Initial Project Planning Report

Date	14 Dec 2024
Team ID	739884
Proje Name SmartLender - ct	Plant seedling classification using Deep learning
Maximum Marks	4 Marks

g

Product Backlog, Sprint Schedule, and Estimation

Use the below template to create a product backlog and sprint schedule

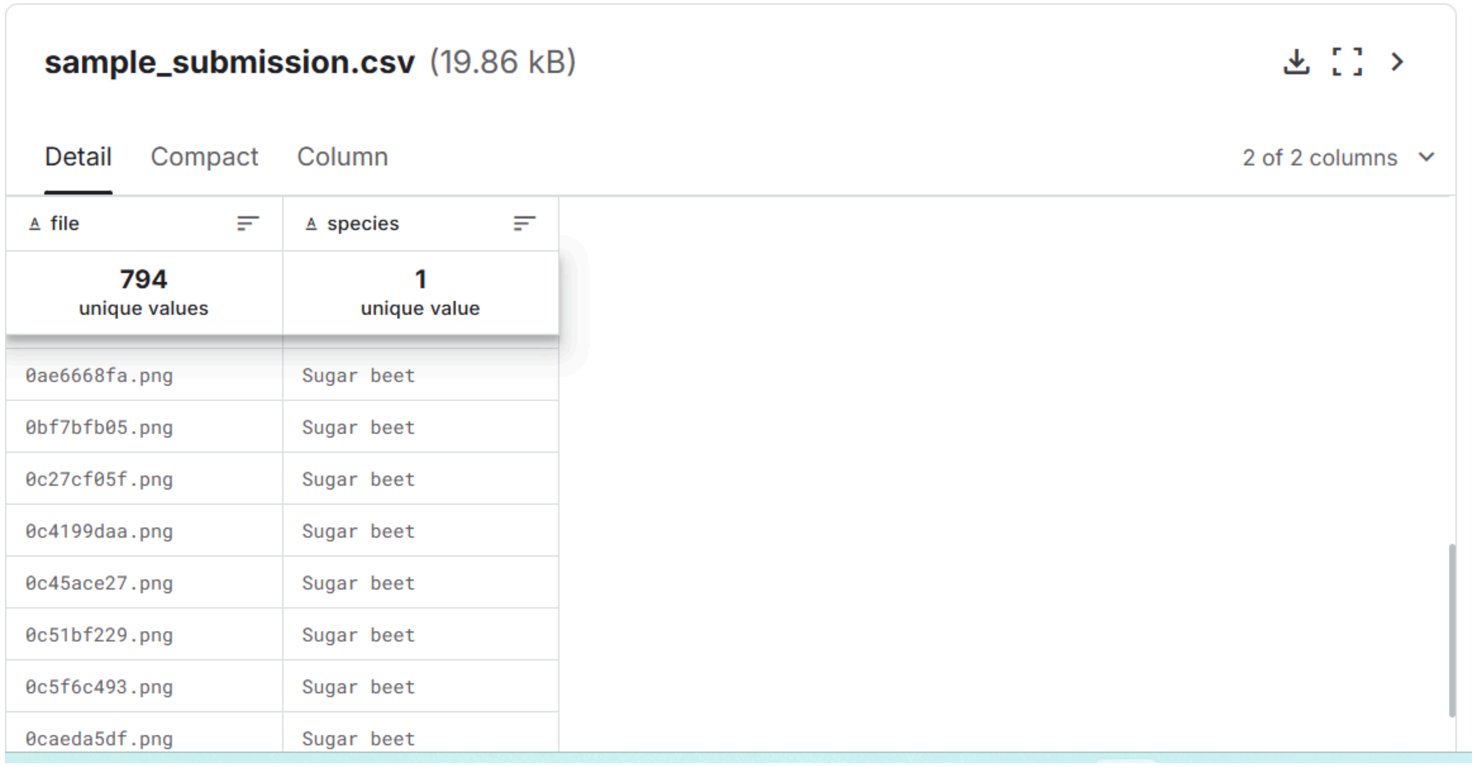
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority		Team Members	Sprint Start Date	Sprint End Date (Planned)
Sprint - 1	Data Collection and Preprocessing	SL-3	Understanding & loading data	Low		sathwika	2024/10/20	2024/10/21
Sprint-1	Data Collection and Preprocessing	SL-4	Data cleaning	High		sathwika	2024/10*/20*	2024/10/21

Sprint -	Data Collection	SL-5	EDA	Medium		sathwika	2024/10	2024/10/21

1	and Preprocessing						/20	
Sprint - 4	Project Report	SL-20	Report	High		fareed	2024/10/21	2024/10/22
Sprint - 2	Model Development	SL-8	Training the model	Medium		Rohan	2024/10/21	2024/10/22
Sprint - 2	Model tuning and testing	SL-13	Evaluating the model	Medium		Rohan	2024/10/21	2024/10/22

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Priority**	**Team Members**	**Sprint Start Date**	**Sprint End Date (Planned)**
Sprint-3	Web integration and Deployment	SL-16	Building Html templates	Low	varsha	2024/10/22	2024/10/24

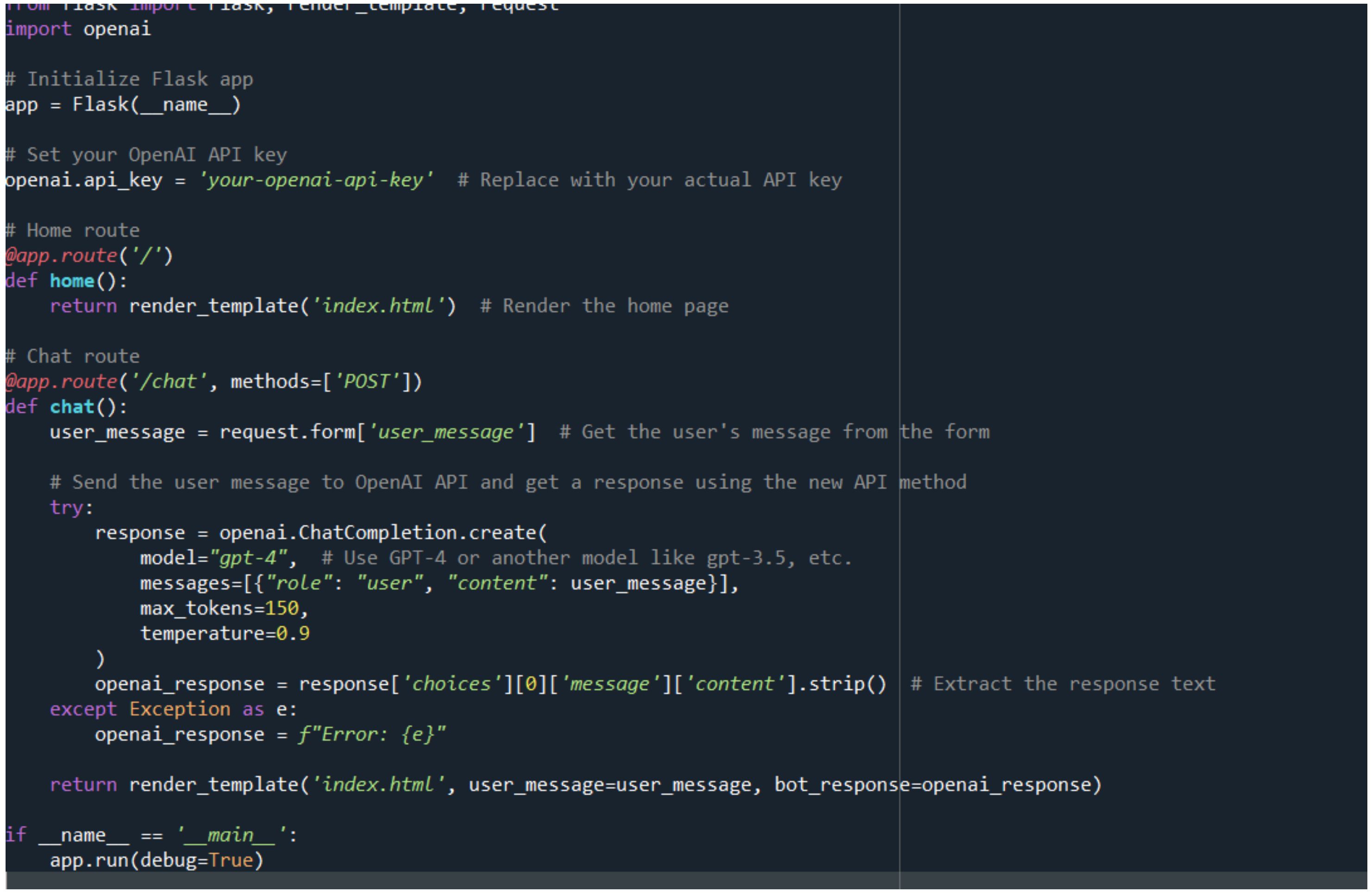
Sprint-3	Web integration and Deployment	SL-17 Local deployment	Local deployment	Medium	varsha	2024/10/24	2024/10/27


sample_submission.csv (19.86 kB)

Detail Compact Column 2 of 2 columns

file	species
794 unique values	1 unique value
0ae6668fa.png	Sugar beet
0bf7bfb05.png	Sugar beet
0c27cf05f.png	Sugar beet
0c4199daa.png	Sugar beet
0c45ace27.png	Sugar beet
0c51bf229.png	Sugar beet
0c5f6c493.png	Sugar beet
0caeda5df.png	Sugar beet

Screenshot:



```python
from flask import Flask, render_template, request
import openai

# Initialize Flask app
app = Flask(__name__)

# Set your OpenAI API key
openai.api_key = 'your-openai-api-key'  # Replace with your actual API key

# Home route
@app.route('/')
def home():
    return render_template('index.html')  # Render the home page

# Chat route
@app.route('/chat', methods=['POST'])
def chat():
    user_message = request.form['user_message']  # Get the user's message from the form

    # Send the user message to OpenAI API and get a response using the new API method
    try:
        response = openai.ChatCompletion.create(
            model="gpt-4",  # Use GPT-4 or another model like gpt-3.5, etc.
            messages=[{"role": "user", "content": user_message}],
            max_tokens=150,
            temperature=0.9
        )
        openai_response = response['choices'][0]['message']['content'].strip()  # Extract the response text
    except Exception as e:
        openai_response = f"Error: {e}"

    return render_template('index.html', user_message=user_message, bot_response=openai_response)

if __name__ == '__main__':
    app.run(debug=True)
```