Question 1 – Functional Requirements - 15 Marks

Here is a list of 20 functional requirements for the online agriculture product store based on the case study provided:

Functional requirements :

Functional requirement define the specific behaviors,functions or operations of a system. They describe what the system should do , outlining the necessary tasks, actions or activities it must perform to achieve its objectives.

Non functional requirement :

Non functional requirement will describe the qualities and attributes of a system , focusing on how the system performs rather than specific behaviors or functions .

**Functional Requirements**

1. **FR0001: Farmer Registration**
   * Farmers should be able to register with the application using their email and create a secure password.
2. **FR0002: Farmer Search for Products**
   * Farmers should be able to search for available products such as fertilizers, seeds, and pesticides.
3. **FR0003: Manufacturer Registration**
   * Manufacturers should be able to register with the application and manage their product listings.
4. **FR0004: Product Upload by Manufacturers**
   * Manufacturers should be able to upload details of their products (fertilizers, seeds, pesticides) to display them to farmers.
5. **FR0005: Product Catalog**
   * The system should display a catalog of all available products with details like name, price, and quantity.
6. **FR0006: Product Search Filter**
   * Farmers should be able to filter search results by price, product type, manufacturer, and availability.
7. **FR0007: Add to Cart**
   * Farmers should be able to add selected products to their shopping cart for purchase.
8. **FR0008: Buy-Later List**
   * Farmers should be able to add products to a buy-later list if they decide not to purchase immediately.
9. **FR0009: Payment Gateway Integration**
   * The system should support multiple payment methods such as cash-on-delivery (COD), Credit/Debit cards, and UPI.
10. **FR0010: Order Confirmation Email**

* After a successful order, the system should send a confirmation email to the farmer with order details.

1. **FR0011: Order Tracking**

* Farmers should be able to track the status of their orders in real-time.

1. **FR0012: User Login and Authentication**

* Both farmers and manufacturers should be able to log in to the system securely using email and password.

1. **FR0013: Password Recovery**

* Users should be able to recover their account passwords via email if they forget them.

1. **FR0014: Product Reviews and Ratings**

* Farmers should be able to leave reviews and ratings for products they have purchased.

1. **FR0015: Notifications for Farmers**

* Farmers should receive notifications for successful orders, pending deliveries, and new product updates.

1. **FR0016: Manufacturer Inventory Management**

* Manufacturers should be able to manage their product inventory, including updating stock levels.

1. **FR0017: Farmer Account Management**

* Farmers should be able to manage their account information, including shipping addresses and payment preferences.

1. **FR0018: Product Availability Alerts**

* The system should notify farmers when out-of-stock products become available again.

1. **FR0019: Transaction History**

* Farmers should be able to view a history of their past transactions, including orders and payments.

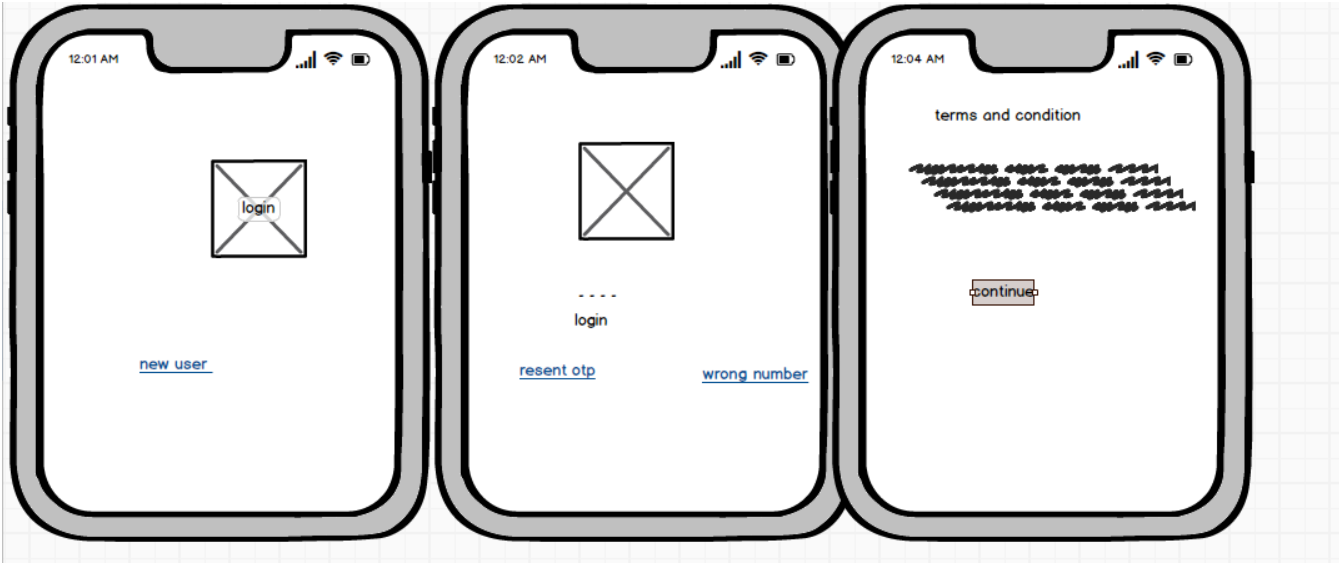
1. **FR0020: Secure Payment Processing**

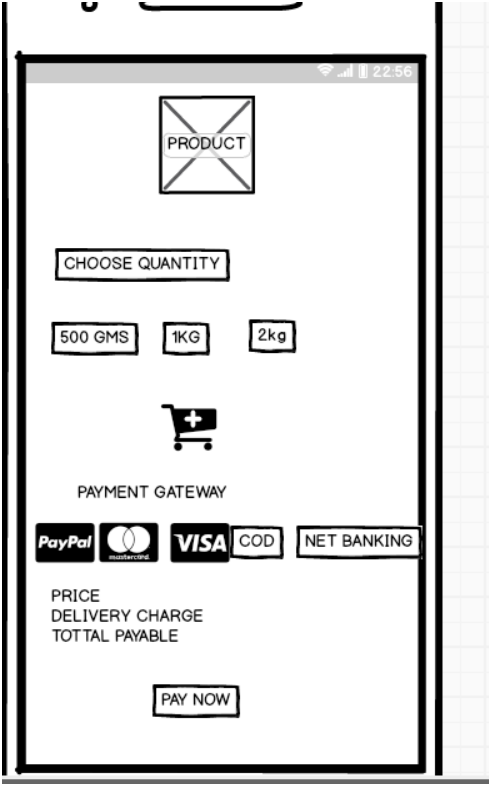
* The system should ensure that all payment transactions are securely processed to protect sensitive information.

These functional requirements will ensure that the application meets the needs of both farmers and manufacturers while maintaining ease of use and functionality.

Question 2–Minimum 5 page designs - 15 Marks Make wireframe and prototypes

Below are the 5 key pages that you can design as wireframes:





**1. Homepage (Farmer Login/Signup)**

* **Sections**:
  + Header with logo, navigation (Home, Products, Contact Us)
  + Login form for returning users (Email, Password, Login Button)
  + Signup button for new users
  + Hero image or banner promoting key features of the platform
  + Footer with links to privacy policy, terms, etc.
* **Key Features**:
  + "Create a New Account" option for new users (Farmers/Manufacturers)
  + Brief description of the platform and benefits

**2. Product Catalog Page**

* **Sections**:
  + Search bar at the top for searching products by name/category
  + Filters (Price range, Product category: seeds, fertilizers, pesticides)
  + Product listing in a grid view (Product image, name, price, add to cart button)
* **Key Features**:
  + Pagination at the bottom
  + Sort options (by price, rating, availability)
  + "Add to Buy Later" and "Add to Cart" buttons for each product

**3. Product Detail Page**

* **Sections**:
  + Product image, name, and detailed description
  + Price, availability, and quantity selection
  + Add to cart and Add to Buy Later buttons
  + User reviews and ratings
* **Key Features**:
  + Related product suggestions below
  + Review form for customers who purchased the product

**4. Shopping Cart Page**

* **Sections**:
  + List of items added to the cart (Product name, quantity, price)
  + Summary section with subtotal, taxes, and total amount
  + Checkout button
* **Key Features**:
  + Option to edit item quantity or remove items from the cart
  + Coupon code input field (if applicable)

**5. Checkout & Payment Page**

* **Sections**:
  + Shipping address form (Name, Address, City, Zip code, Phone number)
  + Payment method selection (Credit/Debit card, UPI, COD)
  + Order summary (Items, Price, Delivery charges)
  + Place order button
* **Key Features**:
  + Payment gateway integration for processing payments
  + "Save Address" option for future orders

You can use tools like **Balsamiq**, or **Microsoft Visio** to create these wireframes.

Microsoft Visio is a diagramming and vector graphic application used to create diagrams and flowchart and other visual representation of complex information.

Balsamiq is a rapid wireframing tool used to create mockups and prototypes of user interface

Axure is more advanced prototyping tool used to create high fidelity interactive wireframe and prototype for web and mobile application

Question 3 – Tools (Visio, Balsamiq) - 15 Marks Make a note of the Tools, which you are using for above concepts.

**Question 3**, you are required to list the tools used for wireframes, diagrams, and prototypes. Here's a detailed note of tools commonly used for such tasks:

**Tools for Wireframing, Prototyping, and Diagram Creation:**

1. **Balsamiq**
   * **Purpose**: Wireframe creation
   * **Description**: Balsamiq is a user-friendly wireframing tool that helps create low-fidelity mockups. It allows designers and BAs to quickly sketch out ideas and create simple, intuitive page layouts.
   * **Usage**: You can use Balsamiq to design the wireframes for pages like the **Homepage**, **Product Catalog**, and **Shopping Cart** for this project.
2. **Microsoft Visio**
   * **Purpose**: Use Case and Activity Diagrams
   * **Description**: Visio is a powerful tool for creating flowcharts, UML diagrams, network diagrams, and data visualizations. It’s excellent for creating **Use Case Diagrams**, **Activity Diagrams**, and other visual representations of system processes.
   * **Usage**: You can use Visio to design **UML diagrams**, **ER diagrams**, and **Data Flow Diagrams** for this project. This will help in visualizing how different parts of the system interact.
3. **Axure RP**
   * **Purpose**: Prototyping and Wireframes
   * **Description**: Axure RP is another powerful tool for creating interactive wireframes and prototypes. It allows the creation of complex, high-fidelity prototypes with dynamic content and interactions.
   * **Usage**: Axure RP can be used to create more advanced wireframes and clickable prototypes, helping stakeholders visualize the product functionality and flow.

**Example of Tool Usage:**

* **Wireframes**: Created using **Balsamiq**
* **Use Case and Activity Diagrams**: Created using **Microsoft Visio**.
* **Prototypes**: Designed with **Axure RP**.
* **Data Flow Diagrams and ER Diagrams**: Created using **MS visio**

These tools will help you complete the required wireframes, diagrams, and prototypes for the project

Question 4 – RTM - 6 Marks A business analyst’s key responsibilities are to keep track of the requirements and make sure that no requirement is missed. Mr. Henry and peter have approached you regarding the current status of the project. How will you tackle this situation? Prepare RTM Req ID Req Name Req Description Design D1 T1 …… T4 UAT FR0001 Farmer Registration Farmers should be able to register with the application FR0002 Farmer Search for Products Farmers should be able to search for available products in fertilizers, seeds, pesticides NFR0101 Page Loading Time Each Page should load within 2 seconds time NFR0102 WCAG 2.1. The system must meet Web Content Accessibility Guidelines WCAG 2.1.

ensure that all requirements are tracked throughout the project lifecycle and that no requirement is missed during development, testing, or deployment.

**Requirements Traceability Matrix (RTM)**

The RTM can be structured like this:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Req ID** | **Req Name** | **Req Description** | **Design (D1)** | **Test Cases (T1 to T4)** | **UAT (User Acceptance Testing)** |
| **FR0001** | Farmer Registration | Farmers should be able to register with the application | Completed | T1, T2 | Pending |
| **FR0002** | Farmer Search for Products | Farmers should be able to search for available products (fertilizers, seeds, pesticides) | In Progress | T1, T3 | Pending |
| **FR0003** | Manufacturer Registration | Manufacturers should be able to register with the application | Completed | T1, T2 | Pending |
| **FR0004** | Product Upload | Manufacturers should be able to upload and display products | In Progress | T1, T4 | Pending |
| **FR0005** | Add to Cart | Farmers should be able to add products to their shopping cart | Completed | T2, T3 | Pending |
| **NFR0101** | Page Lo  ading Time | Each page should load within 2 seconds time | In Progress | T1, T4 | Pending |
| **NFR0102** | WCAG 2.1 Accessibility | The system must meet Web Content Accessibility Guidelines WCAG 2.1 | In Progress | T2, T3 | Pending |

**Explanation:**

* **Req ID**: Unique identifier for each requirement.
* **Req Name**: Short title of the requirement (e.g., Farmer Registration).
* **Req Description**: A brief description of what the requirement entails.
* **Design (D1)**: The status of the design phase for that requirement (e.g., Completed, In Progress).
* **Test Cases (T1 to T4)**: The test cases that verify this requirement. You should list the test cases (T1, T2, etc.) that correspond to the requirement.
* **UAT**: Status of User Acceptance Testing (e.g., Pending, Completed).

**Status Reporting:**

* **Design (D1)** tracks whether the design for the requirement has been completed.
* **Test Cases (T1-T4)** reflect which test cases validate each requirement, ensuring that it functions correctly.
* **UAT** tracks if the requirement has been verified by the end user in the User Acceptance Testing phase.

By preparing the RTM, you can show Mr. Henry and Peter the current status of each requirement, ensuring transparency and making sure no requirement is overlooked.

**Question 5**, you need to create 10 test case documents. A test case document helps ensure that each functionality is thoroughly tested, and the expected results are met. Below is a template for creating test cases along with 10 sample test cases for the online agriculture product store project.

**Test Case document is a detailed outline used by tester to ensure that a software application or system is working as expected**

* **Test Case ID**: Unique ID for each test case.
* **Test Case Name**: Brief description of what is being tested.
* **Preconditions**: Any conditions that must be met before the test is executed.
* **Test Steps**: The sequence of actions to be performed during the test.
* **Expected Result**: The outcome that is expected if the functionality works as intended.
* **Actual Result**: The actual outcome after executing the test.
* **Status**: Pass/Fail.
* **Comments**: Any additional information.

**Test Cases:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID**: TC001 |  |  | **Test Case Name**: Farmer Registration Process |  |  |
| Project ID | 1 |  | Project name | Farmer Registration Process |  |
| PM ID | 5086527 |  | PM | GG |  |
| **Preconditions**: Farmer should have access to the registration page. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Navigate to the registration page. |  |  |  |  |  |
| 2. Enter valid email, password, and other required details. |  |  |  |  |  |
| 3. Click on the "Register" button. |  |  |  |  |  |
| **Expected Result**: User should be successfully registered and redirected to the homepage. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |
|  |  |  |  |  |  |
| **Test Case 2: Farmer Login** |  |  | **Test Case Name**: Login for Existing Farmer |  |  |
| **Test Case ID**: TC002 |  |  |  |  |  |
| Project ID | 2 |  | Project name |  |  |
| PM ID | 5086527 |  | PM |  |  |
| **Preconditions**: Farmer must be registered. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Navigate to the login page. |  |  |  |  |  |
| 2. Enter valid email and password. |  |  |  |  |  |
| 3. Click on the "Login" button. |  |  |  |  |  |
| **Expected Result**: Farmer should be successfully logged in and redirected to the product catalog. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Test Case 3: Search for Products** |  |  | **Test Case Name**: Search Products by Name |  |  |
| **Test Case ID**: TC003 |  |  |  |  |  |
| Project ID | 3 |  | Project name |  |  |
| PM ID | 5086527 |  | PM |  |  |
| **Preconditions**: User should be logged in. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Navigate to the product catalog page. |  |  |  |  |  |
| 2. Enter product name (e.g., "Fertilizer") in the search bar. |  |  |  |  |  |
| 3. Click on the "Search" button. |  |  |  |  |  |
| **Expected Result**: A list of products matching the search term should be displayed. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Test Case 4: Add to Cart** |  |  | **Test Case Name**: Add Products to Cart |  |  |
| **Test Case ID**: TC004 |  |  |  |  |  |
| Project ID | 4 |  | Project name |  |  |
| PM ID | 5086527 |  | PM |  |  |
| **Preconditions**: User should be logged in and browsing the product catalog. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Select a product from the catalog. |  |  |  |  |  |
| 2. Click on the "Add to Cart" button. |  |  |  |  |  |
| **Expected Result**: The selected product should be added to the shopping cart. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Test Case 5: Product Checkout** |  |  | **Test Case Name**: Product Checkout Process |  |  |
| **Test Case ID**: TC005 |  |  |  |  |  |
| Project ID | 5 |  | Project name |  |  |
| PM ID | 5086527 |  | PM |  |  |
| **Preconditions**: User should have items in the cart. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Go to the shopping cart page. |  |  |  |  |  |
| 2. Review the cart and click on the "Checkout" button. |  |  |  |  |  |
| 3. Enter shipping details and select payment method. |  |  |  |  |  |
| 4. Complete the payment. |  |  |  |  |  |
| **Expected Result**: Order should be successfully placed and confirmation email sent. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Test Case 6: Payment Gateway Integration** |  |  | **Test Case Name**: Payment via UPI |  |  |
| **Test Case ID**: TC006 |  |  |  |  |  |
| Project ID | 6 |  | Project name |  |  |
| PM ID | 5086527 |  | PM |  |  |
| **Preconditions**: User should be checking out with valid products. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Select UPI as the payment method. |  |  |  |  |  |
| 2. Enter UPI ID. |  |  |  |  |  |
| 3. Complete the payment process. |  |  |  |  |  |
| **Expected Result**: Payment should be successfully processed, and order placed. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Test Case 7: Order Confirmation Email** |  |  | **Test Case Name**: Order Confirmation Email Sent |  |  |
| **Test Case ID**: TC007 |  |  |  |  |  |
| Project ID | 7 |  | Project name |  |  |
| PM ID | 5086527 |  | PM |  |  |
| **Preconditions**: An order has been placed. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Complete an order. |  |  |  |  |  |
| 2. Check the registered email account for confirmation. |  |  |  |  |  |
| **Expected Result**: Confirmation email with order details should be received. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Test Case 8: View Order History** |  |  | **Test Case Name**: View Transaction History |  |  |
| **Test Case ID**: TC008 |  |  |  |  |  |
| Project ID | 8 |  | Project name |  |  |
| PM ID | 5086527 |  | PM |  |  |
| **Preconditions**: User should be logged in and have previous transactions. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Navigate to the "Order History" section. |  |  |  |  |  |
| 2. Review previous orders. |  |  |  |  |  |
| **Expected Result**: The user’s previous orders should be displayed. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Test Case 9: Page Load Time** |  |  | **Test Case Name**: Page Load Time Performance |  |  |
| **Test Case ID**: TC009 |  |  |  |  |  |
| Project ID | 9 |  | Project name |  |  |
| PM ID | 5086527 |  | PM |  |  |
| **Preconditions**: Access any page of the application. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Open any page (e.g., product catalog). |  |  |  |  |  |
| 2. Measure the load time. |  |  |  |  |  |
| **Expected Result**: The page should load within 2 seconds. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| **Test Case 10: WCAG Accessibility Compliance** |  |  | **Test Case Name**: WCAG 2.1 Accessibility Guidelines |  |  |
| **Test Case ID**: TC010 |  |  |  |  |  |
| Project ID | 9 |  | Project name |  |  |
| PM ID | 5086527 |  | PM |  |  |
| **Preconditions**: Access any user-facing page. |  |  |  |  |  |
| **Test Steps**: |  |  |  |  |  |
| 1. Test the page using an accessibility audit tool. |  |  |  |  |  |
| 2. Check for compliance with WCAG 2.1 guidelines. |  |  |  |  |  |
| **Expected Result**: The system should meet the required accessibility guidelines. |  |  |  |  |  |
| **Actual Result**: As per execution. |  |  |  |  |  |
| **Status**: Pass/Fail. |  |  |  |  |  |
| **Comments**: None. |  |  |  |  |  |

These test cases cover various essential functionalities of the system and ensure the quality of the application. You can expand or modify these based on the specific requirements of your project.

Question 6 – DB Design – 8 Marks After the requirements are thoroughly explained to the entire project team by business analyst, the Database architects have decided to do the database design and also to represent the in-flow and out-flow of data. Draw database schema and ER diagram

the online agriculture product store based on the project requirements. The schema will represent how data is organized and the ER diagram will illustrate the relationships between entities.

**Database Schema:**

DB schema is a blueprint that outline the structure of a database , including its table, fields, relationship, constraints and other characteristics.

Entity relationship diagram is a visual representation of the relationship between entities in a database it depicts the entities such as tables ,attributes and relationship between them

Here’s a basic structure for the database tables:

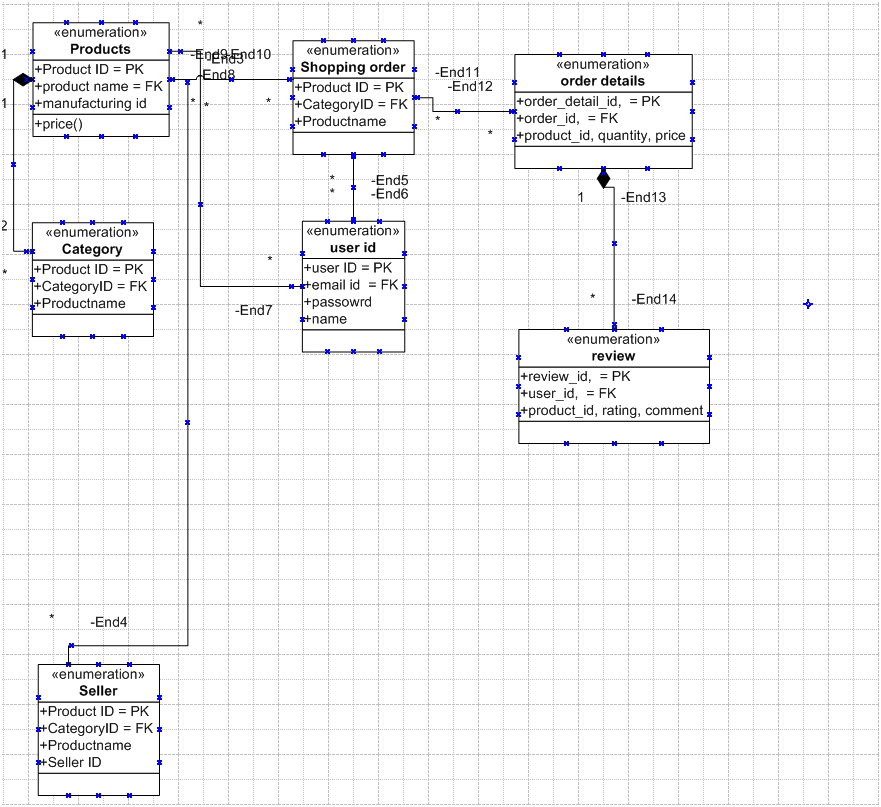
|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| user\_id | INT (PK) | Unique identifier for each user |
| Email | VARCHAR(255) | User email, used for login |
| Password | VARCHAR(255) | User password (hashed) |
| user\_type | VARCHAR(50) | Type of user (Farmer, Manufacturer) |
| Name | VARCHAR(255) | User's full name |
| Phone | VARCHAR(15) | User's contact number |
| Address | TEXT | Shipping address for farmers |
| **Table: Products** |  |  |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT (PK) | Unique identifier for each product |
| product\_name | VARCHAR(255) | Name of the product |
| Category | VARCHAR(100) | Category (Fertilizer, Seed, Pesticide) |
| Description | TEXT | Detailed product description |
| Price | DECIMAL | Price of the product |
| manufacturer\_id | INT (FK) | References user\_id in Users (manufacturer) |
| stock\_quantity | INT | Available quantity in stock |
|  |  |  |
|  |  |  |
| **Table: Orders** |  |  |
| **Column Name** | **Data Type** | **Description** |
| order\_id | INT (PK) | Unique identifier for each order |
| user\_id | INT (FK) | References user\_id in Users |
| order\_date | DATETIME | Date when the order was placed |
| total\_amount | DECIMAL | Total cost of the order |
| payment\_method | VARCHAR(50) | Payment method used (COD, UPI, Card) |
| order\_status | VARCHAR(50) | Status of the order (Pending, Shipped, Delivered) |
|  |  |  |
|  |  |  |
| **Table: Order\_Details** |  |  |
| **Column Name** | **Data Type** | **Description** |
| order\_detail\_id | INT (PK) | Unique identifier for each order item |
| order\_id | INT (FK) | References order\_id in Orders |
| product\_id | INT (FK) | References product\_id in Products |
| quantity | INT | Quantity of the product ordered |
| price | DECIMAL | Price of the product at the time of order |
|  |  |  |
|  |  |  |
| **Table: Reviews** |  |  |
| **Column Name** | **Data Type** | **Description** |
| review\_id | INT (PK) | Unique identifier for each review |
| user\_id | INT (FK) | References user\_id in Users |
| product\_id | INT (FK) | References product\_id in Products |
| rating | INT | Rating provided by the user (1-5) |
| comment | TEXT | Review comment provided by the user |
| review\_date | DATETIME | Date when the review was posted |

**ER Diagram:**

1. **Entities**:
   * Users: Represents both farmers and manufacturers.
   * Products: Represents agricultural products (fertilizers, seeds, pesticides).
   * Orders: Represents an order placed by a farmer.
   * Order\_Details: Represents the individual items in each order.
   * Reviews: Represents reviews and ratings for products.
2. **Relationships**:
   * **Users to Products**: One-to-many relationship (A manufacturer can have many products).
   * **Users to Orders**: One-to-many relationship (A farmer can place multiple orders).
   * **Orders to Order\_Details**: One-to-many relationship (An order can contain multiple products).
   * **Products to Reviews**: One-to-many relationship (A product can have many reviews).

**ER Diagram Structure:**

Here's how the relationships would look:



To represent the flow of data:

* **Manufacturers** upload product data to the **Products** table.
* **Farmers** place orders, which are stored in the **Orders** table, and each item in the order is saved in the **Order\_Details** table.
* After receiving the product, farmers can leave reviews and ratings stored in the **Reviews** table.

**Question 7 – Data Flow Diagram - 3 Marks What is a data flow diagram? Draw a data flow diagram to represent the in-flow and out-flow of data when a Farmer is placing an order for the product**

**What is a Data Flow Diagram (DFD)?**

A **Data Flow Diagram (DFD)** is a graphical representation of how data moves through a system. It shows the **flow of information** between the various components such as processes, data stores, and external entities. DFDs are used to visualize the transformation and flow of data within a system, making it easier to understand and analyze the system’s functionality.It helps analyst and designers to understand the flow of data within system . identify potential bottlenecks or inefficiency and communicate system requirements to stakeholders

**DFD Components:**

* **External Entities:** Represent the sources or destinations of data (e.g., users, other systems).
* **Processes:** Represent actions or transformations that occur within the system.
* **Data Stores:** Represent where data is stored (e.g., databases).
* **Data Flows:** Show how data moves between entities, processes, and data stores.

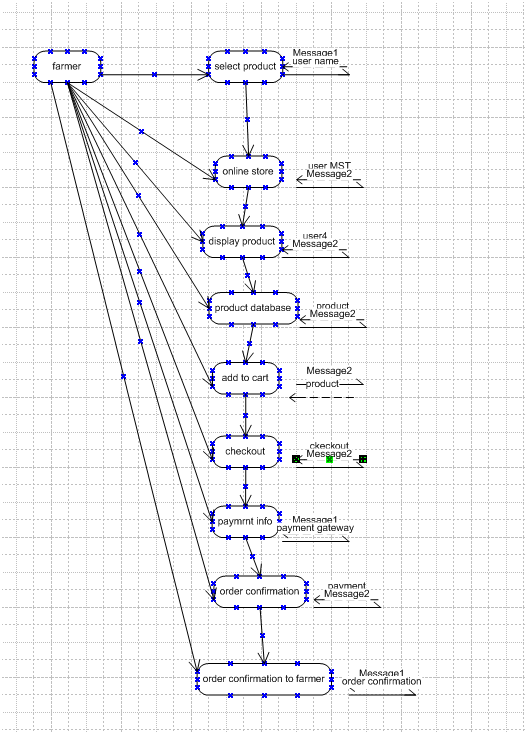
**Data Flow Diagram for Farmer Placing an Order**

This DFD represents the flow of data when a farmer places an order for a product in the online agriculture product store.

**Context Diagram (Level 0):**

* This high-level diagram shows the basic flow between the **Farmer** and the system.

plaintext

****

**Explanation of the Flow:**

1. **Select Product (Process 1):**
   * **Farmer** selects the product they want to purchase.
   * The system **displays product details** from the **Product Database**.
2. **Add to Cart (Process 2):**
   * After selecting a product, the **farmer adds it to their cart**.
   * The **cart data** is updated to reflect the selected product(s).
3. **Checkout (Process 3):**
   * Farmer proceeds to **checkout**.
   * The system requests **payment details** from the farmer and processes the payment through the **Payment Gateway**.
4. **Order Confirmation:**
   * Once payment is successfully processed, an **order confirmation** is sent to the farmer, and the order details are saved in the system.

This data flow diagram captures the interaction between the farmer, system processes, and data storage while placing an order, providing a clear understanding of the data in-flow and out-flow during the order process.

Question 8 – Change Request - 10 Marks Due to change in the Government Taxation structure . we should change the Tax structure How do you handle change requests in a project?

**Handling a Change Request in a Project**

When a change request is received, such as updating the tax structure due to changes in government policies, it is important to handle it in a structured and formal way to ensure that the project remains on track and that the impact of the change is well understood. Here's the step-by-step process to handle the **Change Request (CR)** for modifying the tax structure:

**Steps to Handle the Change Request**

**1. Document the Change Request**

* **Initiation**: The change request is initiated either by the client, a stakeholder, or based on external factors such as the change in government tax laws.
* **Details**: Collect detailed information about the change. In this case, it is the modification of the tax structure. Ensure you have all the specifics, including:
  + New tax rates.
  + Effective date of the tax change.
  + Applicable regions or product categories.

**2. Log the Change Request**

* Enter the change request into the **Change Request Log**. This log tracks all changes requested during the project.
* Assign a **unique Change Request ID** for tracking and communication purposes.
* Update the project documentation with the initial details of the request.

**3. Impact Analysis**

* Perform an **impact analysis** to understand how the change in tax structure will affect various aspects of the project, such as:
  + **Scope**: Identify which parts of the system will be affected (e.g., checkout process, payment gateway, tax calculations in the backend, invoices).
  + **Timeline**: Analyze if this change will affect the project timeline (e.g., development, testing, deployment delays).
  + **Cost**: Determine any additional costs associated with implementing the change (e.g., additional development or testing hours).
  + **Resources**: Identify whether new resources (e.g., tax consultants or developers) are required to implement the change.

**4. Consult Stakeholders**

* Review the change request with key stakeholders, including:
  + The client (e.g., Mr. Henry).
  + Project manager (e.g., Mr. Vandanam).
  + Financial head (e.g., Mr. Pandu) to verify the correctness of the tax update.
* **Discuss the impact** and gather stakeholder approval for proceeding with the change.

**5. Update Project Plans**

* If the change is approved, update all relevant project documents, such as:
  + **Project plan**: Revise timelines, milestones, and tasks to include the tax structure change.
  + **Requirement documents**: Update business and functional requirements to reflect the new tax rules.
  + **Budget**: Update the project budget to account for any cost implications of the change.

**6. Approve the Change**

* Once the stakeholders have agreed, formally approve the change request. This typically requires:
  + Approval from the **client** and **project sponsor**.
  + A sign-off from key stakeholders to document their consent to the changes and potential impacts.

**7. Implement the Change**

* Assign the task of implementing the tax structure change to the development team.
* Ensure the following:
  + Update the **checkout process** to calculate the correct tax based on the new structure.
  + Adjust the **invoices and billing** systems to reflect the new tax rates.
  + Ensure all affected data flows and business logic are modified accordingly.

**8. Test the Change**

* **Test the implementation** thoroughly before deployment:
  + Ensure that tax calculations are correct for various scenarios (e.g., different product categories, regions, discount conditions).
  + Perform **User Acceptance Testing (UAT)** with the client to verify that the system is calculating taxes correctly and is compliant with new tax laws.
  + Confirm the **accuracy of invoices** and payment details after applying the new tax rates.

**9. Deploy the Change**

* Once testing is complete and the client has approved the change, deploy the new tax structure to the **live environment**.
* Ensure that the change is deployed on the agreed-upon **effective date** (i.e., the date the new tax law takes effect).

**10. Communicate the Change**

* Inform all relevant stakeholders that the change has been successfully implemented.
* Update any user manuals or documentation to reflect the new tax structure.
* Provide **support and training** to users if necessary, explaining how the change affects the system's functionality (e.g., how taxes are displayed on invoices).

**Sample Change Request Form**

| **Change Request ID** | **CR001 - Update Tax Structure** |
| --- | --- |
| **Requester** | Government Mandate (Taxation) |
| **Description** | Modify the tax calculation structure in the system to reflect new government tax rates starting from [Date]. |
| **Priority** | High |
| **Impacted Areas** | Payment System, Checkout Process, Invoicing |
| **Impact** | The new tax law affects all product categories (fertilizers, seeds, pesticides). |
| **Estimated Effort** | 40 Development Hours, 20 Testing Hours |
| **Cost Impact** | Potential increase in project cost due to additional development and testing. |
| **Status** | Pending Approval |
| **Approval** | Client Sign-off Required |

**Key Points for Handling the Change Request**

* **Formal Process**: Always handle change requests formally to avoid scope creep or project mismanagement.
* **Impact Assessment**: Understand how the change will affect the scope, cost, and timeline before proceeding.
* **Stakeholder Involvement**: Keep stakeholders informed and involved throughout the process.
* **Testing & Implementation**: Ensure thorough testing before deployment to avoid issues post-implementation.

By following these steps, you ensure the change request is handled smoothly, minimizing disruption to the project while maintaining transparency and stakeholder satisfaction.

Question 9 – Change Request Vs an Enhancement - 5 Marks As the project is in process, Ben and Kevin have contacted you. The reason is to inform you that they want the Farmers to sell their crop yields through this application i.e. Farmers should be able to add their crop yields or products and display to general public and should be able to sell them. They also want to introduce Auction system for their Crop yields. As a BA, what will be your response? Is this a change request or an enhancement???

**Change Request vs. Enhancement**

In the scenario where Ben and Kevin have requested additional functionality for **farmers to sell their crop yields** and **introduce an auction system**, it’s important to distinguish whether this request is a **change request** or an **enhancement**.

**1. Change Request Definition:**

A **change request** refers to a modification to existing functionality that has already been defined or implemented. It typically involves correcting an issue, altering a current feature, or updating the project scope due to unforeseen circumstances.

**2. Enhancement Definition:**

An **enhancement** refers to adding new features or functionalities that were not part of the original project scope. Enhancements usually expand the system’s capabilities, provide additional value, or improve the user experience beyond the initial requirements.

**Analysis of the Request**

The original scope of the project was to facilitate farmers in **purchasing agricultural products** like fertilizers, seeds, and pesticides. The new request from Ben and Kevin introduces entirely **new functionality** for:

* Allowing **farmers to sell their crop yields** through the platform.
* Adding an **auction system** for selling crop yields to the general public.

These features were **not part of the initial project scope** and significantly alter the functionality and user base of the application. The system was initially designed to allow farmers to buy products; now, it would need to accommodate both buying and selling functions, along with the complexity of auction management.

**My Response as a Business Analyst:**

As a Business Analyst, I would classify this request as an **enhancement** for the following reasons:

* **New Functionality**: The ability for farmers to sell their crops and introduce an auction system is **new functionality** that was not part of the original requirements.
* **Expansion of Scope**: This request significantly expands the project’s scope, shifting it from a product purchasing platform to a marketplace where farmers can also **sell** their goods.
* **Additional Development**: The development team will need to build new modules to handle crop sales, auctions, payment processes, and public product listings, which were not initially planned.

**Steps to Handle the Enhancement Request:**

1. **Document the Enhancement**:
   * Clearly document the new features requested: **crop yield sales** and an **auction system**.
2. **Impact Analysis**:
   * Conduct an impact analysis to understand how this enhancement will affect:
     + **Timeline**: This will likely extend the project timeline due to additional development and testing.
     + **Cost**: More resources will be needed for design, development, testing, and implementation.
     + **Scope**: This enhancement changes the project’s overall scope from a purchasing platform to a full marketplace.
3. **Stakeholder Consultation**:
   * Present the impact analysis to the client (Mr. Henry) and other stakeholders (e.g., Peter, Mr. Pandu) to gather their approval for the new functionality and ensure that the additional scope and cost are agreed upon.
4. **Update Project Plan**:
   * If the enhancement is approved, the project plan must be updated to reflect the new requirements, timelines, resources, and deliverables.
5. **Design and Implementation**:
   * Collaborate with the technical team to design and implement the new selling and auction features, ensuring they are integrated smoothly with the existing system functionalities.

**Conclusion:**

This request is clearly an **enhancement**, as it introduces new functionality that was not part of the original project scope. By following a structured process of documenting, analyzing, and consulting with stakeholders, I would ensure that this enhancement is properly managed and integrated into the project.

Question 10 – Estimations - 6 Marks Come up with estimations – How many Manhours required

**Estimating Manhours for the Project**

To estimate the **manhours** required for implementing the project, we need to break the project down into phases and tasks. The major activities involved in this project (which now includes the original scope as well as the new enhancement for farmers to sell crop yields and introduce an auction system) are as follows:

**1. Requirements Gathering and Analysis**

* **Tasks:**
  + Initial meetings with stakeholders (e.g., Mr. Henry, Ben, Kevin).
  + Gather functional and non-functional requirements.
  + Document business requirements and user stories.
  + Conduct stakeholder analysis.
  + Impact analysis for enhancement (crop yield sales and auction system).
* **Estimated Manhours:** 40 manhours

**2. Design Phase**

* **Tasks:**
  + Design wireframes for user interfaces (product catalog, auction page, etc.).
  + Create UML diagrams (use cases, activity diagrams, sequence diagrams).
  + Database design (ER diagrams, schema design for product and auction databases).
  + Create data flow diagrams for the sales and auction process.
* **Estimated Manhours:** 50 manhours

**3. Development Phase**

* **Tasks:**
  + Frontend development (pages for product catalog, registration, cart, auction).
  + Backend development (API integration, database interaction for products and auctions).
  + Implement business logic for tax changes, crop sales, auction system.
  + Implement payment integration for the new sales and auction functionalities.
  + Develop order tracking and confirmation features.
* **Estimated Manhours:** 120 manhours

**4. Testing Phase**

* **Tasks:**
  + Unit testing of individual modules.
  + Integration testing to ensure proper interaction between modules (product search, sales, auctions, payments).
  + Sxgv-User Acceptance Testing (UAT) with stakeholders.
  + Performance testing (e.g., loading time, system response).
  + Security testing for payments and auction data.
* **Estimated Manhours:** 50 manhours

**5. Deployment and Go-Live**

* **Tasks:**
  + Setup and deploy the application on production servers.
  + Ensure smooth transition from testing to production.
  + Setup auction and sales-related database entries.
  + Conduct post-deployment testing (verify all services are working).
* **Estimated Manhours:** 20 manhours

**6. Project Management and Coordination**

* **Tasks:**
  + Schedule and coordinate meetings with stakeholders.
  + Weekly project status updates to client and team.
  + Documentation of changes, decisions, and updates.
  + Coordinate UAT and feedback sessions.
* **Estimated Manhours:** 30 manhours

**Total Estimated Manhours:**

* **Requirements Gathering and Analysis:** 40 manhours
* **Design Phase:** 50 manhours
* **Development Phase:** 120 manhours
* **Testing Phase:** 50 manhours
* **Deployment and Go-Live:** 20 manhours
* **Project Management and Coordination:** 30 manhours

**Grand Total: 310 manhours**

**Assumptions:**

1. The development team is familiar with the technologies being used, and no major learning curve is involved.
2. There are no major delays in receiving feedback or approvals from stakeholders.
3. The estimation assumes a **medium complexity** for the implementation of the auction system and crop sales functionality.
4. The estimations account for both the original scope and the new enhancement requested.

These manhours give a comprehensive view of the time required for the successful completion of the project, including both the original product-purchasing system and the new farmer crop yield sales and auction functionality.

Question 11 – UAT – 6 Marks Project has finally completed all the stages i.e., design, development, testing etc. Now, it is the role of a business analyst to contact the client for testing of the final product and have to successfully complete it. How are you going to handle this situation? And once it is done, what will be the process to close the project? Explain UAT Acceptance process

**Handling the User Acceptance Testing (UAT) and Project Closure**

As the Business Analyst, once the project has completed all stages—**design, development, and testing**—it is time to initiate **User Acceptance Testing (UAT)** with the client. UAT ensures that the final product meets the client’s expectations and business requirements before it goes live.

**Steps to Handle the UAT Process:**

**1. Prepare the UAT Plan**

* **Create a UAT Test Plan**: Define the scope of UAT, key functionalities to be tested, and the testing timeline.
* **Test Scenarios and Scripts**: Develop detailed test scenarios and scripts based on the business requirements, ensuring that all use cases and workflows are covered.
* **Select Testers**: Coordinate with the client to identify the UAT participants (e.g., Mr. Henry, Ben, Kevin) who will execute the tests.
* **Environment Setup**: Ensure that a UAT environment is prepared, which mirrors the production environment. Data related to orders, products, payment methods, and auctions must be realistic but not affect the production environment.

**2. Communicate with the Client**

* **Schedule UAT**: Reach out to the client (e.g., Mr. Henry, key stakeholders) to schedule UAT sessions.
* **Provide Guidelines**: Share UAT test cases, guidelines, and instructions with the client so they know how to execute the test cases and provide feedback.
* **Clarify the Objectives**: Explain that the goal of UAT is to ensure that the system meets the client’s expectations, identifies any gaps, and confirms the readiness of the system for launch.

**3. Coordinate UAT Execution**

* **Oversee UAT Sessions**: Be available during UAT sessions to assist the client and stakeholders if they encounter any issues or have questions about the system.
* **Gather Feedback**: Capture real-time feedback from the testers on the system’s performance, usability, and whether it meets the business requirements.
* **Log Defects or Issues**: If any issues or bugs are discovered during UAT, log them in a **defect tracking system** and assign them for resolution by the development team.

**4. Resolve Issues**

* **Issue Resolution**: Work with the development and testing teams to fix any issues raised during UAT promptly. Ensure that all critical defects are resolved.
* **Retest if Necessary**: Once the issues are resolved, conduct a retest to ensure that the fixes have been properly implemented without introducing new problems.

**5. Obtain UAT Sign-off**

* Once all the test cases are executed and the client is satisfied with the system’s functionality, request **formal sign-off** from the client.
* A **UAT sign-off document** should be prepared, confirming that the system meets all the agreed-upon business requirements and is ready for deployment.

**UAT Acceptance Process**

1. **Preparation**: Ensure UAT scenarios, test cases, and environment are ready.
2. **Execution**: Client and stakeholders perform tests using the system.
3. **Feedback Collection**: Log defects and gather feedback on system performance.
4. **Issue Resolution**: Fix any issues raised during UAT and retest.
5. **Sign-off**: Obtain formal approval from the client, signaling acceptance.

**Example UAT Sign-off Document:**

| **Client Name** | **SOONY Company (Mr. Henry, Ben, Kevin)** |
| --- | --- |
| **Project Name** | Online Agriculture Product Store |
| **UAT Completion Date** | [Insert Date] |
| **Scope** | Final system testing including product catalog, checkout, payment, crop yield sales, and auction functionalities. |
| **Result** | All UAT test cases have been executed and passed successfully. |
| **Sign-off** | [Client Signature] |

**Steps for Project Closure After UAT**

**1. Final Deployment**

* Once UAT is successfully completed, coordinate with the technical team to deploy the system to the **live production environment**.
* Ensure that all features, including new enhancements (e.g., crop yield sales and auctions), are working as intended.

**2. Final Documentation**

* Prepare and deliver all final project documentation to the client, including:
  + UAT results and sign-off documents.
  + User manuals and training materials for end users.
  + Technical documentation for system maintenance.

**3. Knowledge Transfer and Training**

* Provide training to the client’s users and administrators, ensuring they are comfortable using the system.
* Offer knowledge transfer sessions to the client’s IT team to manage the system post-deployment.

**4. Project Closure Meeting**

* Organize a project closure meeting with the client and internal team to formally close the project.
* Discuss the project’s successes, lessons learned, and any final feedback from the client.
* Ensure all financial matters, such as project billing and invoices, are settled.

**5. Project Closure Document**

* Prepare a **Project Closure Document** that summarizes the project, its deliverables, any deviations, final client sign-off, and lessons learned.
* Ensure all stakeholders sign off on the **Project Closure Document**.

**Example Project Closure Document:**

| **Project Name** | **Online Agriculture Product Store** |
| --- | --- |
| **Client** | SOONY Company (Mr. Henry) |
| **Start Date** | [Insert Start Date] |
| **Completion Date** | [Insert Completion Date] |
| **Scope** | Development of an online platform for farmers to purchase and sell products. |
| **Deliverables** | Complete system, user manuals, training materials. |
| **Sign-off** | [Client Signature] |

**Summary of the UAT Acceptance Process:**

* **Contact the client** for UAT planning and execution.
* **Ensure that all test cases** cover the key system functionalities.
* **Resolve issues** raised during UAT and conduct a retest.
* Obtain **formal UAT sign-off** to confirm that the system meets the client’s needs.
* Complete the project with **final deployment**, documentation delivery, and training before obtaining final **project closure sign-off**.

This ensures a smooth transition from testing to production and successful closure of the project.

Question 12 – Project Closure Document - 6 Marks Explain Project closure document

**Project Closure Document**

A **Project Closure Document** is a formal record that marks the end of a project, confirming that all deliverables have been completed, objectives have been met, and the client has accepted the final product. It serves as an official sign-off for project completion and ensures that both the project team and client agree on the outcomes. This document also summarizes the project, highlights key achievements, and records any lessons learned for future reference.

**Key Components of a Project Closure Document**

**1. Project Overview**

* **Project Name**: Clearly state the name of the project.
* **Client Name**: Mention the client or stakeholder for whom the project was developed.
* **Project Start Date**: Include the date when the project was initiated.
* **Completion Date**: Record the final date when the project was completed.
* **Project Manager**: Include the name of the project manager responsible for overseeing the project.
* **Project Objective**: Provide a brief overview of the project’s objectives and goals (e.g., developing an online agriculture store for purchasing and selling agricultural products).

**2. Project Deliverables**

* **Completed Deliverables**: List all deliverables that were produced during the project (e.g., product catalog, order tracking, auction system).
* **Quality Standards**: Mention any quality standards or criteria that the deliverables had to meet (e.g., compliance with Web Content Accessibility Guidelines).
* **Acceptance Criteria**: Reference the acceptance criteria agreed upon with the client and confirm that all criteria have been met.

**3. Milestones Achieved**

* **Key Milestones**: Provide a list of significant milestones completed throughout the project lifecycle (e.g., design phase, development phase, UAT sign-off, final deployment).
* **Timeline**: A summary of the project timeline, including any significant adjustments or extensions during the project.

**4. Client Sign-off**

* **Client Approval**: Confirm that the client has reviewed and accepted all deliverables. Include a section for the client’s formal sign-off, which is a critical aspect of project closure.
* **Sign-off Date**: Record the date when the client officially approved and accepted the final product.
* **Client’s Signature**: Provide space for the client’s signature to formalize acceptance.

**5. Lessons Learned**

* **Project Successes**: Highlight the key successes of the project, including any particular achievements that contributed to the smooth running or positive outcomes of the project.
* **Challenges Encountered**: Document any issues or challenges faced during the project, such as scope changes, technical difficulties, or stakeholder misalignment.
* **Recommendations for Future Projects**: Offer suggestions for future projects based on the lessons learned, which could help improve project management processes or client communication.

**6. Final Project Cost and Budget**

* **Initial Budget**: Provide the original budget allocated for the project.
* **Final Cost**: Compare the final cost with the initial budget and explain any differences, such as additional expenses incurred due to scope changes or delays.
* **Cost Breakdown**: Offer a detailed breakdown of costs for each phase or major task (e.g., design, development, testing).

**7. Post-Project Support and Maintenance**

* **Support Plan**: Outline any agreed-upon post-project support or maintenance services (e.g., technical support for six months, bug fixes, system updates).
* **Responsibilities**: Clearly state who will be responsible for ongoing support (e.g., the client’s internal team or the project team).

**8. Project Documentation**

* **Final Documentation**: Mention all the final documents that were handed over to the client, such as user manuals, technical guides, and project reports.
* **Training Materials**: If applicable, provide a list of training materials or sessions delivered to the client to help them operate or manage the system.

**9. Sign-off and Approvals**

* **Client Approval**: A section where the client formally approves the project closure.
* **Internal Team Sign-off**: Approval from the project manager and key team members, indicating that the project has been completed successfully.

**Example Project Closure Document**

| **Section** | **Details** |
| --- | --- |
| **Project Name** | Online Agriculture Product Store |
| **Client** | SOONY Company (Mr. Henry) |
| **Project Start Date** | January 1, 2023 |
| **Project Completion Date** | March 31, 2024 |
| **Project Manager** | Mr. Vandanam |
| **Objective** | Develop an online platform for farmers to buy and sell agricultural products, including a product catalog, crop yield sales, and an auction system. |

| **Deliverables** | **Status** |
| --- | --- |
| Product Catalog | Completed |
| Payment Integration | Completed |
| Order Tracking | Completed |
| Auction System | Completed |

| **Lessons Learned** | **Comments** |
| --- | --- |
| Successes | Strong communication with stakeholders; on-time delivery of major milestones. |
| Challenges | Scope changes related to adding auction functionality caused delays in development. |
| Recommendations | Implement a more structured change request process to manage future scope changes efficiently. |

| **Final Budget** | **Initial Estimate** | **Actual Cost** |
| --- | --- | --- |
| INR 2 Crores | INR 2.1 Crores | Slight increase due to scope changes (auction functionality). |

| **Post-Project Support** | **Details** |
| --- | --- |
| Support Period | 6 months post-deployment |
| Support Coverage | Bug fixes, minor updates, and customer support |

| **Client Sign-off** | **Client Name** | **Date** | **Signature** |
| --- | --- | --- | --- |
| Mr. Henry (SOONY Company) | [Insert Date] | [Insert Signature] |  |

**Importance of the Project Closure Document**

1. **Formal Closure**: It provides a formal closure to the project, ensuring all agreed deliverables have been met and accepted by the client.
2. **Client Satisfaction**: It confirms the client is satisfied with the work and agrees that the project objectives have been achieved.
3. **Lessons Learned**: By documenting lessons learned, it helps improve future projects by highlighting what worked well and what could be improved.
4. **Financial Accountability**: It confirms the project budget, any deviations, and ensures all financial matters are settled.
5. **Post-Project Handover**: It records any post-project support or maintenance agreements, ensuring a smooth transition.

The **Project Closure Document** ensures that both the project team and the client have a clear understanding that the project has been successfully completed, and all deliverables have been provided according to the requirements. It is a crucial part of project management that formalizes the project’s end and allows both parties to move forward confidently.