**Green Harvest Project System**

**Abstract :**

The "Green Harvest project” website aims to revolutionize the agricultural marketplace by providing a platform for farmers and buyers to seamlessly connect and transact. In contrast to traditional methods of selling produce, which often rely on localized markets and word-of-mouth communication, Green Harvest leverages cutting-edge technology to facilitate direct interactions between farmers and buyers. Inspired by the need for a more efficient and accessible agricultural trade platform, Green Harvest addresses the limitations of existing systems by introducing a novel website approach to product discovery and transaction facilitation. The development methodology employed in creating Green Harvest is rooted in extensive research and analysis of existing agricultural trade systems, as well as insights from research papers and journals .

Additionally, the website integrates live vegetable prices from local markets, providing farmers and buyers with up-to-date information to make informed decisions. Detailed system analysis and design are presented in the report, accompanied by illustrations of system architecture, ER diagrams, class diagrams, and various refinement diagrams. Additionally, the report provides comprehensive details on the system's modules, components, and algorithms, highlighting the technological advancements and implementation details that underpin Green Harvest functionality.

* **Introduction**

In recent years, technology has become an indispensable tool for innovation across various industries, and agriculture is no exception. The Green Harvest project is an ambitious endeavor aimed at leveraging technology to revolutionize agricultural trade by providing a comprehensive platform for farmers and buyers alike.

Traditionally, farmers have faced numerous challenges in marketing their products, often struggling to identify nearby markets or connect with interested buyers. This inefficiency not only leads to income loss but also hampers the growth and sustainability of the agricultural sector. Recognizing these challenges, the Green Harvest website seeks to bridge the gap between farmers and buyers by offering a user-friendly platform for trading agricultural products and services.

**Background Study**

Technological advancements have revolutionized various industries, including agriculture. However, the agricultural sector continues to face challenges such as limited market access, information asymmetry, and inefficient distribution channels. Farmers often struggle to connect with buyers, negotiate fair prices, and access timely information about market trends and demand.

Traditional agricultural trade practices are cumbersome and time-consuming. Farmers are required to physically visit markets or rely on intermediaries to sell their produce, leading to inefficiencies and income loss. Moreover, record-keeping and maintenance of agricultural transactions are often neglected, Profileing in poor traceability and accountability in the supply chain. Recognizing these challenges, the Green Harvest aims to digitize and streamline the agricultural trade process. By providing a user-friendly platform, Green Harvest enables farmers to showcase their products, connect with buyers, and negotiate fair prices from the convenience of their smartphones.

The introduction of Green Harvest timely, considering the increasing demand for fast and convenient services in today's digital era. With Green Harvest, farmers can access a wide range of services, including market information, product listing, and transaction facilitation, all in one centralized platform. This digitalization of agricultural trade not only enhances efficiency and customer satisfaction but also promotes transparency and accountability in the agricultural supply chain.

Despite the presence of numerous traditional marketplaces and intermediaries, the adoption of digitized agricultural services remains low. Green Harvest fills this gap by offering a comprehensive solution that addresses the unique needs of farmers and buyers alike. By providing basic facilities and ensuring a customer-friendly experience, Green Harvest aims to revolutionize agricultural trade and provide farmers with the tools they need to thrive in today's fast-paced world.

* **Problem Statement**

Despite their crucial role in food production, many farmers face significant challenges in connecting with local markets or buyers for their agricultural products. This disconnect often leads to food waste, income loss, and inefficient distribution of produce. Additionally, the reliance on intermediaries in the agricultural supply chain Profiles in farmers receiving minimal profit, as middlemen dominate the trade. Moreover, consumers may miss out on the opportunity to access fresh, locally grown produce due to a lack of information about nearby farms and their offerings. This gap in communication between farmers and consumers not only hampers economic growth but also undermines the potential for sustainable agriculture practices. In response to these pressing challenges, there is an urgent need for innovative solutions that streamline the process of connecting farmers with nearby markets and buyers. A website like Green Harvest has the potential to revolutionize the agricultural supply chain by providing a platform where farmers can directly showcase their products and connect with interested buyers. By eliminating the reliance on intermediaries, Green Harvest aims to empower farmers, improve income opportunities, and ensure consumers have access to fresh, locally sourced produce. In summary, the development of Green Harvest addresses critical issues in the agricultural sector, offering a solution that enhances efficiency, transparency, and sustainability throughout the supply chain.

* **Existing system**

The Agriculture system, which is followed at present, the farmer Used the Product Distributing is manual system. The Farmer consists of Booked Product details and Accounts Details, Printing work that has to be maintained in all aspects are difficult. In the existing system each and every time a reference should be made manually. There are high possibilities to commit errors and mistakes, which leads to produce the wrong statements to the management. Report generation is also not an easy task.

**DRAW BACKS**

Manual Work.

Security of information is low.

Time Consumption.

High Manpower.

* **Proposed System**

The proposed system is named the “Green Harvest Project” shortly named as GHP.

The proposed system will:

* The drawbacks, which are faced during existing system, can be eradicated by using the proposed system.
* The Agriculture System which is proposed now computerizes all the details. Once the Farmer Add the Product details are stored into the computer and Easy to Use Customer View and Buy a Product. Only a single person is enough to maintain all the reports. The security can also be given as per the requirement of the users.
* Their will be an admin interface though which an admin can easily manage farmers, customers and products etc.

**ADVANTAGES OF PROPOSED SYSTEM**

Large volumes of data can be stored with case.

Newly Products stored are updated.

Stored data and procedures can be easily edited.

Reports can be generated with case.

Accurate calculations are made.

* **Goals & Objectives:**

Green Harvest was website approached by a Very big system to develop a web based website to be accessed by the users over the web. Farmers added the products and can be sells, buys, report. The interface of the system will be Customer friendly and as simple as possible yet powerful requiring Report to Farmer. The Agriculture typically comprises fresh produce, products and baked goods Payments along with shelf space reserved for canned.

Web site Green Harvest whose website include finding out the retail price of the things which is kept in the store. By using this system we can minimize the billing time to the store not only accuracy also improved. The price Details and quantity value which is stored in the database

The Green Harvest website is designed to simplify the process of connecting farmers with nearby markets or potential buyers. Through advanced features such as geo-location tracking and product categorization, the website enables farmers to showcase their offerings effectively, while buyers can browse through a diverse range of products and services tailored to their needs.

Following are objectives of our project:

***1. Efficient Administrative Management:***

* Streamline administrative tasks such as registration, and managing financials.

***2. Farmer Services:***

* Provide services including crop management tools, weather forecasts, market insights, and expert advice, empowering farmers with essential resources to optimize their yields and profitability. Connect with us to access tailored solutions that cater to your farming needs and goals.
* Provides an easy to use and understand interface.

***3. Data Accuracy and Security:***

* Ensure the accuracy and security of Product and staff data.
* Implement data backup and recovery mechanisms to prevent data loss.

***4. Customer Services:***

* Provide timely support and solutions, ensuring your farming needs are met efficiently and effectively. Contact us for assistance with orders, product inquiries, or any other concerns—we're here to help you succeed in agriculture.

***5. Accessibility and User-Friendliness:***

* Ensure that the system is user-friendly for both administrators and Products.
* Make the system accessible to all users, including those with disabilities.

***6. Scalability:***

* Design the system to accommodate future growth and changing needs of the university.
* **Scope of the project**

The scope of the Farmer Website project encompasses several key functionalities aimed at facilitating seamless interactions between farmers and buyers within the agricultural marketplace. The website will allow users to register and create profiles, providing essential information about their farming activities or product preferences.

Farmers will have the ability to list their crops, specifying details such as type, quantity, and expected prices.

Real-time market price updates will be accessible to users, empowering them to make informed decisions regarding crop sales. Secure transactions will be facilitated through the integration of reliable payment gateways, ensuring the safety and integrity of financial transactions.

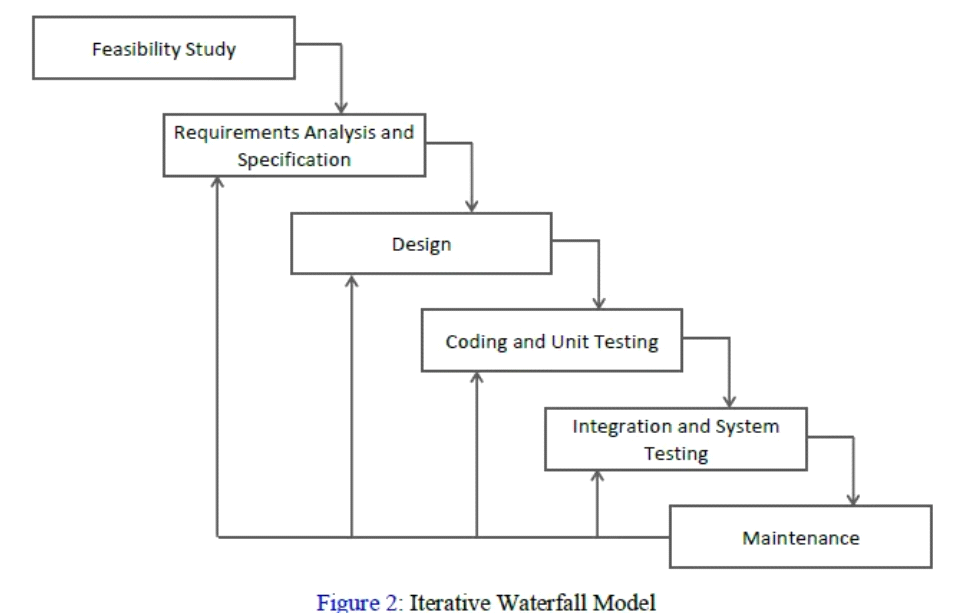
**3.3.1 Limitations**

Despite the comprehensive scope of the Farmer Website project, several limitations must be acknowledged. Market variability poses a challenge, as market prices may fluctuate frequently, impacting the accuracy of price information provided within the website.

Moreover, the reliance on internet connectivity may restrict access to the website in rural areas with limited network coverage. Ensuring the authenticity of user information presents another limitation, as verifying user identities and credentials may prove challenging. Additionally, while efforts will be made to implement robust security measures, including secure transactions and data protection, addressing potential security concerns remains an ongoing challenge that requires continuous vigilance and improvement.

* **Development Methodology**

The model chosen for this system is "Iterative Waterfall model". The iterative waterfall model combines elements of both the waterfall model and iterative development. The iterative waterfall model allows for better communication between team members, better risk management and improved project control. It also provide more flexibility in making changes to the project scope as each phase can be revisited and revised. This is a simple project with the well-defined process and the requirement. The various step of the iterative waterfall model are followed throughout the whole project. This model is understood and incorporate in these project. Thus, the system is developed according to the iterative waterfall model. The various steps of this model is shown in figure below.



**2.2 Software Requirements Specification**

In our project, we have two types of software requirements specifications. We will discuss and explain each of them separately here.

**2.2.1 Functional Requirements**

Requirement Analysis, also known as Requirement Engineering, is the process of defining user expectations for a new software being built or modified. In software engineering, it is sometimes referred to loosely by names such as requirements gathering or requirements capturing. Requirements analysis encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product or project, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements.

1**. Functional Requirements.**

A Functional Requirement (FR) is a description of the service that the software must offer. It describes a software system or its components. A function is nothing but inputs to the software system, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform. Here are some of the functional requirements for Green Harvest:

* **User Registration and Authentication:**
* Users should be able to create accounts and log in securely using email or phone number authentication. The website should verify user identity and prevent unauthorized access to user data.
* **Product Listing and Management:**
* Farmers should be able to list their agricultural products with details such as type, quantity, and price. Farmers should have the ability to edit, update, and remove their product listings as needed.
* **Communication and Transaction Facilitation:**
* The website should facilitate communication between farmers and buyers, allowing them to negotiate prices and arrange transactions.
* **Market Information and Price Updates:**
* Farmers should have access to real-time market information, including prices of agricultural products in various markets.
* **Secure Payment Gateway:**
* The website should integrate with secure payment gateways to enable seamless and secure transactions between farmers and buyers. Users should have confidence in the security of their financial transactions within the website.

**2.2.2 Non-Functional Requirements**

Non-functional requirements for a logo analyzer system describe the characteristics and qualities that the system must exhibit, beyond its functional capabilities. Here are some non-functional requirements for Green Harvest:

* **Response Time:**
* The website should have fast response times, with minimal delays in loading pages or processing transactions.
* **Scalability:**
* The website should be able to handle a large number of users and transactions without significant performance degradation.
* **Reliability:**
* The website should be reliable and available for use consistently, with minimal downtime or system failures.
* **User Interface:**
* The website should have an intuitive and user-friendly interface that is easy to navigate and understand for farmers and buyers.
* **Secure Transactions:**
* The website should utilize secure payment gateways and encryption protocols to protect financial transactions and sensitive information.

**2.2.1 Feasibility Study**

In the feasibility study of Green Harvest, we analyze the technical, operational, and economic aspects to determine the viability of the system.

* **Technical Feasibility:**

The technical feasibility analysis focused on determining whether the development of Green Harvest feasible given the available technology and resources. Our technical team evaluated the compatibility, availability, and cost-effectiveness of the necessary software and hardware components. We conducted tests to gauge the system's performance, scalability, and reliability, considering factors such as the compatibility of mobile website development frameworks, availability of essential APIs, cost-effectiveness of hosting solutions, and scalability to accommodate potential growth.

* **Operational Feasibility:**

Regarding operational feasibility, our aim was to ascertain whether Green Harvest could seamlessly integrate into existing organizational processes and practices. The operational team assessed the potential impact of the proposed system on current workflows, staff training requirements, and user acceptance. We also analyzed the system's maintenance and support needs to ensure its sustainability. Key considerations included integrating Green Harvest with existing agricultural trade practices, conducting user acceptance testing,

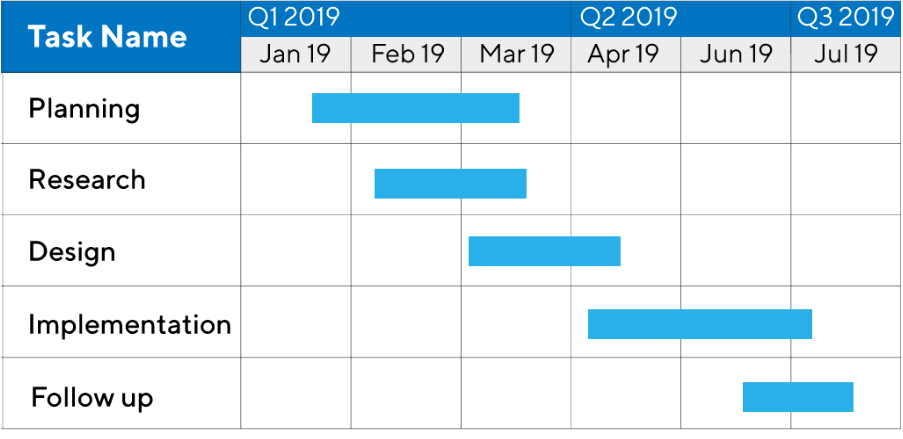
providing training programs for users, and establishing support mechanisms.

* **Economic Feasibility:**

The economic feasibility study sought to determine the financial viability of Green Harvest. We analyzed the development, implementation, and operational costs, comparing them with potential benefits and returns. Additionally, we projected potential revenue streams and assessed long-term profitability, potential risks, and the impact .This analysis encompassed estimating development costs, operational expenses, potential revenue streams, and conducting risk analysis to identify potential financial obstacles and mitigation strategies

* **Account Feasibility:**

For each task of the project, proper estimation and splitting of the time have been done. Overall, the calculated time is sufficient enough to complete project in time.



**2.3 Use Cases Analysis**

Use case analysis involves identifying , defining and analyzing different use cases and scenarios to understand how certain users act in certain situations. It helps in identifying the errors or malfunctioning of a function or to know whether they are behaving as they are supposed to.

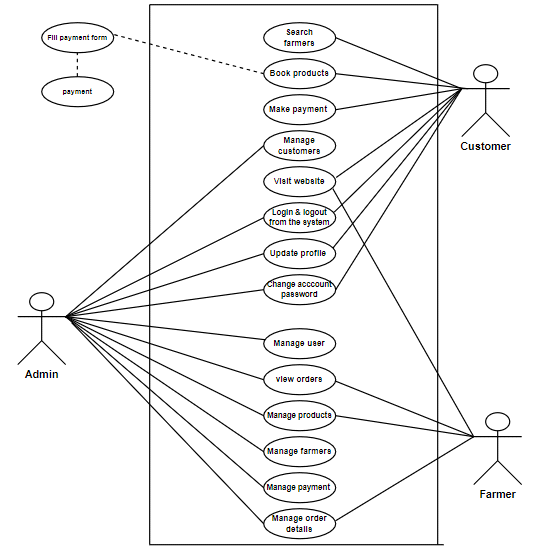
Here is the use case analysis of the **Green Harvest System** :

* **Identify Actors:**
* Identify the different actors (users or external systems) interacting with the University management System. Common actors could include:
* ***Administrator:*** Manages the overall system, including user accounts, courses, and campus facilities.
* ***Farmer*** : Manages Products, and grading.
* ***customer:*** Accesses course information, submits assignments, and views grades.
* **Identify Use Cases:**
* List down the high-level functionalities or actions the system should support, such as adding or removing products, adding or removing farmers and customers, managing products etc.
* **Detail Each Use Case:**
* For each identified use case, provide a detailed description including preconditions, post conditions, normal and alternative flows, exceptions, and potential errors.
* **Use Case Diagram:**

Create use case diagrams to visually represent the relationships between actors and use cases. This diagram shows an overview of the functionality of the system.

The use case diagram for this project is **Fig:2.1.**

* **Prioritize Use Cases:**
* Prioritize use cases based on their importance, relevance to the system's objectives, and frequency of usage.
* **Refine Use Cases Iteratively:**
* As the project progresses, continuously refine and update the use cases based on feedback, changes in requirements, or new insights.
* **Verify and Validate:**
* Ensure that the use cases are accurate, complete, and meet the stakeholders' requirements through verification and validation processes.



**Fig 2.1 : Use Case For Green Harvest System**

**2.3.1 Use Case Brief Details:**

**2.3.1.1 Admin Login:**

|  |  |
| --- | --- |
|  | UC-01 : Admin Login |
| Use Case ID: | UC-01 |
| Use Case Name: | Admin Login |
| Description: | This use case describes the process by which admin will login to its dashboard to access all the details. |
| Actors: | Admin |

**2.3.1.2 Farmer Login:**

|  |  |
| --- | --- |
|  | UC-02 : Farmer Login |
| Use Case ID: | UC-02 |
| Use Case Name: | Farmer Login |
| Description: | This use case describes the process by which famer will login to its dashboard and access its information. |
| Actors: | Farmer |

**2.3.1.3 Customer Login:**

|  |  |
| --- | --- |
|  | UC-02 : customer Login |
| Use Case ID: | UC-03 |
| Use Case Name: | Customer Login |
| Description: | This use case describes the process by which customer will login to their dashboard and access all the details. |
| Actors: | customer |

**2.3.1.4 Payment Management :**

|  |  |
| --- | --- |
|  | UC-04 : Payment Management |
| Use Case ID: | UC-04 |
| Use Case Name: | Payment Management |
| Description: | This use case describes the process by which admin manage  payments of users |
| Actors: | Admin |

**2.3.1.5 Product Management :**

|  |  |
| --- | --- |
|  | UC-05 : Product Management |
| Use Case ID: | UC-05 |
| Use Case Name: | Product Management |
| Description: | This use case describes how products will be added or removed or updated by admins and farmers. |
| Actors: | Admin , Farmer |

**2.3.1.6 Order Management :**

|  |  |
| --- | --- |
|  | UC-06 : Order Management |
| Use Case ID: | UC-06 |
| Use Case Name: | Order Management |
| Description: | This use case describes how orders will be managed by admin. |
| Actors: | Admin |

**2.3.1.7 Categories Management:**

|  |  |
| --- | --- |
|  | UC-07 : Categories Management |
| Use Case ID: | UC-07 |
| Use Case Name: | Categories Management |
| Description: | This use case describes how Categories section will be managed by admins. |
| Actors: | Admins |

**2.3.2 Use Case Detailed Description:**

**2.3.2.1 Admin Login:**

|  |  |
| --- | --- |
|  | UC-01 : Admin Login |
| Use Case ID: | UC-01 |
| Use Case Name: | Admin Login |
| Description: | This use case describes the process by which admin will login to its dashboard to access all the details. |
| Actors: | Admin |
| Pre-conditions: | Internet required , Webapp must be running and user should know their login details. |
| Post – Conditions: | Admin logged in |
| Includes: | WebApp , MySql , Interface |
| Basic Flow: | * Admin will enter the name , cnic and password. * Then it will press the submit button. * Webapp will cross check it with database . * If validated then admin will be logged in. |
| Frequency of Use: | Numerous |

**2.3.2.2 Customer Login:**

|  |  |
| --- | --- |
|  | UC-02 : Customer Login |
| Use Case ID: | UC-02 |
| Use Case Name: | Customer Login |
| Description: | This use case describes the process by which customer will login to its dashboard and access its information. |
| Actors: | Customer |
| Pre-conditions: | Internet required , Webapp must be running and user should know their login details. |
| Post – Conditions: | Customer logged in |
| Includes: | WebApp , MySql , Interface |
| Basic Flow: | * User will enter its CNIC number and password. * WebApp will validate it with the ones stored in database. * If validated then customer will log in. |
| Frequency of Use: | Numerous |

**2.3.2.3 Farmer Login:**

|  |  |
| --- | --- |
|  | UC-02 : Farmer Login |
| Use Case ID: | UC-03 |
| Use Case Name: | Farmer Login |
| Description: | This use case describes the process by which Farmers will login to their dashboard and access all the details. |
| Actors: | Farmers |
| Pre-conditions: | Internet required , Webapp must be running and user should know their login details. |
| Post – Conditions: | Farmer logged in |
| Includes: | WebApp , MySql , Interface |
| Basic Flow: | * Farmer will enter id and password. * WebApp will validate is against the database data. * If validated , Farmer will be logged in. |
| Frequency of Use: | Numerous |

**2.3.2.4 Payment Management :**

|  |  |
| --- | --- |
|  | UC-04 : Payment Management |
| Use Case ID: | UC-04 |
| Use Case Name: | Payment Management |
| Description: | This use case decribes how Payments can be added or removed or updated by admins. |
| Actors: | Admin |
| Pre-conditions: | Internet required , Webapp must be running and admin must be logged in. |
| Post – Conditions: | Payment added , removed or updated. |
| Includes: | WebApp , MySql , Interface |
| Basic Flow: | * Admin will go to the dashboard. * Go in the Payments tab add or remove the payment according to action. * After making changes it’ll save the changes. |
| Frequency of Use: | Rarely |

**2.3.2.5 Product Management :**

|  |  |
| --- | --- |
|  | UC-05 : Product Management |
| Use Case ID: | UC-05 |
| Use Case Name: | Product Management |
| Description: | This use case describes how Products will be added or removed or updated by admins and Farmers. |
| Actors: | Admin , Farmer |
| Pre-conditions: | Internet required , Webapp must be running and admin/Farmer must be logged in.. |
| Post – Conditions: | Product added , removed or updated |
| Includes: | WebApp , MySql , Interface |
| Basic Flow: | * Go to Products tab and operate the function. * Save the changes will be updated in database. |
| Frequency of Use: | Numerous |

**2.3.2.6 Orrder Management :**

|  |  |
| --- | --- |
|  | UC-06 : Order Management |
| Use Case ID: | UC-06 |
| Use Case Name: | Order Management |
| Description: | This use case describes how Orders will be managed by admins. |
| Actors: | Admin |
| Pre-conditions: | Internet required , Webapp must be running and admin must be logged in. |
| Post – Conditions: | Order added , removed or updated. |
| Includes: | WebApp , MySql , Interface |
| Basic Flow: | * Admin will go to the dashboard. * Open the Orders tab. * Click on “add Order” to add , remove to remove and update to update. Save the changes in database. |
| Frequency of Use: | Occasionally |

**2.3.2.7 Categories Management:**

|  |  |
| --- | --- |
|  | UC-07 : Categories Management |
| Use Case ID: | UC-07 |
| Use Case Name: | Categories Management |
| Description: | This use case describes how Categories section will be managed by admin. |
| Actors: | Admin |
| Pre-conditions: | Internet required , Webapp must be running and Order must be logged in. |
| Post – Conditions: | Profile updated |
| Includes: | WebApp , MySql , Interface |
| Basic Flow: | * Admin will go to dashboard in Categories tab. * Admin will upload/update Categories. * Save the changes to the database. |
| Frequency of Use: | Occasionally |

* **ERD**

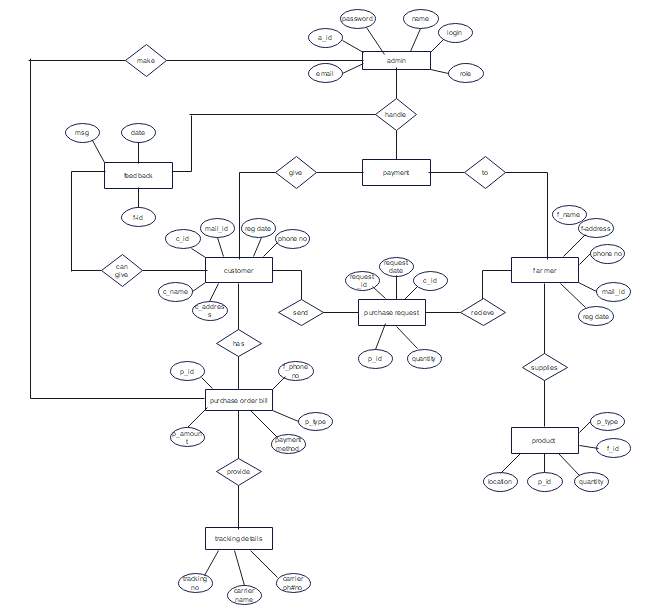


Fig 3.1 : Entity – Relationship Diagram of Green Harvest Project System

* **DFD**
* ***0- Level***

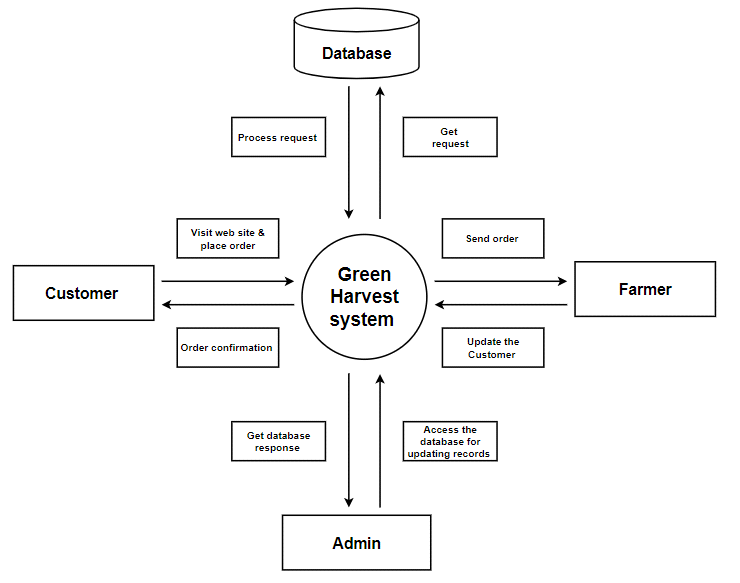


Fig 3.2.1 : Data-Flow Diagram 0- Level

* ***1- Level***

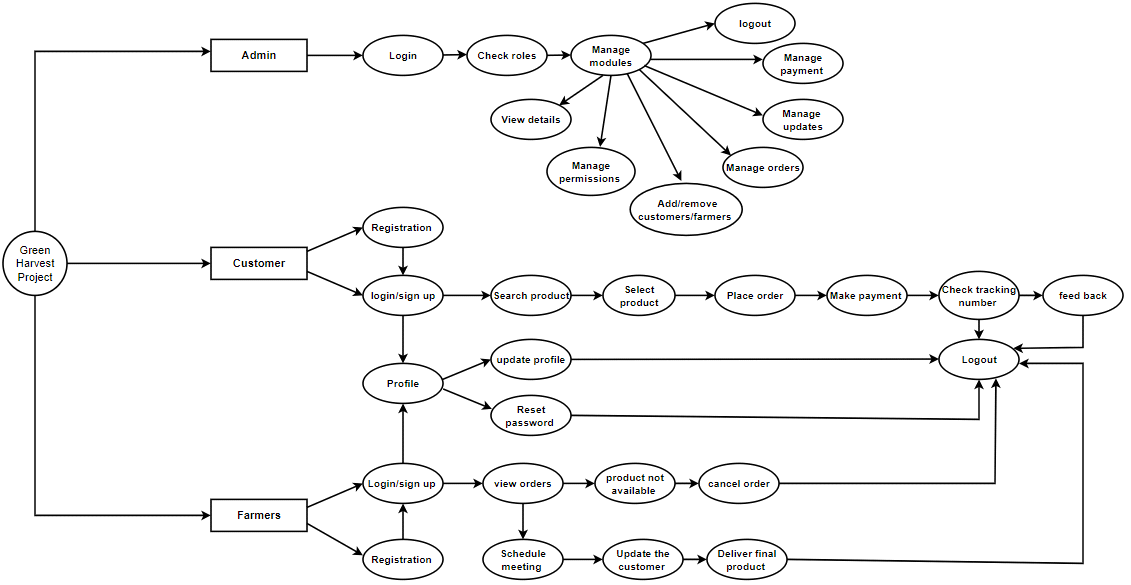
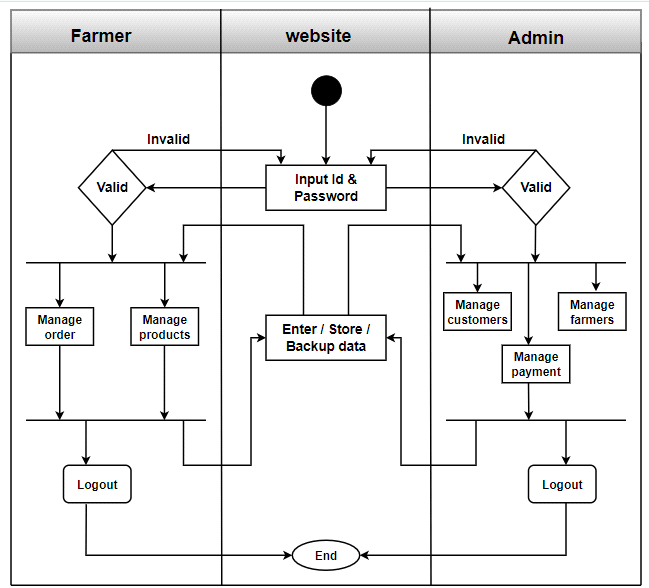


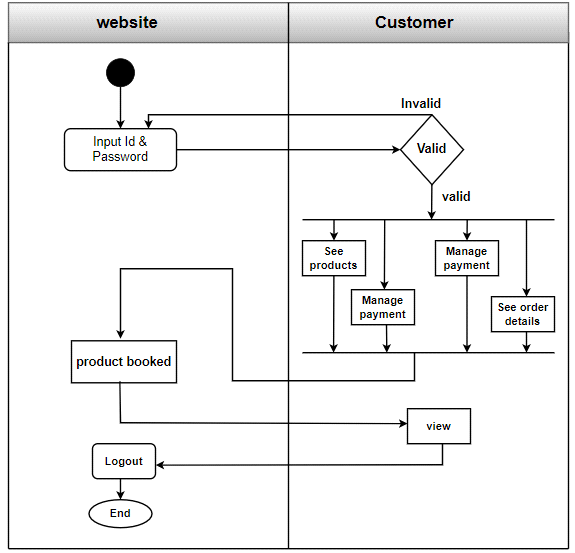
Fig 3.2.2 : Data-Flow Diagram 1- Level

* **Activity Diagram**

**3.3.1. Admin and Order activity diagram**



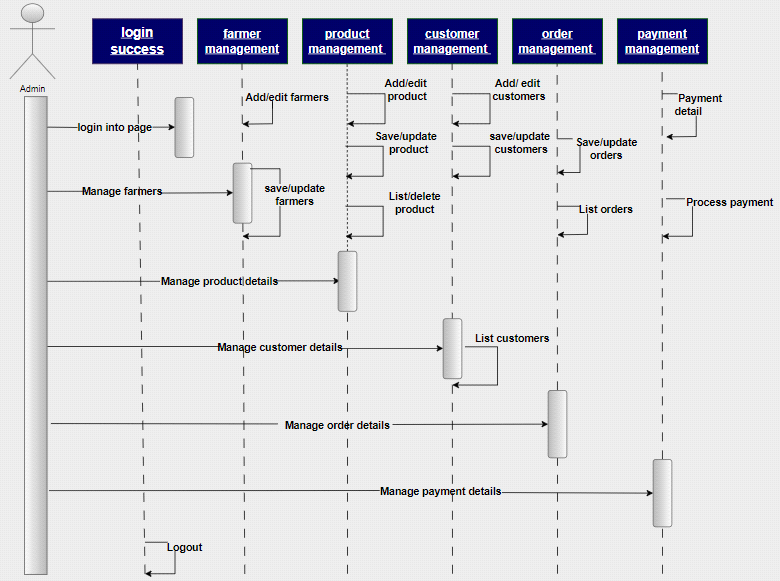
**Fig 3.3.1 : Admin & Order Side Activity Diagram**



**Fig 3.3.2 : Product Side Activity Diagram**

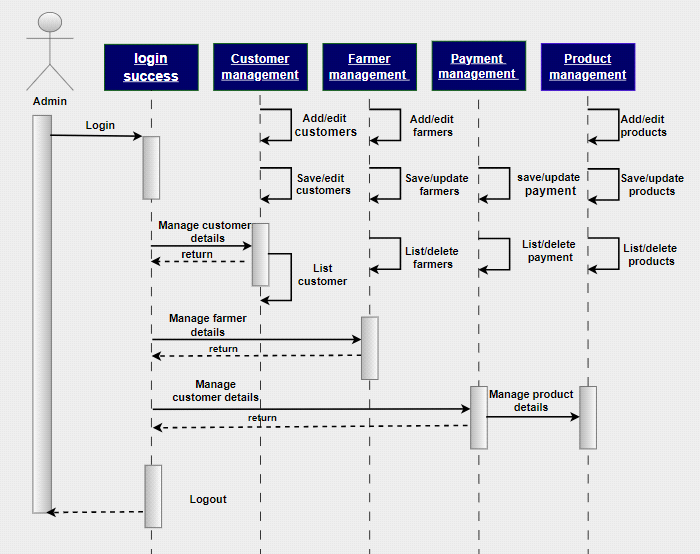
* **Sequence Diagram**

**3.4.1 Admin Sequence Diagram**



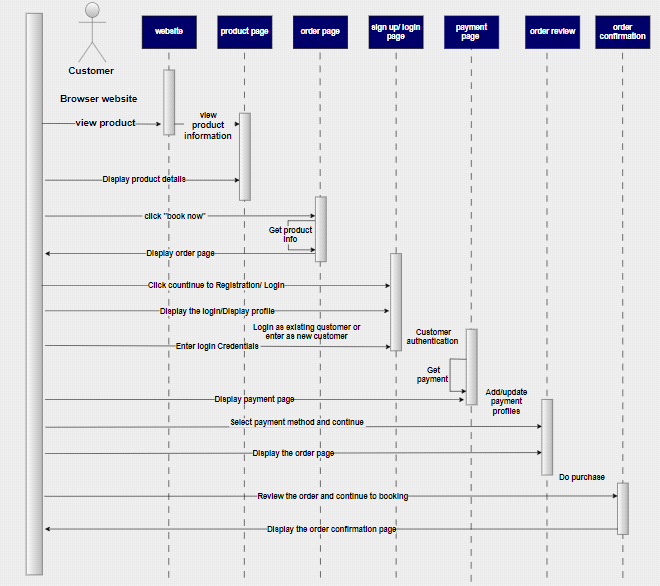
**Fig 3.4.1 : Admin Side Sequence Diagram**

**3.4.2 Order Side Sequence diagram**



**Fig 3.4.2 : Order Side Sequence Diagram**

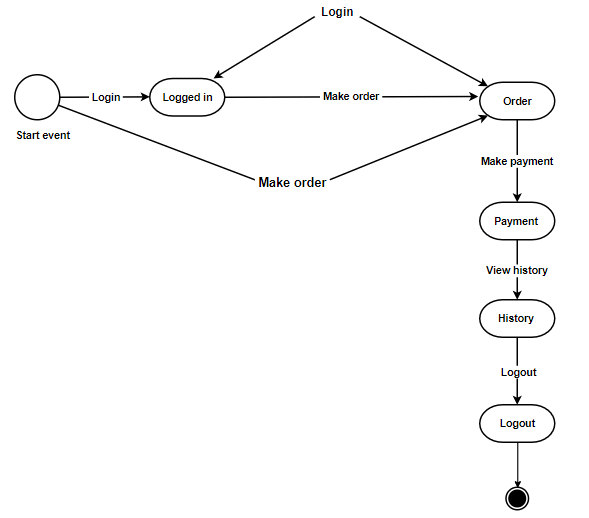
* **Customer Side sequence diagram**



**Fig 3.4.1 : Customer Side Sequence Diagram**

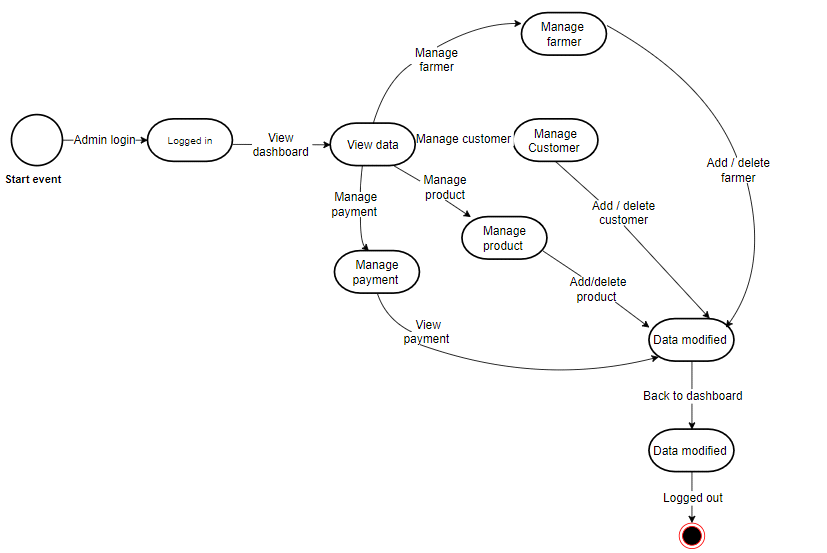
* **State-Transition Diagram**

**3.5.1 customer state-transition diagram**

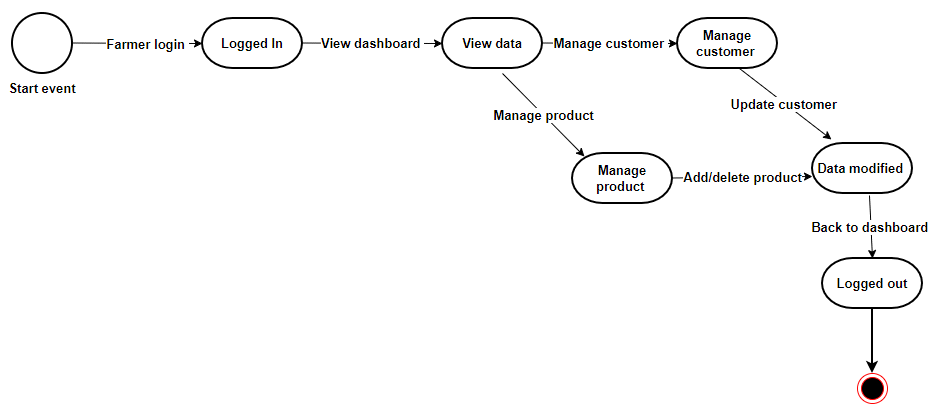


**Fig 3.5: State – Transition Diagram**

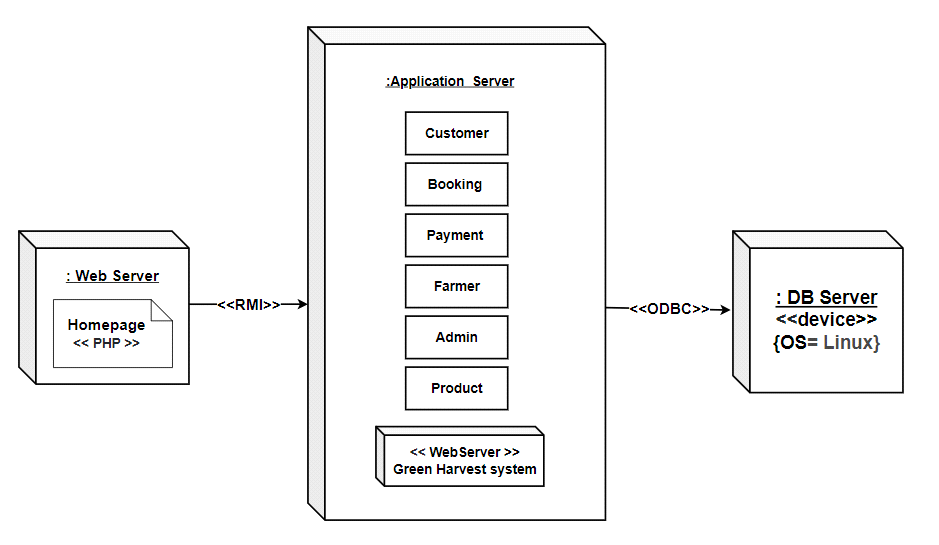
**3.5.3.Admin state transition diagram:**



* **Order state transition diagram**

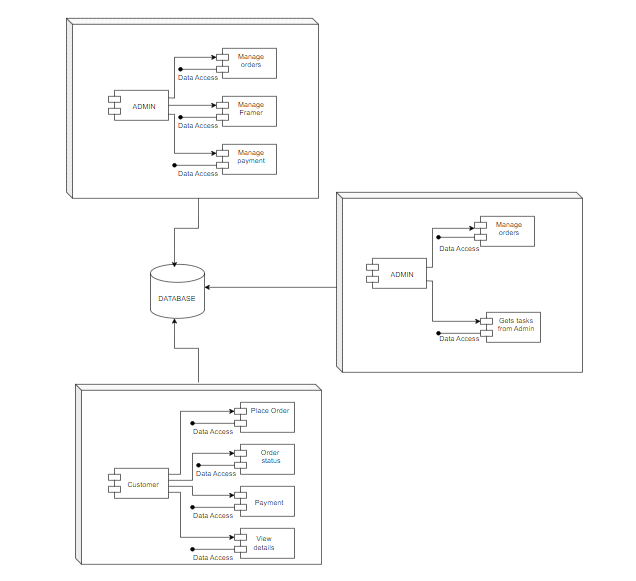


* **Deployment Diagram**



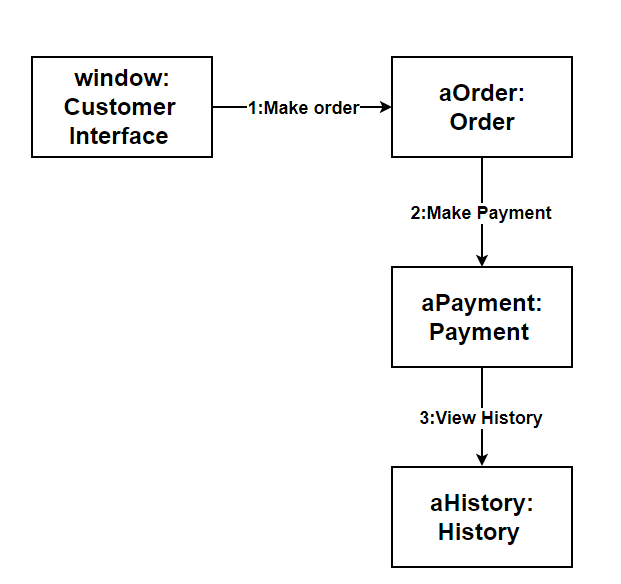
**Fig 3.6 : Deployment Diagram of GHS**

* **Component Diagram**



**Fig 3.7 : Component Diagram of GHS**

* **Collaboration Diagram**



**Fig 3.8 : Collaboration Diagram of GHS**

**4.1 Architectural Design :**

Architectural design represents the structure of data and program components that are required to build a computer based system. It considers the architectural style that the system will take the structure and properties of the components the constitute the system, and the architectural component of a system.

The architectural style we used for our system is a **Layered Architecture.** Where all of the modules are working at different layer. There are total of 4 layers. Layers and its components are as follows:

* ***Presentation Layer:*** It includes the user interface of our Green Harvest system. The users involved are the customers, Orders and admins.
* ***Application Layer:*** It contains the core business logic of our GHS .Its components involves login authentication, customer management , Order management , product management , payment management , order management.
* ***Integration Layer:*** This layer connects our webservice with the database. Its components are the APIs used to connect to the database.
* ***Data Layer :*** This is the core layer where all of our database is stored. Its component is the stored data like Orders, customers, products and payment etc. records and everything.

The diagrammatical view of our system architecture is shown in **Fig 4.1.**

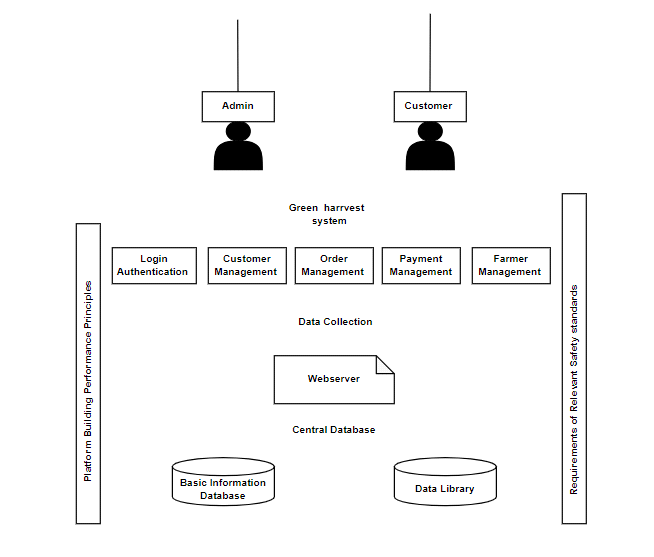


Fig 4.1 : Layered Architecture Design Of GHS

**Chapter 5 System Testing**

* **Test Case Specification**

A test case plan, as it applies to a Flight Reservation System, provides a clear outline of how to validate specific features or functions of the system. Each test case spells out what to put in, what to expect out, the conditions for testing, and what you need before starting.

Here's a standard format for a test case plan tailored to a University management system:

* ***Test Case Code:*** A unique number or code for the test case.
* ***Test Case Description:*** A brief explanation of what this test case is checking.
* ***Testing Goal:*** What we want to achieve or confirm with this test case.
* ***Prerequisites:*** Things that must be in place before we can run this test. For instance, the system setup or having the right permissions.
* ***Input Actions:*** The actions or data we need to input to perform this test.
* ***Expected Results:*** What we should see happening in the system after the test.
* ***Testing Steps:***
* Detailed steps to carry out the test.
* Expected results after each step.
* ***After Testing:*** What the system looks like after this test has been done.
* ***Pass/Fail Criteria:*** What makes this test a success or a failure?
* ***Notes/Comments:*** Any extra information or observations about this test case.
* **Black Box Testing:**

Black box testing is a method of testing software where the tester evaluates how a system works without delving into the inner workings of its code, intricate details, or design. In black box testing, the focus is squarely on checking the inputs and outputs of the system to ensure that the software operates as intended according to established requirements.

A black box test case comprises a series of instructions or test scenarios meticulously crafted to verify the functionality and conduct of the software or system from an external standpoint. These test cases are generated based on the software's outlined specifications, requirements, or user guides.

**Test Case Specification For Black Box:**

***Test Case:***

Following are the test cases designed for our **Flight Reservation System**:

***Login***

|  |  |
| --- | --- |
| TC-01: Login | |
| Test Case ID: | TC-01 |
| Wrote By: | Amna bibi , Rubab tehreem |
| Test Type: | Black Box testing |
| Product Name: | Green Harvest system |
| Test Item: | WebApp , MySQL |
| Documented Date: | 28th july , 2024 |
| Test case description: | This test case is designed for testing admin , farmer and customer login. |
| Operation procedure: | * User will first go to login on the main page for farmer and customer login and for admin it needs to access admin page by adding “/admin” in the search link after the webapp link. * Then user will input credentials in the input field and then click the submit button. * Webapp will validate it and once validated it’ll redirect to their respective dashboards. |
| Pre-conditions: | Internet is required and webApp must be running , and user should also remember their credentials. |
| Post-conditions: | User will be logged in to their respective dashboards. |

***Add products***

|  |  |
| --- | --- |
| TC-02: Add Products | |
| Test Case ID: | TC-02 |
| Wrote By: | Amna bibi , Rubab Tehreem |
| Test Type: | Black Box testing |
| Product Name: | Green Harvest System |
| Test Item: | WebApp , MySQL |
| Documented Date: | 28th july , 2024 |
| Test case description: | This test case is designed to test the add product functionality of the project. |
| Operation procedure: | * Admin first login to the dashboard. * It’ll go to the product tab and then click on “add product “ button. * It fills the necessary information and click on submit button * The data will be saved in the database |
| Pre-conditions: | Internet is required and WebApp must be running. |
| Post-conditions: | Product Will be added . |

***Remove Product***

|  |  |
| --- | --- |
| TC-03 : Remove Product | |
| Test Case ID: | TC-03 |
| Wrote By: | Amna bibi, Rubab tehreem |
| Test Type: | Black Box testing |
| Product Name: | Green Harvest System |
| Test Item: | WebApp , MySQL |
| Documented Date: | 28th july , 2024 |
| Test case description: | This test case is designed to test the functionality of removing the product from the system. |
| Operation procedure: | * Admin will login into the dashboard * It’ll go to the product page. * It’ll click on remove product. * WebApp will delete the product from the database. |
| Pre-conditions: | Internet is required and WebApp must be running. |
| Post-conditions: | product will be removed. |

***Add Farmers***

|  |  |
| --- | --- |
| TC-04 : Add farmers | |
| Test Case ID: | TC-04 |
| Wrote By: | Amna bibi, Rubab tehreem |
| Test Type: | Black Box testing |
| Product Name: | Green Harvest System |
| Test Item: | WebApp , MySQL |
| Documented Date: | 28th july , 2024 |
| Test case description: | This test case is designed to test the working of the “add farmers” Function. |
| Operation procedure: | * First of all admin will login to the dashboard. * Admin will go to the farmer’s tab . * It’ll click on “add farmers” button. * the admin will fill in the form and then submit it . * The webapp will saves the data in the database in the farmer’s table. |
| Pre-conditions: | Internet is required and WebApp must be running. |
| Post-conditions: | The farmers will be added. |

***Remove Farmers***

|  |  |
| --- | --- |
| TC-05: Remove Farmers | |
| Test Case ID: | TC-05 |
| Wrote By: | Amna bibi, Rubab tehreem |
| Test Type: | Black Box testing |
| Product Name: | Green Harvest System |
| Test Item: | WebApp , MySQL |
| Documented Date: | 28th july , 2024 |
| Test case description: | This test case is designed to test the functionality of “remove farmers” button. |
| Operation procedure: | * Admin will login to the dashboard. * Admin will go to the farmer’s tab. * To remove the farmers it’ll click on the “remove” button in front of the farmers it want to remove. * Upon clicking the webapp removed the respective farmers data from the database. |
| Pre-conditions: | Internet is required and WebApp must be running. |
| Post-conditions: | The farmers will be removed. |

***Add Customers***

|  |  |
| --- | --- |
| TC-06 : Add customers | |
| Test Case ID: | TC-06 |
| Wrote By: | Amna bibi, Rubab tehreem |
| Test Type: | Black Box testing |
| Product Name: | Green Harvest System |
| Test Item: | WebApp , MySQL |
| Documented Date: | 28th july , 2024 |
| Test case description: | This test case is designed to test the functionality of “Add customers” button. |
| Operation procedure: | * First of all admin will login to the dashboard. * Admin will go to the teacher’s tab . * It’ll click on “add customer ” button. * the admin will fill in the form and then submit it . * The webapp will saves the data in the database in the customer’s table. |
| Pre-conditions: | Internet is required and WebApp must be running. |
| Post-conditions: | customer will be added. |

***Remove Customers***

|  |  |
| --- | --- |
| TC-07 : Remove Customers | |
| Test Case ID: | TC-07 |
| Wrote By: | Amna bibi, Rubab tehreem |
| Test Type: | Black Box testing |
| Product Name: | Green Harvest System |
| Test Item: | WebApp , MySQL |
| Documented Date: | 28th july , 2024 |
| Test case description: | This test case is designed to test the functionality of :remove teacher “ button. |
| Operation procedure: | * Admin will login to the dashboard. * Admin will go to the customer’s tab. * To remove the customer's it’ll click on the “remove” button in front of the customer it want to remove. * Upon clicking the webapp removed the respective customers data from the database. |
| Pre-conditions: | Internet is required and WebApp must be running. |
| Post-conditions: | customers will be removed. |

***Add/Update products***

|  |  |
| --- | --- |
| TC-09 : Add/Update Products | |
| Test Case ID: | TC-09 |
| Wrote By: | Amna bibi, Rubab tehreem |
| Test Type: | Black Box testing |
| Product Name: | Green Harvest System |
| Test Item: | WebApp , MySQL |
| Documented Date: | 28th july , 2­024 |
| Test case description: | This test case is designed to test the functionality of adding or updating the products of customer. |
| Operation procedure: | * Farmer will login to the dashboard. * It’ll go to the customer it wants to add the product of. * It’ll insert the product data and submit it . * The Webapp will add the product in the database of that customer. |
| Pre-conditions: | Internet is required and WebApp must be running. |
| Post-conditions: | Products will be added / updated. |

**5.3 White Box Testing:**

White box testing is a software testing approach that entails the examination of the internal workings, structure, and design of the software or system. In white box testing for a University management system, the tester possesses an understanding of the system's code and applies this knowledge to devise test cases aimed at thoroughly covering the code paths, branches, conditions, and the overall code structure.

A white box test case consists of a collection of test scenarios and procedures tailored to validate the software based on its internal operations, code flow, and structure. The primary objective is to confirm the accuracy of the code, reveal any logical flaws, and guarantee that all code paths are effectively examined.

**Test Case Specifications :**

Following are the test cases designed for white box testing:

|  |  |
| --- | --- |
| TC-001 : Successful Login | |
| Test Case ID: | TC-001 |
| Test Type: | White Box Testing |
| Documented Date: | 15th October , 2023 |
| Test case Objective: | To ensure that the login functionality correctly validates user credentials. |
| Pre-conditions: | User Account Exists in the system. |
| Test Steps: | * Enter valid user credentials like id and password. * Verify that the user is granted access. * Repeat the steps with invalid credentials. |
| Test Data: | Student login : Roll no : 73431431  Password : st5999 |
| Expected Results: | Student successfully logged in . |
| Actual Result : | Student was successfully logged in and directed to the home page. |
| Pass / Fail criteria: | **Pass:** With correct login credentials , the user logged in as a student successfully.  **Fail:** With wrong credentials , student failed to login. |
| Test Environment: | Web Browser |
| Test case Status: | Pass |

|  |  |
| --- | --- |
| TC-002 : Add Student | |
| Test Case ID: | TC-002 |
| Test Type: | White Box Testing |
| Documented Date: | 15th October , 2023 |
| Test case Objective: | To ensure that the add student functionality works correctly. |
| Pre-conditions: | Admins needs to have internet access and logged in to the dashboard. |
| Test Steps: | * Admin will go to the student dashboard and click on “Add student” button. * Form will open , then enter the details of the student . * Then submit it . * Data will be saved in the database. |
| Test Data: | First name: Aneeqa  Last name : Tahir  Roll no : 11203  Batch no : 5509  Department : BS Maths  Semester : first  Fees Status : Unpaid |
| Expected Results: | Student added successfully. |
| Actual Result : | Student added successfully. |
| Pass / Fail criteria: | **Pass :** Student Added Successfully  **Fail :**  Failed to add student. |
| Test Environment: | Web Browser |
| Test case Status: | Pass |

|  |  |
| --- | --- |
| TC-003 : Add Teacher | |
| Test Case ID: | TC-003 |
| Test Type: | White Box Testing |
| Documented Date: | 15th October , 2023 |
| Test case Objective: | To ensure that the add teacher functionality works correctly. |
| Pre-conditions: | Admins needs to have internet access and logged in to the dashboard. |
| Test Steps: | * Admin will go to the teacher dashboard and click on “Add teacher” button. * Form will open , then enter the details of the teacher . * Then submit it . * Data will be saved in the database. |
| Test Data: | First name: Sadia  Last name : Sharif  CNIC – 123456-123434-3  Department : BS Maths |
| Expected Results: | Teacher added successfully. |
| Actual Result : | Teacher added successfully. |
| Pass / Fail criteria: | **Pass :** Teacher Added Successfully  **Fail :**  Failed to add teacher. |
| Test Environment: | Web Browser |
| Test case Status: | Pass |

**future Enhancements :**

Our current project is a web based solution for managing administration and students with ease . In the future we can make the following enhancements :

* We can add an LMS , learning management system which will allow student to take online classes and mark online attendance.
* We can add library management system which will allow students to donate or borrow books online .
* We can also add servers and make a chatting hub for the students and teachers where they can connect with each other live.