Capstone Project Prep1-Part 2

Online Agriculture Products Store

# **Audits:**

1. Quarterly audits are planned each phase by phase as:

* Q1 - Requirement Gathering phase
* Q2 - Requirement analysis phase
* Q3 - Design phase and Development phase
* Q4 - Testing phase

1. Requirement Gathering phase :

* Audit will check for if all the requirement is collected from the client as per their needs and check if completed or not .
* Checklist includes BRD template Elicitation results report Duplicate requirements report grouping of functionalities/features- Client signoff.

1. Requirement analysis Phase:

* Check if all the customer requirements are analysed and worked accordingly.
* Checklist includes UML Diagrams Business to Functional Requirements mapping Client Signoff RTM Document version control.

1. Design and Development phase:

* Audit will check if software has been designed according to client need.
* Audit will check if website has been created according to customer’s requirement.
* Checklist includes Utilization of Tools Documented evidence on client communication. Stakeholder MOM.
* Checklist includes creating detailed checklist of requirement, Creating timeline and task with list of deliverables and deadlines Meeting with Project development team.

1. Testing Phase:

* Developed application will be tested and verify before handed over to real-time user.
* If any error occurs then again handed over to development team for correction.
* Audit checklist will include Meeting with testers to check on possible outcome, Discussion with QA team on the details such as automation code, where to store the automation code and who will need access to it, who's running the tests; and writing test cases Meeting with QA team to identify where the tests will run.

|  |  |
| --- | --- |
| **Q1** | **Quarterly Audit 1** |
| **Stage 1** | **Requirement gathering 15 weeks (week 1 to week 15)** |
| Completed | 10 weeks (week 1 to week 10) |
| Checklist |  |
|  | BRD template |
|  | Elicitation results reports |
|  | Duplicate requirements report |
|  | Grouping of features - client sign off |
|  | Email communication- To, CC, BCC |
|  |  |
| **Q2** | **Quaterly Audit 2** |
| **Stage 2** | **Requirement Analysis Phase 13 weeks (weeks 16 to week 29)** |
| Completed | 7 weeks (week 16 to week 23) |
| Checklist |  |
|  | UML diagrams |
|  | Business to functional requirements mapping |
|  | Client sign off - documents |
|  | RTM Document version control |
|  | Email communication- To , CC, BCC |
|  |  |
| **Q3** | **Quaterly Audit 3** |
| **Stage 3** | **Design Phase - 10 weeks (week 30 to week 40)** |
| Completed | 7 weeks (week 30 to week 37) |
| Checklist |  |
|  | Utilization of tools |
|  | Documented evidence on client communication |
|  | Stakeholder MOM |
|  | Email communication- To , CC, BCC |
| **Stage 4** | **Development Phase 30 weeks (week 40 to week 70)** |
| Completed | 20 weeks (week 40 -to 60) |
| Checklist |  |
|  | JAD session reports |
|  | End user manual preparation diagram |
|  | BA and developer MOM |
|  | Email communication- To , CC, BCC |
|  |  |
| **Q4** | **Quaterly Audit 4** |
| **Stage 5** | **Testing Phase 20 weeks (week 58- 78)** |
| Completed | 20 weeks (week 58 - week 78) |
| Checklist |  |
|  | Test case summary |
|  | Training report to end users |
|  | Lesson learnt docs |
|  | Email communication- To , CC, BCC |



# **Business Analyst (BA) Approach Strategy:**

Technical Team have assembled to discuss on the Project approach and have finalised to follow 3-tier architecture for this project.



As the **Business Analyst (BA)** for this project, my primary role is to ensure the business requirements are fully understood, accurately documented, and properly communicated to both the stakeholders and the development team. Below is a detailed **BA Approach Strategy** outlining the key steps and processes that I will follow throughout the project’s lifecycle.

1. **Elicitation Techniques**

**Elicitation techniques** refer to the methods or approaches used by Business Analysts (BAs) to gather information, requirements, and feedback from stakeholders, users, and other sources. The goal of these techniques is to ensure that the requirements and expectations for a project or system are clearly understood and documented. Elicitation techniques help the BA collect insights into business needs, challenges, and objectives, which are essential for designing and developing solutions that meet those needs.

Common Elicitation Techniques include Interviews, **Surveys / Questionnaires, Workshops, Document Analysis, Use Case Analysis etc.**

* **Interviews**: Conduct in-depth interviews with stakeholders such as **Peter**, **Kevin**, **Ben**, and the **Committee** members to understand their pain points, requirements, and expectations.
* **Surveys/Questionnaires**: Distribute surveys to gather specific feedback on the types of products (seeds, fertilizers, pesticides) needed by farmers, their preferences, and their pain points.
* **Workshops**: Hold collaborative workshops with stakeholders and the development team to define key business processes and product features. This will help to ensure all requirements are aligned.
* **Document Analysis**: Analyse any existing documents, reports, or systems used by stakeholders (e.g., inventory management, farmer profiles, product catalogues) to understand their workflow.
* **Use Case Modelling**: Develop use cases to understand how the end users (farmers, manufacturers) will interact with the application.

1. **Stakeholder Analysis & RACI/ILS**

Effective stakeholder management is key to ensuring project success. Here’s how I will approach stakeholder analysis:

1. **Stakeholder Identification**: Identify all stakeholders, including:
   * **Primary Stakeholders**: Mr. Henry, Peter, Kevin, Ben (Farmers, and Committee)
   * **Secondary Stakeholders**: APT IT SOLUTIONS Team (PM, Developers, Testers, Network Admin, DB Admin)
   * **Tertiary Stakeholders**: Manufacturer representatives (Fertilizer, Seed, Pesticide Companies)
2. **Stakeholder Analysis:** Categorize stakeholders based on their interest, influence, and impact on the project. This helps prioritize communication and ensure the right stakeholders are consulted at each stage.
3. **RACI Matrix**: Define clear roles and responsibilities for stakeholders using the **RACI** model

* **Responsible**: Who will do the work (e.g., Developers for product features).
* **Accountable**: Who will own the final decision (e.g., Project Manager Mr. Vandanam for overall project delivery).
* **Consulted**: Who will provide input (e.g., BA for business requirements).
* **Informed**: Who needs to be updated on progress (e.g., Stakeholders like Mr. Henry, Peter, Kevin).

1. **ILS (Issue and Log Sheet)**: Track any issues, risks, and decisions that arise throughout the project lifecycle and ensure these are documented and addressed in a timely manner.
2. **Documentation and Requirements Management**

**A. Key Documents to be Written**

1. **Business Requirements Document (BRD):** Capture all functional and non-functional requirements, including the system's expected capabilities, user roles, and key functionalities.
2. **Functional Requirements Document (FRD):** Document specific details on how the system will work, ensuring alignment with business objectives. It includes process flows, use cases, and functional specifications.
3. **User Stories/Use Cases:** Document user interactions with the system from the perspective of the farmer, manufacturer, and system administrators.
4. **System Requirements Specification (SRS):** Ensure that the technical team has a clear understanding of the requirements for designing the solution, in line with the three-tier architecture.
5. **Data Flow Diagrams (DFD):** Illustrate how data flows within the system, showing interactions between the user interface, business logic layer, and database.
6. **Test Plans/Scenarios:** Ensure clear test cases and acceptance criteria are documented for validation during the UAT phase.

**B. Process to Follow for Document Sign-Off**

1. **Review and Validation:** Once the documents (BRD, FRD, etc.) are written, I will review them with the stakeholders to ensure they accurately capture the needs of the project. This will involve working closely with Mr. Pandu (Financial Head) and Mr. Dooku (Project Coordinator).
2. **Feedback Incorporation:** Collect feedback and ensure all suggestions are incorporated before finalizing the documents.
3. **Formal Sign-Off:** The finalized documents will be sent to the stakeholders for formal approval. I will gather their signatures or formal approval via email or through a project management tool.
4. **Approval Process from the Client**

* **Regular Communication**: Throughout the project, I will maintain open channels with **Mr. Henry** and other stakeholders to gather their feedback and approval.
* **Formal Approvals**: At each major milestone (requirements, design, testing, and deployment), I will seek formal sign-off from the stakeholders. These approvals will be documented and stored for reference.

1. **Communication Channels to Establish & Implement**

* **Regular Meetings:** I will organize weekly stand-ups and bi-weekly stakeholder meetings to update the stakeholders on the project’s progress, risks, and upcoming milestones.
* **Project Management Tool:** Use tools for tracking project tasks, bugs, and progress.
* **Email Updates:** For important announcements, feedback, and approvals, I will maintain regular email communication with all stakeholders.
* **Collaborative Tools:** Use Microsoft Teams or other tools for real-time communication among the project team and stakeholders.
* **Documentation Sharing:** Use Google Drive, SharePoint, or a similar platform for document sharing and version control.

1. **Handling Change Requests**

* **Change Request Form**: Any change request received from stakeholders will be logged using a **Change Request Form** (CRF) to capture the nature of the change, the reasons for the change, and its potential impact.
* **Impact Analysis**: I will work with the technical team (developers, testers, etc.) to assess the impact of the change request on the project timeline, budget, and scope.
* **Approval**: Once the impact analysis is complete, I will send the change request to stakeholders for approval. Only after receiving approval will the change be incorporated into the project.

1. **Progress Updates to Stakeholders**

* **Progress Reports**: I will provide **monthly progress reports** detailing completed tasks, upcoming activities, risks, and issues. These reports will be shared with **Mr. Henry** and other stakeholders.
* **KPIs and Metrics**: Monitor key performance indicators (KPIs), such as **development progress**, **budget adherence** and **timeline compliance**, and communicate them during the quarterly audits.

1. **UAT and Client Project Acceptance Form**
2. **User Acceptance Testing (UAT):**

* Once development is complete, I will assist in organizing and facilitating the UAT. This will involve coordinating with the Testers (Jason and Alekya) to ensure the application meets the agreed-upon business requirements.
* I will also help stakeholders, like Peter, Kevin, and Ben, to test the system and provide feedback.

1. **Client Project Acceptance Form:**

* After UAT, I will ensure that the Client Project Acceptance Form is filled out and signed by the relevant stakeholders to confirm that the application is ready for deployment.
* The Client Project Acceptance Form will include details such as: Confirmation that the business requirements have been met.
* A final review of the system’s performance.
* Sign-off from Mr. Henry or any designated stakeholder confirming project acceptance.

**Conclusion:**

The above strategy outlines the necessary steps for ensuring a successful project outcome. By applying appropriate elicitation techniques, conducting thorough stakeholder analysis, maintaining clear communication, and following a structured approach to document management and approvals, I will ensure that the project is delivered on time, within budget, and meets all stakeholder expectations.



# **3-Tier Architecture:**

* The 3-tier architecture is a software design model that divides an application into three logical layers: Presentation Layer, Application Layer, and Data Layer.
* Each layer is responsible for specific functionalities, ensuring separation of concerns, scalability, and maintainability
* In the context of the online agricultural product store that Mr. Henry is planning to build, the **3-tier architecture** provides a structured way to design and organize the system into three separate layers, each with distinct responsibilities.

1. **Application Layer (Presentation Layer)**:

* **Purpose**: The Application Layer, also known as the **Presentation Layer**, is the topmost layer responsible for interacting with the user. This is where farmers, suppliers, and other users will interact with the system through a user interface (UI).
* **Role**: It displays the application’s data to the user and handles user input (e.g., browsing products, adding items to the cart, and placing orders). This layer ensures that the user interface is intuitive and user-friendly, as the target audience (farmers in remote areas) may not be very tech-savvy.
* **Components**:
  1. **Web Interface**: A responsive web interface that allows farmers to browse the available fertilizers, seeds, and pesticides.
  2. **Mobile Application**: A mobile app version of the platform, providing easy access for users with mobile phones.
  3. **Frontend Technologies**: HTML, CSS, JavaScript, React (for web), and mobile app frameworks like React Native or Flutter.

**Example**:

A farmer logs in, browses available products, filters them by price and availability, adds items to the cart, and places an order. This information is then sent to the Business Logic Layer.

1. **Business Logic Layer (Application Logic Layer)**:

* **Purpose**: The Business Logic Layer processes the data between the Application Layer (UI) and the Database Layer. It contains the core functionality and rules that dictate how the system operates.
* **Role**: This layer handles the logic for managing product details, calculating prices, validating user inputs, managing orders, and handling communication between farmers and manufacturers. It ensures that the user’s actions are valid and that business rules (such as order quantity limits, product availability, etc.) are followed. (example - calculating and showing the final billing price after taxes and other additions)
* **Components:**

1. **Programming Languages:** Java, Python, JavaScript, C#, Ruby, etc. Application Programming Interface (API)s that handle requests from the Application Layer and communicate with the Database Layer.
2. **Business Rules:** The logic to calculate pricing, discounts, and shipping costs. It also validates user data like order quantities, delivery locations, and payment methods.
3. **Backend Technologies:** Java (as mentioned, with Java developers like Ms. Juhi), Spring Boot for backend development etc

**Example**:

When a farmer places an order, this layer checks stock availability, calculates the total price (including taxes and discounts), and processes the payment before storing order details in the database.

1. **Database Layer (Data Layer)**:

* **Purpose**: The **Database Layer** is where all the application data is stored, managed, and retrieved. This layer interacts directly with the data storage system (such as a relational or non-relational database) and ensures that data is persisted for later use.
* **Role**: This layer handles the creation, reading, updating, and deletion (CRUD operations) of data. It is responsible for storing all the necessary information, including user profiles, product details, orders, and transactions. The **Database Layer** ensures that the data is structured, secured, and easily accessible when needed by the Business Logic Layer.
* **Components**:
  1. **Relational Databases**: MySQL, PostgreSQL, or Oracle databases could be used to store structured data such as product information, user profiles, and order history.
  2. **NoSQL Databases**: MongoDB or other NoSQL databases might be used to store unstructured or semi-structured data, such as product reviews, inventory data, and logs.
  3. **Data Access Layer**: A layer responsible for interacting with the database to fetch, update, or delete data.

**Example**:   
When a farmer places an order, the Business Logic Layer sends order details to be stored in the database. If they are a new user, a profile is created. Returning users can access their previous orders and details.

**Illustrate 3-Tier Architecture**:



**Flow of Data in 3-Tier Architecture**

1. **User Interaction:**
   * The farmer interacts with the **Application Layer** (web or mobile app) to browse products and place an order.
   * **Example:** The farmer selects a fertilizer, views the price, and adds it to the cart.
2. **Business Logic Processing:**
   * The **Business Logic Layer** processes the request, checking stock, calculating prices, and validating the order.
   * **Example:** It verifies product availability, computes the total cost, and processes the order.
3. **Data Storage:**
   * Once confirmed, the **Business Logic Layer** stores the order details in the **Database Layer** for future reference.
   * **Example:** The order, including shipping details, is saved in the database.
4. **Order Confirmation:**
   * The **Business Logic Layer** sends the order status (success or failure) back to the **Application Layer**, which displays it to the farmer.
   * **Example:** The farmer sees a confirmation message that the order is successfully placed.

**Advantages of 3-Tier Architecture for the Project:**

* **Scalability:** Each layer can scale independently. More servers can be added to the Application Layer for traffic, or the Database Layer can expand for more data
* **Maintainability:** Database or business logic changes don’t impact the user interface, allowing seamless updates.
* **Security:** The database layer is isolated from direct user interaction, which helps in securing sensitive data (e.g., payment details, product inventory).
* **Flexibility:** The modular system allows upgrades or replacements without



1. **BA Approach Strategy for Framing Questions**:



**(5W 1H – SMART – RACI – 3 Tier Architecture – Use Cases, Use case Specs, Activity Diagrams, Models, Page designs).**

As a Business Analyst (BA) in this project, you need to approach the stakeholders carefully and comprehensively to gather the required information for building the online agricultural product store. Before framing any questions, it's crucial to consider several key aspects to ensure that the requirements are clear and detailed, aligning with the stakeholders' needs and expectations. Here's a breakdown of what to keep in mind:

1. **5W 1H (Who, What, Where, When, Why and How):**

The 5W 1H method ensures you gather detailed and broad information to understand the problem and the needs of stakeholders.

* **Who:**
  + Who are the primary users of the application (farmers, manufacturers, delivery agents)?
  + Who will manage the system (admin, support team)?
  + Who is responsible for each feature (e.g., product details, delivery)?
  + Who will maintain and update the application (internal or external teams)?
* **What:**
  + What are the core functionalities (product browsing, ordering, payment)?
  + What types of products (seeds, fertilizers, pesticides) will be listed?
  + What features do stakeholders consider as essential (e.g., search filters, product recommendations)?
  + What are the technical requirements (e.g., mobile app, responsive design, payment gateway)?
* **Where:**
  + Where will the system be used (farmers’ locations, manufacturers’ locations)?
  + Where will the data be stored (cloud, on-premise servers)?
* **When:**
  + When will the system need to be launched (specific milestones, deadlines)?
  + When will users need access to the application (real-time availability or business hours)?
* **Why:**
  + Why is this project important for farmers and manufacturers?
  + Why will this online platform make a difference for remote area farmers?
* **How:**
  + How will the communication happen between farmers and manufacturers (chat, forms, call centre)?
  + How will the product delivery process work (tracking, delivery dates)?

1. **SMART (Specific, Measurable, Achievable, Relevant, Time-bound):**

Ensure that the questions you ask will help clarify specific and measurable goals for the project.

* **Specific:**
  + What specific products need to be listed on the platform?
  + How will the farmers select the products (search filters, categories)?
* **Measurable:**
  + What metrics or KPIs (Key Performance Indicators) will determine success (e.g., number of users, product sales)?
  + How will user satisfaction be measured?
* **Achievable:**
  + Are there technical limitations (e.g., internet connectivity issues in remote areas)?
  + Can the system handle the expected number of users and transactions?
* **Relevant:**
  + How will the platform help farmers in remote areas (better access to products, better prices)?
  + Is there a focus on creating a sustainable and scalable solution?
* **Time-bound:**
  + What is the timeline for launching the platform and subsequent updates?
  + Are there any specific deadlines for product delivery, onboarding, or marketing?

1. **RACI (Responsible, Accountable, Consulted, Informed):**

This framework will help clarify the roles and responsibilities of everyone involved, ensuring clear accountability.

* **Responsible:**
  + Who is responsible for developing the product features (developers, project manager)?
  + Who is responsible for gathering requirements and documentation (you as BA)?
* **Accountable:**
  + Who is accountable for the overall success of the project (Mr. Henry, Mr. Pandu)?
* **Consulted:**
  + Who should you consult when gathering requirements (Peter, Kevin, Ben, other farmers)?
* **Informed:**
  + Who should be kept informed on progress and updates (Mr. Vandanam, Mr. Karthik, Mr. Pandu)?

1. **3-Tier Architecture:**

The system’s architecture will have three main layers: presentation (application), logic (business logic) and database layers.

* **Presentation/Application Layer (UI/UX):**
  + How should the user interface look (simple, intuitive, mobile-friendly)?
  + What features are important for user interaction (easy search, product descriptions, checkout process)?
* **Logic Layer (Business Logic):**
  + How should the logic for order processing work (inventory checks, payment gateway)?
  + What business rules must be followed (product availability, delivery conditions)?
* **Data Layer (Database):**
  + What data needs to be stored (user profiles, product inventory, order history)?
  + How will the data be accessed and managed (database security, query optimization)?

1. **Use Cases & Use Case Specifications:**

A **use case** is a description of how a system or application will be used to accomplish a specific goal from the perspective of an end user. It outlines the interactions between the user (referred to as an "actor") and the system to achieve a particular task or objective.

* **Use Case Examples:**
* A farmer browses products and adds items to their cart.
* A manufacturer submits new product details to the platform.
* An admin monitors and approves product listings.
* **Use Case Specifications:**
  + Define the flow of each use case (normal and alternative flows).
  + What are the preconditions (e.g., user must log in to place an order)?
  + What are the postconditions (e.g., order is placed, and farmer gets a confirmation)?

1. **Activity Diagrams:**

Activity Diagrams are a type of Unified Modelling Language (UML) diagram that visually represents the flow of control or activities in a system. They show the sequence of steps in a process or workflow and how they are connected, helping to model the dynamic aspects of a system. Activity diagrams are commonly used in the analysis and design phases to understand how processes work, identify potential bottlenecks, and ensure all workflows are accounted for. They are especially useful for representing business processes, use case flows or system interactions in a simplified, visual way. They also help to model the flow of actions, such as the order process, product submission and product delivery.

**For example:**

* **Farmer Ordering Flow:**
  + Browse products → Add to cart → Checkout → Payment → Delivery.
* **Manufacturer Product Submission:**
  + Log in → Add product details → Submit for review → Product approval

1. **Models:**

* Create **Entity-Relationship Models (ERMs)** to visualize how data entities interact.
* Create **Class Models** to map out the structure of the application and its components.
* **State Diagrams** could be useful for visualizing the various states of an order (pending, shipped, delivered).

1. **Page Designs:**

* Develop wireframes and mock-ups for the user interface (UI), focusing on the ease of use for farmers.
* Define which pages are essential (product pages, order summary, account management, etc.) and how they will flow.
* Ensure that the design is mobile-friendly, as many farmers in remote areas might use mobile phones for access.

**Conclusion:**

As a Business Analyst, before framing any questions, ensure you have a clear understanding of these areas and adapt your questions accordingly to gather comprehensive requirements. Always consider the 5W 1H framework for clarity, SMART for goal setting, RACI for roles and responsibilities, and the 3-tier architecture for the system design. Additionally, create use cases, activity diagrams, models, and page designs to ensure that the system will be both functional and user-friendly.



1. **Elicitation Techniques:**

Requirement analysis is to dig the information from the stakeholders before starting the project to gather as much as information possible from the stakeholders. Requirement elicitation serves as the foundation in documenting the requirements before starting a project.

There are 11 elicitation techniques to apply accordingly: -

1. **Brainstorming:**

* This technique is used to generate new ideas and find a solution for a specific issue.
* The members included for brainstorming can be domain experts, subject matter experts.
* Multiple ideas and information give us a repository of knowledge and we can choose from different ideas.

1. **Document Analysis:**

* During this step of the requirements elicitation process, business analysts review existing documentation at hand, with the intent of identifying requirements for changes or improvements.
* Examples of document analysis sources include pre-existing project plans, system specifications, process documentation, market research dossiers, customer feedback, meeting minutes, and user manuals.
* Document analysis is performed before scheduling more in-depth requirements elicitation sessions or interviews with stakeholders.

1. **Reverse engineering:**

* In this Technique, any outdated documentation in an existing system, can be reversed to understand what the system does.
* His is an elicitation technique that can extract implemented requirements from the system.
* There are two types of reverse engineering techniques.

1. **Black box reverse engineering:**

The system is studied without examining its internal structure (function and composition of software).

1. **White box reverse engineering:**

The inner workings of the system are studied (analysing and understanding of software code).

1. **Focus Group:**

* By using a focus group, you can get information about a product, service from a group.
* The Focus group includes subject matter experts. The objective of this group is to discuss the top issue and provide information. A moderator manages this session.

1. **Observation:**

* An excellent elicitation technique helps understand requirements based on observations related to process flows and work environments of stakeholders.
* Observation requires a business analyst to go and look at the work – for example, observing the business processes in scope of the project.
* The elicitation technique observation is an effective means of understanding how a user does their job by assessing their work environment.

1. **Workshops**:

* Workshops comprise a group of users or stakeholders working together to identify requirements.
* A requirement workshop is a structured way to capture requirements.
* Workshops are used to scope, discover, define, and prioritize requirements for the proposed system.

1. **JAD (Joint Application Development):**

* This technique is more process-oriented and formal as compared to other techniques.
* These are structured meetings involving end-users, PMs, SMEs.
* This is used to define, clarify, and complete requirements.

1. **Interviews:**

* An interview is a systematic approach to elicit information from a person or group of people.
* This is the most common technique used for requirement elicitation.
* Interview techniques should be used for building strong relationships between business analysts and stakeholders.
* In this technique, the interviewer directs the question to stakeholders to obtain information. One to one interview is the most commonly used technique.

1. **Prototyping:**

* Prototyping is used to identify missing or unspecified requirements.
* In this technique, frequent demos are given to the client by creating the prototypes so that client can get an idea of how the product will look like.
* Prototypes can be used to create a mock-up of sites, and describe the process using diagrams.

1. **Questionnaire & Surveys:**

* For Survey/Questionnaire, a set of questions is given to stakeholders to quantify their thoughts.
* After collecting the responses from stakeholders, data is analysed to identify the area of interest of stakeholders. Questions should be based on high priority risks.

1. **Use case specs:**

* Use cases are an effective and widely used technique for eliciting software requirements.
* The use-case approach focuses on the goals that users have with a system, rather than emphasizing system functionality.
* This technique combines text and pictures to provide a better understanding of the requirements.
* The use cases describe the ‘what’, of a system and not ‘how’. Hence, they only give a functional view of the system.
* The components of the use case design include three major things – Actor, Use cases, use case diagram.



1. **This Project Elicitation Techniques:**

For the Online Agriculture Products Store project, multiple elicitation techniques are suitable. However, selecting the best techniques depends on the specific requirements; stakeholders involved, and project goals.

1. **Prototyping (Best for Usability Validation)**
2. **Why:**
3. Provides stakeholders (like farmers and manufacturers) with a visual representation of the application, making it easier for them to understand and provide feedback.
4. Especially helpful for non-technical stakeholders who may struggle to articulate their requirements clearly.
5. **What to Prototype:**
6. Login/Sign-up workflows.
7. Product catalog with filters and search options.
8. Payment gateway options (UI flow for COD, UPI, and card payments).
9. Delivery tracking interface.
10. **Outcome:**  
    Clear, user-approved designs that minimize ambiguity and set accurate expectations.
11. **Use Case Specifications (Best for Functional Clarity)**
12. **Why:**
13. Captures interactions between users (farmers, manufacturers) and the system in a structured format.
14. Provides a clear understanding of functional requirements and expected system behaviour.
15. **Example Use Cases:**
    1. "Search for a product."
    2. "Login/Sign-up."
    3. "Place an order and make a payment."
    4. "Track delivery."
16. **Outcome:**  
    Detailed documentation that aligns developers and testers with the business requirements.
17. **Brainstorming (Best for Ideation and Innovation)**
18. **Why:**
    1. Encourages creative input from key stakeholders like Peter, Kevin, Ben, and Mr. Henry.
    2. Helps identify additional features or enhancements, such as promotional offers or farmer-specific dashboards.
19. **Topics to Brainstorm:**
    1. Features to improve the farmer's user experience.
    2. Methods to streamline product uploads for manufacturers.
    3. Effective ways to send order notifications and delivery updates.
20. **Outcome:**  
    A rich set of ideas to refine and expand the application's scope.
21. **Document Analysis (Best for Understanding Context and Constraints)**
22. **Why:**
    1. Reviews existing product catalogs, user workflows, and industry guidelines for fertilizers, seeds, and pesticides.
    2. Identifies mandatory fields for manufacturers and ensures compliance with agricultural regulations.
23. **What to Analyze:**
    1. Current formats used by manufacturers for product details.
    2. Industry norms for fertilizers and pesticide labeling.
    3. Existing farmer and manufacturer workflows.
24. **Outcome:**  
    A solid understanding of the domain to ensure the application aligns with stakeholder and regulatory requirements.

* **Best Technique for Each Stakeholder Group**

| **Stakeholder Group** | **Best Elicitation Technique** | **Reason** |
| --- | --- | --- |
| Farmers (Peter, Kevin) | **Prototyping, Brainstorming** | Easy visualization and ideation for usability improvements. |
| Manufacturers (Ben, etc.) | **Document Analysis, Prototyping** | Ensures clarity in product uploads and compliance with guidelines. |
| Mr. Henry (Client) | **Use Case Specs, Prototyping** | Provides structured scenarios and visual feedback for approval. |

* **Recommendation:**

1. Start with Prototyping: Create mock-ups of critical features and gather feedback from farmers and manufacturers.
2. Develop Use Case Specs: Write detailed use cases for login, product search, payment, and order tracking.
3. Conduct a Brainstorming Workshop: Explore additional features and workflows with stakeholders.
4. Perform Document Analysis: Review existing product documentation and industry standards.
5. **10 Business Requirements:**
6. **BR001 –** The platform should have a product catalogue that includes all fertilizers, seeds, and pesticides from different manufacturers and vendors.

**Assumptions:**

* Manufacturers and vendors will upload product data (e.g., name, description, image, price, stock) via a dedicated portal.
* Each product will be categorized accurately into fertilizers, seeds, or pesticides.
* An admin team may review and approve listings before they go live.

1. **BR002 –** The platform should allow farmers to search for products by name, category, and brand.

**Assumptions:**

* Products will be tagged with metadata (name, category, brand) during upload.
* A search bar with filter and sorting features (e.g., price, popularity) will be available.
* Search performance will be optimized for mobile and low-bandwidth usage.

1. **BR003** – The platform should have a login feature for all users including farmers, manufacturers, and vendors.

**Assumptions:**

* All users will use a common login system with role-based access controls.
* Authentication will be token-based (e.g., JWT) and sessions will expire after a set time.
* Login credentials will be encrypted and stored securely.

1. **BR004 –** The platform should allow new users to create an account by submitting their email ID and creating a secure password.

**Assumptions:**

* Email verification via OTP or link will be mandatory for account activation.
* Passwords must follow security protocols (minimum 8 characters, mixed characters).
* Duplicate email IDs will be blocked to prevent multiple accounts.

1. **BR005 –** The platform should have user-friendly interfaces and easy navigation for a better user experience.

**Assumptions:**

* The UI/UX will be designed with a mobile-first approach, prioritizing ease of use for farmers.
* Language support for local/regional dialects may be considered.
* Navigation menus will be simple, with icon-based guides and tooltips.

1. **BR006** – The platform should have a payment gateway that includes COD, credit/debit cards, and UPI options.

**Assumptions:**

* Payment gateways like Razorpay, PayU, or Stripe will be integrated for digital payments.
* COD will be available only in eligible regions based on delivery partner support.
* Payment failures will trigger retry and support mechanisms.

1. **BR007** – The platform should send email confirmation regarding order status to users.

**Assumptions:**

* Users must provide a valid email during signup to receive notifications.
* Email templates will be automated and triggered based on status updates (e.g., order placed, shipped, delivered).
* Emails will be sent via a reliable transactional email service (e.g., SendGrid, Mailgun).

1. **BR008** – The platform should have a delivery tracker to track the whereabouts of the order.

**Assumptions:**

* The system will integrate with third-party logistics APIs (e.g., Delivery, Ship rocket).
* Tracking details will be updated in real-time on both the app and website.
* Users will get tracking links and status updates via SMS or email.

1. **BR009** – The platform should be scalable to accommodate future growth and expansion.

**Assumptions:**

* The backend will be hosted on a scalable cloud platform (e.g., AWS, Azure).
* A microservices or modular architecture will be used to allow independent scaling of services.
* Database systems will support partitioning and horizontal scaling.

1. **BR010** – The platform should have a secure infrastructure to protect user data and prevent data breaches.

**Assumptions:**

* The platform will follow data protection regulations (e.g., GDPR, local laws).
* All sensitive data will be encrypted at rest and in transit.
* Regular security audits, firewalls, and access logs will be maintained.



1. **Assumptions:**

* The application will be accessible on both web and mobile platforms.
* The application will have an easy-to-use interface, as users may not be very familiar with technology.
* The platform will support multiple languages to cater to a wider audience, especially in remote areas.
* Payments can be done via multiple methods, including online and offline (COD).
* Delivery services will be included for product shipping and it will support tracking of orders.
* The manufacturers (companies) will be verified before they can list their products.
* The platform will also include a support/help feature for users who face technical issues.
* Inventory management will be in place to track product availability.
* The system will have security features for protecting sensitive data (user details, financial transactions etc.).

1. **This project Requirements Priority**

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement** | **Description** | **Priority** | **Justification** |
| **BR001** | Product catalogue including fertilizers, seeds, and pesticides from various manufacturers/vendors. | **10** | Core functionality; without this, the platform cannot fulfil its purpose. |
| **BR002** | Search feature by name, category, and brand. | **9** | Enhances usability and ensures users can find products quickly, crucial for user experience. |
| **BR003** | Login feature for all users, including farmers, manufacturers, and vendors. | **9** | Ensures user-specific access and management, foundational for platform security and operations. |
| **BR004** | Account creation for new users by submitting email ID and creating a secure password. | **8** | Essential for on boarding new users but can follow after initial core functionalities are built. |
| **BR005** | User-friendly interface and easy navigation. | **10** | Critical for adoption, especially given farmers may have limited technical skills. |
| **BR006** | Payment gateway with COD, credit/debit cards, and UPI options. | **8** | Important for completing transactions; COD can serve as a fall back if online payment is delayed. |
| **BR007** | Email confirmation regarding order status. | **7** | Nice-to-have; adds convenience but can be implemented after core order functionalities. |
| **BR008** | Delivery tracker to track the whereabouts of the order. | **6** | Enhances user experience but is less critical during the initial implementation phase. |
| **BR009** | Scalable platform to accommodate future growth. | **9** | Vital for long-term sustainability and should be considered during the architectural design phase. |
| **BR010** | Secure infrastructure to protect user data and prevent data breaches. | **10** | Critical to build trust and comply with data protection laws. |

1. **Draw use case diagram.**

To create a use case diagram for the online agricultural product store based on the scenario provided, we can break down the system into several key actors and use cases. Here's a summary of the actors and use cases:

**Actors:**

1. **Farmer**: End users who will purchase fertilizers, seeds and pesticides.
2. **Manufacturer**: Fertilizer, seed and pesticide companies who will list their products for sale.
3. **Administrator**: The person who manages the application, ensuring smooth functioning and managing the system.

**System:**

1. **Database of Agriculture Store**: The online application itself which facilitates all the transactions.

**Use Cases:**

1. **Farmer**:
   1. Register/Log in to the application.
   2. Browse available products (fertilizers, seeds, pesticides).
   3. Search for specific products.
   4. View detailed product information (product description, price etc.).
   5. Select products and add them to the cart.
   6. Place order for products.
   7. Request delivery to their location.
   8. View order history.
   9. Track the order status.
2. **Manufacturer**:
   1. Register/Log in to the application.
   2. Add new products (fertilizers, seeds, pesticides).
   3. Edit product details.
   4. Remove products.
   5. View sales and customer information.
3. **Administrator**:
   1. Log in to the admin panel.
   2. Approve or reject manufacturer/farmer registration.
   3. Manage user accounts (both farmers and manufacturers).
   4. Monitor the performance of the system.
   5. Generate reports.

**System Use Cases:**

1. **Database of Agriculture Store**:
   * Authenticate users (farmers, manufacturers, administrators).
   * Process orders and transactions.
   * Send notifications to users regarding order status.
   * Manage user data securely.

**Here, we mainly concentrate on the Use Case diagrams of the interactions between Primary Actors (Farmer, Manufacturer) and System Admin.**

* **Farmers** can register, browse and select products, view order history, and request delivery.
* **Manufacturers** can add, edit, or remove their products, and view sales data.
* **Administrators** manage the system, user accounts, and ensure smooth operations.
* The **System (Database Agriculture Store)** manages all backend processes, such as user authentication, order processing, and sending notifications.

This use case diagram gives a high-level view of the application and its main interactions.



1. **Use Case Specs:**

|  |  |
| --- | --- |
| **Use Case ID** | **UC001** |
| Use Case Name | Buying a product |
| Actors | Customer, Seller |
| Description | This use case describes how users can make purchase via App |
| Pre - Condition | User should have been registered into the application |
| Post - Condition | Successfully able to login the Account |
| Basic Flow | Step 1: User create and account and login. |
|  | Step 2: User search for a product from the search bar. |
|  | Step 3: same product and related product option from different manufacture will be appeared on the screen. |
|  | Step 4: User select one product, selects the size and quantity of the product and click on "buy now option". |
|  | Step 5: System will take to another page, where total price calculation will be displayed along with the products added to cart. |
|  | Step 6: User click on "Place order button". |
|  | Step 7: User need to to choose the mode of the payment. |
|  | Step 8: User need to enter the banking details and make payment. |
|  | Step 9: User will receive order confirmation on email along with the tracking id. Step 10: Basic flow end here. |
| Alternate Flows | Step 1: User is not able to login and redirected to forgot "Username/Password" page. |
|  | Step 2: If you user is not able to get the right information, he can request for a call from customer care. |
|  | Step 3: once he get connected with the customer care he will explain the issue to the customer care representative, |
|  | Step 4: Customer care will send a link to reset password to his email account. |
|  | Step 5: User will go to that link and system will take to new page, where user will be able to change new password |
|  | Step 6: User will be put a new password. |
|  | Step 7: System will ask to reconfirm the password. |
|  | Step 8: User will be able to login the account now. |
| Exceptions | If internet connectivity lost while doing this use case, system displays " check with your internet connectivity " |
| Frequency of use | High |
| Assumptions | It is assumed that the customer is registered. |
|  | It is assumed that the customer has the computer knowledge. |
|  | It is assumed that the customer has a suitable device to use the APP. |

|  |  |
| --- | --- |
| **Use Case ID** | **UC002** |
| Use Case Name | Exchange of product |
| Actors | Customer, Seller |
| Description | This use case describes how users can exchange a purchased product. |
| Pre - Condition | User should have purchased a product before in order to make a exchange. |
| Post - Condition | Successfully able to exchange the product |
| Basic Flow | Step 1: User login to account via credentials. |
|  | Step 2: User click on "Account" . |
|  | Step 3: System takes to different page with other details. |
|  | Step 4: User select option "Exchange" among those options. |
|  | Step 5: System will take to another page, where recently ordered products will be displayed on the screen. |
|  | Step 6: User has to choose the product which he wants to exchange. |
|  | Step 7: User will get another option where he will be asked- "different size in same product" or "want to buy another product" |
|  | Step 8: User need to choose one of the options and take action according to chosen option. |
|  | Step 9: Once the product is chosen, user will have to click on button "Exchange". |
|  | Step 10: User will get the confirmation on email. |
| Alternate Flows | Step 1: User couldn't find the size which he wanted. |
|  | Step 2: User call customer care agent to get a solution |
|  | Step 3: Agent suggested to wait for the size to be restocked and gave a tentative date or go for similar products. |
|  | Step 4: Agent shares the link of similar products to the registered email of the customer. |
|  | Step 5: User choose the product |
|  | Step 6: User will be put a new password. |
|  | Step 7: System will ask to reconfirm the password. |
|  | Step 8: User will be able to login the account now. |
| Exceptions | If internet connectivity lost while doing this use case, system displays " check with your internet connectivity " |
| Frequency of use | High |
| Assumptions | It is assumed that the customer is registered. |
|  | It is assumed that the customer has the computer knowledge. |
|  | It is assumed that the customer has a suitable device to use the APP. |

|  |  |
| --- | --- |
| **Use Case ID** | **UC003** |
| Use Case Name | Return of product |
| Actors | Customer, Seller |
| Description | This use case describes how users can return a purchased product. |
| Pre - Condition | User should have purchased a product before in order to make a return. |
| Post - Condition | Successfully able to exchange the product |
| Basic Flow | Step 1: User login to account via credentials. |
|  | Step 2: User click on "Account" |
|  | Step 3: System takes to different page with other details. |
|  | Step 4: User select option "Return" among those options. |
|  | Step 5: System will take to another page, where recently ordered products will be displayed on the screen. |
|  | Step 6: User has to choose the product which he wants to return. |
|  | Step 7: User will get another option where he will be asked to provide the bank account number for amount of the returned product to be credited. |
|  | Step 8: User need to enter the account number and submit. |
|  | Step 9: User will get the confirmation on email. |
| Alternate Flows | Step 1: User didn't get the amount in his account within the TAT. |
|  | Step 2: User call customer care agent to ask payment status. |
|  | Step 3: Payment was stuck due to a technical glitch. |
|  | Step 4: User was shared complaint form to be filled. |
|  | Step 5: Once form submitted, user received another TAT on the email of amount to be credited. |
|  | Step 6: User get the payment id in registered email |
| Exceptions | User put the incorrect bank account. |
| Frequency of use | High |
| Assumptions | It is assumed that the customer has a valid bank account number. |
|  | It is assumed that the customer has good internet connectivity. |
|  | It is assumed that the customer has computer knowledge. |

|  |  |
| --- | --- |
| **Use Case**  **ID** | **UC004** |
| Use Case Name | Update the delivery address |
| Actors | Customer, Seller |
| Description | This use case describes how users can update address. |
| Pre - Condition | User should have a valid deliverable postal address. |
| Post - Condition | Successfully able to update address. |
| Basic Flow | Step 1: User login to account via credentials. |
|  | Step 2: User click on "Account". |
|  | Step 3: System takes to different page with other details. |
|  | Step 4: User select option "Update" among those options. |
|  | Step 5: System will take to another page, where mandatory fields like; Apt number, landmark, pin code, city name will be displayed and has to be field. |
|  | Step 6: User need to click on "submit" button. |
|  | Step 7: User can use the updated address for products delivery. |
| Alternate Flows | Step 1: User is not able to update the address. |
|  | Step 2: User will refresh the page. |
|  | Step 3: User gets error again while submitting details. |
|  | Step 4: User use live chat box |
|  | Step 5: User is asked to not leave blank any star marked field. |
|  | Step 6: after updating all mandatory field, address was successfully submitted. |
| Exceptions | User put the incorrect address details like; pin exceeds the maximum number of digits |
| Frequency of use | High |
| Assumptions | It is assumed that the customer has a valid postal address It is assumed that the customer has good internet connectivity. |
|  | It is assumed that the customer has computer knowledge. |
|  | It is assumed, customer understands, what details has to be put in every field. |

|  |  |
| --- | --- |
| **Use Case ID** | **UC005** |
| Use Case Name | Update the new contact number |
| Actors | Customer, Seller |
| Description | This use case describes how users can update/ change new phone number |
| Pre - Condition | User should have a new contact number. |
| Post - Condition | Successfully able to change contact number. |
| Basic Flow | Step 1: User login to account via credentials.  Step 2: User click on "Account".  Step 3: System takes to different page with other details.  Step 4: User select option "Manage your Account" among those options.  Step 5: System will take to another page, where personal details will be displayed. Step 6: User has to click Mobile number  Step 7: User will get a red popup button "CHANGE".  Step 8: OTP will be sent to existing updated number  Step 9: once number is verified with the OTP user put. User can update new contact number.  Step 10: New contact number is successfully updated. |
| Alternate Flows | Step 1: User didn't get the OTP in registered existing number.  Step 2: User restarts the phone.  Step 3: User raised a ticket with the customer care  Step 4: User was shared issue ticket number in the registered email.  Step 5: Issue got fixed with the help of support team  Step 6: contact number is successfully changed. |
| Exceptions | User put the incorrect phone number. |
| Frequency of use | Low |
| Assumptions | It is assumed that the customer has a valid phone number.  It is assumed that the customer has good phone network to receive OTP.  It is assumed that the customer has checked the message inbox for OTP. |

1. **Activity Diagram:**



Figure User Registration Login Activity Diagram



Figure User Login Activity Diagram



Figure Return/Cancel Order Activity Diagram



Figure Seller Login Activity Diagram



Figure Admin Login Activity Diagram