

Crop Recommendation System using AIML

ON

Submitted in partial fulfillment of the requirements
of the degree of

**Bachelor of Engineering
(Information
Technology)**

By

Riya Varyani - Roll No (61)

Under the guidance of

Dipti Karani



Department of Information Technology

**VIVEKANAND EDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY,
Chembur, Mumbai 400074**

(An Autonomous Institute, Affiliated to University of Mumbai)

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature

) Sonam Chhabadiya
09

Abstract

This project is a full-stack responsive food ordering website developed using Flask and MongoDB. The platform provides users the ability to browse food items, add them to a cart, and place online orders with secure payments through Stripe. It also includes a contact form powered by Formspree for managing custom and bulk orders. The backend is efficiently designed with Flask and MongoDB, while the frontend uses HTML5, CSS3, Bootstrap, and Jinja2 templates for dynamic rendering. The project incorporates secure user authentication using Flask-Login and offers an admin dashboard for managing menu items. With its modern tech stack and thoughtful user-centric design, the project streamlines the food ordering experience and provides a foundation for scalable growth.

Contents

1. Introduction

- 1.1 Introduction
- 1.2 Objective
- 1.3 Feasibility Study
- 1.4 Organization of the report

2. Design and Implementation

- 2.1 Block Diagram
- 2.2 Hardware Requirements
- 2.3 Software Requirements
- 2.4 Front End Development Framework and Libraries
- 2.5 Back End Development and Database Management

3. Results and Discussion

- 3.1 Results of Implementation
- 3.2 Google Analysis

4. Conclusion

- 4.1 Conclusion

Chapter 1:

Introduction

1.1 Introduction

This project is a fully responsive Crop Recommendation System developed using Flask, an intuitive Python-based web framework, combined with powerful machine learning algorithms. It is designed to assist farmers, agricultural officers, or individuals in identifying the most suitable crops to cultivate, based on soil characteristics, real-time weather conditions, and other key environmental factors.

1.2 Objectives:

- To recommend suitable crops based on soil and real-time weather data using machine learning.
- To integrate OpenWeatherMap API for live weather inputs.
- To build a secure, responsive web app with user authentication.
- To visualize data and predictions through interactive charts.
- To promote smart, sustainable, and data-driven farming practices.

1.3 Feasibility Study:

The proposed food ordering web application is technically and operationally feasible for small and mid-scale restaurants looking to digitize their services. Built using open-source technologies like Flask and MongoDB, it ensures cost-effectiveness, flexibility, and scalability. The responsive design allows smooth access across all devices, while the user-friendly interface simplifies tasks such as menu management and order tracking. Economically, it reduces development and maintenance costs and avoids third-party commissions, helping increase profit margins. It also supports secure authentication and responsible data handling. With the growing demand for digital solutions in the food industry, this project is a practical and market-ready solution to improve customer experience and streamline operations.

1.4 Organization of the report

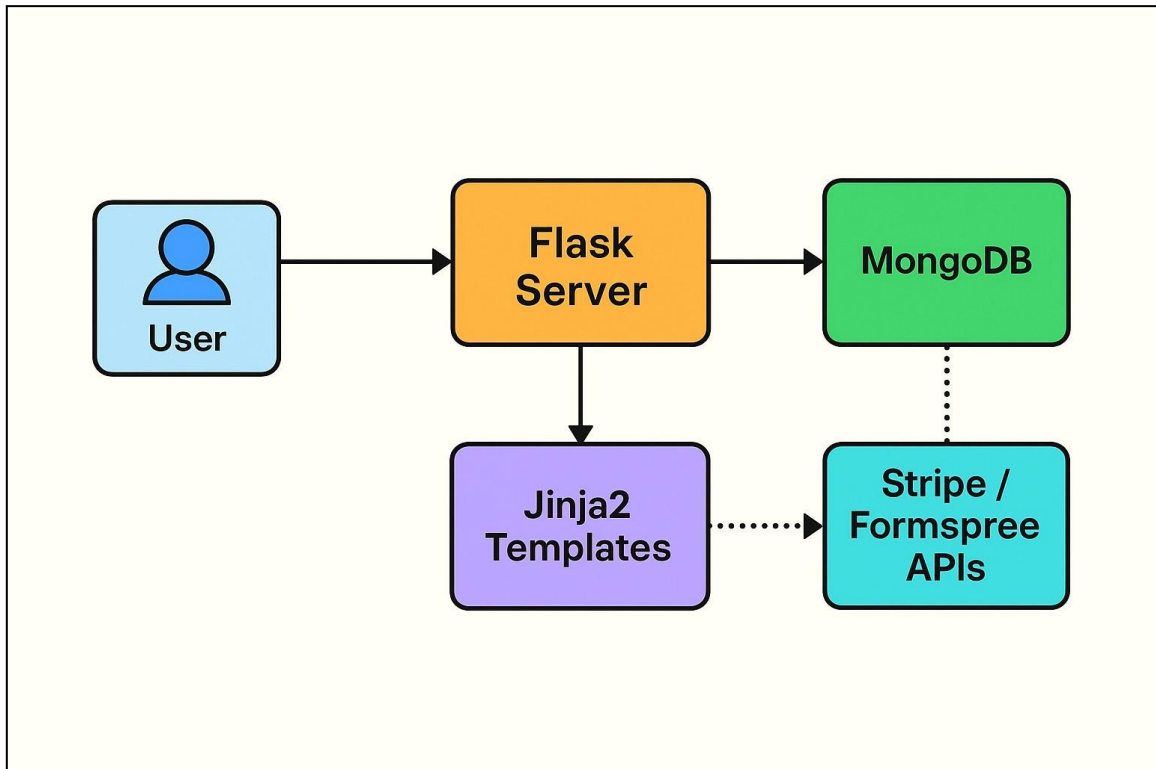
The Crop Recommendation System is organized into key modules for efficient functionality:

1. **User Interface:** A responsive front-end where users input soil and location details, with login and registration features.
2. **Weather Integration:** Fetches real-time weather data via the **OpenWeatherMap API**.
3. **Machine Learning:** Processes user and weather data to provide crop recommendations.
4. **Recommendation Display:** Shows the most suitable crops along with insights like charts.
5. **Admin/Backend:** Allows administrators to manage data, retrain models, and receive alerts.
6. **Database:** Stores user profiles, prediction history, and weather data logs.
7. **Scalability:** The system can be expanded to include mobile apps, AI tools, and IoT sensor integration.

Chapter 2:

Design and Implementation

2.1 Block Diagram



1. User Interface (Frontend Layer)

Users (Customers):

They interact with the platform through a web browser. Users can browse the menu, add food items to their cart, and proceed to place an order.

Technologies Used:

- HTML5, CSS3, and Bootstrap for building a responsive and visually appealing interface.
- Jinja2 template engine to dynamically render HTML pages with data passed from the Flask backend.

2. Flask Server (Backend Layer)

Flask Framework:

Acts as the core controller of the system. It manages routing, request-response handling, session management, and acts as the bridge between the frontend and database.

Authentication System:

Implemented using Flask-Login to allow secure user registration, login, logout, and session control. Certain routes are protected and only accessible to authenticated users.

Order Management:

Handles the logic for cart functionality, order processing, and transferring order data to the payment gateway.

Contact Form Integration:

A contact form is provided for users to place bulk orders or inquiries. This is integrated using Formspree, which directly sends submitted queries to the admin's email.

3. Real-Time Weather

Open weather Map API:

The **OpenWeather API** provides real-time and forecasted weather data, which can be useful for applications that need to deliver weather information to users

4. Database Layer

- **MongoDB:**
A NoSQL database used to store user details, menu items, and order history. It supports flexibility and scalability.
- **MongoDB Compass:**
A GUI tool for managing and visually monitoring collections such as users, food items, and orders

2.2 Hardware Requirements

- **CPU:** Dual-core or Quad-core processor
- **RAM:** 4 GB or higher
- **Storage:** SSD with at least 100 GB of free space
- **Network:** High-speed internet connection

- **User Devices:** Any modern desktop or laptop with a browser
-

2.3 Software Requirements

Languages and Technologies Used:

- **Frontend:** HTML5, CSS3, Bootstrap, Jinja2
 - **Backend:** Python, Flask
 - **Database:** MongoDB (NoSQL)
-

2.4 Front End Development Frameworks and Libraries:

- **HTML5 & CSS3:** For structuring and styling web pages
 - **Bootstrap:** For responsive layout and component design
 - **Jinja2:** For rendering dynamic content from backend to frontend
 - **JavaScript (Basic):** For enhancing interactivity in UI components
-

Tools and Editors:

- **Visual Studio Code:** Code editor used for development
- **Python (3.x):** Backend programming language
- **MongoDB Compass:** GUI for managing and visualizing database content
- **Git & GitHub:** Version control and project collaboration
- **Browser:** Google Chrome or any modern browser for testing and running the app

2.5 Backend Development & Database Management:

- **Flask:** Python microframework used to build backend routes, logic, and API interactions
 - **MongoDB:** NoSQL database for storing food items, orders, and user data
 - **Flask-Login:** For user authentication and session management
 - **Open Weather Map API:** For real time weather information for users
-

Development and Hosting (Optional for Future Deployment):

- **Hosting Platforms:**
 - **Render / Heroku / PythonAnywhere** (for Flask app deployment)
 - **MongoDB Atlas:** For hosting MongoDB in the cloud
 - **Netlify (optional):** If frontend were separated from Flask

Chapter 3:

Results

Hosted Link : <https://cropwise61.onrender.com>

Github Link: <https://github.com/riyav702/Crop-Recommendation---Copy>

3.1 Results of Implementation:

Login Page:

Login to CropWise

Login

Don't have an account? Signup here

Signup for CropWise

Signup


Already have an account? Login here


Home Page:


CropWise

Smart Crop Recommendation

An intelligent system to help farmers grow better. Enter your soil and climate values, and we'll tell you what to grow!


Input Data
Fill in values for N, P, K, temperature, pH, humidity, and rainfall.


Get Recommendations
Our model matches your inputs with suitable crops.


Grow Smarter
Use insights to plan better, smarter farming decisions.

Get Started Now

Get Crop Recommendation:

Crop Recommendation System

Nitrogen (N)

Phosphorus (P)

Potassium (K)

Temperature (°C)

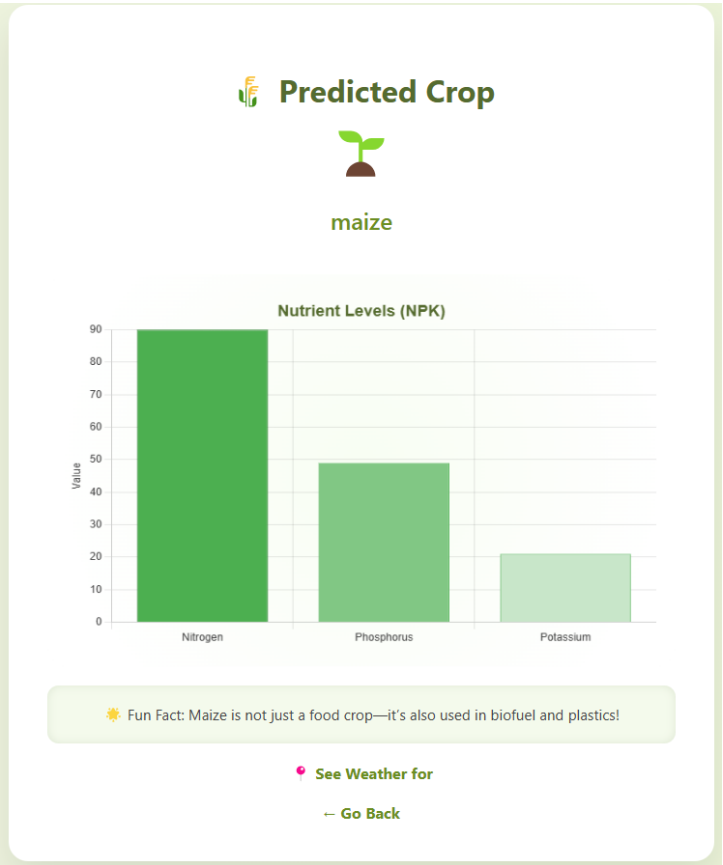
Humidity (%)

Soil pH

Rainfall (mm)

Predict Crop

Predicted Crop:



Crop Suitability based on weather:

Check Crop Suitability Based on Weather

mumbai


Get Crop Advice


Weather in Mumbai

Temperature: 28.5 °C

Humidity: 65%

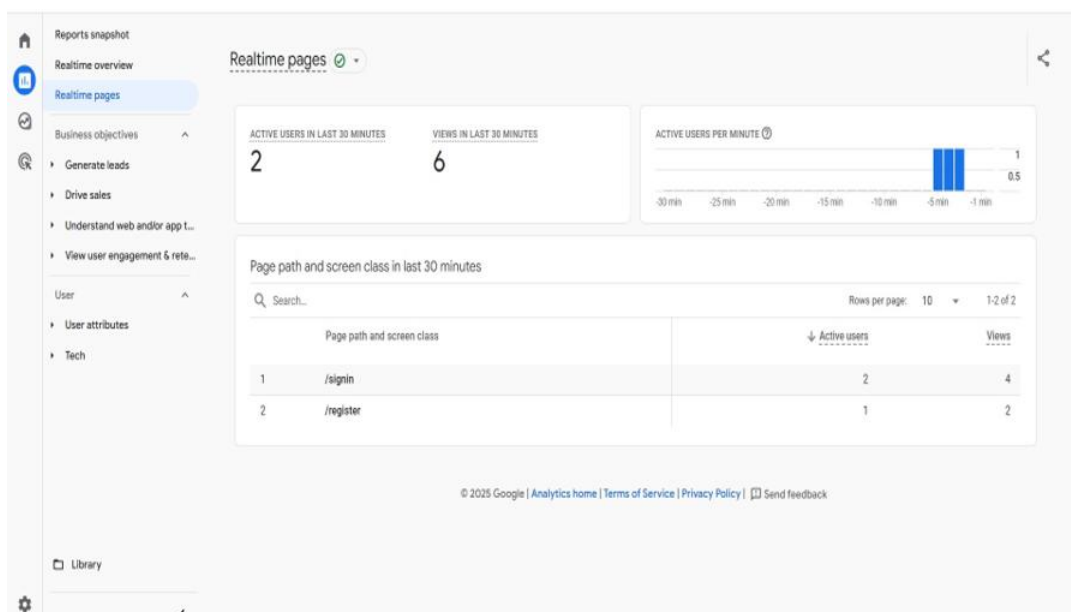
Condition: clear sky

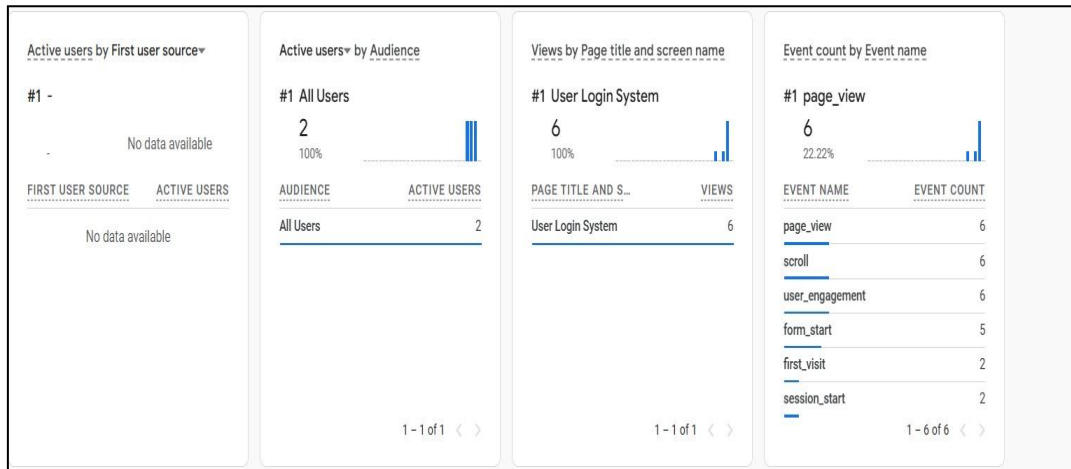
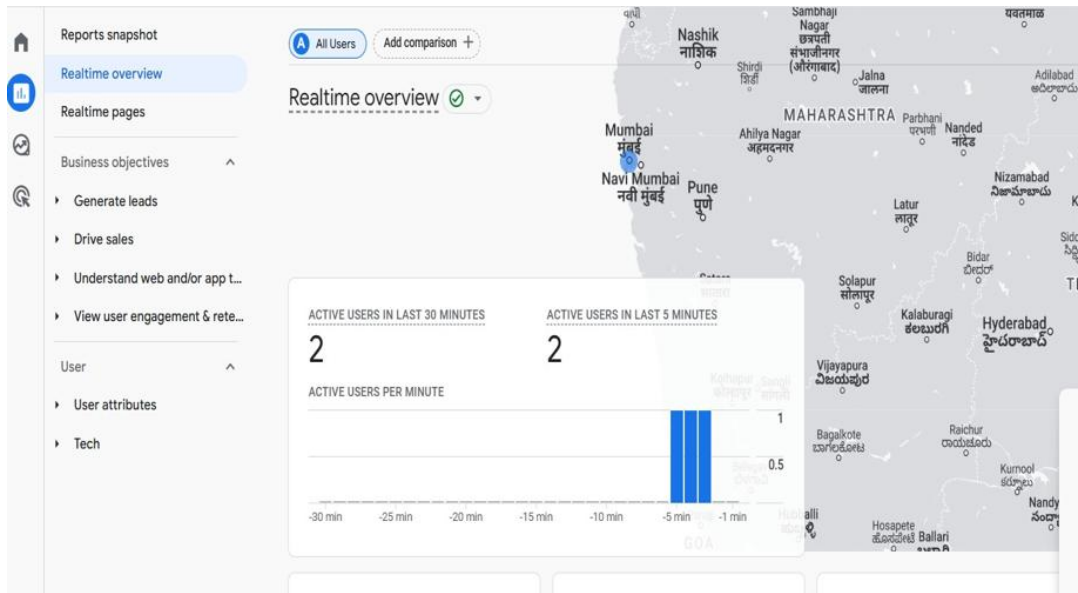


 **Recommended Crops:**

- Rice
- Tomato

3.2 Google Analytics





Chapter:4

4.1 Conclusion

The Crop Recommendation System effectively combines machine learning and real-time weather data to provide valuable insights to farmers, enabling them to make informed decisions on the most suitable crops to grow based on their local conditions. By integrating Flask with MongoDB for seamless data storage and retrieval, the system offers a robust and scalable solution. Additionally, the user-friendly interface, secure authentication, and interactive data visualizations make it accessible and practical for users of all technical backgrounds. This system not only aids in optimizing crop yields but also contributes to the digitization of agricultural practices, paving the way for smarter, data-driven farming in the future..