WHEAT-SHIELD

KASHIF HUSSAIN

MUHAMMAD ZUBAIR AWAN

Spring 2024

Department of Computer Science
COMSATS University Islamabad
Attock Campus-Pakistan

COMSATS University Islamabad Attock Campus
Attock Pakistan

WHEAT-SHIELD

By

Kashif Hussain	**CIIT/FA20-BCS-019/ATK**
Muhammad Zubair Awan	**CIIT/FA20-BCS-041/ATK**

Supervisor

Mr. Waseem Khan

Bachelors of Science in Computer Science (2020-2024)

COMSATS University Islamabad Attock Campus
Attock Pakistan

WHEAT-SHIELD

A project presented to

COMSATS University, Islamabad

In partial fulfillment of the
requirement for the degree of

Bachelors of Science in Computer Science
(2020-2024)

By

Kashif Hussain	**CIIT/FA20-BCS-019/ATK**
Muhammad Zubair Awan	**CIIT/FA20-BCS-041/ATK**

DECLARATION

We hereby declare that this software, neither whole nor as a part has been copied out from any source. It is further declared that we have developed this software and accompanied report entirely on the basis of our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other, we will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

Kashif Hussain
FA20-BCS-019

Muhammad Zubair
FA20-BCS-041

CERTIFICATE OF APPROVAL

It is to certify that the final year project of BS (CS) "**Wheat Shield**" was developed by **Kashif Hussain (CIIT/FA20-BCS-019)** and **Muhammad Zubair (CIIT/FA20-BCS-041)** under the supervision of "Mr. Waseem Khan" and that in (their/his/her) opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Computer Sciences.

1. External Examiner ----------

2. Supervisor ----------

Mr. Waseem Khan

Lecturer

Department of Computer Science
COMSATS University Islamabad

Attock Campus

3. Head of the Department ----------

Dr. Zahoor ur Rehman
Associate Professor HOD(CS)

Department of Computer Science
COMSATS University Islamabad

Attock Campus

Executive Summary

In agriculture, monitoring crop health, weather conditions, and managing agricultural commerce are crucial for maximizing productivity and ensuring crop safety. Manual monitoring and management can be time-consuming, inaccurate, and lack real-time efficiency, leading to potential losses and suboptimal decision-making. To address these issues, the Wheat Shield Project has been developed. It is a comprehensive, stand-alone application that streamlines disease detection, weather monitoring, agricultural commerce, and location-based services, thereby providing farmers with an integrated solution to manage their agricultural activities effectively. The Wheat Shield Project consists of several modules: disease detection, where users can upload images or provide input data for analysis and receive results indicating the detected disease and relevant information; weather monitoring, which displays real-time weather updates based on the user's location, provides weather forecasts for upcoming days, and sends alerts for weather conditions affecting crop health; agricultural commerce, allowing users to browse and create listings for agricultural products; and location-based services, identifying nearby medical shops and agricultural supply stores, offering directions and contact information, and enabling location-based searches for specific products or services. Additionally, the Wheat Shield Project includes a comprehensive disease report feature that detects and identifies diseases, generates detailed reports with impact analysis and recommended actions, and provides historical data for future prevention strategies. This embedded system operates on a web browser, integrates with various api, and uses advance deep learning model capabilities, ensuring that farmers have access to real-time, actionable information.

Acknowledgement

All praise is to Almighty Allah who bestowed upon us a minute portion of His boundless knowledge by virtue of which we were able to accomplish this challenging task.

We are greatly indebted to our project supervisor “Mr. Muhammad Waseem”. Without their personal supervision, advice and valuable guidance, completion of this project would have been doubtful. We are grateful to them for their encouragement and continual help during this work.

And we are also thankful to our parents and family who have been a constant source of encouragement for us and brought us with the values of honesty & hard work.

Kashif Hussain

Muhammad Zubair

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1 Introduction

1.1 Vision Statement

For farmers seeking efficient disease detection, weather monitoring, and agricultural commerce solutions, Wheat Shield is a web-based application that provides accurate disease identification, real-time weather updates, and a seamless e-commerce platform. Unlike traditional methods, Wheat Shield empowers farmers with timely insights and resources, enhancing crop management practices and fostering sustainable agricultural development.

1.2 Related System Analysis/Literature Review

Our Wheat Shield project has some related systems that have field in common and here below is complete literature review of those apps.

Table 1: System analysis

Application Name	**Weakness**	**Proposed Project Solution**
Plantix	No e-commerce functionality	Wheat Shield offers a comprehensive e-commerce platform.
AgriApp	Lack of integration with weather data.	integrates real-time weather monitoring to enhance decision-making for farmers.
Climate FieldView	Proprietary platform with subscription	Wheat Shield provides a free usage of the platform for detecting and monitoring disease.

1.3 Project Deliverables

Wheat Shield project provides these deliverables:

1.3.1 Disease Detection Module:

Algorithm for detecting disease and User interface for uploading images Output displaying data.

1.3.2 Weather Monitoring Module:

Integration with weather API and User interface for displaying weather.

1.3.3 E-commerce Platform:

User interface for browsing agricultural products and Functionality for users to create accounts, list products, like products and make purchases.

1.3.4 Location-based Services:

Integration with mapping APIs to provide location-based services and displaying nearby medical shops and agricultural supply stores.

1.3.5 User Authentication and Management:

User registration and Role-based access control for administrators and users.

1.3.6 Documentation

Comprehensive documentation outlining the project's objectives, methodologies, and technical details. User manuals and guides for using each module of the web application.

1.3.7 Testing and Quality Assurance:

Test cases and scripts for functional, integration, and usability testing and Quality assurance reports ensuring the reliability and stability of the application.

1.3.8 Deployment Package:

Packaged application ready for deployment on web servers and deployment scripts for automating the deployment process.

1.4 System Limitations/Constraints

There are some limitations and constraints for the proposed project "Wheat Shield":

LI-1: Accuracy of Disease Detection: Despite efforts to be precise, there might still be mistakes in identifying diseases due to differences in picture quality or surroundings.

LI-2: Weather Data Availability: The reliability of weather updates depends on getting good data from outside sources. Problems with internet connection or wrong data from these sources might make the weather module less reliable.

LI-3: Geographic Coverage: The location-based services, such as identifying nearby medical shops and agricultural supply stores, may be limited to certain geographic regions where relevant data is available. Users in areas with limited data coverage may experience reduced functionality in this aspect.

LI-4: E-commerce Security: While efforts will be made to ensure the security of transactions on the e-commerce platform, there may still be risks associated with online payments, such as data breaches or fraudulent activities, which are inherent to conducting business transactions online.

LI-5: Device and Browser Compatibility: Compatibility with different devices (desktops, tablets, smartphones) and web browsers (Chrome, Firefox, Safari, etc.) may vary, leading to potential user experience inconsistencies or limitations in functionality across different platforms.

LI-6: Resource Constraints: Limited resources, such as server capacity, bandwidth, and development time, may impose constraints on the scalability and performance of the web application, particularly during periods of high usage or rapid expansion of user base Tools and Technologies

Here are all the hardware/software tools and technologies with version number which will be used in implementation of the project.

Table 2: Tools and Technologies for Proposed Project

Tools And Technologies	**Tools**	**Version**	**Rationale**
	Visual Studio Code	2024	Code editor
	Technology	**Version**	**Rationale**
	React JS	latest	Front-end Development
	JavaScript	ECMAScript 6	Programming language for client-side scripting
	Redux	latest	State management library for ReactJS applications
	MongoDB	latest	NoSQL database for storing application data
	Bootstrap	latest	Back-end Development

	Python	latest	Back-end Development

1.5 Relevance to Course Modules

1.5.1 Machine Learning and Artificial Intelligence (AI):

The disease detection module of "Wheat Shield" utilizes machine learning algorithms to identify septoria and stripe rust in wheat plants. Students who have studied machine learning and AI will find practical application of concepts such as classification algorithms and image recognition.

1.5.2 Web Development:

"Wheat Shield" is a web application developed using ReactJS for the front-end and Node.js for the back-end. Students who have studied web development will find relevance in concepts such as HTML, CSS, JavaScript, server-side scripting, and client-server communication.

1.5.3 Database Management Systems (DBMS):

MongoDB is used as the database management system for storing application data in "Wheat Shield." Students who have studied DBMS will understand database design principles, querying techniques, and database integration within web applications.

1.5.4 Software Engineering:

Concepts of software engineering, such as requirements gathering, system design, testing, and project management, are applied throughout the development lifecycle of "Wheat Shield." Students will recognize the importance of following a structured approach to software development to ensure the project's success.

1.5.5 Data Communication and Computer Networks:

Understanding of computer networks is essential for integrating location-based services, weather APIs, and payment gateways within "Wheat Shield." Students will appreciate the significance of

network protocols, API communication, and security considerations in web application development.

1.5.6 Data Structures and Algorithms:

The disease detection module of "Wheat Shield" involves processing and analyzing large datasets using algorithms for image recognition and pattern matching. Students will recognize the importance of efficient algorithms and data structures in solving real-world problems.

Chapter 2: Problem Definition

2 Problem Definition

2.1 Problem Statement

Farmers face challenges in timely disease and effect crop management, impacting crop yield and agricultural sustainability. Traditional methods of disease identification are often time-consuming and rely on manual observation, leading to delved response and potential crop damage. Additionally, farmers may lack access to real-time weather updates and agricultural resources, hindering informed decision-making. "Wheat Shield" aims to address these challenges by providing a comprehensive web application for disease detection, weather monitoring, and agricultural commerce. By leveraging machine learning algorithms, farmers can accurately identify diseases such as septoria and stripe rust in wheat plants, enabling prompt intervention and minimizing crop losses. Integration with weather APIs ensures farmers have access to up-to-date weather forecasts, empowering them to make informed decisions regarding crop management practices. Furthermore, the e-commerce platform facilitates seamless transactions for buying and selling agricultural products, enhancing accessibility and efficiency in the agricultural marketplace. "Wheat Shield" ultimately aims to optimize crop management processes, improve crop health, and contribute to sustainable agricultural practices.

2.2 Problem Solution

"Wheat Shield" addresses the pressing challenges faced by farmers in timely disease detection and effective crop management. Traditional methods reliant on manual observation often lead to delayed responses and potential crop damage, impacting crop yield and sustainability. By leveraging advanced machine learning algorithms, "Wheat Shield" offers accurate identification of wheat diseases like septoria and stripe rust, enabling prompt intervention to minimize losses. Integration with real-time weather APIs provides farmers with crucial insights, empowering informed decision-making in crop management practices. Additionally, the e-commerce platform facilitates seamless transactions for buying and selling agricultural products, enhancing accessibility and efficiency in the agricultural marketplace. Through educational resources and community collaboration features, "Wheat Shield" fosters knowledge exchange and peer support

among farmers, further optimizing crop management processes. With robust data security measures and scalability, "Wheat Shield" is poised to revolutionize agricultural practices, contributing to improved crop health and sustainable agricultural development.

2.3 Objectives of the Proposed System

The objectives of the proposed "Wheat Shield" system are:

BO-1: Improve Disease Detection Accuracy: Achieve a minimum accuracy rate of 90% in identifying Septoria and stripe rust in wheat plants to facilitate timely intervention and minimize crop losses.

BO-2: Enhance Crop Management Efficiency: Provide farmers with actionable insights and recommendations based on disease detection results and weather forecasts to optimize crop management strategies and maximize yield.

BO-3: Increase Access to Agricultural Resources: Facilitate convenient access to essential agricultural resources, such as medical shops and agricultural supply stores, by providing location-based services within the application.

BO-4: Foster Agricultural Commerce: Create a vibrant marketplace for buying and selling agricultural products, promoting accessibility and efficiency in agricultural transactions.

BO-5: Ensure User Satisfaction and Engagement: Maintain high levels of user satisfaction and engagement by delivering a user-friendly and reliable application experience, supported by comprehensive documentation and responsive customer support.

BO-6: Ensure Data Security and Compliance: Implement robust security measures to safeguard user data and transactions, ensuring compliance with relevant regulations and standards.

BO-7: Foster Sustainable Agricultural Practices: Encourage adoption of sustainable agricultural practices by providing farmers with tools and insights to optimize resource utilization and minimize environmental impact.

BO-8: Support Scalability and Growth: Design the system architecture to be scalable and adaptable, capable of accommodating future growth in user base and functionality enhancements.

BO-9: Promote Continuous Improvement: Commit to ongoing updates and enhancements based on user feedback and technological advancements, continuously improving the functionality and usability of the "Wheat Shield" system.

2.4 Scope

The scope of the "Wheat Shield" project encompasses a holistic solution aimed at revolutionizing various aspects of crop management for farmers. The main functionalities include disease detection, weather monitoring, agricultural commerce, and location-based services. The disease detection module utilizes advanced machine learning algorithms to accurately identify common wheat diseases such as Septoria and stripe rust, aiding farmers in timely intervention and minimizing crop losses. The weather monitoring module integrates real-time weather data from external APIs, providing farmers with up-to-date forecasts and alerts to optimize crop management decisions in response to changing weather conditions. The agricultural commerce platform serves as a user-friendly marketplace for buying and selling agricultural products, enhancing accessibility and efficiency in agricultural transactions. Location-based services enable farmers to conveniently access nearby resources such as medical shops and agricultural supply stores, facilitating seamless integration into their daily operations. User authentication and management functionalities ensure secure access to the application and personalized user experiences. Comprehensive documentation, compliance with regulations, scalability, reliability, and a commitment to continuous improvement are integral parts of the project scope. By encompassing these functionalities, "Wheat Shield" aims to empower farmers with advanced tools and insights, foster sustainable agricultural practices, and contribute to the efficiency and resilience of agricultural operations.

2.5 Modules

2.5.1 Module 1: Disease Detection

FE-1: Upload images or provide input data for disease detection.

FE-2: Display results indicating the detected disease and relevant information.

2.5.2 Module 2: Weather Monitoring

FE-1: Display real-time weather updates based on the user's location.

FE-2: Provide weather forecasts for upcoming days.

FE-3: Send alerts for weather conditions affecting crop health.

2.5.3 Module 3: Agricultural Commerce

FE-1: Browse agricultural products available for sale.

FE-2: Create listings for selling agricultural products.

2.5.4 Module 4: Location-based Services

FE-1: Identify nearby medical shops and agricultural supply stores.

FE-2: Provide directions and contact information for identified locations.

FE-3: Enable location-based search for specific products or services.

2.5.5 Module 5: User Management

FE-1: Register new users and manage user accounts.

FE-2: Assign roles and permissions to users based on their responsibilities.

FE-3: Monitor user activity and access logs for security purposes.

2.5.6 Module 6: Content Management

FE-1: Manage content related to disease information, weather updates, and agricultural products.

FE-2: Create and update educational resources for farmers.

FE-3: Review and moderate user-generated content, such as product listings and reviews.

2.5.7 Module 7: Analytics and Reporting

FE-1: Generate reports on disease detection statistics, weather patterns, and sales data.

FE-2: Analyze trends and patterns to identify areas for improvement and optimization.

FE-3: Provide insights and recommendations for enhancing the platform's effectiveness.

2.5.8 Module 8: System Administration

FE-1: Configure system settings and preferences.

FE-2: Manage integrations with external APIs and services.

FE-3: Monitor system performance and troubleshoot technical issues.

Chapter 3: Requirement Analysis

3 Requirement Analysis

3.1 User classes and characteristics

Here are the user and their characteristics mentioned below in form of table.

Table 3: user classes

User	Description
Farmer	Farmers are the primary users of the "Wheat Shield" system, responsible for managing their wheat crops and utilizing the application for disease detection, weather monitoring, and agricultural commerce. They may vary in experience level, from novice to experienced farmers.
Agricultural Expert	Agricultural experts may include agronomists, researchers, or consultants who provide expertise and guidance to farmers. They may use the system to analyze disease patterns, provide recommendations, and offer educational resources to farmers.
E-commerce user	E-commerce users are individuals or businesses interested in buying or selling agricultural products through the platform. They may include farmers looking to sell their crops, agricultural suppliers, or consumers seeking agricultural products.
System Administrator	System administrators are responsible for managing and maintaining the "Wheat Shield" platform. They oversee user accounts, system configurations, integrations with external services, and ensure the system's security and performance.

3.2 Requirement Identifying Technique

3.2.1 Use case (use case diagram)

A visual representation illustrating interactions between system actors and functionalities, depicting various scenarios of user-system interaction

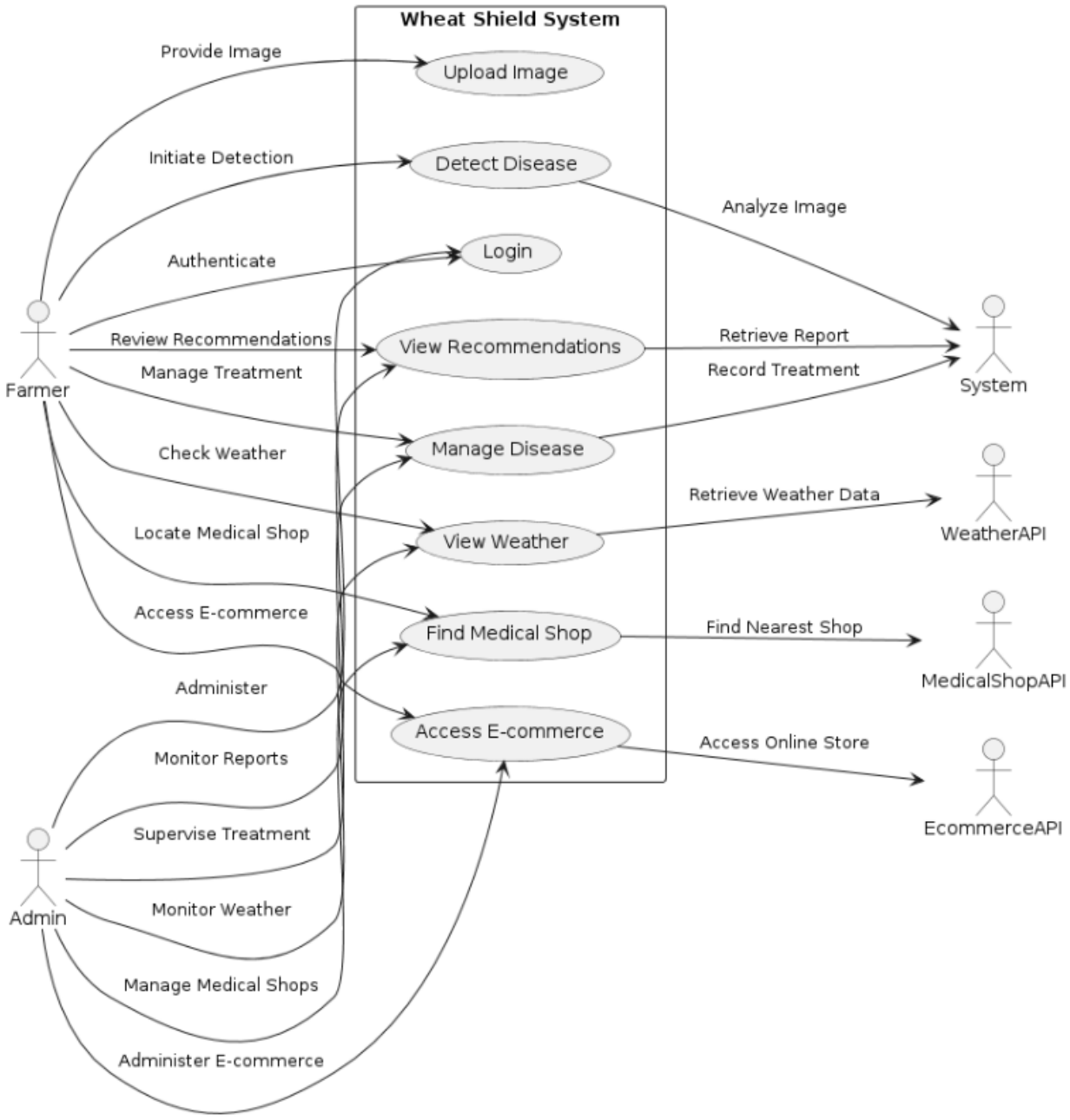

Figure 1: Use Case

3.2.2 Detail use case:

An in-depth description of specific interactions between system actors and functionalities, providing a comprehensive understanding of the steps involved in achieving a particular system goal or task.

Table 4: Use Case

Use Case ID	UC-1
Use Case Name	Detect and Manage Wheat Diseases
Actors	Primary Actor: Farmer Secondary Actor: System
Description	A Farmer accesses the Wheat Shield system, uploads an image of a wheat plant showing signs of disease, and requests disease detection. The System analyzes the image for disease detection and provides the farmer with a detailed report including the detected disease, its effects on the wheat plant and recommended treatments.
Trigger	The Farmer indicates the need to detect and manage diseases affecting their wheat crops.
Preconditions:	PRE-1: The Farmer is logged into the Wheat Shield system. PRE-2: The Farmer has uploaded a clear image of a wheat plant showing symptoms of disease.
Postconditions	POST-1: The disease detection report is generated and stored in the Wheat Shield system. POST-2: The Farmer receives a detailed report including the detected disease, its effects, and recommended treatments.
Normal Flow	1. Farmer uploads an image of a wheat plant showing symptoms of disease. 2. System analyzes the uploaded image for disease detection. 3. System generates a disease detection report. 4. System displays the disease detection to the Farmer.
Business Rules	BR-1: Disease detection accuracy is based on the quality and clarity of the uploaded image.

	BR-2: Recommended treatments are based on expert agricultural knowledge and research.
Assumptions	1. Farmers have basic knowledge of using the Wheat Shield system. 2. The Wheat Shield system has access to a comprehensive database of wheat diseases and treatments.

3.2.3 Event- response table.

A structured representation detailing system responses corresponding to specific events or triggers, facilitating clear comprehension of system behavior under varying conditions.

Table 5: Event response

Event	System State	Response
Image uploaded	Idle	Start image processing to detect diseases. Analyze the uploaded image for signs of septoria and stripe rust. Generate a disease report based on the analysis results.
Disease detected	Image Processing	Display the detected disease to the user. Provide information on the effects of the disease on wheat plants. Recommend suitable pesticides or medicines for treatment.
Weather information updated	Idle	Retrieve the latest weather information for the current location. Update the weather display with the new information.
Product browsing	Idle	Allow users to browse through the list of available products. Display product details including name, description, and price. Provide options to view more details or purchase the product.
Product purchase	Idle	Process the purchase transaction. Update the inventory to reflect the purchased product.
Crop selling	Idle	Allow farmers to list their crops for sale. Provide a platform for buyers to browse and purchase listed crops. Facilitate the transaction process between sellers and buyers.

3.2.4 Storyboarding:

We implemented storyboarding to pre-visualize our project by creating a sequence of drawings that outlined each scene and action.

Figure 2 Access web app

3.2.4.1 Login:

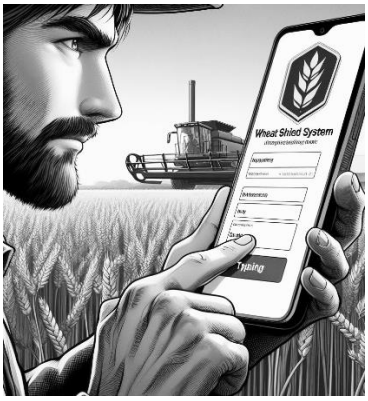

Figure 3 login

3.2.4.2 Uploading picture

Figure 4 Uploading picture

3.2.4.3 Recommendations Generated:

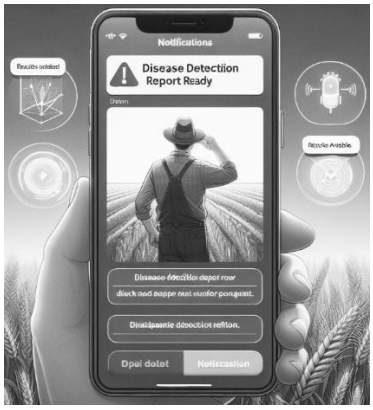

Figure 5 Recommendations

3.2.4.4 Treatment:

Figure 6 Treatment

3.3 Functional Requirements

Descriptions of specific functionalities and features that the system must possess to satisfy user needs and achieve project objectives.

3.3.1 Functional Requirement X

A specific capability or behavior that the system must exhibit to fulfill user needs or meet project objectives, as outlined in the project specifications

Table 6: FR-1

Identifier	FR-1
Title	Image Upload
Requirement	The user shall be able to upload images of wheat plants for disease detection.
Source	User requirement

Rationale	Users need a convenient way to input data for disease detection, and image upload is a common and intuitive method.
Priority	High

Table 7: FR-2

Identifier	FR-2
Title	Input Data Submission
Requirement	The user shall be able to submit input data (e.g., numerical values or textual descriptions) for disease detection if image upload is not feasible.
Source	User requirement
Rationale	Provides flexibility for users who may not have access to image upload capabilities or prefer alternative input methods
Priority	Medium

Table 8: FR-3

Identifier	FR-3
Title	Disease Identification
Requirement	The system shall analyze the uploaded images or input data to identify common wheat diseases, including Septoria and stripe rust.
Source	System requirement
Rationale	Core functionality of the disease detection module, enabling users to diagnose plant diseases accurately.
Priority	High

Table 9: FR-4

Identifier	FR-4

Title	Error Handling: Invalid Image Format
Requirement	If the uploaded image format is invalid (e.g., not supported or corrupted), the system shall display an error message prompting the user to upload a valid image format (e.g., JPEG, PNG).
Source	System requirement
Rationale	Ensures user awareness and prompt action in case of invalid inputs, improving user experience.
Priority	Medium

Table 10: FR-5

Identifier	FR-5
Title	Error Handling: Unavailable Disease Database
Requirement	If the system's disease database is unavailable or outdated, the system shall notify the user and prompt them to try again later or contact support for assistance.
Source	System requirement
Rationale	Provides transparency and guidance to users in case of technical issues with the disease detection functionality.
Priority	Low

3.4 Non-Functional Requirements

3.4.1 Reliability

REL-1: The "Wheat Shield" system shall have a mean time between failures (MTBF) of at least 500 hours, where a failure is defined as any event that disrupts the core functionalities of disease detection, weather monitoring, or agricultural commerce.

REL-2: In the event of a software failure, the system shall automatically generate an error report and notify system administrators via email within 5 minutes of the occurrence, detailing the nature of the failure and any potential impact on system performance.

REL-3: To protect against system failures, the "Wheat Shield" system shall implement regular data backups scheduled to occur daily during off-peak usage hours, with backups stored securely in an off-site location.

REL-4: Error detection mechanisms shall be embedded within critical system components to monitor for abnormal behavior or unexpected events, with alerts triggered for immediate investigation and resolution by system administrators.

3.4.2 Usability

USE-1: The "Wheat Shield" user interface shall adhere to industry-standard design principles and guidelines, ensuring ease of learning and intuitive navigation for users with varying levels of technical expertise.

USE-2: The system shall provide clear and concise error messages in case of user input errors, guiding users toward corrective actions and facilitating error recovery without undue frustration or confusion.

USE-3: Accessibility features such as keyboard shortcuts and screen reader compatibility shall be implemented to ensure inclusivity and accommodate users with disabilities.

3.4.3 Performance

PER-1: 95% of webpage requests made to the "Wheat Shield" system shall complete within 3 seconds from the time the user initiates the request, as measured over a typical internet connection with a bandwidth of 10 Mbps or higher.

PER-2: The system shall support concurrent user sessions for up to 1000 users without experiencing degradation in response time or system performance, as verified through load testing simulations conducted quarterly.

3.4.4 Security

SEC-1: The "Wheat Shield" system shall implement encryption mechanisms using industry-standard protocols to secure data transmission between the client and server components, mitigating the risk of unauthorized interception or tampering.

SEC-2: User authentication and authorization processes shall be enforced through robust security measures, including password hashing, multi-factor authentication, and session management, to prevent unauthorized access to sensitive data and functionalities.

SEC-3: Access to system resources and administrative privileges shall be restricted based on role-based access control (RBAC) principles, with granular permissions assigned to users based on their designated roles and responsibilities within the system.

3.5 External Interface Requirements

3.5.1 User Interfaces Requirements

Specifications outlining the design elements, layout, and interactive features necessary for intuitive and efficient user interaction with the system

- **GUI Standards and Style Guides:**
  - Implemented industry-standard GUI design principles such as Material Design for web applications or Human Interface Guidelines for mobile apps.

- **Fonts, Icons, and Images:**
  - Used a readable and accessible font such as Roboto and utilized standard icon sets like Material Icons and optimized images for web to ensure fast loading times.

- **Color Schemes:**
  - Chose color palette with high contrast for readability and accessibility and used color combinations that comply with WCAG guidelines to ensure accessibility for all users, including those with color vision deficiencies.

- **Screen Layout and Resolution Constraints:**
  - Designed layouts using a responsive grid system such as Bootstrap or CSS Grid to ensure compatibility with various screen sizes.

- **Standard Buttons:**
  - Used clear and consistent labels for buttons and ensured that buttons are easily distinguishable and clickable.

- **Message Display Conventions:**
    - Use a consistent style for displaying messages, such as using a colored box for error messages and a neutral color for informational messages.

3.5.2 Software interfaces

Specifications detailing the communication protocols, data formats, and interactions required for seamless integration between the system and external software components or services.

SI-1: Disease Detection Algorithm

- **Connection:** The "Wheat Shield" system shall integrate with the Disease Detection Algorithm version 2.0.
- **Description:** The system shall utilize the Disease Detection Algorithm to analyze images of wheat plants and identify common diseases such as septoria and stripe rust.
- **Interface:** The system shall transmit images or input data to the Disease Detection Algorithm through an API endpoint.
- **Dependency:** The proper functioning of the disease detection module relies on the accuracy and performance of the Disease Detection Algorithm.

SI-2: Weather API

- **Connection:** The "Wheat Shield" system shall utilize the Weather API version 3.5 for real-time weather updates.
- **Description:** The system shall retrieve weather data, including temperature, humidity, and precipitation forecasts, based on the user's location.
- **Interface:** The system shall make HTTP requests to the Weather API endpoint to fetch weather information.
- **Dependency:** Timely and accurate weather updates are essential for providing relevant recommendations and alerts to farmers.

SI-3: Crop Data Database

- **Connection:** The "Wheat Shield" system shall connect to the Crop Data Database version 1.2.
- **Description:** The system shall retrieve crop-related data, including planting schedules, growth stages, and yield predictions, from the Crop Data Database.

- **Interface:** The system shall establish a database connection using SQL queries to retrieve and store crop data.
- **Dependency:** Access to comprehensive and up-to-date crop data is crucial for providing tailored recommendations and insights to farmers.

SI-4: Leaflet Maps API

- **Connection:** The "Wheat Shield" system shall integrate with the Leaflet Maps API
- **Description:** The system shall utilize the Leaflet Maps API for location-based services, including mapping medical shops, agricultural supply stores, and weather stations.
- **Interface:** The system shall make API calls to the Leaflet Maps API to fetch location data and display maps within the application.
- **Dependency:** Accurate mapping and location services enhance the user experience and facilitate navigation for users.

SI-5: React JS Framework

- **Connection:** The "Wheat Shield" web application shall be built using the React JS framework version 17.
- **Description:** React JS shall be used for frontend development to create dynamic and interactive user interfaces.
- **Dependency:** The proper functioning of the web application relies on the stability and performance of the React JS framework for rendering UI components and managing state.

3.5.3 Communications interfaces

CI-1: Web Browser Interface

- **Requirement:** The "Wheat Shield" system shall provide a web browser interface for users to access the application.
- **Description:** Users shall interact with the system through a web browser interface, accessing features such as disease detection, weather monitoring, agricultural resources, and e-commerce functionalities.
- **Supported Browsers**: The web browser interface shall be compatible with modern web browsers, including Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.

- **Cross-Platform Compatibility**: The interface shall be responsive and compatible with various operating systems, including Windows, macOS, iOS, and Android.
- **User Authentication**: The system shall implement secure user authentication mechanisms to verify user identities and protect sensitive data.

CI-2: Network Protocols

- **Requirement:** The "Wheat Shield" system shall utilize standard network protocols for data transmission between client and server components.
- **Description:** The system shall communicate over the internet using protocols such as Hypertext Transfer Protocol (HTTP) and Hypertext Transfer Protocol Secure (HTTPS) to ensure secure and reliable data exchange.
- **Data Encryption:** Communication between client and server components shall be encrypted using Transport Layer Security (TLS) to prevent unauthorized interception or tampering.

Chapter 4: Design and Architecture

4 Design and Architecture

4.1 Architectural Design

Develop a modular program structure and explain the relationships between the modules to achieve the complete functionality of the system. This is a high-level overview of how the system's modules collaborate with each other in order to achieve the desired functionality.

Don't go into too much detail about the individual subsystems. The main purpose is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together.

Provide a diagram showing the major subsystems and their connections.

- In initial design stage create Box and Line Diagram for simpler representation of the systems
- After finalizing architecture style/pattern diagram (MVC, Client-Server, Layered, Multi-tiered) create a detailed mapping modules/components to each part of the architecture

To view example of box and line diagram and architecture styles, see Appendix B.

4.2 Design Models

Design Models for Object Oriented Development Approach

Architectural blueprints and representations utilized during the development phase, emphasizing object-oriented principles such as encapsulation, inheritance, and polymorphism to design robust and scalable software systems

4.2.1 Activity Diagram

Figure 7: Activity Diagram

4.2.2 Sequence Diagram

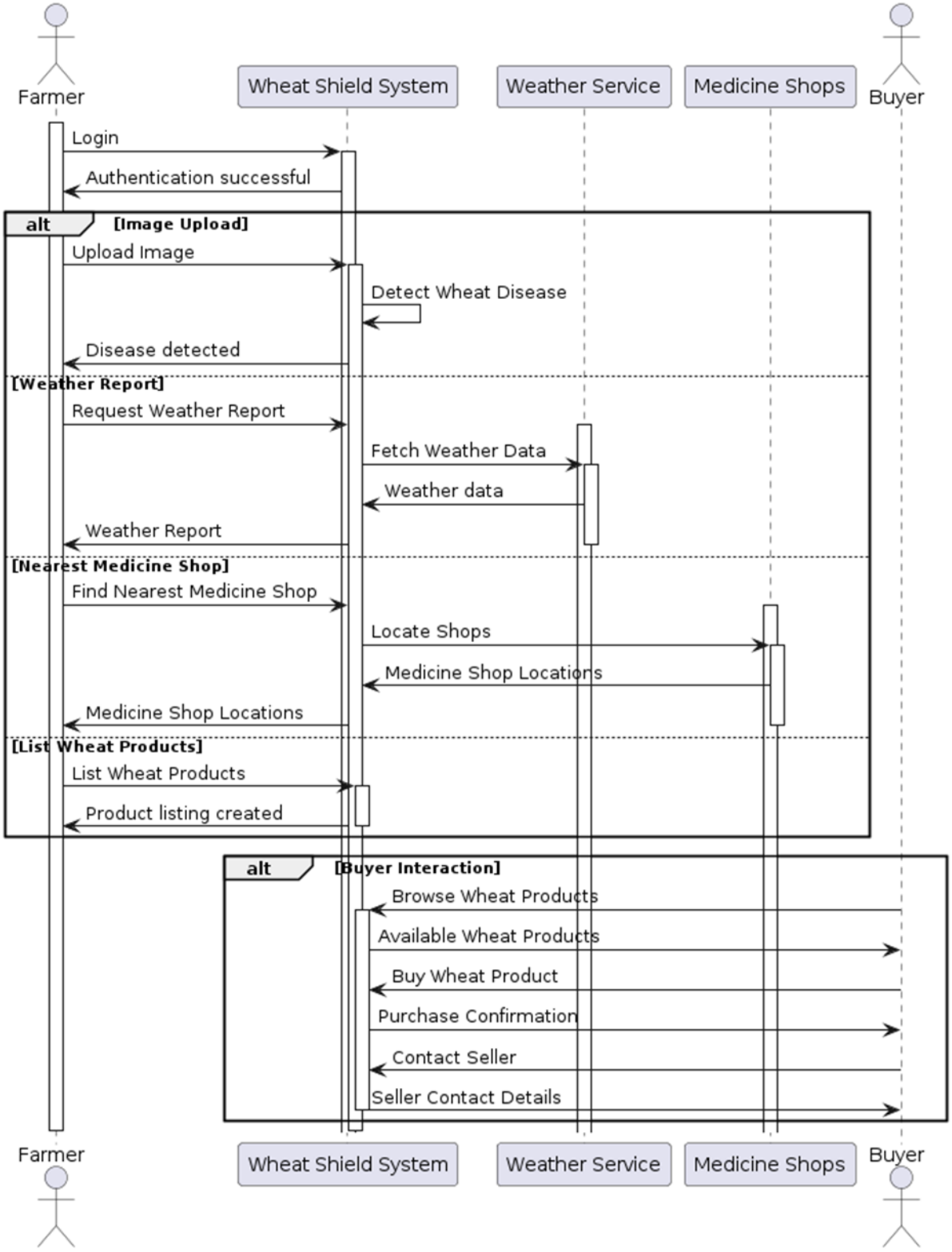

Figure 8: Sequence Diagram

4.3 Data Design

4.3.1 User Data:

This includes information about users such as their profiles, preferences, and login credentials. User data may be stored in a database table with fields such as user ID, username, password (hashed for security), email, role, and any additional attributes relevant to the system.

4.3.2 Disease Data:

Disease data consists of information related to different diseases, their symptoms, causes, and diagnostic criteria. This data may be organized into a database table with fields such as disease ID, disease name, symptoms, causes, treatments, and any related references or links.

4.3.3 Product Data:

Product data pertains to the items available for purchase on the e-commerce platform. This data may include details such as product ID, name, description, price, availability, and images. Product data can be stored in a database table with appropriate fields to represent each attribute.

4.3.4 Weather Data:

Weather data comprises information about current weather conditions, forecasts, and historical weather patterns. This data may be obtained from external APIs or services and stored in the system's database or retrieved in real-time when needed.

4.4 Human Interface Design

Describe the functionality of the system from the user’s perspective. Explain how the user will be able to use your system to complete all the expected features and the feedback information that will be displayed for the user.

4.4.1 Screen Objects and Actions

4.4.1.1 Login Screen:

- *Screen Objects*: Username field, Password field, Login button, Forgot password link.
- *Actions*: Users enter their credentials (username and password) and click on the Login button to authenticate. The Forgot password link allows users to reset their password if they forget it.

4.4.1.2 Home Screen:

Screen Objects: Navigation menu, Search bar, service, Monitor, Test, location and shop widget.
Actions: Users can navigate to different sections of the application using the navigation menu. They can search for products using the search bar. The featured products section displays highlighted items. The weather widget provides real-time weather updates.

4.4.1.3 Shop Screen:

Screen Objects: Product image, Product name, Description, Price, Add to Cart button, Quantity selector.
Actions: Users can view detailed information about a product, including its image, description, and price. They can adjust the quantity of the product and add it to their shopping cart using the Add to Cart button.

4.4.1.4 Shopping Cart Screen:

Screen Objects: List of items in the cart, Subtotal, Proceed to Checkout button, Remove item button.
Actions: Users can view the items added to their shopping cart, adjust quantities, and remove items if needed. The Subtotal indicates the total cost of items in the cart. Users can proceed to checkout to complete their purchase.

4.4.1.5 Location Screen:

Screen Objects: input location.

Actions: Users enter the current location so in response to which the application will respond and will give the result based on the input.

4.4.1.6 Disease Detection Screen:

Screen Objects: Upload image button, Result display area, Diagnosis report button.

Actions: Users can upload an image of a plant affected by a disease. The system processes the image and displays the detection result along with recommendations. Users can download or print the diagnosis report.

Chapter 5: Implementation

5 Implementation

5.1 Algorithm

Here is the Convolutional Neural Network (CNN) algorithm presented in a step-by-step manner:

5.1.1 Convolutional Neural Network (CNN) Algorithm

5.1.1.1 Input:

An image represented as a three-dimensional array of pixel values (height, width, color channels).

5.1.1.2 Output:

A probability distribution over the possible classes for classification tasks or relevant predictions for other tasks (e.g., bounding boxes for object detection).

5.1.1.3 Steps:

1. **Input Layer:**
   Accept the input image of dimensions (H times W times C), where (H) is height, (W) is width, and (C) is the number of color channels.

2. **Convolutional Layer:**
   - For each filter (k) (where (k) ranges from 1 to the number of filters (K)):
   - Slide the filter over the input image.
   - Perform element-wise multiplication between the filter and the input image region covered by the filter.
   - Sum the results to produce a single value in the feature map.
   - Move the filter according to the stride parameter and repeat until the entire image is covered.
   - Apply padding if necessary, to maintain the spatial dimensions.
   - Output a set of feature maps.

3. **Activation Function (ReLU):**
    - Apply the Rectified Linear Unit (ReLU) function to each element of the feature maps: [f(x) = max (0, x)]
    - Replace all negative values with zero to introduce non-linearity.

4. **Pooling Layer (Max Pooling):**
    - For each region of the feature map defined by the pooling size (e.g., (2 times 2)):
    - Select the maximum value from the region.
    - Down sample the feature map by taking the maximum value for each region.
    - Output the pooled feature maps with reduced spatial dimensions.

5. **Repeat Steps 2 to 4:**
    - Repeat the convolutional, ReLU, and pooling layers multiple times to extract higher-level features. The number of repetitions depends on the depth of the network.

6. **Flattening:**
    - Convert the final set of pooled feature maps into a one-dimensional vector.

7. **Fully Connected Layer:**
    - Pass the flattened vector through one or more fully connected layers:
    - Each fully connected layer computes a weighted sum of its inputs and passes the result through an activation function (typically ReLU).
8. **Output Layer:**
    - For classification tasks, use a softmax activation function in the final layer to produce a probability distribution over the classes:

$$\mathrm{P(y = i \mid x)} = \frac{ez_i}{\sum_{j=1}^{C} ez_i}$$

9. **Loss Function and Optimization:**
    - Define a loss function (e.g., cross-entropy loss for classification tasks).
    - Use an optimization algorithm (e.g., stochastic gradient descent) to minimize the loss function by updating the weights of the network through backpropagation.

5.2 External APIs/SDKs

Application Programming Interfaces (APIs) or Software Development Kits (SDKs) provided by third-party vendors or platforms, allowing developers to integrate external functionalities, services, or data sources into their own software applications.

Table 11: Details of APIs used in the project

Name of API and version	**Description of API**	**Purpose of usage**	**List down the API endpoint/function/class in which it is used**
FastAPI	Web framework for building APIs with Python	Handling model prediction requests	/ping, /predict
Express.js	Web framework for Node.js	Handling database CRUD operations and user authentication	/search, /like-product, /add-product, /get-products, /signup, /login, etc.
TensorFlow	Machine learning library for building and training models	Making predictions using pre-trained model	Model prediction function/class
Weather API	Third-party API for fetching weather information	Fetching weather data for weather-based features	Weather API endpoints for retrieving weather information

5.3 User Interface

5.3.1 Home Screen

Figure 9: Home Screen

5.3.2 Login Screen

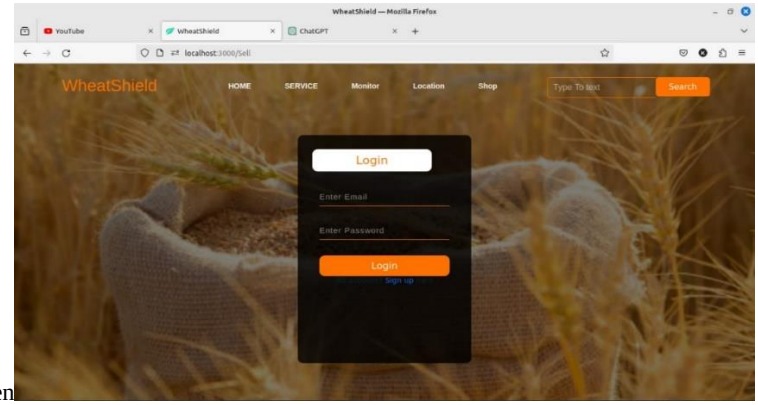

Figure 10: Login Screen

5.3.3 Image Upload

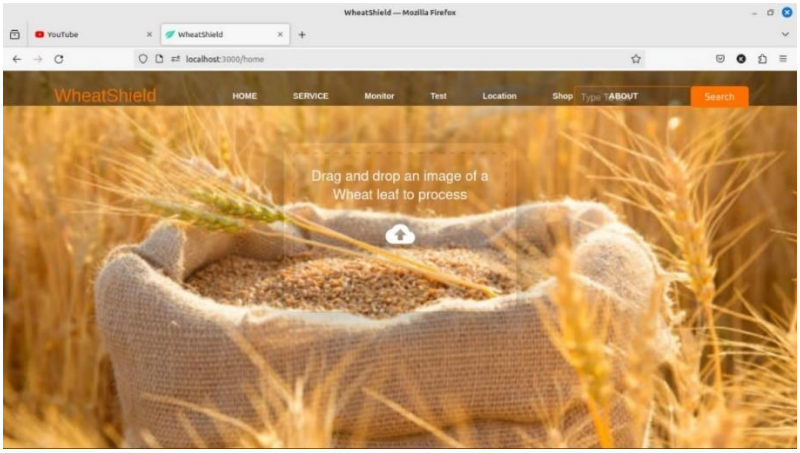

Figure 11: Uploading Screen

5.3.4 Recommendations

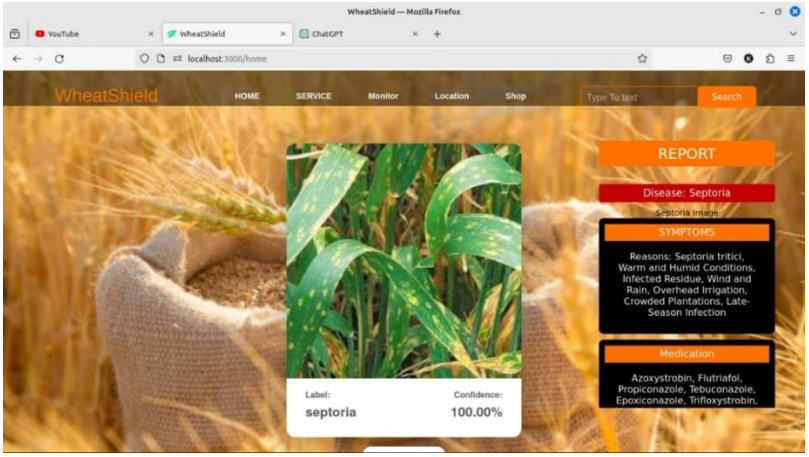

Figure 12: Recommendation Screen

5.3.5 Weather

Figure 13: Weather Screen

5.3.6 Shop

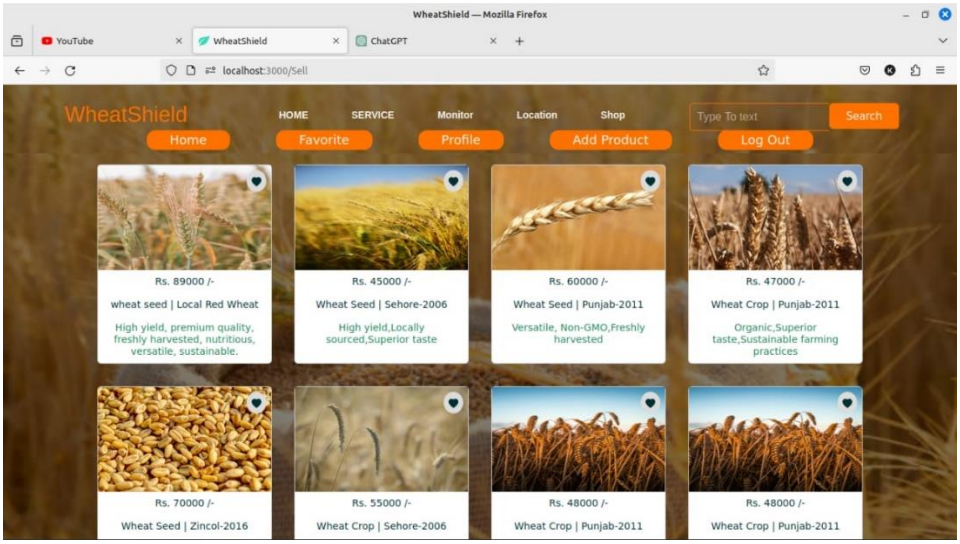

Figure 14: E-commerce shop

Chapter 6: Testing and Evaluation

6 Testing and Evaluation

6.1 Unit Testing

- **Unit Testing 1:** Login as Patient with valid and invalid credentials
- **Testing Objective:** To ensure the login form is working correctly with valid and invalid credentials/inputs.

Table 12: Unit testing

No.	Test case/Test script	Attribute and value	Expected result	Result
1	Check the email field of login to validate that it takes proper email	Email: muhammadzubair16.k@gmail.com	Validates email address and moves cursor to next textbox	Pass
2	Check the email field of login to validate that it displays error message.	Email: kashifhussain.jk@gmail.com	Highlights field and displays error message	Pass

6.2 Functional Testing

The functional testing will take place after the unit testing. In this functional testing, the functionality of each of the module is tested. This is to ensure that the system produced meets the specifications and requirements.

Functional Testing 1: Login with different roles (Management, Patient, Doctor)

Objective: To ensure that the correct page with the correct navigation bar is loaded.

Table 13: Functional testing

No.	Test case/Test script	Attribute and value	Expected result	Actual result	Result
1.	Login as a 'Management' member.	Username : zubair Password: @1234	Main page for the Management is loaded with the Management navigation bar.	Logged in and redirected to management main page.	Pass
2.	Login as a 'Farmer' member.	Username: Ali Password: 1234	Main page for the Farmer is loaded with the Farmer navigation bar.	Logged in and redirected to farmer main page.	Pass

6.3 Rules Testing

Table 14: Ruling testing

Condition 1: Weather Conditions	Condition 2: Soil Moisture Level	Condition 3: Plant Symptoms	Condition 4: Location	Action: Disease Detected
Rainy	High	Yellow spots on leaves	Rural Area	Septoria
Sunny	Low	Red spots on leaves	Urban Area	Stripe Rust
Overcast	Medium	Black spots on leaves	Suburban Area	Septoria
Sunny	High	White powdery spots on leaves	Hillside region	Stripe Rust
Rainy	Low	No visible symptoms	Plains region	No disease detected
Sunny	High	Wilting of leaves	Remote area	Septoria
Overcast	Medium	Stunted growth	Forested area	No disease detected

6.4 Integration Testing

Table 15: Integration testing

No.	Test case/Test script	Attribute and value	Expected result	Actual result	Result
1.	Weather Data Integration	Request Weather Data for Prediction	Successfully retrieved	Successfully retrieves weather data for the specified location and date.	Pass
2.	Weather Data Integration	Generate Disease Prediction Based on Weather	Successfully generated	Successfully generates disease prediction based on the provided weather data.	Pass
3.	Disease Prediction and Notification	Generate Disease Prediction for Notification	Successful retrieval	Successfully generates disease prediction for the specified location and date.	Pass
4.	User Authentication and Data Retrieval	Authenticate User	Success and access granted	Successfully authenticates the user and provides access to data retrieval functions.	User authenticated successfully and granted access to data retrieval module.

Chapter 7: Conclusion and Future Work

7 Conclusion and Future Work

7.1 Conclusion

In conclusion, the development and testing of the Wheat Shield project have been completed successfully. The system has been designed and implemented to provide farmers with valuable insights into wheat disease prediction based on weather data. Through the integration of various modules, including weather data retrieval, disease prediction, user authentication, and notification, Wheat Shield offers a comprehensive solution to help farmers make informed decisions and protect their crops from potential diseases.

The testing phase, including unit testing, functional testing, integration testing, and business rules testing, has ensured the reliability and functionality of the system. Each component has been thoroughly examined to ensure seamless operation and accurate results.

Overall, Wheat Shield represents a significant advancement in agricultural technology, offering farmers a powerful tool to monitor and mitigate the impact of diseases on their wheat crops. With further refinement and optimization, Wheat Shield has the potential to become an indispensable asset for farmers worldwide.

7.2 Future Work

In the future, several enhancements and extensions can be considered to further improve the Wheat Shield system:

Enhanced Prediction Models: Develop more sophisticated disease prediction models using advanced machine learning techniques to improve the accuracy of predictions.

Real-time Data Updates: Implement real-time data updates for weather and disease information to provide farmers with the most up-to-date insights and recommendations.

Mobile Application: Develop a mobile application version of Wheat Shield to provide farmers with convenient access to the system from their smartphones or tablets.

Integration with IoT Devices: Integrate Wheat Shield with IoT devices and sensors installed in the field to collect additional data points for better analysis and prediction.

Multi-language Support: Add support for multiple languages to make Wheat Shield accessible to farmers from different regions and linguistic backgrounds.

Chapter 8: References

8 References

[1] TensorFlow, 2023. [Online]. Available: https://www.tensorflow.org/.

[2] Google, "Google Colab," 2023. [Online]. Available: https://colab.research.google.com/.

[3] MongoDB, " MongoDB," 2024. [Online]. Available: https://www.mongodb.com/.

[4] FastAPI, "FastAPI," 2023. [Online]. Available: https://fastapi.tiangolo.com/.

[5] OpenWeather, "OpenWeather," 2024. [Online]. Available: https://openweathermap.org/api.

[6] Kaggle, " Kaggle," 2023. [Online]. Available: https://www.kaggle.com/datasets/olyadgetch/wheat-leaf-dataset.

[7] V. S. Code, "Visual Studio Code," 2023. [Online]. Available: https://code.visualstudio.com/.

[8] N. L. o. Medicine, "PubMed," 2023. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/.

[9] ISO, "ISO," [Online]. Available: https://www.iso.org/home.html.