**Nurturing Process - Capstone Project1 – Part -3/3**

**Question 1 – Functional Requirements (15 Marks)**

**Answer:**

Functional requirements define the specific behaviour or functionality of the system. Here is a list of **20 functional requirements** for the Online Agriculture Products Store:

| **Req ID** | **Requirement Name** | **Requirement Description** | **Priority** |
| --- | --- | --- | --- |
| **FR001** | **Farmer Registration** | Farmers should be able to register with the application using email and password. | 9 |
| **FR002** | **Manufacturer Registration** | Manufacturers should be able to register and create accounts. | 9 |
| **FR003** | **Login & Authentication** | Both farmers and manufacturers should be able to log in securely using their credentials. | 10 |
| **FR004** | **Product Catalog** | Farmers should be able to browse products (fertilizers, seeds, pesticides). | 10 |
| **FR005** | **Product Search** | Farmers should be able to search for products based on keywords, categories, and filters. | 9 |
| **FR006** | **Product Details View** | Farmers should be able to view detailed product descriptions, images, and pricing. | 9 |
| **FR007** | **Add to Cart** | Farmers should be able to add selected products to their cart. | 10 |
| **FR008** | **Buy Later List** | Farmers should be able to save products in a "Buy Later" list for future purchases. | 8 |
| **FR009** | **Order Placement** | Farmers should be able to place an order and receive order confirmation. | 10 |
| **FR010** | **Payment Options** | System should support multiple payment options: COD, Credit/Debit cards, UPI. | 10 |
| **FR011** | **Order Tracking** | Farmers should be able to track the status of their order through a delivery tracker. | 9 |
| **FR012** | **Email & SMS Notifications** | Farmers should receive order confirmation, dispatch, and delivery notifications via email/SMS. | 8 |
| **FR013** | **Manufacturer Product Upload** | Manufacturers should be able to upload and manage their product listings. | 9 |
| **FR014** | **Inventory Management** | Manufacturers should be able to update stock availability. | 8 |
| **FR015** | **Reviews & Ratings** | Farmers should be able to leave reviews and ratings for purchased products. | 7 |
| **FR016** | **Return & Refund Process** | Farmers should be able to request returns and refunds for eligible products. | 8 |
| **FR017** | **Customer Support Chat** | System should have a support feature where farmers can raise queries or complaints. | 7 |
| **FR018** | **Admin Dashboard** | Admins should be able to monitor orders, payments, and user activity. | 9 |
| **FR019** | **Report Generation** | Admins should be able to generate sales, inventory, and user engagement reports. | 7 |
| **FR020** | **Multi-Language Support** | The application should support multiple languages for ease of use. | 6 |

**Question 2 – Minimum 5 Page Designs**

**Answer:**

Here are 5 page designs that I made using the SW Balsamiq.

1. Registration

****

1. **Login**



1. **Home Page**

****

1. **Product Landing Page**

****

1. **Payment Page**

****

**Question 3 – Tools Used**

**Answer:**

I have used MS Visio, Balsamiq and Axure Pro. Here are the details about it:

1. **MS Visio**: Microsoft Visio is a powerful diagramming tool used for creating UML diagrams, flowcharts, and process maps. It helps Business Analysts visualize Use Case Diagrams, Activity Diagrams, and Data Flow Diagrams (DFD). With an intuitive drag-and-drop interface, Visio makes it easy to design system workflows and document business processes efficiently. Its very simple to use as each component of use case or activity diagram are available and can be used by simple drag-and-drop.

Features & Benefits for Business Analysts:

UML & Flowchart Diagrams: Supports Use Case Diagrams, Activity Diagrams, Data Flow Diagrams (DFDs), and more.

Drag-and-Drop Functionality: Easily create diagrams using pre-built templates and shapes.

Integration with Microsoft Tools: Works seamlessly with Excel, Word, and PowerPoint for documentation.

Collaboration Features: Allows multiple users to work on the same diagram in real-time.

Custom Stencils & Icons: Enables users to design industry-specific process models.

Use Case in the Online Agriculture Store Project:

Use Case Diagram: Illustrates interactions between farmers, manufacturers, and the system.

Activity Diagram: Shows the step-by-step flow of a farmer purchasing products.

DFD: Represents how data moves between users, the database, and external entities.

1. **Balsamiq:** Balsamiq is a wireframing tool designed for quickly creating low-fidelity UI mockups. It is widely used by Business Analysts, UX designers, and product managers to design application layouts before development begins. The tool focuses on simplicity, making it easy to create sketches of web and mobile applications. I’ve used it for creating a wireframe for Online Agri Store. It is simple to use as it has drag and drop option available and it can create a black and white mockup of your app or website it can also be used to show the client what will be the response if you clicked a button, which can be done by adding the response to the link.

Features & Benefits for Business Analysts:

Drag-and-Drop Interface: Users can easily add UI elements like buttons, text boxes, and images.

Pre-Built Components: Includes commonly used UI components, such as navigation bars, modals, and dropdowns.

Rapid Prototyping: Enables quick iteration of design ideas before finalizing requirements.

Collaboration Support: Teams can share wireframes and get feedback instantly.

Export Options: Wireframes can be exported as PDFs or PNGs for documentation.

Use Case in the Online Agriculture Store Project:

Homepage Design: A simple layout displaying product categories and a search bar.

Product Listing Page: A wireframe showing product images, descriptions, and filters.

Checkout Page: A structured design for payment options and order summary.

1. **Axure Pro:** Axure RP is a powerful prototyping tool used for creating interactive wireframes, mockups, and functional prototypes. Unlike Balsamiq, which focuses on low-fidelity sketches, Axure RP allows for dynamic content, conditional logic, and user interactions, making it ideal for designing complex applications.

Features & Benefits for Business Analysts:

Interactive Prototyping: Allows the creation of clickable prototypes with real-world functionality.

Dynamic Panels & States: Helps simulate complex UI behaviors like dropdowns, pop-ups, and sliders.

Logic-Based Interactions: Supports conditions (e.g., "if user selects COD, show address fields").

Team Collaboration: Enables multiple stakeholders to review and comment on prototypes in real time.

Integration with Other Tools: Works well with Figma, Sketch, and Adobe tools for advanced UI/UX design.

Use Case in the Online Agriculture Store Project:

Login Page: A prototype where users enter credentials and experience real-time validation.

Shopping Cart: A fully interactive cart that updates dynamically as users add or remove products.

Order Tracking Page: A functional prototype showing real-time status updates for deliveries.

**Question 4 – Requirement Traceability Matrix (RTM)**

**Answer:**

A Requirement Traceability Matrix (RTM) is a document used to ensure that all business requirements are tracked throughout the development process. It helps in verifying that each requirement is properly implemented, tested, and validated.

| **Req ID** | **Req Name** | **Req Description** | **Design** | **Code** | **UT (Unit Testing)** | **CT (Component Testing)** | **ST (System Testing)** | **SIT** | **UAT (User Acceptance Testing)** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **FR001** | **User Registration** | **Users should be able to register themselves.** | **Complete** | **Complete** | **Complete** | **Complete** | **Complete** | **Complete** | **Incomplete** |
| **FR002** | **User Login & Authentication** | **Users should log in securely using email/password or OTP.** | **Complete** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR003** | **Product Search** | **Users should search for products using keywords or filters.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR004** | **Product Browsing** | **Users should browse products by categories and tags.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR005** | **Product Details Page** | **Users should view product details, images, and descriptions.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR006** | **Add to Cart** | **Users should add products to the cart before purchase.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Complete** | **Incomplete** |
| **FR007** | **Cart Management** | **Users should remove/edit items in the cart.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR008** | **Checkout & Payment** | **Users should complete purchases using multiple payment options.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR009** | **Order Tracking** | **Users should track their orders and receive updates.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR010** | **User Profile Management** | **Users should update personal details and passwords.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR011** | **Seller Registration** | **Farmers should register as sellers to list their products.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR012** | **Product Listing for Farmers** | **Farmers should be able to add product details for selling.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR013** | **Auction Feature for Farmers** | **Farmers should be able to auction their crops.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR014** | **Bid Management** | **Buyers should be able to place bids on auctioned products.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR015** | **Notification System** | **Users should receive notifications for bids, purchases, and deliveries.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR016** | **Reviews & Ratings** | **Buyers should be able to rate and review products.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR017** | **Discount & Coupon System** | **Users should apply discount codes during checkout.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR018** | **Order Cancellation & Refund** | **Users should be able to cancel orders and request refunds.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR019** | **Tax Calculation** | **The system should apply taxes based on government rules.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |
| **FR020** | **Customer Support Chat** | **Users should chat with support for queries or complaints.** | **Complete** | **Complete** | **Complete** | **Incomplete** | **Incomplete** | **Incomplete** | **Incomplete** |

How to Use RTM in the Project?

1. Ensures Complete Coverage – Every requirement is linked to design, development, testing, and UAT.
2. Tracks Changes – Helps in impact analysis if requirements are modified.
3. Validates Testing – Ensures all features are properly tested before UAT.
4. Client Communication – Helps answer stakeholder queries about project progress.

**Question 5 – 10 Test Case Documents**

**Answer:**

**Here are 10 test case documents:**

**Test Case 1: Search for a Product**

| **Test Case ID** | **TC001** | **Test Case Name** | **Search Functionality** |
| --- | --- | --- | --- |
| **Project ID** | **PRJ001** | **Project Name** | **Online Shopping App** |
| **PM ID** | **PM123** | **PM Name** | **John Doe** |
| **Test Strategy ID** | **TS001** | **Tester ID** | **TST001** |
| **Test Plan ID** | **TP001** | **Tester Name** | **Jane Smith** |
| **Test Schedule ID** | **SCH001** | **Date of Test** | **20-March-2025** |
| **Scenario** | **User searches for a product** |  |  |
| **Link to that page** | **Search Results Page** |  |  |
| **Input Data** | **"Blue rugged jeans"** |  |  |
| **Expected Behavior** | **Display 15 designs** |  |  |
| **Actual Behavior** | **TBD** |  |  |
| **Comments** | **TBD** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Test Case 2: Add Product to Cart**

| **Test Case ID** | **TC002** | **Test Case Name** | **Add to Cart** |
| --- | --- | --- | --- |
| **Project ID** | **PRJ001** | **Project Name** | **Online Shopping App** |
| **PM ID** | **PM123** | **PM Name** | **John Doe** |
| **Test Strategy ID** | **TS001** | **Tester ID** | **TST002** |
| **Test Plan ID** | **TP002** | **Tester Name** | **Mark Wilson** |
| **Test Schedule ID** | **SCH002** | **Date of Test** | **20-March-2025** |
| **Scenario** | **User adds an item to the cart** |  |  |
| **Link to that page** | **Product Page** |  |  |
| **Input Data** | **Select "Winter Wear - Jackets" and click "Add to Cart"** |  |  |
| **Expected Behavior** | **Item is added to the cart and confirmation message is displayed** |  |  |
| **Actual Behavior** | **TBD** |  |  |
| **Comments** | **TBD** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Test Case 3: Login Functionality**

| **Test Case ID** | **TC003** | **Test Case Name** | **Login with Credentials** |
| --- | --- | --- | --- |
| **Project ID** | **PRJ001** | **Project Name** | **Online Shopping App** |
| **PM ID** | **PM123** | **PM Name** | **John Doe** |
| **Test Strategy ID** | **TS001** | **Tester ID** | **TST003** |
| **Test Plan ID** | **TP003** | **Tester Name** | **Sarah Lee** |
| **Test Schedule ID** | **SCH003** | **Date of Test** | **20-March-2025** |
| **Scenario** | **User logs into the system** |  |  |
| **Link to that page** | **Login Page** |  |  |
| **Input Data** | **Email:** **user@example.com****, Password: 12345** |  |  |
| **Expected Behavior** | **User is successfully logged in and redirected to the homepage** |  |  |
| **Actual Behavior** | **TBD** |  |  |
| **Comments** | **TBD** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Test Case 4: User Registration**

| **Test Case ID** | **TC004** | **Test Case Name** | **Register New User** |
| --- | --- | --- | --- |
| **Project ID** | **PRJ001** | **Project Name** | **Online Shopping App** |
| **PM ID** | **PM123** | **PM Name** | **John Doe** |
| **Test Strategy ID** | **TS001** | **Tester ID** | **TST004** |
| **Test Plan ID** | **TP004** | **Tester Name** | **David Miller** |
| **Test Schedule ID** | **SCH004** | **Date of Test** | **20-March-2025** |
| **Scenario** | **User registers an account** |  |  |
| **Link to that page** | **Registration Page** |  |  |
| **Input Data** | **Name, Email, Password, Confirm Password** |  |  |
| **Expected Behavior** | **Account is created, and a verification email is sent** |  |  |
| **Actual Behavior** | **TBD** |  |  |
| **Comments** | **TBD** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Test Case 5: Apply Discount Code**

| **Test Case ID** | **TC005** | **Test Case Name** | **Discount Application** |
| --- | --- | --- | --- |
| **Project ID** | **PRJ001** | **Project Name** | **Online Shopping App** |
| **PM ID** | **PM123** | **PM Name** | **John Doe** |
| **Test Strategy ID** | **TS001** | **Tester ID** | **TST005** |
| **Test Plan ID** | **TP005** | **Tester Name** | **Emily Johnson** |
| **Test Schedule ID** | **SCH005** | **Date of Test** | **20-March-2025** |
| **Scenario** | **User applies a discount code at checkout** |  |  |
| **Link to that page** | **Checkout Page** |  |  |
| **Input Data** | **Discount Code: SAVE20** |  |  |
| **Expected Behavior** | **20% discount applied to total price** |  |  |
| **Actual Behavior** | **TBD** |  |  |
| **Comments** | **TBD** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Test Case 6: Place an Order**

| **Test Case ID** | **TC006** | **Test Case Name** | **Order Placement** |
| --- | --- | --- | --- |
| **Scenario** | **User completes the order process** |  |  |
| **Input Data** | **Selects items, enters shipping details, and completes payment** |  |  |
| **Expected Behavior** | **Order is placed and confirmation email is sent** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Test Case 7: Cancel an Order**

| **Test Case ID** | **TC007** | **Test Case Name** | **Order Cancellation** |
| --- | --- | --- | --- |
| **Scenario** | **User cancels an order before shipment** |  |  |
| **Input Data** | **Click "Cancel Order"** |  |  |
| **Expected Behavior** | **Order gets canceled, and refund is processed** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Test Case 8: Track an Order**

| **Test Case ID** | **TC008** | **Test Case Name** | **Order Tracking** |
| --- | --- | --- | --- |
| **Scenario** | **User tracks an order from the order page** |  |  |
| **Input Data** | **Order ID** |  |  |
| **Expected Behavior** | **Order status is displayed** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Test Case 9: Auction System - Bid Placement**

| **Test Case ID** | **TC009** | **Test Case Name** | **Bid Placement on Auction** |
| --- | --- | --- | --- |
| **Scenario** | **User places a bid on an auction item** |  |  |
| **Input Data** | **Bid Amount** |  |  |
| **Expected Behavior** | **Bid is placed, and user gets confirmation** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Test Case 10: Logout Functionality**

| **Test Case ID** | **TC010** | **Test Case Name** |  **Logout** |
| --- | --- | --- | --- |
| **Scenario** | **User logs out of the system** |  |  |
| **Input Data** | **Click "Logout"** |  |  |
| **Expected Behavior** | **User is redirected to login page** |  |  |
| **Result (Pass/Fail)** | **TBD** |  |  |

**Question 8 – Change Request**

**Due to change in the Government Taxation structure. we should change the Tax structure How do you handle change requests in a project?**

**Answer:**

Handling a Change Request in a Project (Due to Government Taxation Changes)

When there is a change in government taxation rules, it requires modifications in the system, impacting tax calculations, invoices, and payments. As a Business Analyst, I will follow a structured Change Request (CR) Management Process to ensure smooth implementation.

1. Identify and Document the Change

* Understand the new tax structure (e.g., GST, VAT, or other regulations).
* Identify which system components are affected (Checkout, Invoice, Reports, etc.).
* Gather inputs from Finance, Legal, and Compliance teams to confirm requirements.

Example Documentation:

"The system should now calculate tax as per the revised GST structure of 18% instead of 12% on agricultural products."

2. Impact Analysis

* Assess how the tax change affects:
	+ Product pricing and final amount at checkout
	+ Invoices and receipts generated for customers
	+ Reports and analytics in the Admin Dashboard
	+ Tax calculation logic in the payment system
* Estimate effort, cost, and resources required for implementation.

Tools Used:

* Impact Analysis Matrix
* Stakeholder Discussion Meetings

3. Get Stakeholder Approval

* Discuss findings with:
	+ Project Manager – To assess project timelines.
	+ Finance and Legal Team – To ensure compliance with regulations.
	+ Development Team – To evaluate technical feasibility.
* Prepare a Change Request Document (CRD) for approval.

Change Request Document Example:

| Field | Details |
| --- | --- |
| Change Request ID | CR-2025-002 |
| Requested By | Finance Department |
| Change Description | Update tax rate from 12% to 18% in all transactions |
| Impact Area | Checkout, Invoices, Reports |
| Priority | High |
| Estimated Effort | 30 developer hours, 10 testing hours |
| Approval Status | Pending / Approved |

4. Implement the Change

* Assign tasks to Developers for modifying tax calculation logic.
* Update Database Tables (if tax rates are stored dynamically).
* Ensure APIs and Payment Integrations reflect the new tax structure.

Development Approach:

* Code changes in pricing and checkout module
* Update invoice templates with the new tax format
* Modify reporting dashboards to align with new tax rules

5. Test and Validate the Changes

* Conduct Regression Testing to check the impact on:
	+ Order calculations
	+ Payment processing
	+ Invoice generation
* Perform User Acceptance Testing (UAT) with the Finance Team.
* Ensure that transactions comply with government audit requirements.

Testing Types:

* Unit Testing – Validate tax computation logic
* System Testing – Ensure changes do not break existing features
* UAT – Validate by Finance and Business teams

6. Deploy the Change and Monitor

* Release the update in a staging environment first.
* Deploy the update to production after stakeholder approval.
* Closely monitor initial transactions for errors or discrepancies.
* Provide support and hotfixes if issues arise post-deployment.

Deployment Checklist:

* Update Help and Support documents for customers.
* Communicate tax changes via email notifications to users.
* Validate reports for tax compliance.

Handling a Change Request properly ensures:

* Compliance with legal and financial regulations.
* Smooth integration without breaking existing features.
* Stakeholder confidence through proper impact analysis.

**Question 9 – Change Request Vs an Enhancement**

Is This a Change Request or an Enhancement?

In this case, adding the ability for Farmers to sell their crop yields and introducing an auction system would be considered an enhancement rather than a change request.

Reason:

* A change request typically involves modifying an existing feature due to defects, compliance issues, or stakeholder requests for changes in functionality.
* An enhancement, on the other hand, involves adding new features to improve the system beyond its original scope.

Since the initial application was designed for selling clothes and accessories, introducing a new category (Farmers selling crops) and a new functionality (Auction System) extends the capabilities of the application, making it an enhancement rather than a simple change request.

Steps to Handle the Enhancement as a Business Analyst:

1. Document the Requirements:
	* Gather detailed specifications from Ben and Kevin regarding how the Farmers' marketplace and Auction System should function.
	* Define user roles such as Farmers, Buyers, and Administrators and their permissions.
2. Conduct Feasibility Analysis:
	* Work with the Development Team to assess the technical feasibility of implementing these new features.
	* Identify database changes, UI/UX updates, and API integrations required for these enhancements.
3. Analyse Benefits, Risks, and Costs:
	* Evaluate the business impact of adding a new category of sellers.
	* Assess potential challenges, such as fraud prevention, payment integration, and scalability concerns.
4. Get Stakeholder Approval:
	* Present the Enhancement Proposal to project sponsors and decision-makers.
	* Ensure alignment with the overall business strategy before implementation.

Conclusion:

This request qualifies as an enhancement since it introduces new functionalities beyond the original project scope. As a Business Analyst, it is essential to document, analyse, and validate the enhancement before proceeding with development.

**Question 10 – Estimations**

**Come up with estimations – How many Manhours required**

**Answer:**

Total Manhours Estimation for the Entire Project

To estimate the manhours required for the complete project, we will break it down into key phases:

1. Requirement Gathering & Analysis
2. UI/UX Design
3. Development
4. Testing & QA
5. Deployment & Support

Each phase consists of multiple tasks, and we will allocate time accordingly.

1. Requirement Gathering & Analysis

| Task | Resources Required | Estimated Hours |
| --- | --- | --- |
| Initial meetings with stakeholders | Business Analyst (BA) | 16 hours |
| Document business requirements (BRD) | BA | 20 hours |
| Feasibility study and impact analysis | BA, Architect, Developers | 24 hours |
| Finalize functional and non-functional requirements | BA | 16 hours |
| Total for Requirement Gathering & Analysis | 76 hours |  |

2. UI/UX Design

| Task | Resources Required | Estimated Hours |
| --- | --- | --- |
| Wireframe design for all modules | UI/UX Designer | 30 hours |
| Create interactive prototypes | UI/UX Designer | 20 hours |
| Get feedback and update designs | UI/UX Designer | 16 hours |
| Design final UI mockups and assets | UI/UX Designer | 24 hours |
| Total for UI/UX Design | 90 hours |  |

3. Development

| Task | Resources Required | Estimated Hours |
| --- | --- | --- |
| Backend architecture setup | Backend Developer | 40 hours |
| Database design and schema creation | Database Engineer | 36 hours |
| API development for core functionality | Backend Developer | 60 hours |
| Frontend development (User and Admin portals) | Frontend Developer | 80 hours |
| Payment gateway and transaction handling | Backend Developer | 40 hours |
| Farmers' product listing and auction system | Backend and Frontend Developers | 50 hours |
| Admin dashboard for system monitoring | Frontend Developer | 40 hours |
| Integration with third-party services | Backend Developer | 30 hours |
| Code review and optimization | Developers | 24 hours |
| Total for Development | 400 hours |  |

4. Testing and QA

| Task | Resources Required | Estimated Hours |
| --- | --- | --- |
| Create test plan and test cases | QA Testers | 24 hours |
| Functional testing (all features) | QA Testers | 50 hours |
| Regression testing | QA Testers | 40 hours |
| Performance and load testing | QA Testers | 36 hours |
| Security testing | QA Testers | 24 hours |
| Bug fixes and retesting | Developers, QA | 50 hours |
| Total for Testing and QA | 224 hours |  |

5. Deployment and Support

| Task | Resources Required | Estimated Hours |
| --- | --- | --- |
| Staging server setup | DevOps Engineer | 12 hours |
| User Acceptance Testing (UAT) | BA, QA, Developers | 30 hours |
| Fix issues from UAT feedback | Developers | 20 hours |
| Deploy to production | DevOps Engineer | 10 hours |
| Post-deployment monitoring and hotfixes | Developers, QA | 30 hours |
| Total for Deployment and Support | 102 hours |  |

Final Estimation Summary

| Phase | Total Manhours |
| --- | --- |
| Requirement Gathering and Analysis | 76 hours |
| UI/UX Design | 90 hours |
| Development | 400 hours |
| Testing and QA | 224 hours |
| Deployment and Support | 102 hours |
| Grand Total | 892 hours |

Resource Allocation and Timeline

If we assume a team of six members (Business Analyst, UI/UX Designer, two Developers, one QA Tester, one DevOps Engineer) working eight hours per day, the estimated timeline would be:

* 892 / (6 × 8) = 18.58 working days (approximately 19 to 20 business days).

To account for unforeseen delays, stakeholder feedback, and minor revisions, the total duration may extend to 22 to 25 business days.

**Question 11 – UAT**

**Answer:**

User Acceptance Testing (UAT) Process

User Acceptance Testing (UAT) is the final stage of testing before a system goes live. It ensures that the software meets business requirements and functions as expected in real-world scenarios. The process involves multiple steps to verify and validate the system with end-users.

1. Planning

* Create a blueprint for UAT testing, defining the scope, objectives, and test criteria.
* Identify key stakeholders, including business users and testers.
* Establish minimum acceptance standards for approving the test results.

2. Designing

* Develop UAT test cases based on real-world business scenarios.
* Ensure test cases cover functionalities, workflows, and edge cases.
* Prepare test data that closely resembles actual production data.

3. UAT Testers

* Select end users or business representatives to perform testing.
* Provide training and documentation to help testers understand how to execute test cases.
* Ensure testers are familiar with expected system behavior to identify deviations.

4. Bug Fixing

* Record and track any defects or discrepancies found during testing.
* The development team works on fixing reported bugs and issues.
* Perform retesting after bug fixes to ensure resolution.

5. Sign-Off

* Once all major defects are resolved, testers confirm that the software meets business expectations.
* Stakeholders provide formal approval to proceed with the deployment.
* The system is marked ready for go-live, and final preparations for production deployment begin.

Final Takeaway

A well-structured UAT process ensures that the system is fully functional, meets business needs, and is ready for real-world usage. Proper planning, execution, and validation lead to a successful deployment with minimal post-launch issues.

**Question 12 – Project Closure Document**

Project Closure Document

Project Name: Online Agricultural Marketplace with Auction System
Project Duration: [Start Date] – [End Date]
Prepared By: Tarang Ovhal
Date: [Date of Document Preparation]

1. Project Overview

The project aimed to develop an online platform where farmers could sell their crop yields directly to consumers and participate in an auction system. The platform included features for product listing, order management, payment integration, and bidding functionality for auction-based sales.

2. Achievements

* Successfully developed and deployed the online marketplace for farmers.
* Implemented an auction system allowing competitive bidding on agricultural products.
* Integrated a secure payment gateway for seamless transactions.
* Designed a user-friendly interface for farmers and buyers.
* Ensured compliance with government taxation rules and e-commerce regulations.

3. Lessons Learned

* Clear and early requirement gathering is essential to avoid scope creep.
* Involvement of end-users (farmers) in the testing phase helped identify usability improvements.
* Continuous stakeholder communication streamlined decision-making.
* Agile methodology helped in managing changes effectively.

4. Quality Assurance

* Conducted rigorous testing, including unit testing, system testing, and UAT.
* Identified and fixed critical bugs before deployment.
* Ensured system stability, performance optimization, and compliance with business requirements.

5. Resource Utilization

* Development Team: [Number of Developers]
* Testing Team: [Number of Testers]
* Business Analysts: [Number of BAs]
* Project Manager: [Name]
* Total Man-Hours: [Estimated Man-Hours Spent]

6. Risk Management

* Addressed potential security vulnerabilities in the payment system.
* Managed timeline constraints by implementing Agile development cycles.
* Mitigated risk of low user adoption by conducting training and awareness programs for farmers.

7. Challenges

* Initial resistance from farmers in adapting to the digital platform.
* Ensuring seamless integration of the auction system with the existing platform.
* Handling real-time tax calculations based on changing government policies.
* Managing multiple stakeholder expectations while staying within project timelines.

Project Closure Approval

This document serves as the official record of project completion. All objectives have been met, and the project is now closed.

Approved By: [Client/Project Sponsor Name]
Date: [Approval Date]