A customer can make a payment either by Card or by Wallet or by Cash or by Net banking.

Q1. Draw a Use Case Diagram

Answer:

Use Case Diagram: Use case diagram is a visual representation of the interactions between users (actors) and a system. Here the below use case diagram shows how the primary actors and secondary actors interacts with the systems.

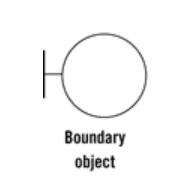


Q2. Derive Boundary Classes, Controller classes, Entity Classes.

Answer:

(i) Boundary Class:

Boundary Classes are responsible for interacting with the external environment, such as the user interface or external systems. They define the interface through which the system communicates with the outside world. The purpose of boundary class is to handle input/output and to ensure that the system does not directly expose its internal logic to the user or external systems.



In this case, all use cases are boundary class. These actors are primary actors. Primary actors means the actors who initiate the use case and interact with the system.

Boundary class are customer registration, customer login, customer logout.

(ii) Controller Class:

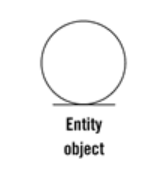
Controller Classes serve as the intermediary between the boundary and entity classes. They handle the application's workflow and implement the logic to control the interaction between boundary and entity objects. The purpose of controller class is to arrange the flow of information between boundary and entity classes. The controller handles user requests (from boundary classes) and invokes appropriate operations on the entities.



In this case, the use case will reflect the actions that the controller class is responsible for performing. Controller classes are registration controller, login controller, payment controller, credentials controllers, Net banking controllers, Email controller, and logout controller.

(iii) Entity Class:

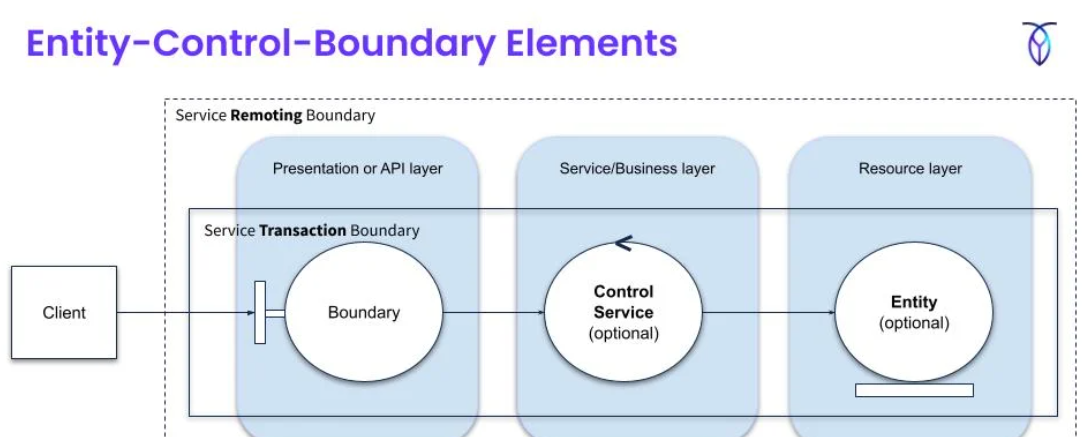
Entity Classes represent the core business logic and data of the application. They encapsulate the application's key objects, their behaviours, and the rules that govern them. The purpose of Entity Class is to hold the data and perform business logic, often interacting with a database or storage system to persist and retrieve data.



In this case, the use cases interact with the entities through the controller class. Entity classes are customer, bank server, cash, card, net banking.

Q3. Place these classes on a three tier Architecture.

Answer:



In 3 tier architecture there are three layers. It organize applications into three logical and physical computing tiers. The Application layer, Business Logic Layer and the database tier, where the data associated with application is stores.

1. Application layer: It is responsible for user interface and user experience. It handles all interaction with user and present data to them. Here we can place boundary class like display application screen, display customer registration page, display customer login page, display payment option boundary, and its full functionality. Application layer send request to business logic layer.
2. Business logic layer: This layer acts as an intermediary between the presentation layer and the database layer – layer contains business rule and logic of the application. It process the user request from the application layer and makes decision based on the business rules. It validate the user data, processing data and implement the business logic. This layer communicate with data base layer to fetch the data. Here we can place Controller class like registration controller, login controller, payment initiated controller, credentials controllers, Net banking controllers, Email controller, wallet controller, and logout controller.
3. Database layer: This layer is responsible for storing and retrieving data from database management systems. It is also known as the data layer or the server layer. It is for storing and managing data and ensures data security and integrity. In this layer we can place Entity Class like Customer, bank account, beneficiary, bank server, wallet, cash, debit/credit card, net banking.

Q4. Explain Domain Model for Customer making payment through Net Banking.

Answer:

Domain model is similar to the entity relationship model. The tables are connected to each

other. In the below diagram, the customer table is connected to bank table, which is why

the customer is able to make payment.

Customer table is also connected to payment table, because he should make the payment. Now the

payment is done by net banking, so payment table is connected to net banking table.

The account is in the bank, so the account table is connected to the bank table. The authentication

table is connected to both net banking table and bank table, because authencation is to be performed

there.

Also, the authentication table is connected to transaction table, because authentication will be done

while transaction.

Difference between ER diagram and domain model-

ER Model – do not have attributes inside the box Domain Model- do have attributes mentioned inside

the box

ER Model – it is a data modelling technique used in database design to represent table

Domain Model- it is a conceptual model that represents real world entities

Domain model is similar to the entity relationship model. The tables are connected to each

other. In the below diagram, the customer table is connected to bank table, which is why

the customer is able to make payment.

Customer table is also connected to payment table, because he should make the payment. Now the

payment is done by net banking, so payment table is connected to net banking table.

The account is in the bank, so the account table is connected to the bank table. The authentication

table is connected to both net banking table and bank table, because authencation is to be performed

there.

Also, the authentication table is connected to transaction table, because authentication will be done

while transaction.

Difference between ER diagram and domain model-

ER Model – do not have attributes inside the box Domain Model- do have attributes mentioned inside

the box

ER Model – it is a data modelling technique used in database design to represent table

Domain Model- it is a conceptual model that represents real world entities

Domain model is similar to the entity relationship model. The tables are connected to each

other. In the below diagram, the customer table is connected to bank table, which is why

the customer is able to make payment.

Customer table is also connected to payment table, because he should make the payment. Now the

payment is done by net banking, so payment table is connected to net banking table.

The account is in the bank, so the account table is connected to the bank table. The authentication

table is connected to both net banking table and bank table, because authencation is to be performed

there.

Also, the authentication table is connected to transaction table, because authentication will be done

while transaction.

Difference between ER diagram and domain model-

ER Model – do not have attributes inside the box Domain Model- do have attributes mentioned inside

the box

ER Model – it is a data modelling technique used in database design to represent table

Domain Model- it is a conceptual model that represents real world entities

Domain modeling is the tool used for constructing visualization of the objects, associations, attributes, as well as dynamic features within a particular domain.

* It plays a significant role in the sense that it assists software engineers and designers deepen understanding and knowledge about the domain for them to create solutions that are well-aligned with the real world.
* Domain model design contains a blueprint that forms the structure and provides the functionality of any software system.



Q5. Draw a sequence diagram for payment done by Customer Net Banking.

Answer:

Sequence diagram is a key component of Unified Modeling Language (UML) used to visualize the interaction between objects in a sequential order. It focuses on how objects communicate with each other over time, making it an essential tool for modeling dynamic behavior in a system. Sequence diagrams illustrate object interactions, message flows, and the sequence of operations, making them valuable for understanding use cases, designing system architecture, and documenting complex processes.



Q6. Explain Conceptual Model for this Case.

Answer:

A conceptual model is an abstract representation that outlines the key concepts and their relationships within a specific system or domain. It provides a high-level view of how a system works, focusing on its essential components, interactions, and structures, without getting into detailed specifications or implementation.



The Conceptual Model is designed to ensure security, authentication, and successful completion of the payment process.

Q7. What is MVC architecture? Explain MVC rules to derive classes from use case diagram and guidelines to place classes in 3-tier architecture.

Answer:

MVC architecture:

The Model-View-Controller (MVC) framework is an architectural pattern that separates an application into three main logical components Model, View and Controller.

View: Represent the presentation layer of the application. Displays the information from the Model to the user, handling the visual presentation on the screen without directly manipulating data.

Model: Represent the data and business logic of the application. Responsible for storing and managing application data, including data validation and business logic related to that data.

Controller: Acts as an intermediary between Model & View. Acts as the intermediary, receiving user input, deciding which Model operations to perform based on that input, and then selecting the appropriate View to display the updated data.

Rules of MVC Architecture:

1. Combination of one actor and a use case results in one boundary class.
2. Combination of two actors and a use case result in two-boundary class.
3. Combination of three actors and results in three-boundary class.
4. Use case will result in a controller class.
5. Each actor will result on one entity class.

In this case of online payment, below classes are derived from use case.

1. Model classes: Customer, Payment, Net banking, Cash, Card. (Entity class)
2. View classes: Registration view, Login view, Payment option view, Net Banking view, Bank Selection view, Credentials view, Payment Amount view, Payment confirmation view, Logout view. (Boundary Class)
3. Controller class: Login controller, Payment option controller, Nat banking Controller, Bank Selection controller, Credentials controller, Payment Amount Controller, Payment confirmation controller, Logout controller. (Controller class)

Guidelines to Place Classes in 3-Tier Architecture

A 3-Tier Architecture consists of three layers:

1. Presentation Layer: This is where the user interacts with the system. This layer is nothing but a user interface. View is inside this layer. It is responsible for displaying information and receiving the input form the user.
2. Business Logic Layer: This contains the business logic and data validation. Model and controller are inside. Controller handles the user input, process the request and co-ordinates between model & view.
3. Data base Layer: This layer is responsible for interacting with database, files or external systems to store and retrieve data.

Q8. Explain BA contributions in project (Waterfall Model – all Stages)

Explain BA contributions in project (Waterfall Model–all Stages)

:

A waterfall model is very old and traditional model in IT industries. It is a progressive

implementation of the projects which is divided into different phrases of SDLC.

Explain BA contributions in project (Waterfall Model–all Stages)

:

A waterfall model is very old and traditional model in IT industries. It is a progressive

implementation of the projects which is divided into different phrases of SDLC.

Explain BA contributions in project (Waterfall Model–all Stages)

:

A waterfall model is very old and traditional model in IT industries. It is a progressive

implementation of the projects which is divided into different phrases of SDLC.

A waterfall model is very old and traditional model in IT industries. It is a progressive

implementation of the projects which is divided into different phrases of SDLC

A waterfall model is very old and traditional model in IT industries. It is a progressive

implementation of the projects which is divided into different phrases of SDLC

In the **Waterfall Model**, Business Analysts (BAs) contribute at each stage of the project lifecycle to ensure that the system being built aligns with the business needs. The BAs ensure requirements are accurately captured, communicated, and validated, and they play a key role in helping the team meet business objectives. Below is a detailed breakdown of BA contributions at each stage along with the artifacts they typically produce or work with:

1. Requirement Gathering

From the initial stage, there is an involvement of the business analyst. Business Analyst is responsible for preparing Business Requirement Document.

Artifacts: Business Requirement Document

1. Requirement analysis:

Requirement analysis plays a critical role in the success of any software development project. It sets the foundation for the entire development process by ensuring that both the development team and stakeholders have a clear and shared understanding of what needs to be built.

Artifacts: Functional Requirement Specifications, Software Requirement Specifications, Requirement Traceability Matrix, Stakeholder Analysis.

1. Designing

In this phase, the architect will start designing the system based on the business analyst’s input and requirement document. Business Analyst helps him to clear his doubts about the requirements.

Artifacts: Design documents and UML diagrams get ready in this phase.

1. Development: This phase is quite lengthy as the core development start in this phase. Developers start product development based on the requirement document prepared by BA. Developers may ask questions to BA regarding the requirement and BA needs to answer their questions as and when required.

Artifacts: BA will help in providing clarification to developers on requirements.

1. Testing: After coding, the testing phase will start. In this phase, business analyst help the testing team to understand the requirements so that they will build proper functional test cases. Business Analyst has to review whether the test case covering the whole functionality.

Artifacts: Test Case & test Results

1. Deployment: Once the code is developed and tested, it is ready to deploy in the production environment. Business Analyst will verify the product is delivered as per the requirements and it is meeting the business needs.

Artifacts: Implementation Review Document

1. Maintenance: Once the implementation is done the team has to give support by installing patches, handling change request etc. Business Analyst is a person who knows every nook and corner of the project. So every change request has to reviewed by him and based on his inputs and reports the team will respond.

Artifacts: User satisfaction review, change request review.

Throughout the Waterfall process, the BA ensures that the final product is aligned with the business objectives by actively participating in requirement gathering, design, development, testing, deployment, and post-implementation phases.

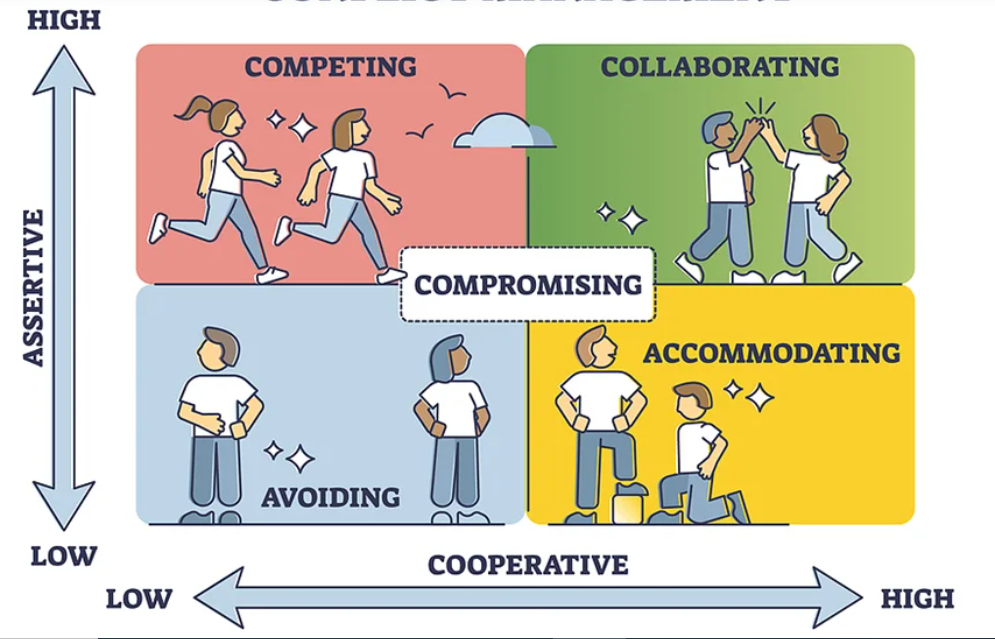
They produce key Artifacts like BRDs, FRDs, use cases, test plans, and training manuals, and they assist in change management by documenting and managing change requests. BAs ensure that all requirements are traceable from the initial phase to the final deliverables, ensuring the system addresses all business needs effectively.

Q9. What is conflict management? Explain using Thomas – Kilmann technique.

Answer:

Conflict management refers to the process of identifying, addressing, and resolving conflicts in a way that benefits all parties involved. It involves understanding the nature of the conflict, applying strategies to minimize negative effects, and fostering cooperation to reach a resolution.

One widely used model for conflict management is the Thomas-Kilmann Conflict Mode Instrument (TKI). This technique, developed by Kenneth Thomas and Ralph Kilmann in the 1970s, identifies five different styles or strategies people use to approach conflicts. These styles are based on two dimensions: assertiveness (the extent to which you try to satisfy your own concerns) and cooperativeness (the extent to which you try to satisfy the concerns of others).



Below are the conflict-handling styles from the Thomas-Kilmann model:

1. High Assertiveness and high cooperativeness: Collaboration – Means working together to find a solution.
2. High Assertiveness and low cooperativeness: Competition – Means defensive that is standing for your individual beliefs and trying to win.
3. Low assertiveness and high cooperativeness: Accommodation – Stakeholder will prioritise their needs over others.
4. Low assertiveness and low cooperativeness: Avoidance – means ignoring the conflict.

The Thomas-Kilmann model emphasizes the need for flexibility and encourages individuals to manage conflict in a way that can lead to healthier, more productive outcomes.

Q10. List down the reasons for project failure.

Answer:

Project failure in software engineering can stem from a variety of factors. These failures often arise from poor planning, miscommunication, technical challenges, and lack of effective management. Below are the of common reasons for project failure.

1. Unclear Project Requirements

Lack of clear, well-defined requirements can lead to confusion, misunderstandings, and missed expectations. If stakeholders do not communicate their needs clearly or changes are not properly managed, the final product may not meet the intended goals.

2. Poor Project Planning

Inadequate planning can result in unrealistic timelines, insufficient resources, and a lack of risk management strategies. Poor planning can also lead to missing key project milestones and deadlines.

3. Lack of Skilled Resources

Not having the right skills within the team can hinder project success. If team members lack necessary technical expertise or experience with specific technologies, the project may fail to meet its objectives.

4. Poor Communication

Communication breakdowns among stakeholders, team members, and project managers can lead to misunderstandings, lack of coordination, and missed deadlines. This includes failure to communicate updates or project changes effectively.

5. Scope Creep

Uncontrolled changes or continuous addition of new features (without adjusting timelines or budgets) leads to scope creep. When the scope of the project grows beyond the original plans, it can overwhelm resources and timelines.

6. Inadequate Risk Management

Failure to identify, assess, and mitigate risks early in the project can lead to unexpected challenges. Without risk management plans, problems like technical failures, staffing issues, or unforeseen market shifts can derail the project.

7. Poor Time Management

If project timelines are not managed effectively, tasks may take longer than expected, and critical milestones might be missed. Inadequate estimation of time needed for tasks or a lack of contingency time can disrupt progress.

8. Inadequate Testing and Quality Assurance

Insufficient testing or skipping quality assurance processes can result in bugs, performance issues, and security vulnerabilities in the final product. This can happen due to time constraints, lack of resources, or oversight.

9. Lack of Stakeholder Engagement

If stakeholders are not adequately involved or engaged throughout the project, their needs might not be fully understood or addressed. This can lead to misalignment between what the developers produce and what the stakeholders expect.

10. Technological Challenges

Incompatibilities, outdated technologies, or incorrect tool choices can impact the project’s ability to succeed. If the technology is not suited to the project needs or if the team is unfamiliar with the tools, it may result in inefficiencies or failures.

Each of these factors, individually or in combination, can contribute to the failure of a software project. By recognizing and addressing these issues early, teams can better navigate challenges and increase the chances of successful project delivery.

Q11. List the Challenges faced in projects for BA.

Answer:

Business analysts (BAs) play a crucial role in bridging the gap between stakeholders and technical teams in any project. However, they often face several challenges throughout the project lifecycle. Here are some of the common challenges that business analysts encounter in projects.

1. Unclear Requirements

Business requirements are often unclear, incomplete, or poorly defined, making it difficult for BAs to gather accurate and actionable information. This leads to misunderstandings, scope creep, and misalignment with business objectives, ultimately causing delays and rework.

2. Scope Creep

Managing scope changes and ensuring that new requirements are properly evaluated and integrated into the project is a constant challenge. Without proper change control, the scope can expand unintentionally. Scope creep can lead to delays, budget overruns, and resource strain, causing the project to go off course.

3. Lack of Proper Documentation

Incomplete, ambiguous, or poorly structured documentation can hinder communication and prevent clear understanding of the requirements, processes, and deliverables. Poor documentation can lead to misunderstandings, missed requirements, and quality issues in the final product.

4. Time Constraints

BAs are often expected to deliver comprehensive analysis and documentation under tight timelines. Rushed work can compromise the quality of the analysis and decision-making. Incomplete analysis or rushed documentation can lead to errors, gaps in requirements, or delays in project delivery.

5. Communication Breakdowns

Effective communication between business stakeholders, technical teams, and vendors is essential. Poor communication can result in misunderstandings and missed expectations. Miscommunication can lead to incorrect requirements being delivered, delays, and frustration from both stakeholders and development teams.

6. Balancing Business Needs with Technical Constraints

Business analysts need to bridge the gap between business goals and technical feasibility. Sometimes business needs conflict with the technical capabilities or limitations of the solution. Unmet expectations, delays, or compromised solutions may result if the BA fails to effectively balance these aspects.

7. Lack of Stakeholder Engagement

If stakeholders are not actively engaged or available to provide timely feedback, it becomes difficult for the BA to gather necessary insights and make informed decisions. Poor engagement can lead to gaps in requirements, misalignment of expectations, and a product that does not fully meet stakeholder needs.

8. Technical Complexity

BAs may struggle to understand highly technical details, especially when working on complex systems or with new technologies. This can create difficulties in accurately translating business needs into technical specifications. Misunderstanding technical limitations or capabilities can result in unrealistic requirements or flawed solutions.

9. Inadequate Training or Knowledge

Lack of proper training in new methodologies, tools, or industry practices can be a barrier for BAs to effectively perform their tasks. Inadequate knowledge or skill gaps can lead to ineffective analysis, poorly defined requirements, and a failure to leverage best practices.

10. Quality Assurance and Testing

Business analysts often have limited involvement in testing and quality assurance, yet their requirements must be accurately translated into the final product. If the requirements are not clear or precise, testing can be compromised. Inaccurate requirements lead to defects in the system, failure to meet business expectations, and potential rework in later stages.

These challenges highlight the complexity of the business analyst’s role. By developing effective communication, management, and problem-solving skills, BAs can help mitigate these issues and increase the likelihood of project success.

Q12. Write about Document Naming Standards.

Answer:

Document Naming Standards are a set of guidelines and conventions designed to standardize how documents are named within an organization, project, or team. These standards ensure consistency, clarity, and ease of organization for files and documents, making it easier for teams to locate, reference, and manage them. In environments like software development, business analysis, or project management, document naming conventions are crucial for maintaining an efficient workflow.

Example: If we have a project with the ID “PROJ123” and we are working with a Business Requirement Document.

Project ID: PROJ123

Document type: BRD

Version: 1.0

Date: 2025/10/02

The document identifier would be PROJ123\_BRD\_1.0\_2025/10/02

By following document naming standards, Business Analysts can ensure that their work is organized, traceable, and easy to manage. This consistency not only helps BAs themselves but also facilitates collaboration across teams, reducing errors, miscommunication, and the risk of losing important documents. Implementing clear and well-thought-out naming conventions is essential for any software engineering project, making it more efficient and effective overall.

Q13. What are the Do’s and Don’ts of a Business analyst?

Answer:

|  |  |  |
| --- | --- | --- |
| Sr. No. | Do’s | Don’t’s |
| 1 | Consult an SME for clarification in requirements. | Never say NO to the client. |
| 2 | Go to the client with a plain mind with no assumptions. Listen carefully and completely until the client is done and then you can ask queries. | There is no word as “By Default” |
| 3 | Try to extract maximum leads to the solution from the client himself. | Never imagine anything in terms of GUI. |
| 4 | Concentrate on the important requirement. | Don’t interrupt the client when he is giving you a problem. |
| 5 | Question everything. | Never try to give a solutions to the client straight away with your previous experience and assumptions. |
| 6 | Don’t Promise Solutions You Can’t Deliver | Do Focus on Value Delivery |

Q14. Write the difference between packages and sub-systems.

Answer:

Packages: A package is a collection or grouping of related classes, interfaces, and other components (such as modules or sub-packages) that share common functionality. It is mainly used for organizing code within a software project.

In programming languages like Java, packages help avoid name conflicts and improve code maintainability by logically grouping related classes.

Sub-system:

A sub-system refers to a more complex, independent system or module that exists within a larger system. It usually provides a specific set of functionalities or services and interacts with other sub-systems in the overall system.

A sub-system may include multiple packages, libraries, and even external components, and often operates as a cohesive unit within the software architecture.

Q15. What is Camel-casing and explain where it will be used.

Answer:

Camel Casing is a naming convention used in used in Computer Programming. It is used for naming variables, functions and identifications. This style resembles the humps of a camel, hence the name "camel case."

For example:

firstName

getUserInfo

userProfilePicture

calculateTotal

getUserData

In Camel Casing, the first words starts with a lower case letter and each subsequent word begins with an uppercase letter.

Q16. Illustrate Development server and what are the accesses does business analyst has?

Answer:

A Development Server is an environment where software developers can build, test and debug applications or features before they are released to a production environment. It allows teams to deploy, test, and troubleshoot code in a controlled setting.

A development server refers to a dedicated environment or server that is used during the software development process.

As a Business Analyst, we have limited access only.

1. Read-Only Access to Data – Reports, Test Results, User Interface Access
2. Approval, feedback & documentation – BA can access development server for UAT where they perform testing from a business perspective.
3. Communication with developers – BA may request specific queries or logs to be provided from the development server.

A Business Analyst's role is more about ensuring that the system being developed meets business goals. They might have limited or read-only access to the development server, primarily for activities like verifying data integrity, reviewing test results, or running basic business-oriented tests (such as User Acceptance Testing).

In summary, the BA will not usually have full access to code or backend functionalities but will be able to validate features, view reports, and provide feedback during development and testing.

Q17. What is Data Mapping?

Answer:

Data Mapping is the process of connecting data from one source to another. It is like creating a guide or map that shows how data in one place corresponds to data in another place.

This is especially important when you are moving data between different systems or databases to ensure that the data stays consistent and accurate.

Rules of Data Mapping:

1. One-to-One Mapping:

A field in the source model corresponds to exactly one field in the target model.

1. One-to-Many Mapping:

A field in the source system may map to multiple fields in the target system.

1. Many-to-One Mapping:

Multiple fields in the source model are mapped to a single field in the target system.

1. Complex Mapping:

The mapping involves conditions, calculations, or transformations, such as converting data types (e.g., converting a string to a date) or merging fields.

Steps in Data Mapping:

1. Identify Source and Target Systems: Understand the structure, format, and fields of both the source and target systems.
2. Define Data Elements: List the fields from the source system that need to be transferred to the target system.
3. Create the Mapping Rules: Decide how each field in the source system will map to fields in the target system (e.g., one-to-one, one-to-many, etc.).

Define any necessary transformations, such as data type conversion, concatenation, or calculations.

1. Test the Mapping: After defining the mapping, test the data flow between systems to ensure accuracy and completeness.
2. Execute the Data Transfer: Once the mapping is verified, perform the actual data migration, integration, or synchronization.
3. Monitor and Maintain: Continuously monitor data flow and correct any issues that arise due to changes in data structure or system updates.

In summary, **data mapping** is a fundamental process for enabling different systems, databases, or applications to work together effectively by ensuring that data is correctly transferred and transformed according to predefined rules.

Q18. What is API. Explain how you would use API integration in the case of your application Date format is dd-mm-yyyy and it is accepting some data from Other Application from US whose Date Format is mm-dd-yyyy.

Answer: An API (Application Programming Interface) is a set of rules, protocols, and tools that allows different software applications to communicate with each other. It defines how requests and responses should be structured and enables different systems to interact, exchange data, and perform operations without having to understand each other's internal workings.

Summary of Steps for Date Format Integration:

Step 1: When receiving data from the external system, identify the date field and parse it (assume the format is mm-dd-yyyy).

Step 2: Convert the date to dd-mm-yyyy format (using programming language functions).

Step 3: Store or process the date in the desired format (dd-mm-yyyy).

Step 4: If sending data back, convert the date from dd-mm-yyyy to mm-dd-yyyy before sending.

Step 5: Implement error handling for invalid dates.

Step 6: Test the conversion thoroughly.

By implementing this approach, you can seamlessly integrate date data from the external system, ensuring that it is correctly formatted and understood in your application without causing errors or confusion for end-users.