

Capstone Project-3 Prep-1

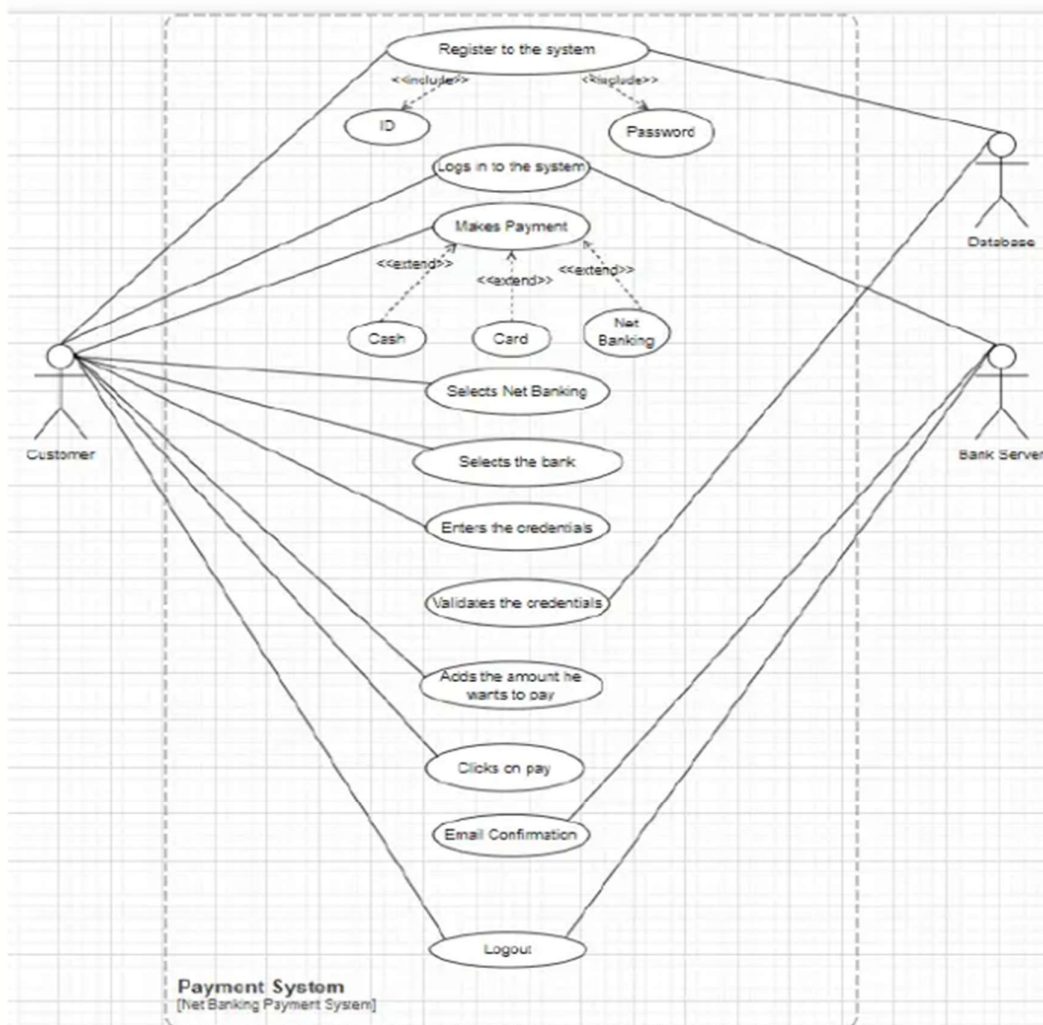
Srishti Gupta

Batch -05/10/2024

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

Question 1- Draw the Use case diagram

Answer-



Question 2- Derive Boundary Classes, Controller classes, Entity Classes.

Answer-

Boundary classes: Boundary Classes are a type of class in software design, particularly in Object-Oriented Analysis and Design, which act as intermediaries

between the system and external entities. These external entities could be users, external systems, or devices. **The role of a boundary class is to handle the interaction between the system and the outside world.**

Boundary class (All use cases)

[**Combination of 1** actor and a use case is one boundary class]

[**Combination of 2** actors and a use case is two boundary class]

[**Combination of 3** actors and a use case is three boundary class]

and so on and those actors should be primary actors. Primary actors means the actors who initiate the use case and interact with the system.

Eg: Customer Registration, Customer Login, Bank Server Login, Customer Logout and Bank Server Logout.

Controller classes: Controller classes are a type of class in Object-Oriented Analysis and Design (OOAD) that handle the flow of control and manage the interactions between Boundary classes (responsible for system interfaces) and Entity classes (responsible for data and business logic). Their primary role is to process incoming requests, coordinate activities, and direct data between other components of the system.

Controller class (handles user input and process the data) Use case will be considered as the controller classes

Eg: Registration Controller, Login Controller, Payment Controller, Credentials Controller, Net Banking Controller, Email Controller and Logout Controller

Entity Classes: Entity classes are a type of class in software design, particularly in Object-Oriented Analysis and Design (OOAD), that represent the core business objects of a system. These classes encapsulate data and the logic (business rules) that operates on that data

Entity Class (All actors) Each Actor will be considered as one entity. Customer, Bank Server, Cash, Card and Net banking

Eg: Customer, Bank Server, Cash, Card and Net banking

Question 3 - Place these classes on a three tier Architecture.

Answer: -Tier Architecture is a software design pattern that separates an application into three logical layers:

Presentation Tier (user interface)- handles user interaction,

Application Tier (business logic),- processes business rules and workflows,

Data Tier (database)- stores and retrieves information. This separation improves scalability, maintainability, flexibility, and security, allowing each tier to be developed, managed, and scaled independently.

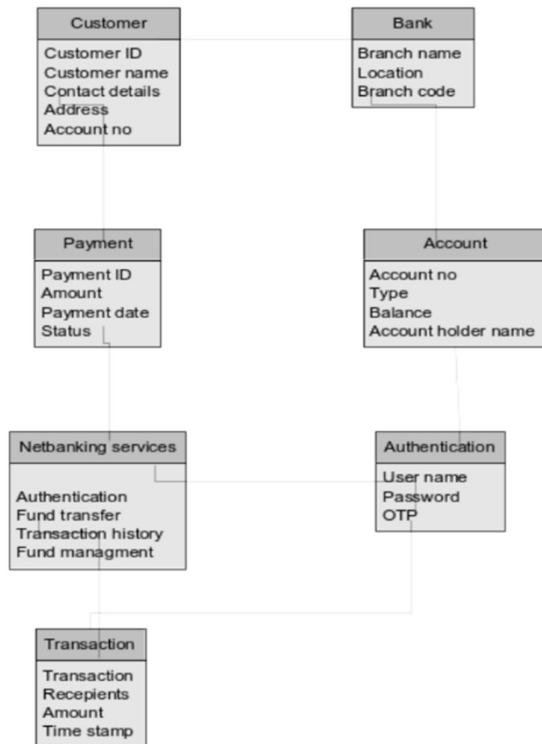
Application Layer	Customer Registration Customer Login Bank Server Login
Business Logic Layer (Primary actors associated with the Boundary class)	Customer Bank Server
Data Layer (All the entity classes)	Customer Bank Server Cash Card Net banking

Question 4. Explain Domain Model for Customer making payment through Net Banking

Answer: A domain model is a conceptual representation of the key entities, their attributes, relationships, and behaviours within a specific problem domain. **It serves as a blueprint to understand and design the core functionality of a system by modelling real-world objects and their interactions.** The domain model is typically visualized using UML (Unified Modelling Language) class diagrams or similar tools and is central to object-oriented software design. It bridges the gap between the business requirements and the technical implementation.

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

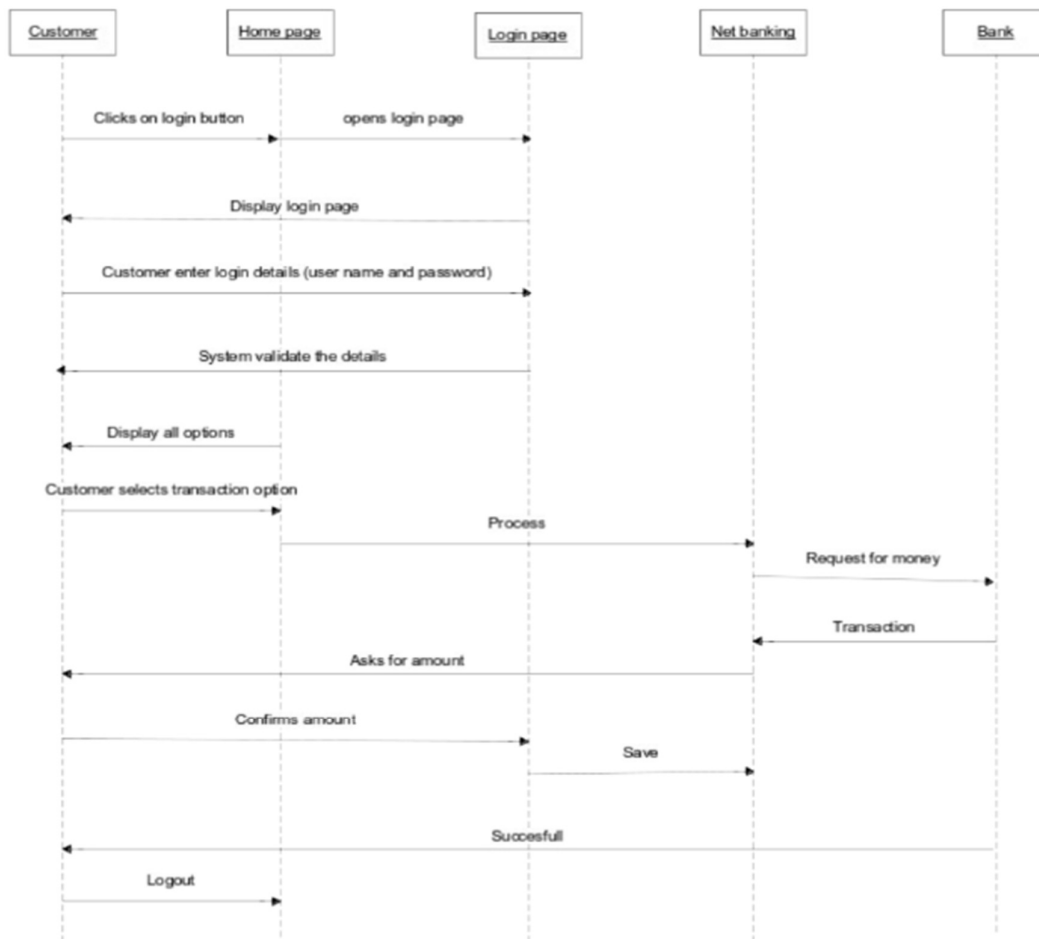
In the below diagram, the customer table is connected to bank table, which is why the customer is able to make payment. Customer table is also connected to payment table, because he should make the payment. Now the payment is done by net banking, so payment table is connected to net banking table. The 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 authentication is to performed there. Also the authentication table is connected to transaction table, because authentication will be done while transaction.



Question 5. Draw a sequence diagram for payment done by Customer Net Banking

Answer: This diagram shows how the objects in the system interact and communicate with each other with time to achieve specific task. Developer will draw this. It is used to show the flow of messages, events or actions between the objects of the system.

Below diagram helps to visualize the behaviour of the system.



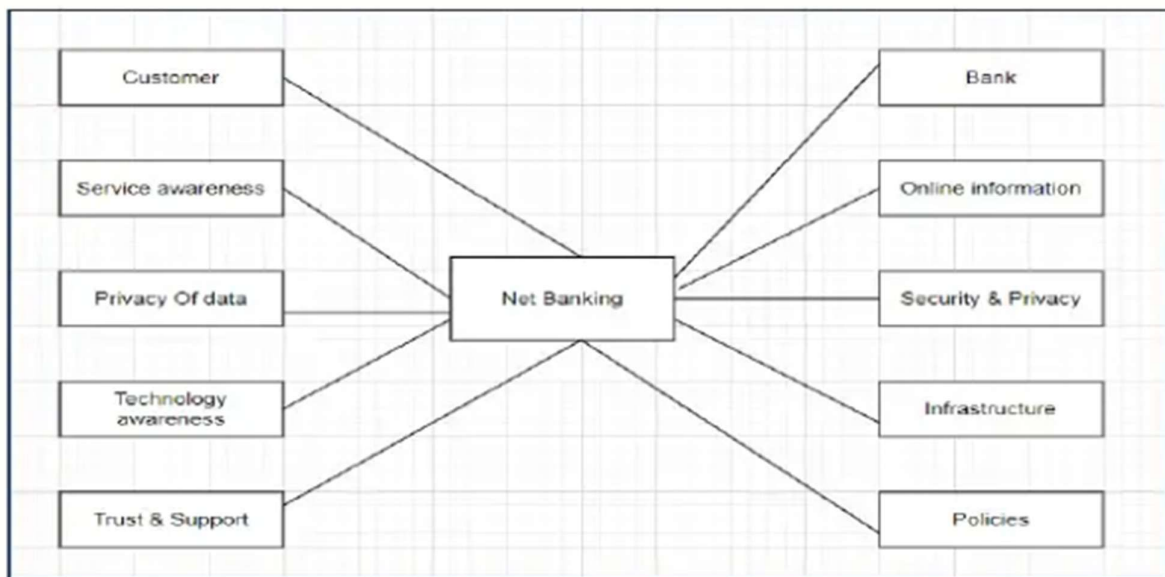
Question 6. Explain Conceptual Model for this Case?

Answer: A Conceptual Model is a **high-level, abstract** representation of the system or domain that illustrates the key concepts, entities, relationships, and overall structure of the problem space without focusing on technical details or implementation specifics. It serves as a bridge between the business perspective and the technical design.

The relationships between these entities can be described as follows:

1. **Customer** - This node represents the customers or users of net banking services.
2. **Service Awareness** - Customers should be aware of the available net banking services and their features.
3. **Privacy of Data** - The importance/significance of this node is to protect the privacy and confidentiality of customer data in the context of net banking.
4. **Technology Awareness** - The significance of this node is that customers should be aware and comfortable with the underlying technology used in net banking services.
5. **Trust & Support** - This node indicates that the bank provides such good services that it will help to enhance the customer's trust.

6. **Bank** - This node represents a service provider responsible for offering net banking services.
7. **Online Information** - This aspect highlights the importance of providing accurate and up-to-date online information about net banking services to customers.
8. **Security & Privacy** - the bank should adapt the security policies which will help the customers to keep their data related to their transaction secure and private.
9. **Infrastructure** - This component suggests that the underlying technological infrastructure, including hardware and software systems, plays an important role in enabling net banking services
10. **Policies** - This node represents the various policies and regulations that govern the implementation and operation of net banking services, ensuring compliance and customer protection.



Question 7. 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 is a design pattern where, the application is divided into 3 logical parts-Model, View and Controller. Each of these parts will have specific responsibility.

Model - The Model represents the data and the business logic of the application. Model is responsible for multiple tasks like managing the application's data, performing data validation, implementing business rules, and handling data access operations. Model does not depend on how the data is presented or how the user interacts with the application.

The model class is known about all the data that is needed to be displayed. This layer corresponds to the data-related logic that the user works with. It represents the

data that is being transferred between View and Controller. It can add or retrieve the data from the database.

It responds to the controller's request because the controller cannot interact with the database by itself. The model interacts with the database and give the requested data. All the model classes are nothing but the entities. Model classes are represented as entity class.

View - The View is responsible for presenting the data to the user for handling the user interface. The View can be a web page, a desktop application window, or any other form of user interface. It receives input from the user and passes it to the Controller for processing.

It represents the presentation of the application. View refers to the model. It takes the data from the Model and renders it in a way that is suitable for the user's display or interaction.

For rendering the data, it uses query method. View does not depend upon application logic. View class are represented as boundary class.

Controller - The Controller acts as an intermediary between the Model and the View. It receives input from the user (via the View), processes the input by invoking the appropriate methods in the Model, and then updates the View with the new data or state. The Controller handles user interactions, interprets user input, and translates it into instructions for the Model or the View. It coordinates the flow of data between the Model and the View, ensuring that they remain separated and independent of each other. Whenever the user requests for anything, that request first goes to the controller. Controller works on the user's request.

Takes input from the user/ client. It interacts with the model and view. Controller class represents as use case. Controller acts as a mediator between model and database.

Controller cannot directly get the data from the database. So controller interacts with the model.

Advantages of MVC - MVC has the feature of scalability, which in turn helps the growth of application. The components are easy to maintain. A model can be used by multiple views that provide reusability of code. By using MVC, the application becomes more manageable. As all the three layers are different and independent, they are maintained separately

Rules to derive the classes from use case diagram –

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

Consider the example of Online shopping application with the following use case:

Model Classes - Customer, Payment, Net Banking, Card, Cash

View Classes - Login View, Payment Option View, Net Banking View, Bank Selection View, Credentials View, Payment Amount View, Payment Confirmation View, Logout View

Controller Classes- Login Controller, Payment Option Controller, Net Banking Controller, Bank Selection Controller, Credentials Controller, Payment Amount Controller, Payment Confirmation Controller, Logout Controller

Guidelines to place classes in 3-tier architecture-

Presentation Layer - This layer is nothing but a user interface. View is inside this layer. These classes interact directly with the user. Presentation layer is responsible for displaying information and also receiving the input from the user.

Application Layer - This layer is nothing but business logic. Model and controller are inside this layer. Controller handles the user input, process the request and co-ordinates interaction between the model and view.

Data Layer - Classes which are responsible for data access and storage are in this layer. It contains the classes which interacts with the database, files and other data sources.

Question 8. Explain BA contributions in project (Waterfall Model - all Stages)?

Answer: Waterfall model is useful in the situation where the **project requirements** are **well defined** and the **project goals** are **clear**.

Waterfall model follows sequential approach.

In this model each phase is completed entirely and then only moved to the next phase.

Waterfall model relies on documentation to ensure that the project is well defined and project team is working toward clear goals. Once that particular phase has been completed and once we move to the next phase, we cannot go back to the previous phase to make changes.

This model is stable for the projects when the requirements are clear
Requirements Gathering -

First, the stakeholders are identified. In this phase, all the requirements are gathered from the stakeholder. BA and Project Manager participate in this phase. After completing this phase, BRD will be generated.

Requirements Analysis - The requirements are analysed to understand the scope of the project. Analysing means the BA will check all the requirements, if he finds conflicting requirements then the BA will talk to the concerned stakeholder to clear it, remove the ambiguous requirements. BA will prepare functional requirement. The

document which contains the functional requirements is called (FRS). [Functional Requirement Specifications].

Technical team will prepare non- functional requirement.

The document which contains the non-functional requirements is called (SSD).

[Supplementary Support Document]

BA will combine FRS and SSD to form SRS. [Software Requirement Specifications]

BA will prepare RTM by referring SRS.

Design - After the requirements are cleared, Design phase starts. This has a detailed design document

This has detailed document that outlines the software architecture, user interface, and system components.

HDD, ADD and solution document will be generated here. [High level Design Doc.]

BA Collaborate with designers, architects, and developers to translate requirements into system design.

BA Ensure that the design aligns with the documented requirements and addresses stakeholder needs

Development - The Development phase include implementation. It involves coding the software based on the design specifications. Programmers or developer are involved in this phase. Here BA acts as a mediator between the development team and the stakeholders.

BA clarifies the requirements, check if the development is going on right track or not. BA also participates in scrum meetings

Testing - In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects.

- Testers are involved in this phase.
- Test documents are generated here.
- BA works with the testing team to ensure that the solution meets the requirements.
- BA facilitate UAT.
- BA helps the users to know the functionality of the system and also helps them to use the system.

Deployment - Once the software has been tested and approved, it is deployed to the production environment.

BA ensures that there is smooth transition from development phase to the production phase.

Implementation - This is the final stage of waterfall model.

- It involves running the code for the very first time in production phase.
- Release manager handles this phase.
- BA will update documentation and requirements specifications to reflect changes in the system over time

Maintenance - Running the code for second time in the production phase is called maintenance. This is done by support team.

Question 9. What is conflict management? Explain using Thomas – Kilmann technique?

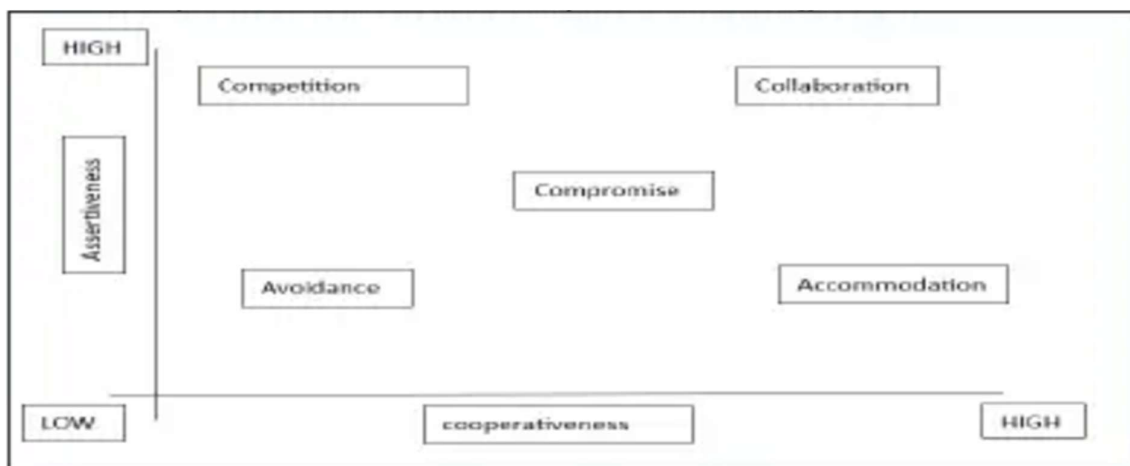
Answer: **Conflicts** can occur due to various reasons, such as differences in goals, values, personalities, resources, or communication breakdowns.

Conflict is an inevitable part of any workplace. So it is important to resolve it to promote learning and growth.

Conflict management is nothing but the process of **identifying** and **addressing** **conflicts** in a **healthy** and **constructive manner**.

- It consists of strategies and techniques aimed at resolving disputes, disagreements, or differing perspectives among individuals or groups.
- By identifying the conflicts efficiently, it will in turn be helpful to reduce negative impact and increase positive impact.
- It is a process or skill to and crave ways to handle the disagreement.

Thomas - Kilmann approach is widely used to recognize the approaches for conflict management.



Y axis- assertiveness x axis- cooperativeness

High assertiveness and High cooperativeness - Collaboration- means working together to find solution.

High assertiveness and Low cooperativeness - Competition- means defensive that is standing for your individual beliefs and trying to win.

Low assertiveness and High cooperativeness - Accommodation- stakeholder will prioritize their needs over others.

Low assertiveness and Low Cooperativeness - Avoidance- means ignoring the conflict assertiveness - the extent to which the person attempts to satisfy his own concerns.

Assertiveness- the extent to which the person attempts to satisfy his own concerns.

Cooperativeness- the extent to which the person attempts to satisfy the other persons concerns.

Question 10. List down the reasons for project failure?

Answer: Reasons for project failure are

- **Improper requirement gathering**- If the requirements of the project are not gathered correctly, then this can lead to project failure.
- **Lack of stakeholder involvement**- A project can fail if the stakeholders are not participating in the process. The stakeholders input and feedback plays very important role to meet the goals.
- **Ineffective or less communication**- If there are communication issues between stakeholders, team members then this can lead to misunderstandings or delays in project or even can lead to project failure.
- **Continuous change in the requirement**- If the requirements keep on changing frequently, this can also lead to project failure. Because the scope of the project will also keep on changing which will lead to project failure.
- **Poor risk management**- Poor risk management can also lead to project failure. The team fails to identify the risks and do the risk mitigation, which can lead to unexpected challenges or delays in project. Lack of user involvement. Lack of executive support.
- **Unrealistic expectations**- means the goals that cannot be achieved or the goals that are out of scope
- **Improper planning**- The project can fail if the planning is not done properly. The milestones, goals should be discussed. If there is no proper planning, then team may face difficulties in addressing the issues or to track the progress.
- **Insufficient resources**- Insufficient resources can also lead to project failure. The project may fail due to lack of technology knowledge or lack of finances.

Question 11. List the Challenges faced in projects for BA?

Answer:

- Lack of training.
- Obtaining sign-off on the requirement.
- Change management.

- Co-ordination between developers and testers.
- Conducting meeting.
- Making sure status report is effective.
- Driving clients for UAT completion.
- Making sure that the project is going on right track and delivered as per the timelines without any issues.
- Gathering clear and unambiguous requirements can be challenging.
- Unable to understand what stakeholder is trying to convey.
- Scope creep- change in requirement or scope of the project during the project lifecycle can lead to scope creep.
- Managing the stakeholder with conflicting interest can be a difficult task for BA
- BA may face difficulties in understanding the requirements if the domain is not familiar to him.
- Poor communication between stakeholder and BA can affect the process of gathering the information.
- Technical complexity.

Question 12. Write about Document Naming Standards?

Answer: File Naming Standards are used to save the file with particular name or format. This is important in sharing and keeping track of data files.

The following are the best standards in Naming Convention –

1. It should be Named Consistently.
2. File names should be short (<25 characters)
3. Avoid special characters or spaces in a file name.
4. Use Capital and Underscores instead of spaces or slashes.
5. Use date format as per ISO 8601: YYYYMMDD.
6. Include a version number.
7. Write down naming convention.

We must consider following naming conventions -

- Date of Creation
- Short Description
- Work
- Location
- Project name or number
- Sample
- Analysis
- Version Number

For example - We have a project with ID "PROJ45fIBANK" and we are working with Requirement Specification Document then –

Project ID - PROJ45fIBANK

Document Type - REQ

Version - 1.0

Date - 2024-12-18

Then the naming convention of the document will be "PROJ45fIBANK-REQ-1.0-2024-12-18".

Question 13. What are the Do's and Don'ts of a Business analyst?

Answer:

- Never say "NO" to the client.
- There is no word called as "BY DEFAULT".
- Never imagine anything in terms of GUI.
- Question everything in the world
- Go to the client with plain mind i.e. with no assumptions.
- Listen to the client very carefully and after he is done, then ask question.
- Don't interrupt the client.
- Never try to give solutions to the client right away.
- Try to concentrate only on important and required things.
- Be like a lotus in mud- if a client comes with a fancy requirement, then talk to the project manager first.
- Requirement hurried-project buried.
- Never criticize the stakeholder.
- Always appreciate the stakeholder even for small efforts.

Question 14. Write the difference between packages and sub-systems?

Answer:

Packages - Packages are the collection of components which are not reusable in nature.

Example - Application development companies work on Packages.

It is a group of classes or use cases that are used to organize model elements.

Packages can be nested within other packages. These are used as containers to organize elements. It is very useful to represent system architecture

Sub Systems - Sub Systems are the collection of components which are reusable in nature.

Example - Product development companies work on Sub Systems.

It is logical grouping of related components. It is collection of classes, packages, libraries and other sub systems that work together to deliver a specific set of functionalities.

Difference between Packages and Sub Systems are

A package is a collection of headers and source files that provide related functionality. A subsystem is a collection of one or more packages.

Question 15. What is camel-casing and explain where it will be used?

Answer: Camel-casing refers to the naming convention of variable, parameters or properties.

Here, multiple words are combined together.

In camel-casing, the starting letter of first word starts with small letter and other words first letter starts with capital letter.

Ex- first Name, last Name

In BA, camel-casing is used in requirements documentation.

In requirement documentation, BA often use camel-casing to name the entities like use case, features, user stories like validate Customer Details, calculate Interest Rate, etc. Business rules, which should be satisfied by the system use camel-casing.

While documenting business process or workflows, camel-casing can be used to individual in steps. This will help maintain consistency in the document.

The database tables name also uses camel-casing.

Requirement naming- camel casing is used in requirement document also, to name the functional and non-functional requirements.

By using camel casing in the documents, it helps to maintain consistency in the entire document and also increases readability.

Question 16. Illustrate Development server and what are the accesses does business analyst has?

Answer:

Development server - A Development Server is a computer or environment where **developers build, test, and deploy software applications** or services. It is a dedicated machine or instance used for development purposes, often running on a local or remote server.

A development server allows developers to code and test their applications before moving them to production.

A Business Analyst can have below types of access based on the needs -

1. **Read Only Access** - BA may be granted with the ReadOnly access to the development server. This will allow them to view the user interface of the application, navigate through the features and also they will be able to observe the behaviour of the application.
2. **Limited Access** - Depending upon the project needs, the BA's will be granted limited access to the specific modules in the application.

3. **Limited Configuration Access** - Means BA have the authority to make changes in certain areas of application where they have the access.

Question 17. What is Data Mapping?

Answer:

- The database contains multiple tables in it.
- There may come a scenario, where we need to map the data from one table to another.
- Data mapping is necessary in cases where we want **quick manner**.
- Data mapping is nothing but a **process to establish connection between multiple data sources**.
- The purpose of data mapping is to ensure that the data is accurately transferred or converted into different format.

The main purpose of data mapping is-

Data integration- While combining the data from different sources, it ensures that the data is properly matched.

Data Migration- While migrating the data from legacy system (source) to the new system (destination), the data elements are mapped accurately into the new system. Required techniques are applied to convert the data into the format that is required by the new system.

Data Transformation-Data transformation means converting the data from one format to other. In data mapping, data transformation plays very important role which ensures that the data of legacy system (source) is mapped correctly to the data in new system (destination).

Question 18. 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: Application Programming Interface or **API** - Application Programming

Interface (API) is a set of rules, protocols, and tools that allow different software applications to communicate and interact with each other. APIs define how different software components should interact, enabling one system or service to access features or data from another without needing to understand the internal workings of the other system.

Key components of API are described below -

1. **API Endpoint:** A specific URL or URI (Uniform Resource Identifier) that represents a specific function or resource available via the API.
 - a. Example: <https://api.example.com/users> (Endpoint for fetching user data).

2. **HTTP Methods (Verbs):** These methods define the type of operation to be performed on the resource:
 - a. GET: Retrieve data from the API.
 - b. POST: Send data to the API, usually to create a new resource.
 - c. PUT: Update an existing resource.
 - d. DELETE: Remove a resource.
3. **Request Headers:** Metadata sent along with the request, such as authentication tokens, content types, or session IDs.
4. **Request Body:** Data sent along with the request, usually in JSON or XML format, that contains the necessary information for the API to process.
5. **Response Body:** The data returned by the API after processing the request, typically in a structured format like JSON or XML.
6. **Authentication and Authorization:** Many APIs require security mechanisms to verify the identity of the user or system making the request (e.g., via API keys, OAuth, or JWT tokens).

APIs are used in Third-Party Integrations, Mobile and Web Applications, Cloud Services, IoT (Internet of Things), Automation and Workflow.

Benefits of APIs -

1. **Efficiency:** APIs allow businesses and developers to reuse existing software, components, or services, speeding up development time.
2. **Scalability:** APIs enable systems to scale by allowing new services or components to be added easily without disrupting existing systems.
3. **Integration:** APIs allow for the seamless integration of external services, platforms, or data sources, enabling systems to communicate and share information across different environments.
4. **Security:** APIs can offer controlled access to services or data, enabling security features like authentication, rate limiting, and encryption to ensure safe data transmission.
5. **Modularity:** APIs promote modular design by allowing applications to be broken down into smaller, independent services, which can be updated or replaced without affecting the whole system.

For the above scenario, we can follow below procedure –

1. **Establish API communication** - set up API communication between your application and other application to exchange data.
2. **Do Data formatting-** while sending the data from one application to another, convert the date format from dd-mm-yyyy to mm-dd-yyyy.
3. **While receiving the data from another application,** parse the data and extract the date, month and year and re-arrange them accordingly.
4. **Perform Data Validation** and ensure that the converted date remains in a valid format.