NLTK

NLTK (Natural Language Toolkit) is a powerful, open-source Python library used extensively for natural language processing (NLP). It provides a wide range of tools for working with human language data, including tokenization, parsing, classification, stemming, tagging, and more. NLTK is widely used for tasks like text analysis, sentiment analysis, and building language models.

WEBBROWSER

The webbrowser library in Python is a standard library module that provides a simple way to control and interact with web browsers directly from Python code. This library allows you to open URLs in the default web browser and supports multiple browser types.

FLASK

Flask is a lightweight, flexible web framework for Python that is widely used to create web applications, APIs, and microservices. Known for its simplicity and ease of use, Flask provides a basic structure for developing web applications without the additional overhead of more full-featured frameworks like Django.


INPUT

```
#html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Jarvis - Music Recommendation Bot</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
</head>
<body>
    <div class="container">
        <h1>Jarvis - Music Recommendation Bot</h1>
        <form method="POST">
            <input type="text" name="user_input" placeholder="Enter your text here..."
required>
            <button type="submit">Submit</button>
        </form>
        <div class="message">
            <p>{{ message }}</p>
        </div>
    </div>
</body>
</html>
```

```css
body {
    font-family: Arial, sans-serif;
    background-color: #e0f7fa;
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

.container {
    background-color: #ffffff;
    padding: 30px;
    border-radius: 10px;
    box-shadow: 0 4px 15px rgba(0, 0, 0, 0.2);
    max-width: 500px;
    width: 100%;
    text-align: center;
    transition: all 0.3s ease-in-out;
}

.container:hover {
    transform: translateY(-5px);
}

h1 {
    margin-bottom: 20px;
    color: #00796b;
}
```

```css
input[type="text"] {
    padding: 10px;
    border: 1px solid #ccc;
    border-radius: 5px;
    font-size: 16px;
}

button {
    padding: 10px;
    border: none;
    background-color: #00796b;
    color: white;
    border-radius: 5px;
    cursor: pointer;
    font-size: 16px;
    transition: background-color 0.3s ease;
}

button:hover {
    background-color: #004d40;
}

.message {
    margin-top: 20px;
    color: #555;
}
```

```python
from textblob import TextBlob
import webbrowser
from pytube import Search

app = Flask(__name__)

def analyze_mood(text):
    blob = TextBlob(text)
    sentiment_polarity = blob.sentiment.polarity

    if sentiment_polarity > 0.5:
        return "excited"
    elif sentiment_polarity > 0.2 and sentiment_polarity <= 0.5:
        return "happy"
    elif sentiment_polarity > 0 and sentiment_polarity <= 0.2:
        return "content"
    elif sentiment_polarity < 0 and sentiment_polarity >= -0.2:
        return "sad"
    elif sentiment_polarity < -0.2 and sentiment_polarity >= -0.5:
        return "depressed"
    elif sentiment_polarity < -0.5:
        return "angry"
```

main.py Share Run

```python

def recommend_music(mood, music_type, language, attempt=0, preference=None):
    if preference:
        search_query = preference
    else:
        search_query = f"{music_type} songs in {language} for when you are feeling {mood}"
    search = Search(search_query)
    video_url = search.results[attempt].watch_url
    return video_url
```

```
    if request.method == "POST":
        user_input = request.form["user_input"]
        print(f"User input: {user_input}")
        if not hasattr(index, 'mood'):
            index.mood = analyze_mood(user_input)
            print(f"Analyzed mood: {index.mood}")
            return render_template("index.html", message=f"You seem
                to be feeling {index.mood}. What type of music would
                you like to listen to (e.g., pop, rock, classical)?"
                )
        elif not hasattr(index, 'music_type'):
            index.music_type = user_input
            print(f"Selected music type: {index.music_type}")
            return render_template("index.html", message="Great! Now
                , what language do you prefer for your music?")
        elif not hasattr(index, 'language'):
            index.language = user_input
```

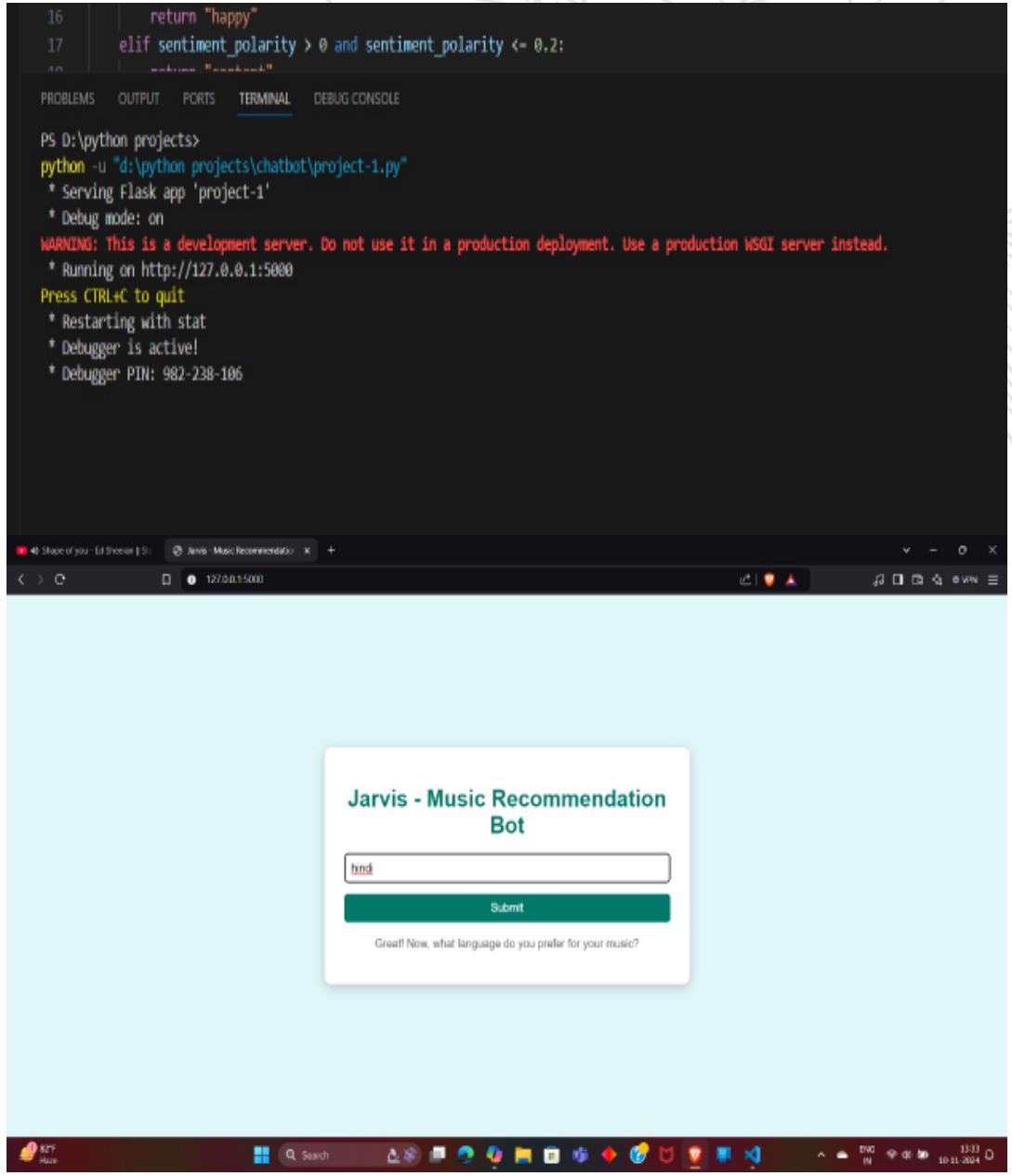
main.py — Share — Run

```
            print(f"Selected language: {index.language}")
            return redirect(url_for("recommendation"))
        elif hasattr(index, 'waiting_for_feedback') and index
            .waiting_for_feedback:
            index.waiting_for_feedback = False
            if user_input.lower() in ["no", "nah", "nope"]:
                print("User did not like the recommendation")
                index.attempt += 1  # Try the next recommendation
                return redirect(url_for("recommendation"))
            else:
                print("User liked the recommendation")
                delattr(index, 'mood')
                delattr(index, 'music_type')
                delattr(index, 'language')
                return render_template("index.html", message="Great!
                    Enjoy your music.")
        elif hasattr(index, 'waiting_for_preference') and index
            .waiting_for_preference:
            index.waiting_for_preference = False
            index.preference = user_input
            index.attempt = 0
            print(f"User preference: {index.preference}")
            return redirect(url_for("recommendation"))
    return render_template("index.html", message="Hi, I am Jarvis. I
        am your music assistant and I will help you find the perfect
        music. How are you feeling today?")

```

```
77 @app.route("/recommendation")
78 def recommendation():
79     try:
80         recommendation = recommend_music(index.mood, index
              .music_type, index.language, index.attempt, index
              .preference)
81         webbrowser.open(recommendation)
82         index.waiting_for_feedback = True
83         print(f"Music recommendation: {recommendation}")
84         return render_template("index.html", message=f"Here's some
              {index.language} music for you: {recommendation}. Did
              you like it?")
85     except IndexError:
86         # If no more recommendations are available
87         return render_template("index.html", message="I'm out of
              suggestions for now. Would you like to tell me your
```

output


return "happy"
elif sentiment_polarity > 0 and sentiment_polarity <= 0.2:
PROBLEMS
OUTPUT
PORTS
TERMINAL
DEBUG CONSOLE
PS D:\python projects>
python -u "d:\python projects\chatbot\project-1.py"
* Serving Flask app 'project-1'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 982-238-106
Jarvis - Music Recommendation Bot
hindi
Submit
Great! Now, what language do you prefer for your music?

Thank you