Intelligent Crop Planning and Management using ML

Harinder Kaur

Department of Computing Studies & Information System, Douglas College

CSIS 4495: Applied Research Project

Dr. Padmapriya Arasanipalai Kandhadai

February 24, 2025

https://youtu.be/wj-8-0fIl5c

Intelligent Crop Planning and Management

Introduction

Agricultural productivity is heavily influenced by environmental conditions such as soil health and weather patterns. Farmers often struggle with selecting the optimal crops to cultivate based on real-time conditions. This project aims to build a machine learning-powered recommendation system that suggests the most suitable crops based on land area, location, soil conditions, and weather data. The system will also predict the time required from sowing to harvesting and the system will recommend suitable crops based on a user's land and location and offer continuous guidance throughout the crop growth cycle. By using real-time weather and soil APIs and predictive analytics, this project aims to enhance decision-making in agriculture, improving yield predictions.

With the availability of real-time weather, soil, and UV index data, the system will enable data-driven decision-making and predictive analytics for improved agricultural outcomes. Current farm management practices often only rely on manual monitoring and historical trends, which can lead to manual errors or not an optimal resource utilization. With the availability of real-time weather, soil, and UV index data, this system will improve agricultural outcomes.

Problem Statement

Farmers often struggle to determine the most suitable crops for their land, given fluctuating soil conditions, weather patterns, and climate change. Additionally, they lack real-time insights into when to irrigate and when to expect harvest readiness. The absence of a structured decision-support system results in inefficient resource utilization, reduced crop yields, and financial losses. This research aims to address the following key questions:

1. How can real-time soil and weather data be used to recommend the best crops for a given land area?
2. How can machine learning models predict the duration from sowing to harvest for different crops?
3. How can a weekly advisory system provide farmers with updates on irrigation and crop management actions based on real-time data?

Limitations Addressed in the Research

Previous research has demonstrated the use of remote sensing, soil analysis, and weather forecasting in precision agriculture. These studies have explored various machine learning models for crop prediction, including Decision Trees, Random Forests, etc. However, many of these models rely only on historical datasets and do not leverage real-time environmental data for continuous decision-making. Furthermore, existing solutions often require expensive IoT-based soil sensors, making them inaccessible to small-scale farmers. This research aims to bridge this gap by using real-time API-based soil and weather data to provide crop recommendations and without requiring physical sensor installations.

Hypothesis and Assumptions

There are two hypotheses for this project:

1. Hypothesis 1: Real-time soil moisture, temperature, and weather data can accurately predict the best crops for a specific location.
2. Hypothesis 2: Machine learning models can accurately estimate sowing-to-harvest time with high accuracy based on environmental conditions.

This research project assumes that the farmers will be able to accurately input their location and the data provided by APIs like soil moisture, temperature, and weather data is real-time and accurate.

Summary of the Initially Proposed Research Project

The initial proposal aimed to develop an AI-driven crop recommendation and farm management system that helps farmers make data-driven decisions based on real-time weather and soil conditions. The system would use machine learning models to recommend the best crops for a given land area and location, estimate the time from sowing to harvesting, and provide weekly management updates based on live API data.

Changes to the proposal

1. *Soil Data Source*

Initially, the project planned to use AgroAPI for soil data. However, AgroAPI only covers Europe, the United States, some parts of Canada. To address this limitation, I implemented Selenium to scrape soil temperature and moisture data from https://soiltemperature.app.

Justification. The change was necessary because AgroAPI's limited coverage would have restricted the system's usability for farmers outside Europe and the United States. Selenium was chosen as it allows scraping data from a more globally accessible source.

2. *Land Area Consideration*

In the proposal, the plan was to use land area as a factor for crop recommendations. The recommendations would be based on farmland area a farmer has and hence the results would be different for every user. The recommendations are now based solely on weather, location, and planting date.

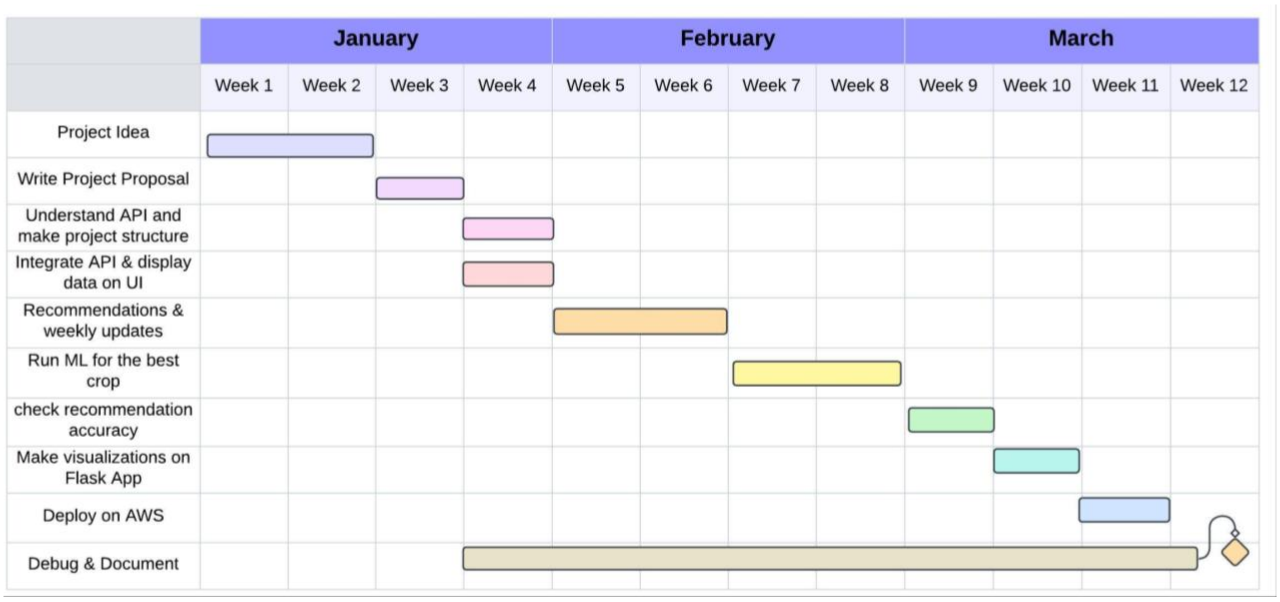
Justification. The inability to find a dataset that included land area as a feature necessitated this change. Therefore, the focus was on weather, location, and planting date as the primary factors for crop recommendations, as these were the readily available data points.

3. *Machine Learning Model*

I implemented a machine learning model that suggests the best crop based on N, P, K levels, and rainfall, but this model has not yet been integrated into the Flask app for recommendations.

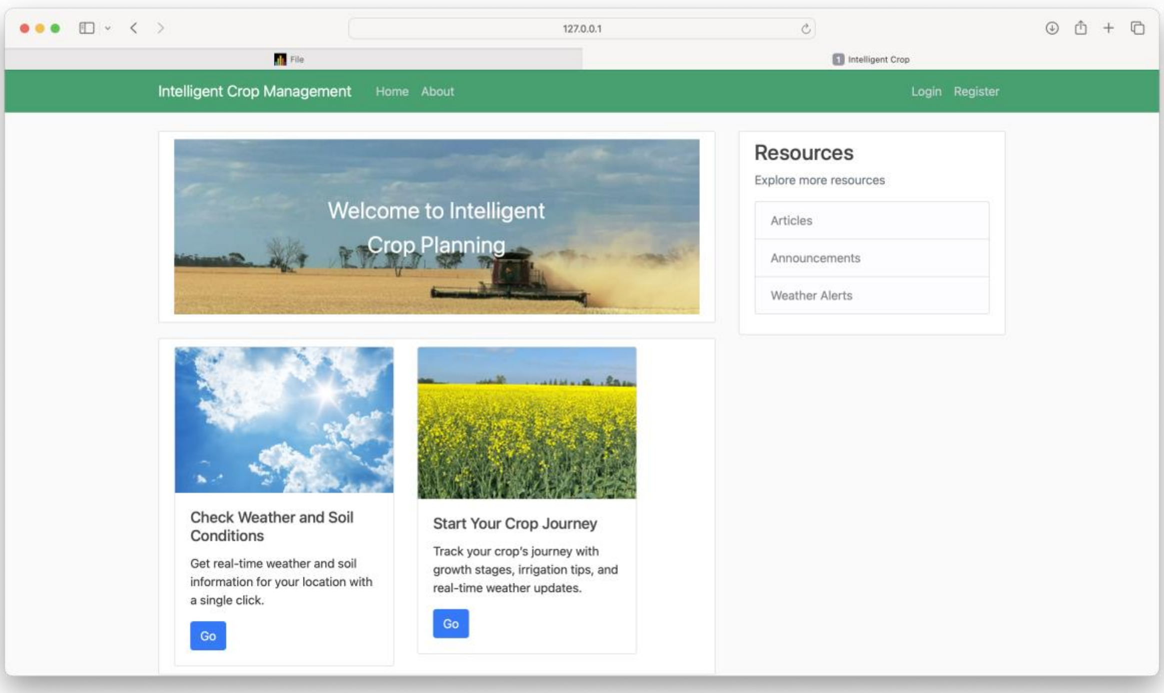
Justification. The inability to get real time values of N,P, and K. However, this can be solved by prompting the user the manually input the values.

Project Timeline



Implemented Features

Home Page



Welcome to Intelligent Crop Planning

Check Weather and Soil Conditions

Get real-time weather and soil information for your location with a single click.

Start Your Crop Journey

Track your crop's journey with growth stages, irrigation tips, and real-time weather updates.

Code.

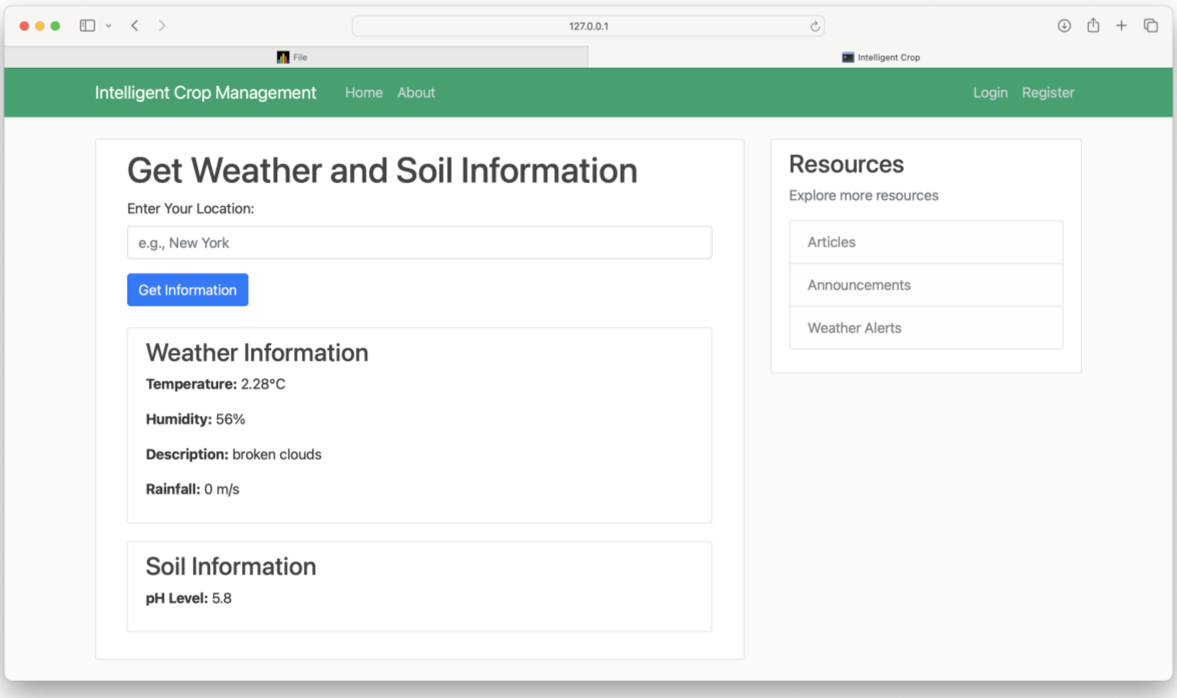
HTML.

```
{% extends "layout.html" %}
{% block content %}
        <article class="media content-section">
            <div style="position: relative; text-align: center; color: white;">
                <img src="/static/background.jpeg" class="img-fluid" alt="Responsive
image">
                <div style="position: absolute;
                  top: 50%;
                  left: 50%;
                  transform: translate(-50%, -50%);
                  color: white;
                  font-size: 30px;">
                  Welcome to Intelligent Crop Planning
                </div>
            </div>
          <br/>

        </article>

        <article class="media content-section">
        <div class="row">
            <div class="col-sm-6">
                <div class="card" style="width: 18rem;">
                    <img class="card-img-top" src="/static/weather.jpeg" alt="Card image
cap">
                    <div class="card-body">
                        <h5 class="card-title">Check Weather and Soil Conditions</h5>
                        <p class="card-text">Get real-time weather and soil information
for your location with a single click.</p>
```

```
                        <a href="/weather" class="btn btn-primary">Go</a>
                    </div>
                </div>
            </div>
            <div class="col-sm-6">
                <div class="card" style="width: 18rem;">
                    <img class="card-img-top" src="/static/crop_rec.jpeg" alt="Card image
cap">
                    <div class="card-body">
                        <h5 class="card-title">Start Your Crop Journey </h5>
                        <p class="card-text">Track your crop's journey with growth
stages, irrigation tips, and real-time weather updates.</p>
                        <a href="/care-plan" class="btn btn-primary">Go</a>
                    </div>
                </div>
            </div>

        </div>
        </article>

{% endblock content %}
```

Get Weather and Soil Information

This feature returns weather information such as temperature, humidity, rainfall and soil pH of user's entered location.



Get Weather and Soil Information

Enter Your Location:

e.g., New York

Weather Information

Temperature: 2.28°C

Humidity: 56%

Description: broken clouds

Rainfall: 0 m/s

Soil Information

pH Level: 5.8

Resources

Explore more resources

Articles

Announcements

Weather Alerts

Code.

```
{% extends "layout.html" %}
{% block content %}
    <article class="media content-section">
        <div class="container">
        <h1>Get Weather and Soil Information</h1>
        <form action="/weather" method="POST">
            <div class="form-group">
                <label for="location">Enter Your Location:</label>
                <input type="text" class="form-control" id="location" name="location"
placeholder="e.g., New York" required>
            </div>
            <button type="submit" class="btn btn-primary">Get
Information</button><br/><br/>
        </form>

        <!-- Display Weather Information -->

        {% if weather_data %}
        <article class="media content-section">
        <div class="weather-info">
            <h3 class="info-title">Weather Information</h3>
            <p><strong>Temperature:</strong> {{ weather_data.temperature }}°C</p>
            <p><strong>Humidity:</strong> {{ weather_data.humidity }}%</p>
            <p><strong>Description:</strong> {{ weather_data.description }}</p>
            <p><strong>Rainfall:</strong> {{ weather_data.rain }} m/s</p>
        </div>
        </article>
        {% endif %}


        <!-- Display Soil Information -->

        {% if ph_value %}
        <article class="media content-section">
        <div class="soil-info">
            <h3 class="info-title">Soil Information</h3>
            <p><strong>pH Level:</strong> {{ ph_value }}</p>
        </div>
        </article>
        {% endif %}

    </div>
    </article>
{% endblock content %}
```

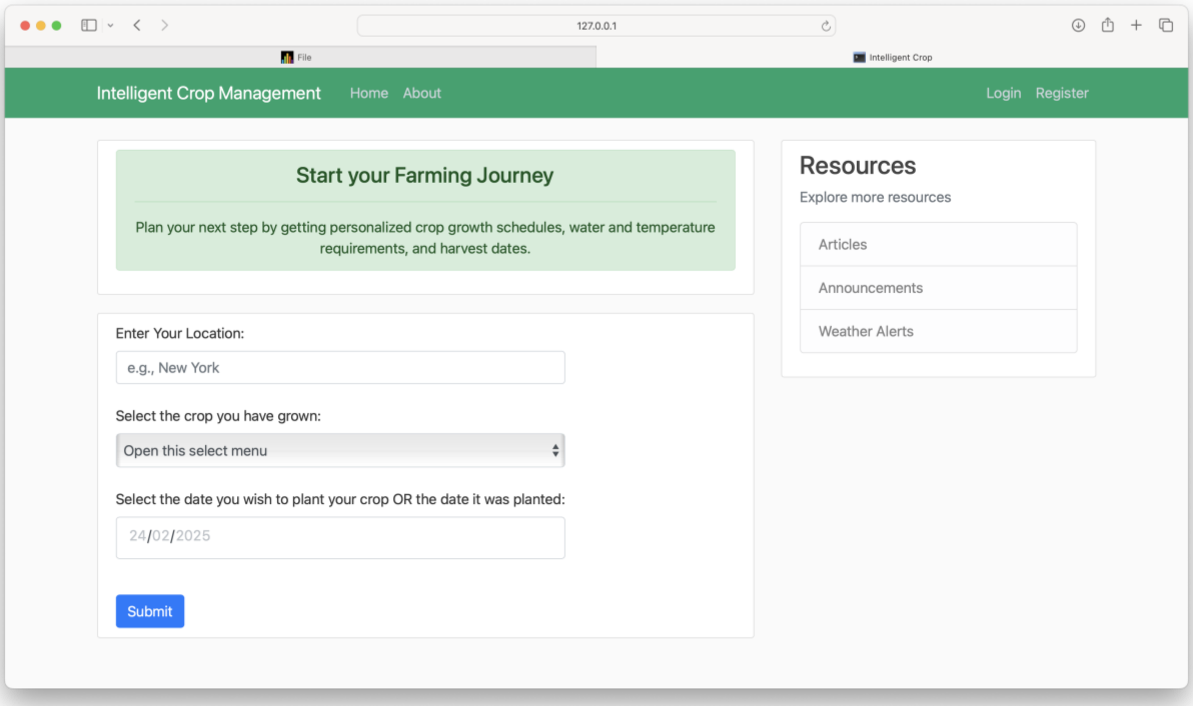
Python.

```
api_key='ea284063eb75d6986cbf37b5f104a552' # <====API KEY=======|

def get_weather_data(api_key, location):
    base_url = 'http://api.openweathermap.org/data/2.5/weather?'
    complete_url = base_url + 'q=' + location + '&appid=' + api_key + '&units=metric'
    response = requests.get(complete_url)
    return response.json()
```

```python
# Method to get weather data from the 'OpenWeather' API's JSON response
========================
def parse_weather_data(data):
    if data['cod'] == 200:  # Check HTTP 'OK' Status Code
        main = data['main']
        coord = data['coord']
        lon = coord['lon']
        lat =coord['lat']

        if 'rain' in data: # If there is no rain today
            rain_json = data['rain']
            rain = rain_json['1h']
        else:
            rain = 0

        weather_desc = data['weather'][0]['description']
        weather_data = {
            'lon': lon,
            'lat': lat,
            'temperature': main['temp'],
            'humidity': main['humidity'],
            'description': weather_desc,
            'rain':rain
        }
        return weather_data
    else:
        return {'Error': data.get('message', 'Unable to fetch data')}


@app.route('/weather', methods=['GET', 'POST'])
def weather():
    weather_data = None
    ph_value = None
    if request.method == 'POST':
        location = request.form['location']

        weather_json= get_weather_data(api_key, location)
        if weather_json.get('cod') == 200:
            weather_data = parse_weather_data(weather_json)
        else:
            weather_data = {'Error': weather_data.get('message', 'Unable to fetch data')}


        ph_value = get_ph_value(weather_json)
    return render_template('weather.html', weather_data=weather_data, ph_value=ph_value)
```

Plan Crop Journey

This feature lets the user input the crop they are interested in, their location and date. Based on these variables, ideal time for the growth of that crop and weekly care schedule is displayed.



Start your Farming Journey

Plan your next step by getting personalized crop growth schedules, water and temperature requirements, and harvest dates.

Resources

Explore more resources

Articles

Announcements

Weather Alerts

Enter Your Location: e.g., New York

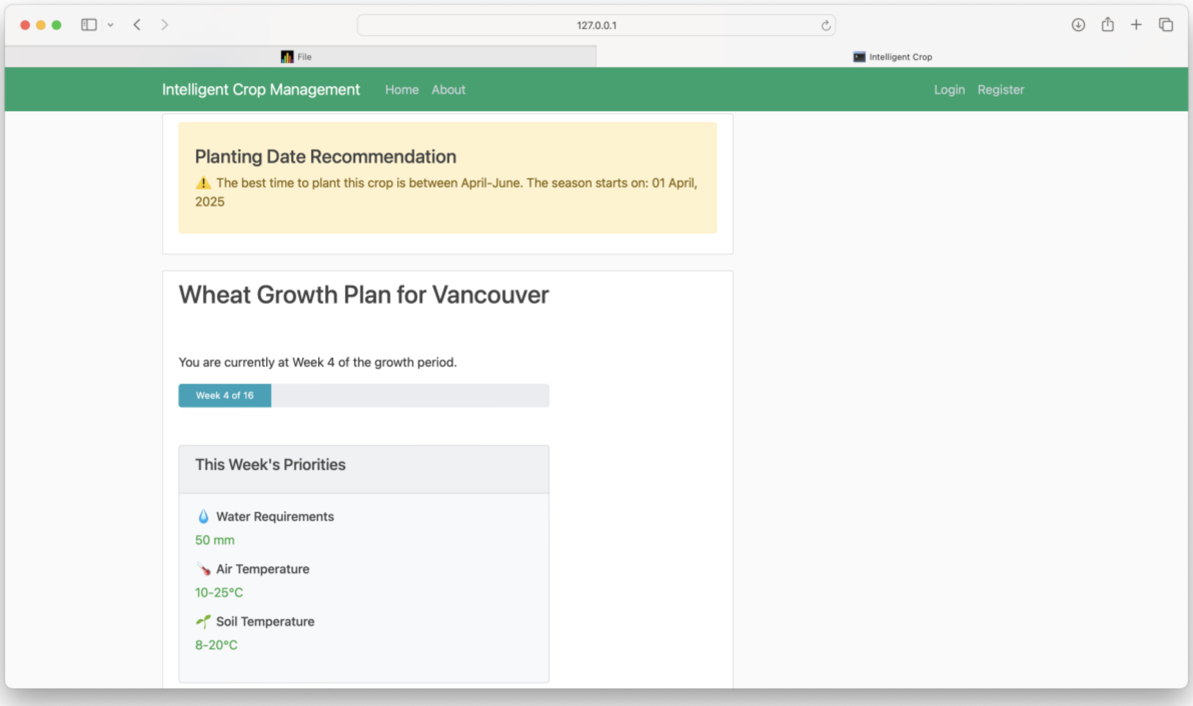
Select the crop you have grown: Open this select menu

Select the date you wish to plant your crop OR the date it was planted: 24/02/2025

HTML Code.

```
{% extends "layout.html" %}
{% block content %}
       <article class="media content-section">
              <div class="alert alert-success" role="alert">
            <center>
          <h4 class="alert-heading">Start your Farming Journey</h4>
          <hr>
          <p class="mb-0">Plan your next step by getting personalized crop growth
schedules, water and temperature requirements, and harvest dates.</p>
          </center>
        </div>
       </article>
       <article class="media content-section">
               <form action= "/care-plan" method="POST">
                      <div class="form-group">
                             <label for="location">Enter Your Location:</label>
                        <input type="text" class="form-control" id="location"
name="location" placeholder="e.g., New York" required><br/>


                           <label for="crop">Select the crop you have grown:  </label>
                           <select class="form-control" aria-label="Default select
example" id="crop" name="crop" required>
                               <option selected>Open this select menu</option>
                               <option value="rice">Rice</option>
                               <option value="wheat">Wheat</option>
                               <option value="maize">Maize</option>
                               <option value="apple">Apple</option>
                               <option value="banana">Banana</option>
                               <option value="blackgram">Blackgram</option>
                               <option value="chickpea">Chickpea</option>
                               <option value="coconut">Coconut</option>
                               <option value="coffee">Coffee</option>
                               <option value="cotton">Cotton</option>
                               <option value="grapes">Grapes</option>
                               <option value="jute">Jute</option>
                               <option value="kidneybeans">Kidney Beans</option>
                               <option value="lentil">Lentil</option>
                               <option value="mango">Mango</option>
                               <option value="mothbeans">Moth Beans</option>
                               <option value="mungbean">Mung Bean</option>
                               <option value="muskmelon">Muskmelon</option>
                               <option value="orange">Orange</option>
                               <option value="papaya">Papaya</option>
                               <option value="pigeonpeas">Pigeon Peas</option>
                               <option value="pomegranate">Pomegranate</option>
                               <option value="watermelon">Watermelon</option>
                           </select><br/>

                    <label for="planting_date">Select the date you wish to plant your
crop OR the date it was planted:</label>
                    <input type="date" class="form-control" id="planting_date"
name="planting_date" required
                        pattern="\d{4}-\d{2}-\d{2}"><br/>

                      </div>
                      <button type="submit" class="btn btn-primary">Submit</button>
               </form>
       </article>
    {% if future_planting_message %}
            <article class="media content-section">
                <p></p>
              <div class="alert alert-warning">
                <p></p>
                <h4>Planting Date Recommendation</h4>
                <p>{{ future_planting_message }}</p>
              </div>
            </article>
    {% endif %}

    {% if harvest_message %}
    <article class="media content-section">
        <div class="harvest-alert">
            <h2>{{ harvest_message }}</h2>
        </div>
    </article>
    {% endif %}

    {% if recommendations %}

    <article class="media content-section">

        <div class="soil-info">
            <h2>{{ recommendations.crop }} Growth Plan for {{ location }}</h2><br/><br/>
            <div class="progress-container">
                <p> You are currently at Week {{ recommendations.current_week }} of the
growth period.</p>
                <div class="progress" style="height: 30px;">
                    <div class="progress-bar bg-info" style="width: {{
(recommendations.current_week/recommendations.total_weeks)*100 }}%">
                        Week {{ recommendations.current_week }} of {{
recommendations.total_weeks }}
                    </div>
                </div>
            </div><br/><br/>

            <div class="card bg-light mb-3" style="max-width: 60rem;">
```

```
              <div class="card-header">
                <h5 class = "card-title">This Week's Priorities </h5></div>
              <div class="card-body text-success">
                <div class="current-week">
                    <div class="metric-box">
                        <h6>💧 Water Requirements</h6>
                        <p>{{ recommendations.weekly_plan[0].water }} mm</p>
                    </div>
                    <div class="metric-box">
                        <h6>🌡 Air Temperature</h6>
                        <p>{{ recommendations.weekly_plan[0].air_temp }}</p>
                    </div>
                    <div class="metric-box">
                        <h6>🌱 Soil Temperature</h6>
                        <p>{{ recommendations.weekly_plan[0].soil_temp }}</p>
                        {% if soil_temperature %}
                        <small>Current: {{ soil_temperature }}°C</small>
                        {% endif %}
                    </div>
                </div>

              </div>
            </div>
            <br/><br/>

            <h3>Upcoming Schedule</h3>
            <div class="weekly-plan">
                <table class="table table-striped">
                    <thead>
                        <tr class="table-success">
                            <th>Week</th>
                            <th>Water (mm)</th>
                            <th>Air Temp (°C)</th>
                            <th>Soil Temp (°C)</th>
                        </tr>
                    </thead>
                    <tbody>
                        {% for week in recommendations.weekly_plan %}
                        <tr class="{% if week.week == recommendations.current_week
%}table-active{% endif %}">
                            <td>Week {{ week.week }}</td>
                            <td>{{ week.water }}</td>
                            <td>{{ week.air_temp }}</td>
                            <td>{{ week.soil_temp }}</td>
                        </tr>
                        {% endfor %}
                    </tbody>
                </table>
            </div><br/>

            <div class="card border-info mb-3" style="max-width: 60rem;">
              <div class="card-header">
                <h5 class="card-title">Harvest Information</h5>
              </div>
              <div class="card-body text-success">
                <p class="card-text">Expected Harvest Date: {{ harvest_date }}</p>
                <p class="card-text">Weeks Remaining: {{ weeks_remaining }}</p>
              </div>
            </div>


            {% if recommendations.actions %}
            <h3>Immediate Actions that you can take</h3>
            <ul class="action-list">
                {% for action in recommendations.actions %}
                <li>{{ action }}</li>
                {% endfor %}
            </ul>
            {% endif %}


        </div>
    </article>
    {% endif %}

{% endblock content %}
```



127.0.0.1

Intelligent Crop Management Home About Login Register

Planting Date Recommendation

⚠️ The best time to plant this crop is between April-June. The season starts on: 01 April, 2025

Wheat Growth Plan for Vancouver

You are currently at Week 4 of the growth period.

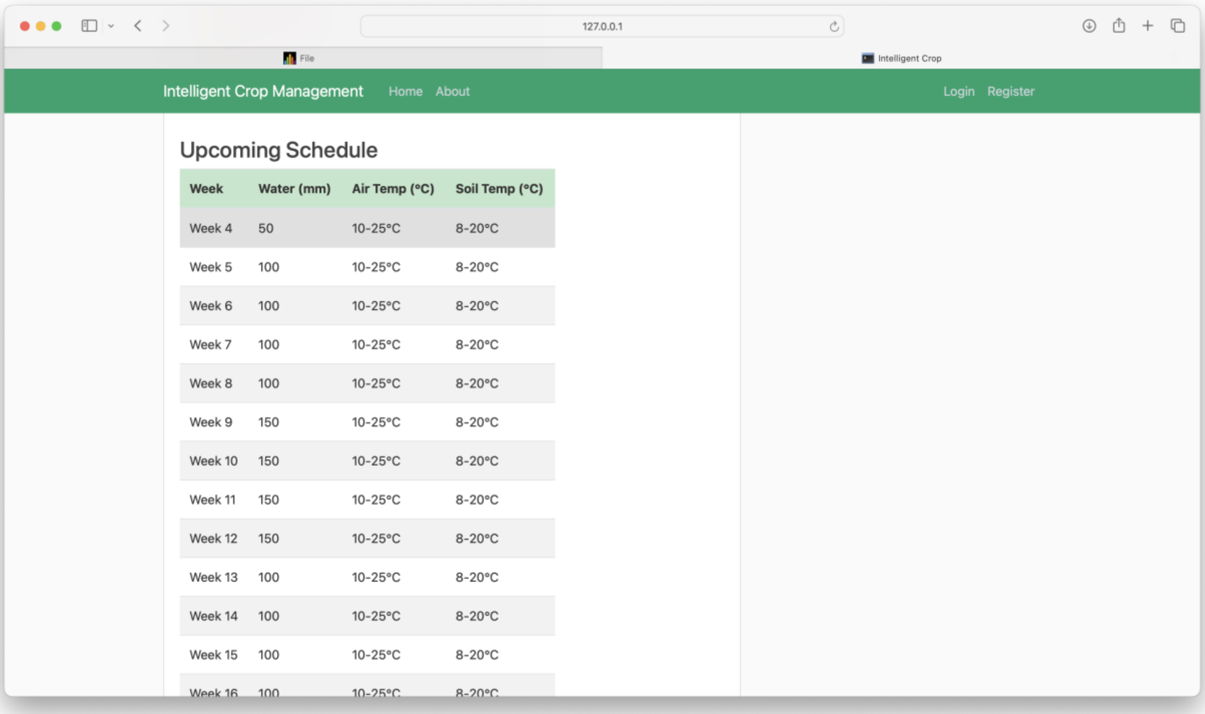
Week 4 of 16

This Week's Priorities

💧 Water Requirements
50 mm

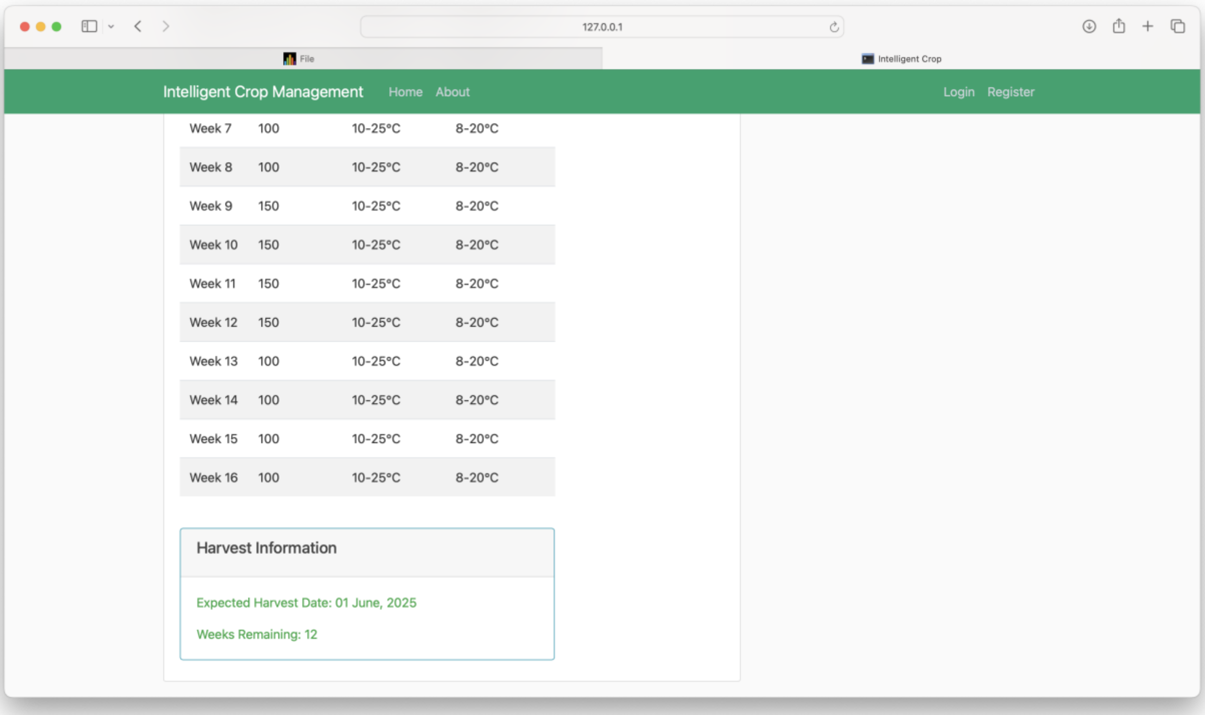
Air Temperature
10-25°C

🌱 Soil Temperature
8-20°C



Upcoming Schedule

Week	Water (mm)	Air Temp (°C)	Soil Temp (°C)
Week 4	50	10-25°C	8-20°C
Week 5	100	10-25°C	8-20°C
Week 6	100	10-25°C	8-20°C
Week 7	100	10-25°C	8-20°C
Week 8	100	10-25°C	8-20°C
Week 9	150	10-25°C	8-20°C
Week 10	150	10-25°C	8-20°C
Week 11	150	10-25°C	8-20°C
Week 12	150	10-25°C	8-20°C
Week 13	100	10-25°C	8-20°C
Week 14	100	10-25°C	8-20°C
Week 15	100	10-25°C	8-20°C
Week 16	100	10-25°C	8-20°C



Week 7	100	10-25°C	8-20°C
Week 8	100	10-25°C	8-20°C
Week 9	150	10-25°C	8-20°C
Week 10	150	10-25°C	8-20°C
Week 11	150	10-25°C	8-20°C
Week 12	150	10-25°C	8-20°C
Week 13	100	10-25°C	8-20°C
Week 14	100	10-25°C	8-20°C
Week 15	100	10-25°C	8-20°C
Week 16	100	10-25°C	8-20°C

Harvest Information

Expected Harvest Date: 01 June, 2025

Weeks Remaining: 12

Python.

```
@app.route('/care-plan', methods=['GET', 'POST'])
def care_plan():
    if request.method == 'POST':
        # Get user inputs
        selected_crop = request.form['crop'].strip().lower()  # Clean input
        location = request.form['location']
        planting_date = datetime.strptime(request.form['planting_date'], '%Y-%m-%d')
        current_date = datetime.now()

        # Fetch weather data
        weather_json = get_weather_data(api_key, location)
        if weather_json.get('cod') != 200:
            return render_template('care_plan.html', error="Location not found")
        weather_data = parse_weather_data(weather_json)
        lon = weather_data.get('lon', 0)


        # Load crop data
        df_crop = pd.read_csv('csv-files/growth-conditions.csv')
        df_crop['Crop'] = df_crop['Crop'].str.strip().str.lower()  # Clean CSV data

        # Validate crop exists
        crop_filter = df_crop['Crop'] == selected_crop
        if not crop_filter.any():
            return render_template('care_plan.html',
                                   error=f"Crop '{selected_crop}' not found in database.
Please select a valid crop.")

        crop_data = df_crop[crop_filter].iloc[0]  # Safe after validation

        # Get ideal planting months
        print(lon, 'and')
        if lon >= 0:
            hemisphere = 'Northern'
        else:
            hemisphere = 'Southern'
        if hemisphere == 'Northern':
            start_month = crop_data['NH_Planting_Start']
            end_month = crop_data['NH_Planting_End']
        elif hemisphere == 'Southern':
            start_month = crop_data['SH_Planting_Start']
            end_month = crop_data['SH_Planting_End']

        # Calculate next valid planting date
        current_year = datetime.now().year
        current_month = datetime.now().month
        if start_month <= end_month:
            valid_months = list(range(start_month, end_month + 1))
        else:
            valid_months = list(range(start_month, 13)) + list(range(1, end_month + 1))

        candidates = [m for m in valid_months if m > current_month]
        if candidates:
            next_valid_month = min(candidates)
            next_year = current_year
        else:
            next_valid_month = valid_months[0]
            next_year = current_year + 1

        suggested_date = datetime(next_year, next_valid_month, 1)
        message = None

        # Validate planting date
        is_ideal = (
            (start_month <= planting_date.month <= end_month)
            if start_month <= end_month
            else (planting_date.month >= start_month or planting_date.month <= end_month)
        )

        # Check if the user's selected date is within or outside the season
        if planting_date > current_date and not is_ideal:
            suggested_date = datetime(next_year + 1, next_valid_month, 1)
            message = (
                f"⚠️ The best time to plant this crop is between
{calendar.month_name[start_month]}-{calendar.month_name[end_month]}. "
                f"Consider planting next year on: {suggested_date.strftime('%d %B, %Y')}"
            )
        elif planting_date <= current_date and not is_ideal:
            message = (
                f"⚠️ The best time to plant this crop is between
{calendar.month_name[start_month]}-{calendar.month_name[end_month]}. "
                f"The season starts on: {suggested_date.strftime('%d %B, %Y')}"
            )
        else:
            message = (
                f"You're currently on the optimal timeline for planting! The best time to
grow is between {calendar.month_name[start_month]}-{calendar.month_name[end_month]}. "
                f"You can start planting this crop."
            )

        # Calculate timeline
        elapsed_days = (current_date - planting_date).days
        elapsed_weeks = elapsed_days // 7
        current_week = max(elapsed_weeks + 1, 1)  # Ensure minimum week 1
        growth_months = crop_data['Growth Season (Months)']
        total_weeks = int(growth_months * 4)
        total_days = int(growth_months * 30)
        harvest_date = planting_date + timedelta(days=total_days)

        # Generate recommendations
        recommendations = {
            'crop': selected_crop.title(),
            'current_week': min(current_week, total_weeks),
            'total_weeks': total_weeks,
            'weekly_plan': [],
            'actions': []
        }

        # Weekly plan logic
        for week in range(recommendations['current_week'], total_weeks + 1):
            month = ((week - 1) // 4) + 1
            month = max(min(month, 12), 1)  # Clamp between 1-12

            water_key = f'Water Needs (mm) - Month {month}'
            if water_key not in crop_data:
                continue

            weekly_plan_entry = {
                'week': week,
                'water': crop_data[water_key],
                'air_temp': f"{crop_data['Min Temp (°C)']}-{crop_data['Max Temp
(°C)']}°C",
                'soil_temp': f"{crop_data['Min Soil Temp (°C)']}-{crop_data['Max Soil
Temp (°C)']}°C"
            }
            recommendations['weekly_plan'].append(weekly_plan_entry)

        # Get Soil data
        try:
            ph_value = get_ph_value(weather_json)
            soil_data = fetch_soil_data_selenium(weather_data['lon'],
weather_data['lat'])
```

```
            # ... (soil data parsing logic)
        except Exception as e:
            ph_value = 5.8
            soil_temperature = soil_moisture = None

        return render_template(
            'care_plan.html',
            recommendations=recommendations,
            harvest_date=harvest_date.strftime('%d %B, %Y'),
            weeks_remaining=total_weeks - recommendations['current_week'],
            future_planting_message=message,
            ph_value=ph_value,
            location=location,
            today=datetime.now().strftime('%Y-%m-%d')
        )

    return render_template('care_plan.html', today=datetime.now().strftime('%Y-%m-%d'))
```

Work Log

Date	Number of Hours	Work Done
February 11, 2025	1	Worked on designing the home page
Februray 12, 2025	2	Used a new dataset to compare user's conditions to ideal crop conditions
February 13, 2025	3	Implemented the new methods in front end, i.e., duration of the growth of a crop, changes needed in soil or weather to enhance crop growth and irrigation recommendations
February 14, 2025	2	cleaned the dataset and merged the new data into existing csv
February 15, 2025	2	Used the new dataset to generate ideal growth timeline
February 17, 2025	2	Displayed the weekly schedule on the front end of the flask app
February 18, 2024	3	Spend this time on handling run time errors created by null and inconsistent API responses
February 20,2025	2	Worked on designing the crop planning page
February 21,2025	1.5	Debugging any errors and improving css and styling