PROJECT REPORT

ON

AGRO AI

Submitted in the partial fulfilment of the requirement for the award of degree

of

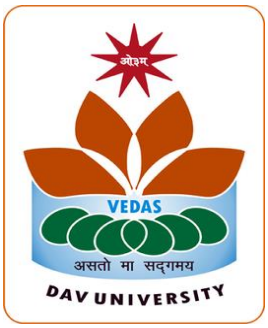
BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE

BATCH

(2022-2026)



Submitted to:

Ms. Bindu Bansal

Submitted by:

Shivani

12200256

B.Tech(CS&AI)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DAV UNIVERSITY

JALANDHAR-PATHANKOT NATIONAL HIGHWAY,NH 44,

SARMASTPUR PUNJAB,144012

TRAINING CERTIFICATE

CERTIFICATE

This is to certify that Ms. SHIVANI D/O NIRBHA SINGH (Student of DAV UNIVERSITY,JALANDHAR) has partially completed the 6-weeks Industrial Training during the period from June 2025 to July 2025 in our Organization/Industry as a Partial Fulfillment of Degree of BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE.She was trained in the field of DEGREE OF BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE.

DR. RAHUL HANS

(Head of Computer Science Dept)

DAV UNIVERSITY,JALANDHAR

DECLARATION

I,SHIVANI, hereby declare that the Industrial Training Report entitled ("AGRO AI") is an authentic record of my own work as requirements of 6-weeks industrial Training during the period from June 2025 to July 2025 for the award of degree of B.TECH (CS&AI), DAV UNIVERSITY ,under the guidance of (MR. HARITESH CHAUHAN) .

SHIVANI

ABSTRACT

The Crop Recommendation System is a machine learning-based project designed to assist farmers in selecting the most suitable crop based on real-time soil and climatic conditions. This system utilizes a dataset containing essential parameters such as nitrogen, phosphorus, potassium (N P K), temperature, humidity, pH, and rainfall. By analyzing these features using supervised learning algorithms, the model predicts the best crop for cultivation, improving decision-making and agricultural productivity. The system aims to promote sustainable farming, reduce resource wastage, and support modern, data-driven agriculture practices. It bridges the gap between traditional farming and intelligent automation in agriculture.

ACKNOWEDGEMENT

First, I express my gratitude to all those who helped us in various stages of the development of this project. would like to express my sincere gratitude indebtedness to DR. Rahul Hans (Coordinator) of DAV University for allowing me to undergo the summer training of 45 days at O7 Services. I am also thankful to all faculty members of Department of Computer Science and Engineering, for their true help, inspiration and for helping me for the preparation of the final report and presentation.

Last but not least, I pay my sincere thanks and gratitude to all the Staff Members for their support and for making our training valuable and fruitful.

COMPANY PROFILE

O7 Services is ISO 9001:2015 Certified company founded in 2015. They specialise in a variety of services including Web Development, Mobile Application Development, Custom Software Development, UI/UX Designing, Hosting services, Digital Marketing, Registration of Domain Names with various extensions, AMC & MMC Services, Bulk SMS and voice calls. O7 Services offers advanced IT solutions supporting the entire business cycle - from consulting to system development, deployment, quality assurance, and 24x7 support. With over 9 years of experience, the company aims to form long-lasting strategic partnerships with clients, offering affordable prices, timely delivery, and measurable business results. Their headquarter is located in Jalandhar with a branch office in Hoshiarpur.

Some of the products developed by O7 Services include Vehicle Tracking System, Invoice Software, School Management System, Hospital Management System, Parents - Teacher Communication App, Fee Management system, Task Management System, Online Food Ordering App, Security App, Admission system, Inventory Software, and Car Servicing App. Additionally, O7 Services provides training programs including 6 Weeks/6 Months Industrial Training, Project-Based Training, Corporate Training, and Job-Oriented Courses Training, covering major IT trends such as Full Stack Development (MEAN/MERN), Flutter, Kotlin Android, Swift UI (iOS), Firebase, Python, Angular, React Js, Vue Js, Node Js, ASP.NET, .NET Core, PHP, Laravel, CodeIgniter, Software Testing, Cloud Computing, Blockchain, DevOps, Data Science, Artificial Intelligence, Machine Learning, UI/UX Designing, Digital Marketing, WordPress, Linux, CCNP, CCNA Security, Network Security, Cyber Security, MCSE, MCITP, Java, Spring, Hibernate, C/C++, Photoshop, Adobe Illustrator, Figma, CorelDraw and many more.

+91- 8437365007, +91-181-5015007

E-Mail: enquiry@o7services.com , hr@o7services.com

www.o7services.com


A.C.C.
APEX
SINCE 1953
TRUMATH
ERYO
astro river
CAPITOL HOSPITAL
HR
The Tool Co.
KSE
Krishna Smart Education
FesoMe
Mekmetme
Sales & Buys
SUD
TRACKING
GURMAT JEEVAN
JAACH
Handl
Be Positive
CENTER
BILANDI
BRAND K
CENTURION
Nikhil PTE
Right Click Jobs
A JOB PORTAL FOR FRESHER'S
THE GRAND
livevalue
TRUE SCOOP
ROMASIDHU
Miracle Mushroom
Nature for Health
TRADITIONAL
KITCHEN
SRI
Shoring & Scaffolding
LOCAL FENCING

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

INTRODUCTION

Agriculture is one of the most important sectors in India, supporting the livelihood of millions of people. Agriculture is the practice of growing plants and raising animals for food, fiber, and other products we use every day. It is one of the oldest and most important activities for human survival. Agriculture provides the food we eat, the clothes we wear, and many materials for shelter and medicine.
Farmers use knowledge about soil, weather, and crops to grow plants efficiently and sustainably. Modern agriculture also uses technology and data to improve crop yields, protect the environment, and support communities.

1.1 Problem Definition:

Farmers need to select the most suitable crop to grow based on the quality of their soil and local weather conditions. However, without proper knowledge or tools, choosing the right crop can be difficult and may result in poor yield or loss.The problem is to create a system that can predict the best crop to plant by analyzing soil nutrients (like nitrogen, phosphorus, potassium), temperature, humidity, pH level, and rainfall. This prediction will help farmers make informed decisions to improve crop yield and reduce losses.

The core problem farmers face is identifying the best crop that matches the characteristics of their land ate. Growing a crop unsuitable for local conditions can lead to:

- Farmers need to select the best crop to grow based on their land's soil and weather conditions.
- Different crops require different amounts of nutrients (Nitrogen, Phosphorus, Potassium), temperature, humidity, pH, and rainfall to grow well.
- Without accurate guidance, farmers may plant unsuitable crops, leading to low yields and financial loss.
- Wasting resources like water, fertilizers, and labor on the wrong crop affects both the environment and farmer income.
- There is a need for a reliable system that can analyze soil and environmental data to recommend the most suitable crop.

1.2 Objective

The primary objective of this dataset and its analysis is to develop a robust system for optimizing agricultural decisions by understanding the relationship between soil characteristics, climatic conditions, and crop yields. Specifically, the goals include:

- Create a helpful system that tells farmers which crop to grow based on the nutrients in their soil and the weather around them. This system uses information like soil nutrients, temperature, humidity, rainfall, and soil pH to make smart suggestions.
- Learn how important soil nutrients like Nitrogen, Phosphorus, and Potassium affect crop growth. Different crops need different amounts of these nutrients, so understanding this helps the system recommend the right crop for the soil.
- Save water, fertilizers, and other resources by recommending crops suited to local soil and weather. This way, farmers don’t waste money or harm the environment by trying to grow crops that need extra chemicals or irrigation.
- Promote eco-friendly and sustainable farming practices. By planting crops that naturally fit their environment, farmers can protect the soil and nature, ensuring their land stays healthy for future farming.
- Support researchers and policymakers by providing data that helps improve crop planning and agro development.

CHAPTER 2

DATA PREPROCESSING

DATA PREPROCESSING :

Data preprocessing is the process of preparing raw data for analysis by cleaning and transforming it into a usable format.

DATA PREPROCESSING STEPS:

1.Data Collection and Loading:

- The dataset is collected from resources and contains features like:
- ➢ N, P, K – levels of nitrogen, phosphorus, and potassium in the soil.
- ➢ Temperature, Humidity, Rainfall, and pH – environmental factors.
- ➢ Label – the type of crop suitable for those conditions.
- The dataset is loaded into a data analysis tool (like Python using pandas) for further processing.

2. Data Inspection

- Understand the structure and content of the dataset:
- ➢ How many rows and columns are there?
- ➢ What are the data types (e.g., integer, float, string)?
- This helps in identifying issues early (like wrong data types or mixed values).

3. Handling Missing Data

- Check if any values are missing (null or NaN).
- Missing values can affect model training and lead to poor performance.
- Strategies to handle missing data:
- ➢ Remove rows or columns with missing values (if few).

4. Removing Duplicates

- Duplicate rows can mislead the model by giving too much importance to certain data points.
- Identifying and removing duplicates ensures data quality and balance.

5. Outlier Detection and Handling

- Outliers are unusually high or low values that don't follow the trend.
- For example, a rainfall value of 2000 mm when the usual range is 100–300 mm.
- Outliers can be removed or treated to prevent them from skewing the model.

6. Encoding Categorical Variables

- Machine learning models cannot work with text data directly.
- The "label" column (crop names) needs to be converted into numbers using encoding methods:

➢ Label Encoding assigns each crop a unique number.

➢ One-Hot Encoding creates binary columns for each crop (if needed).

7. Feature Selection and Separation

- Separate the dataset into:

➢ Features (x) – input variables like N, P, K, temperature, etc.

➢ Target (y) – the crop type (label) we want to predict.

- This helps prepare the data for machine learning models.

8. Train-Test Splitting

- Split the dataset into two parts:

➢ Training Set – used to train the model.

➢ Test Set – used to evaluate how well the model performs.

- Usually done in a ratio like 80:20 or 70:30.

9. Feature Scaling (Normalization or Standardization)

- Different features have different units and ranges (e.g., temperature in °C, rainfall in mm).
- Feature scaling ensures that all values are on a similar scale, which improves model performance.

➢ Standardization scales values to have zero mean and unit variance.

➢ Normalization scales values between 0 and 1.

CHAPTER-3

EXPLORATORY DATA ANALYSIS

EDA (Exploratory Data Analysis): EDA means understanding the data before building a machine learning model. It helps us find patterns, errors, and relationships between different columns in the data Why

EDA is Important for Crop Recommendation?

- To understand which soil and weather conditions suit which crop.
- To check if there are any missing or wrong values in the dataset.
- To see how each value (like nitrogen, temperature) affects crop growth.
- Helps in choosing the right features for the machine learning model.

Key Steps in EDA

1.Check Data Info: Look at all the columns: nitrogen, phosphorus, potassium, pH, temperature, humidity, rainfall, and crop name. Make sure all columns have correct data types (numbers, text).

Example: import pandas as pd

```
df = pd.read_csv ("Crop_recommendation.csv")
df.info()
```

2. Look for Missing Values: Make sure the dataset is clean (no blank cells or missing information). Example: df.isnull().sum()

3. Summary of Numbers: Use .describe() to find average, minimum, and maximum values of each column.

Example: df.describe()

4. See Crop Distribution: Find out how many times each crop appears. This helps check if all crops are equally present (balanced dataset).

Example: df ['label']. value_counts()

5. Use Graphs: Histograms show how values are spread (e.g., most soil has pH between 6–7).Boxplots show outliers (unusual values).Heatmaps show which values are related (e.g., does temperature affect humidity?).

Example: **Boxplot to Find Outliers in Rainfall**

```
sns.boxplot(x=df['rainfall'])
plt.title("Rainfall Outliers")
plt.show()
```

6. Compare Features: Use scatter plots or pair plots to see how two or more features change with different crops. Example: rice needs more rainfall, wheat needs moderate nitrogen.

Example:

```
sns.scatterplot(x='temperature', y='humidity', hue='label', data=df)
plt.title("Temperature vs Humidity by Crop")
plt.show()
```

CHAPTER-4

INTRODUCTION TO LANGUAGES,IDE AND TOOLS

4.1 INTRODUCTION TO PYTHON

Python is a very popular general-purpose interpreted, interactive, object-oriented, and high-level programming language. Python is a high-level, interpreted programming language known for its simplicity and readability. Python emphasizes code readability and clarity, making it an excellent choice for beginners and experienced programmers alike.

Applications that can be created using Python programming:

■ **Web Applications:** Python is widely used for developing web applications due to its simplicity and the availability of powerful frameworks like Django, Flask, and Pyramid. These frameworks provide tools and libraries for building scalable and feature-rich web applications, including content management systems (CMS), e commerce platforms, social media platforms, and more. Python's integration with web technologies such as HTML, CSS, and JavaScript makes it a popular choice for full stack web development.

■ **Data Science and Machine Learning Applications:** Python is extensively used in data science and machine learning applications due to its rich ecosystem of libraries and tools such as NumPy, Pandas, Matplotlib, and scikit-learn. Data scientists and machine learning engineers use Python to perform tasks such as data analysis, visualization, modeling, and deployment of machine learning models. Python frameworks like TensorFlow and PyTorch enable the development of deep learning models for complex tasks such as image recognition, natural language processing, and reinforcement learning.

4.2 INTRODUCTION TO IDE

An Integrated Development Environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development. It typically includes a source code editor, build automation tools, and a debugger, along with other features that streamline the development process. IDEs are designed to enhance productivity by providing an all-in-one environment for writing, testing, and debugging code.

- **PyCharm:** PyCharm is a powerful Integrated Development Environment (IDE) developed by JetBrains specifically for Python development. It offers a comprehensive set of features designed to enhance productivity and streamline the development process for Python programmers.
- **Visual Studio Code:** Visual Studio Code (VS Code) is a lightweight, open-source code editor developed by Microsoft. While it's not a full-fledged IDE like PyCharm, it offers many features that make it a powerful tool for Python development.
- **Jupyter:** Jupyter is an open-source project that provides tools and standards for interactive computing, notably through Jupyter Notebooks. While Jupyter supports over 40 programming languages, Python is one of its primary and most widely used languages.

4.3 TOOLS USED FOR DEVELOPMENT OF PROJECT

4.3.1 Python: Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.

4.3.2 Pandas: Pandas is a python package providing fast, flexible, and expressive data structures designed to make working with "relational" or "labelled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis/manipulation tool available in any language. It is already well on its way toward this goal. The two primary data structures of pandas, Series (1-dimensional) and DataFrame (2-dimensional) handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering.

Here are just a few of the things that pandas does well:

- Easy handling of missing data (represented as NaN) in floating point as well as non floating point data
- Size mutability: columns can be inserted and deleted from DataFrame and higher dimensional objects
- Robust IO tools for loading data from flat files (CSV and delimited), Excel files, databases, and saving / loading data from the ultrafast HDF5 format.

4.3.3 NumPy: NumPy is a Python library for numerical computing that provides support for large, multidimensional arrays and matrices, along with a collection of high-level mathematical functions to operate on these arrays. It is a fundamental library in the scientific computing ecosystem in Python, and is widely used in fields such as data science, machine learning, physics, and engineering.

Some of the key features of NumPy include:

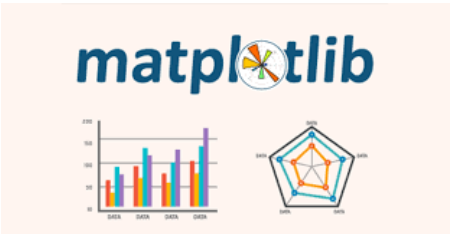
- A rich set of functions for performing mathematical operations on arrays, such as element-wise operations, matrix multiplication, and linear algebra operations .
- Efficient handling of large data sets and complex calculations using optimized C code and memory management techniques .

4.3.4 Streamlit: Streamlit is an open-source Python library that allows you to create interactive web applications for data visualization, machine learning, and data science. It provides a simple and intuitive way to build user interfaces without requiring extensive knowledge of web development technologies like HTML, CSS, or JavaScript.

Here are some key features and benefits of Streamlit:

- **Interactive Data Visualization:** Streamlit integrates well with popular data visualization libraries like Matplotlib, Plotly, and Altair, allowing you to create interactive and visually appealing plots and charts.
- **Machine Learning Model Deployment:** Streamlit provides a convenient way to deploy and showcase machine learning models by creating interactive user interfaces around them.
- **File Upload and Download:** Streamlit allows users to upload and download files directly from the application, making it easy to work with data files.

4.3.5 Matplotlib: Matplotlib is one of the most widely used in Python for making charts and graphs. It's like the foundation for many other visualization tools including Seaborn. With Matplotlib can create a wide range of static (non-interactive) plots such as line graphs, bar charts, pie charts, histograms and scatter plots. It’s very flexible and powerful, but it can be a little more complex compared to Seaborn or Plotly, especially for beginners. Scatter plots can show relationships between soil nutrients and crop yield, histograms can display the distribution of rainfall over different seasons, and box plots can highlight variations in soil pH across different farmland regions.



CHAPTER 5

REQUIREMENT ANALYSIS

5.1 FEASIBILITY STUDY:

- **Executive Summary:**

➢ The goal is to build an AI-based system that suggests the best crop to grow based on soil and climate data.

➢ It uses machine learning models trained on the Crop and Soil Dataset.

➢ The system can benefit farmers by increasing productivity and reducing losses.

- **Market Feasibility:**

➢ Large market: millions of farmers worldwide.

➢ Growing interest in digital and smart farming tools.

➢ Potential users: farmers, agriculture extension workers, government bodies.

➢ Can improve food security and reduce crop failures.

➢ Increasing smartphone and internet penetration supports adoption.

- **Technical Feasibility:**

➢ Inputs needed: Nitrogen (N), Phosphorus (P), Potassium (K), Temperature, Humidity, pH, Rainfall.

➢ Dataset includes these features linked to recommended crops.

➢ Common machine learning algorithms to use: Decision Trees, Random Forest, KNN, SVM, Neural Networks.

- **Financial Feasibility:**

➢ Development cost low due to open-source tools and existing datasets.

➢ Maintenance requires periodic updates and model retraining.

➢ Potential funding from agricultural grants and government schemes.

➢ Revenue models: subscription, consulting, ads, or integration with agri-input suppliers.

➢ Savings for farmers by reducing crop failure and unnecessary fertilizer use.

- **Operational Feasibility:**

➢ Simple operation: farmers enter soil and weather info; system gives crop suggestion.

➢ Integration possible with existing agricultural advisory services.

➢ Can work offline or with intermittent internet.

➢ Training required for farmers to use the system effectively.

➢ Support team needed for troubleshooting and updates.

- **Legal and Regulatory Feasibility:**

➢ Data used is mostly environmental, so minimal privacy issues.

➢ Must comply with data protection laws if personal user data is collected.

➢ Need to ensure transparency and fairness of AI recommendations.

➢ May require certifications if integrated into government programs.

5.2 SDLC MODEL

The Software Development Life Cycle (SDLC) is a structured process used by developers and project teams to design, develop, test, and deploy software systems. It ensures the software meets requirements, is delivered on time, and is maintainable.

- ## Phases of SDLC

1. Requirement Gathering and Analysis

This is the first and most important phase. It involves gathering information about what the system should do from the stakeholders (farmers, developers, users).

For This Project:

- Understand user needs: "I want to enter soil details and get crop suggestions."
- Identify features: data input, model prediction, clean output.
- Final output: a document listing functional & non-functional requirements.

2. System Design

In this phase, the system’s architecture is designed. It includes DFDs, database design, algorithm design, and interface layout.

For This Project:

- Create DFD (Level 0/1)
- Design how data flows from user → model → output
- Choose ML algorithm (e.g., Random Forest, Decision Tree)
- Plan the UI/UX design for inputs and outputs

3. Implementation (Coding)

This is the actual coding phase. Developers write code for front-end, back-end, and machine learning model training.

For This Project:

- Use Python and libraries like pandas, sklearn, numpy, matplotlib
- Train ML model using dataset (features: N, P, K, temperature, etc.)
- Optional: Build UI using Streamlit, Flask, or HTML/CSS

4. Testing

Testing ensures the software works as expected, with no bugs. Types of testing include unit testing, integration testing, system testing, etc.

For This Project:

- White Box Testing: Test internal code logic (e.g., ML model)
- Black Box Testing: Give input data and check if correct crop is predicted

5. Deployment

Once the software is tested, it is deployed to a server or user environment so it can be used by end-users.

For This Project:

- Deploy model on a web app (Streamlit)
- Host on local server, GitHub Pages, or cloud platforms (optional)

6. Maintenance

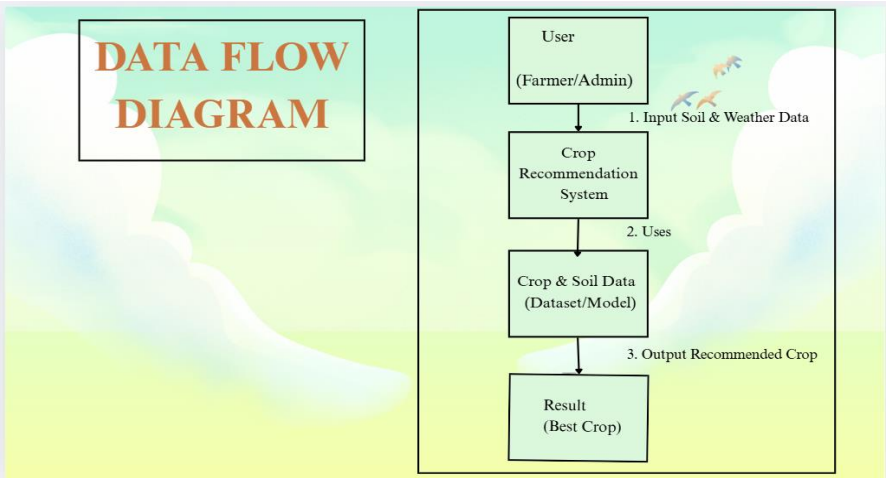
Software may need updates, bug fixes, or new features over time. Maintenance includes monitoring, feedback handling, and future improvements.

For This Project:

- Add new crops or weather conditions
- Retrain model with more updated soil data
- Fix UI issues or bugs based on user feedback

5.3 DFD

Data Flow Diagram (DFD) represents the flow of data within information systems. Data Flow Diagram (DFD) for the Crop and Soil Dataset visualizes how soil and environmental data is collected, processed, and used to recommend suitable crops. The process starts with external entities such as farmers or agricultural sensors providing input data including soil nutrients (nitrogen, phosphorus, potassium), temperature, humidity, pH, and rainfall. This raw data is fed into a data preprocessing process, where it is cleaned, normalized, and prepared for analysis. The processed data is stored in a data repository, acting as the central dataset for model training and evaluation. Analytical processes, such as machine learning algorithms, access this dataset to train predictive models that learn patterns between soil conditions and crop suitability. Once the model generates crop recommendations, the results flow back to the farmer or agricultural advisor through an output interface. This DFD outlines the logical movement of data from collection to recommendation, ensuring transparency and systematic handling of data within the crop recommendation system.



CHAPTER 6

HARDWARE AND SOFTWARE REQUIREMENTS

6.1 HARDWARE REQUIREMENTS: -

- Processor: Intel i3,i5 or higher
- RAM: Minimum 8 GB
- Storage: At least 256 GB
- Operating System: Windows, macOS, or Linux

6.2 SOFTWARE REQUIREMENTS: -

1. Operating System:

- Windows Operating System
- Linux (Ubuntu preferred)

2. Web Browser:

- Google Chrome
- Browsers are used to access platforms like Google Colab, GitHub, or view the deployed web application.

3. Text Editor / IDE

- Visual Studio Code

4. Programming Language:

- Python 3.7 or higher
- Most suitable for data science and machine learning tasks due to its large number of libraries and ease of use.

5. Python Libraries:

- **NumPy:** Used for handling numerical data. Helps perform fast mathematical operation.
- **Pandas:** Helps read and process data from the dataset. Used to clean and organize the data in table format (DataFrame).
- **Matplotlib:** Used to create basic graphs and charts. Helps visualize the dataset (like nutrient levels or crop frequency).
- **Scikit-learn:** Main library for machine learning. Used to train models like Decision Tree or Random Forest. Also helps evaluate model accuracy.

CHAPTER 7

IMPLEMENTATION, TESTING AND MAINTENANCE

7.1 Implementation:

❖ **Data Collection**

- Downloaded the Crop and Soil Dataset from Kaggle.
- The dataset contains important soil attributes such as pH, nitrogen (N), phosphorus (P), potassium (K), temperature, humidity, and moisture content.

❖ **Data Processing**

- Loaded the dataset using pandas in Python.
- Cleaned the data by checking for missing values, removing duplicates, and correcting any inconsistencies.
- Performed basic exploratory data analysis (EDA) to understand the distribution and relationship between soil properties and suitable crops.

❖ **Data Analysis**

- Analyzed the cleaned data and created clear paragraphs summarizing key insights, such as:

➢ Which soil conditions are ideal for specific crops.

➢ How factors like pH or nutrient levels affect crop suitability.

❖ **Streamlit App Development**

- Used Streamlit, an open-source Python framework, to build an interactive web app.
- The app allows users to:

➢ Input soil parameters (e.g., pH, N, P, K).

➢ Get crop recommendations based on the input.

➢ Visualize soil data using charts and plots for better interpretation.

7.2 Software Testing:

- ➢ Software testing is like checking your homework before you submit it — you check if everything is correct.
- ➢ It's a process to find mistakes (bugs or defects) in the software or model.
- ➢ If there are mistakes in your data analysis or prediction model, it may give wrong crop suggestions — so testing helps avoid that.
- ➢ The main goal: Make sure your project meets the requirements, works properly, and does what users need.

- **Why do Bugs Happen?**

- ➢ Bugs can come from mistakes made by the developer while writing the code or designing the system.
- ➢ Sometimes bugs appear because of wrong calculations, wrong input data, or poor integration between modules.
- ➢ For example, if you wrongly map soil pH values to crops, the prediction may be completely wrong.

- **What Happens if There is a Bug?**

- ➢ If there's a bug, your program gives wrong output. For example, it may suggest a crop that won't grow in that soil.
- ➢ So, testing finds the bugs early and you fix them to prevent failure in the real world.
- ➢ This saves time, money, and effort.

- **Levels of Testing:-**

❖ **Unit Testing**

- Smallest part of testing.
- Test each unit or piece separately.
- Example: One unit could be your function that calculates soil moisture.
- It uses White-box testing, which means you look inside the code and check how it works.

❖ **Integration Testing**

- After unit testing, you combine units.
- Example: Your soil data module and crop suggestion module.
- The goal is to check if data flows correctly between parts.

❖ **System Testing**

- Now test the full system as one big piece.
- This checks if your project follows all the requirements: e.g., “Should recommend suitable crops for given soil data”.
- Here, you do performance testing, security testing, and functional testing.
- Make sure the system works on different devices or with different data inputs.

❖ **Acceptance Testing**

- Also called User Acceptance Testing (UAT).
- Final testing by real users.
- Example: A farmer uses the system and checks if it helps them.
- If they approve, your project is ready for production (real use).

7.2.1 Test Planning: It is the process of creating a detailed document that outlines the strategy, objectives, resources, schedule, scope, and approach for testing activities in a software project. It is a critical phase of the **Software Testing Life Cycle (STLC)** that ensures testing is methodical, efficient, and effective.

Here are the essential software testing steps every software engineer should perform before showing their work to someone else.

1. Basic functionality testing

Begin by making sure that every button on every screen works. You also need to ensure that you can enter simple text into each field without crashing the software. You don't have to try out all the different combinations of clicks and characters, or edge conditions, because that's what your testers do and they're really good at that.

2. Static code analysis

There are tools that can perform analysis on source code or bytecode without executing it. These static code analysis tools can look for many weaknesses in the source code, such as security vulnerabilities and potential concurrency issues. Use static code analysis tools to enforce coding standards, and configure those tools to run automatically as part of the build.

3. Unit testing

Developers will write unit tests to make sure that the unit (be it a method, class, or component) is working as expected and test across a range of valid and invalid inputs. In a continuous integration environment, unit tests should run every time you commit a change to the source code repository, and you should run them on your development machine as well. Some teams have coverage goals for their unit tests and will fail a build if the unit tests aren't extensive enough.

4. Single-user performance testing

Some teams have load and performance testing baked into their continuous integration process and run load tests as soon as code is checked in. This is particularly true for back-end code. But developers should also be looking at single-user performance on the front end and making sure the software is responsive when only they are using the system.

7.2.2 Test Design:

The test design component addresses the need to define the number of tests to be performed, the ways that testing will be approached (paths, functions), and the test conditions that need to be exercised. Test design standards need to be defined and followed.

An effective test program, incorporating the automation of software testing, involves a mini development lifecycle of its own, complete with strategy and goal planning, test requirement definition, analysis, design, and coding. Similar to software application development, test requirements must be specified before test design is constructed. Test requirements need to be clearly defined and documented, so that all project personnel will understand the basis of the test effort.

7.5 Maintenance:

❖ **Bug Fixes:** Address and resolve any issues or bugs identified during testing or reported by users. Prioritize and fix bugs promptly to maintain system reliability and user satisfaction.

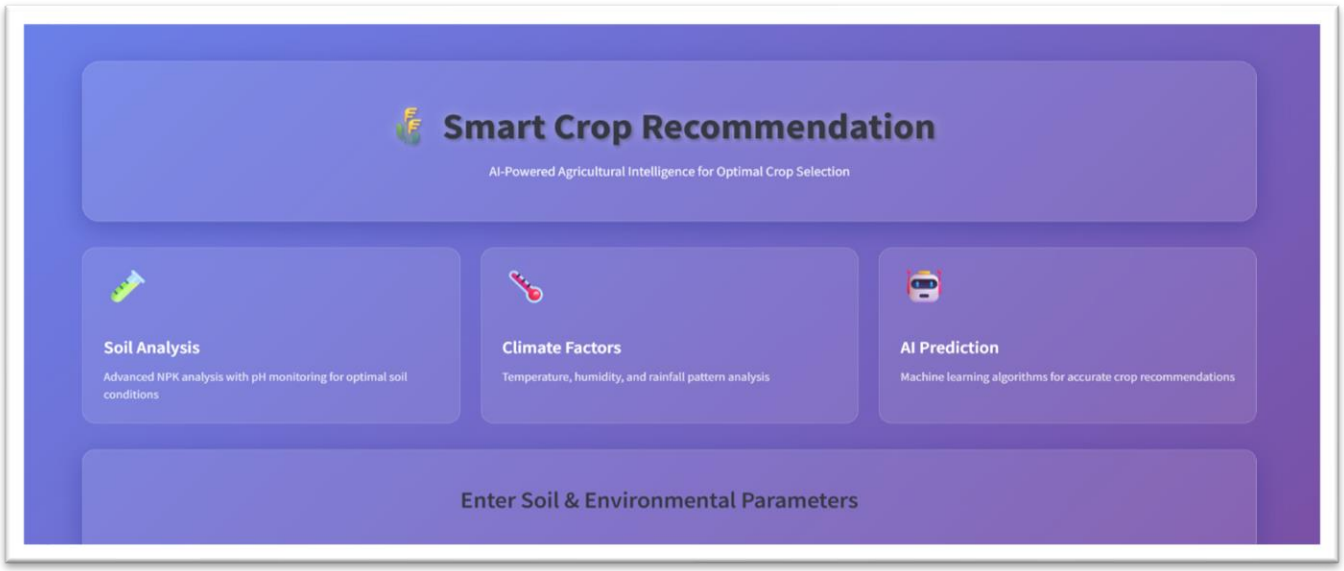
❖ **Updates and Enhancements:** Continuously improve the system by implementing new features, enhancements, and optimizations based on user feedback and emerging requirements. Prioritize updates based on user needs and market trends.

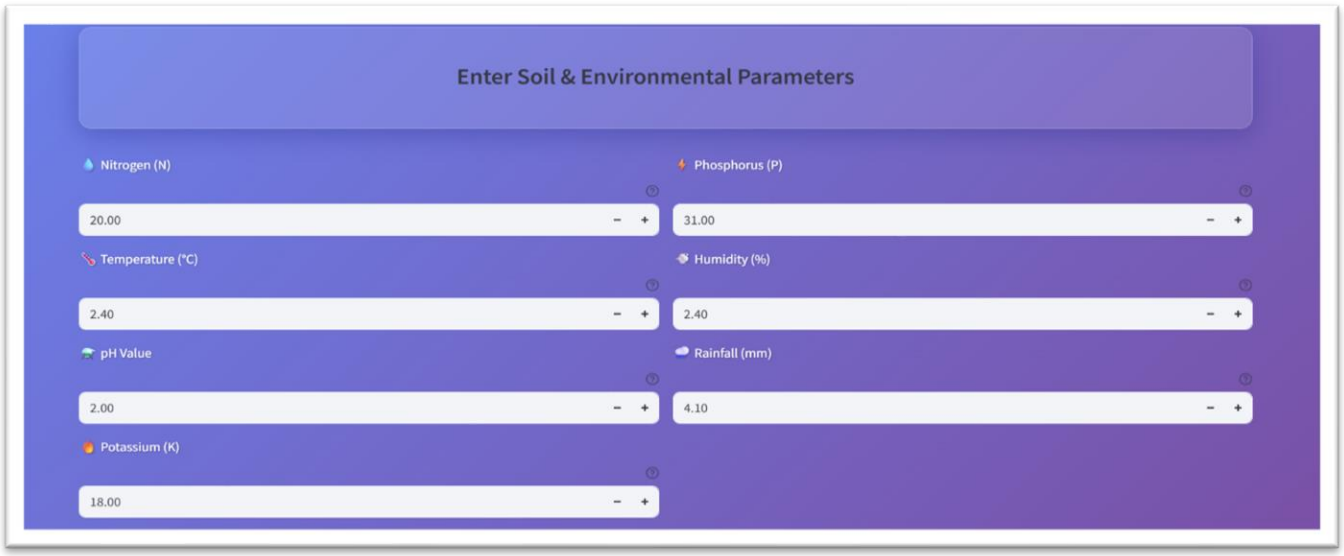
❖ **Performance Optimization:** Monitor and optimize the system's performance over time to ensure efficient resource utilization, minimize latency, and improve overall responsiveness.

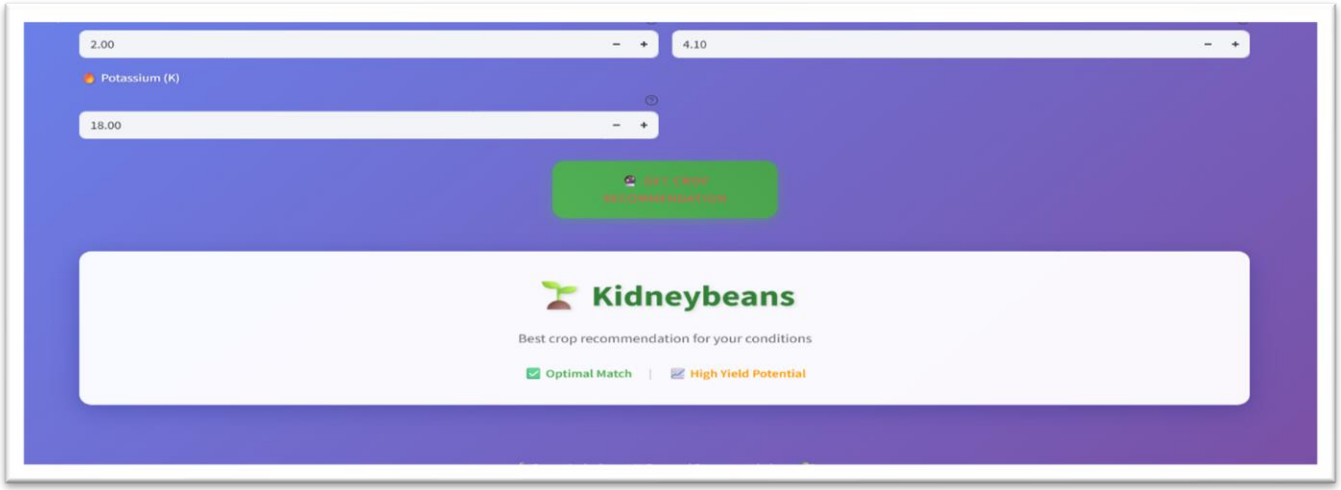
❖ **Security Updates:** Stay vigilant against security threats and vulnerabilities by applying security applying security updates, patches, and fixes to the system.

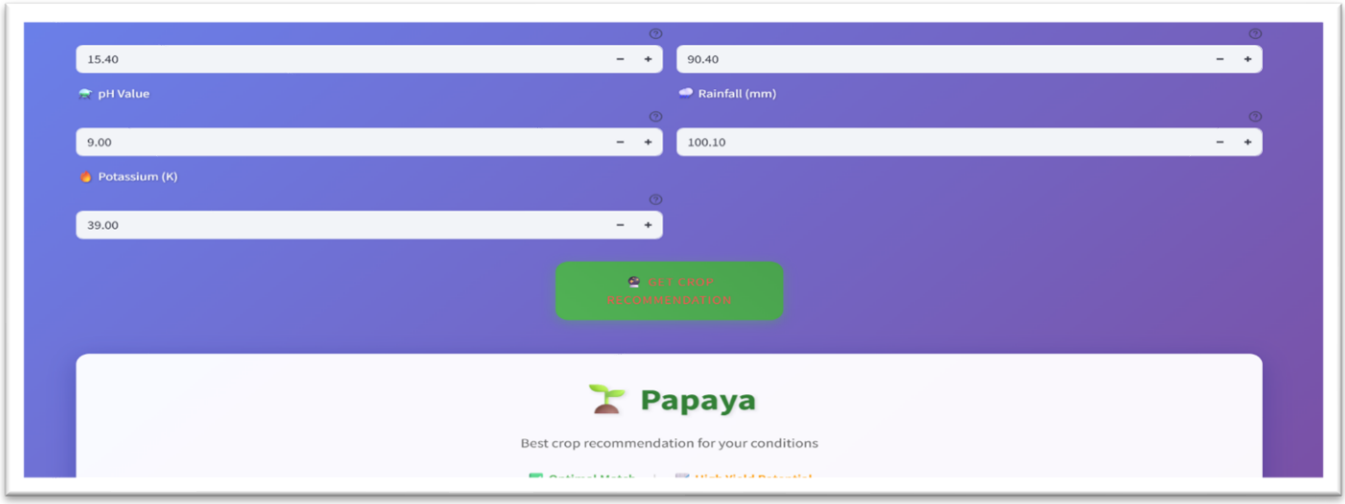
CHAPTER-8
SCREENSHOTS

SCREENSHOTS :






2.00
4.10
Potassium (K)
18.00
Kidneybeans
Best crop recommendation for your conditions
Optimal Match
High Yield Potential


15.40
90.40
pH Value
Rainfall (mm)
9.00
100.10
Potassium (K)
39.00
GET CROP RECOMMENDATION
Papaya
Best crop recommendation for your conditions

CHAPTER- 9
SUMMARY,CONCLUSION AND FUTURE SCOPE

Summary:

The Crop and Soil Dataset is a structured agricultural dataset designed to support machine learning applications in smart farming. It contains detailed information about essential soil nutrients—Nitrogen (N), Phosphorus (P), and Potassium (K)—as well as environmental factors such as temperature, humidity, pH level, and rainfall. These features are critical for crop health and productivity. The dataset's target variable is the recommended crop type, based on the input conditions. By training AI/ML models on this dataset, developers can build systems that predict the most suitable crop for a specific set of soil and climatic conditions. Project helps farmers choose the most suitable crop based on real-time soil and climate conditions. This supports data-driven decision-making in agriculture, helping farmers improve crop selection, optimize yield, and reduce the risk of crop failure.

Conclusion:

The Crop Recommendation System using the Crop and Soil Dataset successfully demonstrates how AI and machine learning can revolutionize agriculture. By analyzing soil nutrients and environmental factors, the model accurately suggests the most suitable crop for cultivation. This project supports farmers in making informed decisions, improving productivity, conserving resources, and promoting sustainable farming practices. Ultimately, it bridges the gap between traditional agriculture and modern data-driven solutions.

Future Scope:

- **AI-Driven Fertilizer and Pest Management:** Extend ML models to suggest optimal fertilizers and predict pest/disease outbreaks. Helps reduce chemical usage and improves sustainable farming practices.
- **AI Chatbots and Voice Assistants:** Enable voice-based interaction for low-literacy farmers using NLP in local languages. Makes technology accessible to a wider user base.
- **Multi-Season Crop Rotation Planning:** Suggest crop sequences based on nutrient cycles and soil health improvement. Prevents soil degradation and improves farm sustainability.

CHAPTER- 10

BIBLIOGRAPHY

- To Find Python and IDE Use:
  https://www.geeksforgeeks.org/python/python-programming-language-tutorial/
- To read libraries(Numpy,Pandas):
  https://www.almabetter.com/bytes/tutorials/python/popular-python-libraries
- To read tools (Pytorch):
  https://www.geeksforgeeks.org/python/difference-between-pytorch-and-tensorflow/
- To read documentary of Streamlit:
  https://www.geeksforgeeks.org/python/a-beginners-guide-to-streamlit/
- To read documentary of Matplotlib:
  https://www.geeksforgeeks.org/python/python-introduction-matplotlib/