Project 2 –

1. **Audits**

Quarterly audits for a BA in a project are essential to ensure compliance, identify risks, and track the progress of deliverables. The quarterly audits (Q1, Q2, Q3, Q4) are designed to ensure that the project stays on track and aligns with the business objectives and stakeholder expectations.

1. **Q1 Audit – Planning and Initiation**

This level focuses on ensuring all requirements gathering and requirements analysis are properly documented and aligned with business goals. It identifies any gaps in early planning and stakeholder involvement.

**Phases covered** are - Requirements Gathering and Requirements Analysis

**BA’s Role in Q1 Audit:**

* To ensure that all the business and technical requirements (e.g., fertilizers, seeds, pesticides purchasing and delivery functionalities) are gathered, documented and signed by stakeholders. (Including Mr. Henry, Peter, Kevin, and Ben)
* BA needs to verify that the project scope, timelines and deliverables are properly noted.
* BA should ensure that stakeholder inputs are included and approved.
* Requirements document, scope Document should be there.

1. **Q2 Audit – Design and Initial Development**

This level evaluates development progress, testing of initial phases, and adherence to timelines. Also ensures that coding and system architecture align with defined requirements.

**Phases covered are –** Design Phase, Development Phase - D1, Testing Phase - T1

**BA’s Role in Q2 Audit:**

* BA ensures the design phase has been reviewed and approved by stakeholders.
* BA needs to ensure that the design aligns with documented requirements. (e.g., user-friendly interfaces for farmers)
* Confirm that development of D1 aligns with the approved design.
* Validate that T1 (associated with D1) is executed, and initial feedback is captured and resolved.

1. **Q3 Audit – Development Progress and Testing**

This level reviews system integration, extensive testing, and preparation for deployment. This level also focuses on addressing risks, ensuring quality, and verifying system readiness for launch.

**Phases covered are –** Development Phases - D2, D3 and Testing Phases - T2, T3

**BA’s Role in Q2 Audit:**

* Validate the progress of development (D2 and D3 to ensure requirements are being implemented correctly.
* Audit T2 and T3 to confirm critical functionalities are tested.
* Review defect logs and ensure issues from T1 and T2 are being resolved efficiently.
* BA needs to ensure that all requirements are being addressed during testing.

1. **Q4 Audit – Finalisation and UAT (User Acceptance Testing)**

This level concentrates on final testing, user acceptance, deployment, and post-launch evaluation. Also ensures that deliverables meet objectives and gathers insights for future improvements.

**Phases covered are –** Development Phase – D4 and Testing Phase – T4 and UAT

**BA’s Role in Q2 Audit:**

* Audit the final phase of development (D4) to ensure all requirements have been implemented.
* Validate that T4 tests all key functionalities and that defects are resolved before UAT begins.
* Focus on UAT to ensure that the final deliverable meets business objectives.
* Verify documentation, including user manuals and training materials, is complete.
* Ensure proper handovers to operational teams and that all stakeholders are satisfied with the deliverables.

1. **BA Approach Strategy**
2. **3-Tier Architecture**

The 3-tier architecture is a widely used software architecture that organizes applications into three layers or tiers. Each tier focuses on a specific function by making the system easier to manage and develop. The changes in one layer do not affect the others, which helps with faster updates and modifications. Each layer has a distinct role for improving scalability, maintainability, and reusability. This structure makes applications easier to build, update, and scale over time.

Each tier has its own responsibility (UI, business logic, and data management), making the system modular, easier to manage, and scalable. Modifications to one layer (like UI redesigns or database structure changes) do not require changes in the other layers.

The tiers are:

1. **Presentation Tier (User Interface)**

This is the topmost layer where users interact with the application. It is the client-facing layer. It consists of the user interface (UI) and handles input/output, such as web browsers or mobile apps.

* This is the top layer that interacts with the user.
* Its main job is to collect input from users and display the relevant data to them.
* It sends user requests to the Logic Tier and displays the results back to the user.
* Example: A web page or a mobile app where users browse agricultural products like fertilizers, seeds, or pesticides.

1. **Application Tier (Logic)**

The middle layer that processes the business logic, validates inputs, and manages the application’s core functionality. It acts as a bridge between the user interface and the database.

* It acts as an intermediary between the user interface (Presentation Tier) and the data (Data Tier).
* The Logic Tier validates the inputs, processes requests, and sends necessary data to the Data Tier, then returns the processed results to the Presentation Tier.
* This layer contains the core functionality, such as handling user authentication, managing business rules, and interacting with external services.
* It also allows for flexibility in modifying business processes or adding new features without changing the data or presentation layer.
* Example: The logic for validating stock availability, calculating discounts, handling payment processing, and managing delivery requests is all handled by the Logic Tier. If the farmer requests to buy a product, the Logic Tier checks whether the product is in stock and calculates the total price (with any applicable discounts or taxes).

1. **Data Tier (Database)**

The layer responsible for storing and managing data in a database. It retrieves, stores, and updates data as requested by the Logic Tier.

* The Data Tier is responsible for storing and managing the application’s data.
* This layer is responsible for maintaining persistent data such as user profiles, order history, product details, pricing, and stock levels. It ensures that data is securely stored and efficiently retrieved.
* The Data Tier ensures data integrity and consistency while interacting with the Logic Tier to provide the correct data when requested.
* It provides centralized data storage, which simplifies data management and ensures consistency across the application.
* Example: A database that stores details about products (seeds, fertilizers, pesticides), user profiles, orders, delivery information, and pricing. The data layer responds to requests from the Logic Tier, such as retrieving a list of available products or updating an order’s status.

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

When a Business Analyst (BA) frames questions to ask stakeholders, it's important to ensure that the questions are clear, purposeful, and aligned with the project’s objectives. It helps in understanding the context and gathering complete information by addressing Who is involved, What needs to be done, When it happens, Where it occurs, Why it’s important, and How it will be implemented.

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

This will help the BA to ask comprehensive and open-ended questions that cover all aspects of a scenario.

* Who: Who are the key stakeholders involved? Who will use the system? Who will be responsible for a task?
* What: What are the requirements or challenges? What functionalities should the system have?
* When: When should certain features or tasks be completed? What is the timeline?
* Where: Where will the application be used (web, mobile, location-specific requirements)?
* Why: Why is this feature needed? Why are certain processes in place?
* How: How will the system be used? How will data flow through the system?

Example:

Who are the users who will access the product catalog?

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

SMART is a framework for setting clear, actionable, and realistic goals. It ensures that objectives are Specific, Measurable, Achievable, Relevant, and have a Time-bound deadline to make them more focused and easier to track.

* **Specific:**

The goal should be clear and unambiguous, addressing exactly what needs to be achieved. For example, "Increase product sales" is too vague, but "Increase product sales by 10% in the next quarter" is specific.

Is the question clear and detailed enough to avoid ambiguity?

* **Measurable:**

You should be able to quantify progress or completion. Measurability ensures that you can track and evaluate success, such as "Increase website traffic by 20% in the next six months."

Can you measure the success or outcome of the question’s answer?

* **Achievable:**

The goal must be realistic given the available resources and constraints. It should be challenging, but not so difficult that it’s unattainable. For example, setting a goal to increase sales by 500% might not be achievable in a short timeframe.

Is the question asking for something realistic given the resources?

* **Relevant:**

The goal should align with broader business objectives and be meaningful. It should matter to both the project and the organization.

Does the question directly relate to the project’s goals or stakeholders' needs?

* **Time-bound:**

Every goal needs a deadline. Setting a timeframe helps create urgency and keeps the project on track. For example, “Launch new mobile app feature by the end of Q2” is time-bound.

Example:

What specific product categories should be available on the agriculture store by the end of Q2?

What is the target delivery time for each order after placing it?

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

RACI is a matrix used to define roles and responsibilities in a project. It clarifies who is Responsible for performing tasks, Accountable for decision-making, consulted for input, and Informed about progress or decisions.

* **Responsible:** Who will do the task or work?

The person or people who perform the work to achieve a task or deliverable. They are responsible for executing the activity and ensuring the task is completed.

Example: A developer responsible for coding a new feature.

* **Accountable:** Who is ultimately responsible for the success of the task?

The person who is ultimately accountable for the success or failure of the task. There should only be one accountable person per task. They ensure that the work is done correctly and meets standards.

Example: The project manager who oversees the overall success of the feature rollout.

* **Consulted:** Who needs to be consulted for input or feedback during decision-making?

These are the people who need to be consulted for input or advice. Their feedback or expertise is important during the process, but they are not directly involved in completing the work.

Example: A subject matter expert who provides guidance on the feature’s functionality.

* **Informed:** Who needs to be informed of progress or decisions?

These are the people who need to be kept informed about the progress or outcomes of the task, but they do not participate in the work itself.

Example: Senior stakeholders who are kept informed about the project’s status and updates.

1. **3-Tier Architecture**

3-Tier Architecture divides an application into three layers.

* **Presentation Layer (Client Tier):** The user-facing layer (UI/UX) where the user interacts with the system (e.g., web or mobile app). It handles input and output but does not process or store data.

Example: The app where farmers browse products and make purchases.

* **Logic Layer (Application Tier):** This middle layer processes business logic. It acts as an intermediary between the user and the database, validating input, processing data, and executing business rules.

Example: Logic that checks if products are in stock or calculates pricing and discounts.

* **Data Layer (Database Tier):** The backend layer responsible for storing, retrieving, and managing data. It ensures that data is safely stored in databases and accessed by the logic layer when required.

Example: A database storing product details, order history, user information, etc.

1. **Use Cases** Use cases describe how a user interacts with a system to achieve a specific goal. They outline the functional requirements of a system from the user's perspective, providing a clear understanding of how the system should behave in various scenarios. The BA needs to understand user roles, actions, and outcomes.

Example: What are the primary actions a farmer should be able to perform when ordering products on the agriculture platform?

What happens when a farmer tries to purchase a product that is out of stock?

1. **Use Case Specifications**

Use case specifications provide a more detailed and structured description of a use case. It includes step-by-step actions, alternate scenarios, preconditions, postconditions, and exceptions, ensuring that the system's behavior is well-defined.

1. **Activity Diagrams**
2. **Models and Page Designs**