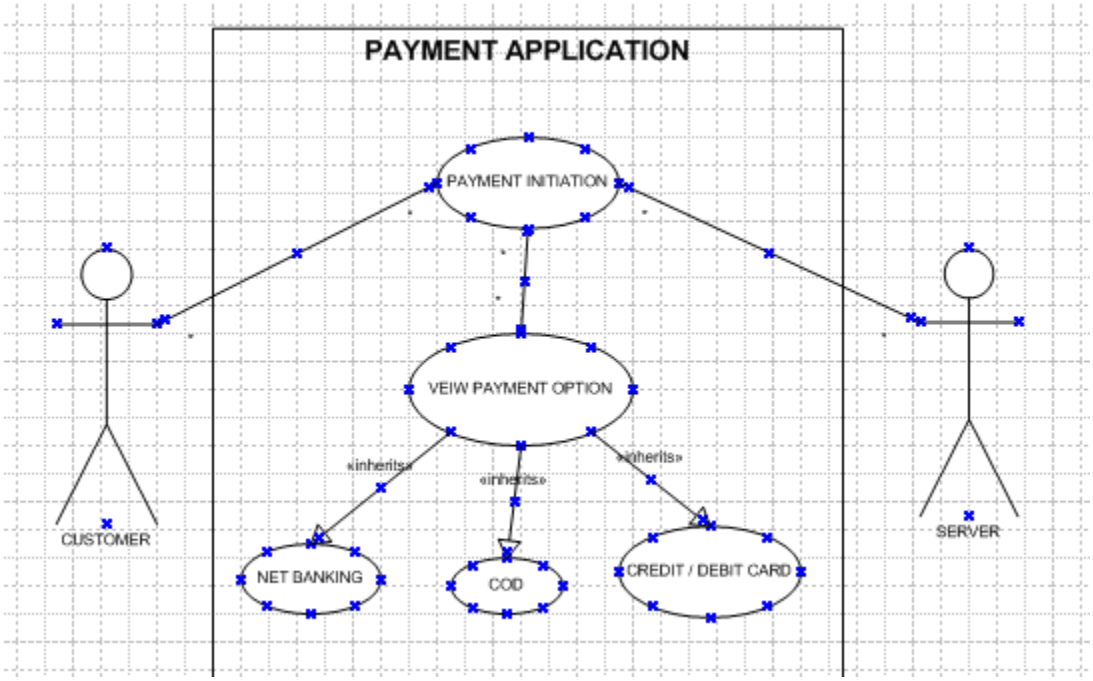
CAPSTONE PREP 3 PART 1 ANSWERS

1Q Draw a Use Case Diagram

Ans A Use Case Diagram is a type of diagram in the Unified Modelling Language (UML) that visually represents the interactions between users (actors) and a system.



2Q Derive Boundary Classes, Controller classes, Entity Classes.

Ans

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.	Customer Registration Customer login Bank server login Customer logout Bank server logout
Controller class (handles user input and process the data)	Registration Controller Login Controller Payment Controller

Use case will be considered as the controller classes	Credentials Controller Net Banking Controller Email Controller Logout Controller
Entity Class (All actors) Each Actor will be considered as one entity.	Customer Bank Server Cash Card Net banking

3Q Place these classes on a three tier Architecture

Ans In a three-tier architecture, the payment options such as card, wallet, cash, and net banking can be categorized and placed logically across the three layers: Presentation Layer, Business Logic Layer, and Data Access Layer

Application Layer (Place primary actors associated boundary Place controller class)	Customer registration Customer login Bank server Login
Business logic layer	Customer Bank server
Data Layer (All the entity classes)	Customer Bank server Card Cash Net baking wallet

In three tier architecture, the application layer handles the user interface,
The business logic layer manages the business logic and coordinates between the other tiers,
The data tier handles data storage and retrieval.

4Q Explain Domain Model for Customer making payment through Net Banking

Ans Domain Modelling is also known as Conceptual Modelling. A Conceptual Model depicts the concepts (idea, thing or object) that are easily identifiable in the problem description.

Domain modelling is also known as conceptual modelling. A conceptual model depicts the concepts (idea, thing or object) that are easily identifiable in the problem description.

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, bcoz he should make the payment.

Now the payment is done by netbanking, so payment table is connected to netbanking 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, bocz authencation is to performed there.

Also the authentication table is connected to transaction table, becoz 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 tables.

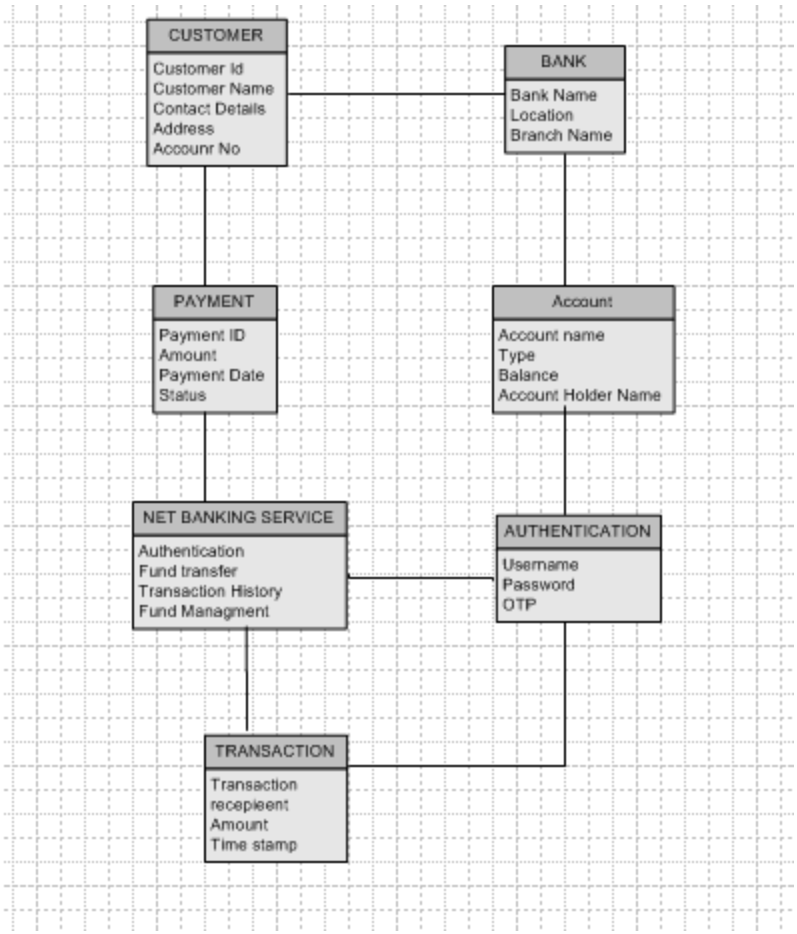
Domain Model- it is a conceptual model that represents real world entities.

ER Model-focuses on relationships required for storing and retrieving the data

Domain Model- It focuses on capturing the behaviour of application

ER Model-primarily used in database design

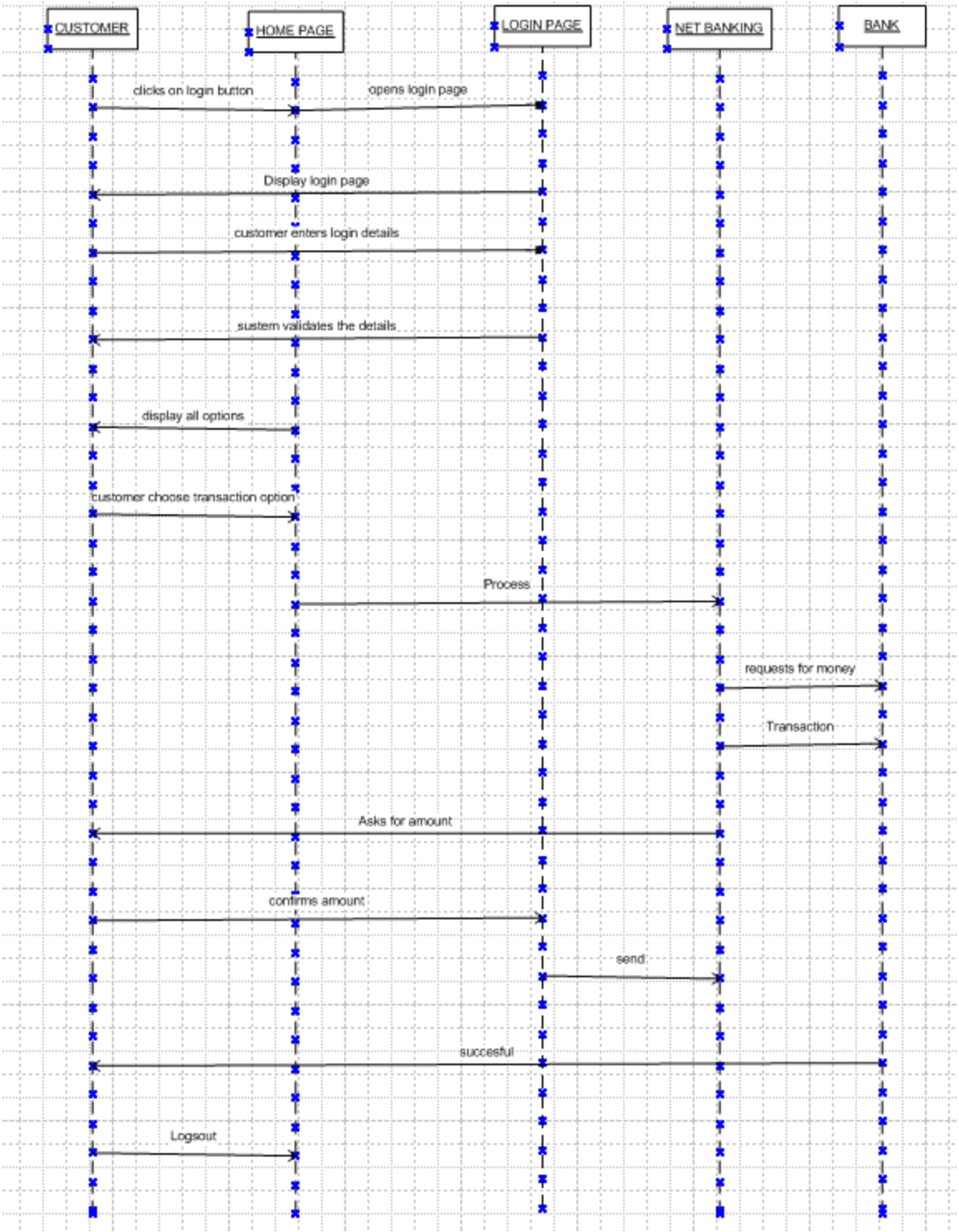
Domain Model- used throughout the software development lifecycle



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

Ans The sequence diagram is used primarily to show the interactions between classes in the sequential order in which those interactions occur.

This diagram shows how the objects in the system interact and communicate with each other with time to achieve specific task.



Q6 Explain Conceptual Model for this Case

Ans A conceptual model is a high-level representation of a system that helps in understanding, visualizing, and communicating the essential aspects of a domain.

Its Provides a clear and simplified view of the domain, making it easier to understand.

Key elements of the conceptual model are :

Entities:

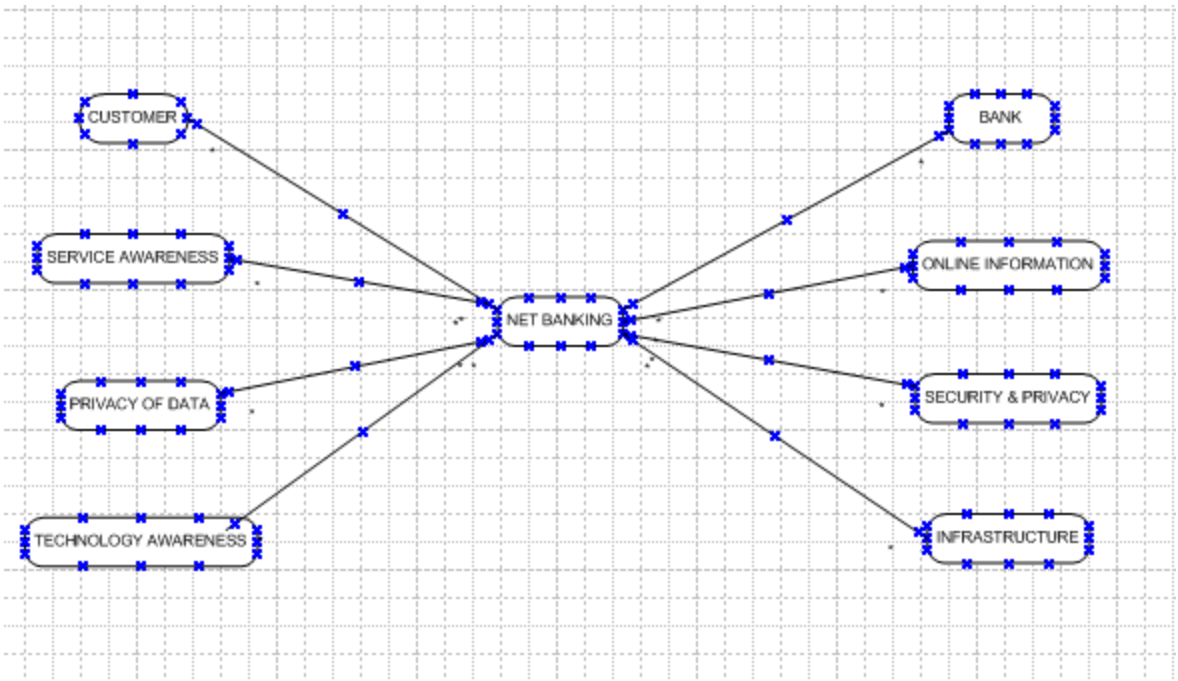
The key entities in this system are Customer, Payment, and Payment Method. The Customer entity represents individuals making payments and is characterized by attributes such as Customer ID, Name, and Contact Information. The Payment entity captures details of the transaction and includes attributes like Payment ID, Amount, Date, and Status. The Payment Method entity identifies the medium through which the payment is made, such as Method Type (Net Banking, Card, Wallet, or Cash) and Provider Name for methods like Net Banking.

Attributes:

Each entity contains attributes that define its properties. For Customer, key attributes are CustomerID (unique identifier), Name, and Email. For Payment, important attributes are Payment ID (unique transaction ID), Amount (value of the transaction), Timestamp, and Transaction Status. The Payment Method entity's attributes include Method Type, which categorizes the type (e.g., Net Banking), and additional attributes like Bank Name or Wallet Provider to specify details relevant to the chosen method.

Relationships:

Relationships connect the entities and establish their interaction. A Customer can initiate one or more Payments, creating a one-to-many relationship between Customer and Payment. Each Payment is linked to a specific Payment Method, forming a many-to-one relationship between Payment and Payment Method. In the case of net banking, the relationship includes a reference to the customer's Bank Name and may involve additional validations, such as Account Number or Authentication Code.



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

Ans MVC Architecture MVC (Model-View-Controller) Architecture is a software design pattern that separates an application into three interconnected components: the Model, which manages the application's data and business logic; the View, which handles the user interface and presentation of data; and the Controller, which processes user input, updates the model, and determines how information is displayed in the view. This separation of concerns improves modularity, scalability, and maintainability by allowing each component to be developed, tested, and modified independently while working cohesively to handle user interactions and system workflows.

3 Parts of MVC architecture are described below -

1. Model-

a. Represents the application's core logic and data.

b. Responsible for retrieving, storing, and processing data (e.g., through databases or APIs).

c. Independent of the user interface.

2. View-

a. Represents the presentation layer or the user interface,

b. Displays data from the model to the user and sends user input to the controller.

c. Examples: Web pages, mobile app screens, or desktop GUIs.

3. Controller -

a. Acts as the intermediary between the model and the view.

b. Handles user input, processes it, and determines the appropriate response.

c. Updates the model or view as needed.

Advantages of MVC-

1. MVC has the feature of scalability, which in turn helps the growth of application.

2. The components are easy to maintain.

3. A model can be used by multiple views that provide reusability of code.

4. By using MVC, the application becomes more manageable.

5. 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 a use case results in one boundary class.

2. Combination of Two actors and a use case results in two boundary class.

3. Combination of Three actors and a use case results in three boundary class.

4. Use case will result in a controller class.

5. Each actor will result in one entity class.

For example, we take scenario of customer making payment either by card or wallet or by cash or net banking -

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

2. View Classes - Login View, PaymentOptionView, NetBankingView, BankSelection View, Credentials View, PaymentAmountView, PaymentConfirmationView, LogoutView

3. Controller Classes - LoginController, PaymentOptionController, NetBankingController, BankSelectionController, CredentialsController, PaymentAmountController, PaymentConfirmationController, LogoutController

Guidelines to place identified MVC classes in a 3 Tier Architecture -

1. Place all entity classes in DB layer

2. Place Primary actor associated Boundary class in Application Layer

3. Place Controller class in application layer.

4. If Governing body influence or reusability is there with any of the remaining Boundary class place them in Business Logic layer or else place them in Application layer.

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

Ans The Waterfall Model is a linear and sequential software development methodology where each phase must be completed before moving to the next. The model typically includes stages like Requirements Analysis, System Design, Implementation, Testing, Deployment, and Maintenance. It emphasizes thorough documentation and planning at every step. This model is ideal for projects with well-defined requirements that are unlikely to change, as it does not allow for much flexibility or iteration. While straightforward and easy to manage, its rigidity can be a drawback in dynamic or complex projects where requirements evolve over time.

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 participates 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 founds 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 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-

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-

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.

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

Ans 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 find creative ways to handle the disagreement.

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

Y axis- assertiveness , x axis-co-operativeness

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

Moderate assertiveness, moderate cooperativeness-Comprising-When both parties need to give and take.

In situations where time or resources are limited.

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.

5 steps to conflict Management-

Identify the conflict

Discuss the details

Agree with the root problem

Check for every possible solution for the conflict Negotiate

The solution to avoid the future conflict

Q10 List down the reasons for project failure

Ans Reasons for the project failure-

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 fall due to lack of technology knowledge or lack of finances

11Q List the Challenges faced in projects for BA

Ans

- 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

12Q Write about Document Naming Standards

Ans All Documents will be named using some standards like
[projectID][DocumentType]V[x]D[y].ext

For example: We have a project with ID "PROJ456BANK" and we are working with Requirement Specification Document then -

Project ID-PROJ456BANK

Document Type - REQ

Version - 1.0

Date-2024-12-18

Then the naming convention of the document will be **"PROJ456BANK-REQ-1.0- 2024-12-18"**

13Q What are the Do's and Don'ts of a Business analyst

Ans

Do's	Don'ts
Consult a SME for clarifications in requirements.	Never say 'NO' to clients.
Go to the client with 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".
Try to extract maximum leads to the solution from the client himself.	Never imagine anything in terms of GUI
Concentrate on important requirements.	Don't interrupt the client when he is giving you the problem.
Question the existence of existence or question everything.	Never try to give solutions to the client straight away with your previous experience and assumptions.

14Q Write the difference between packages and sub-systems

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

Example - Application development companies work on Packages.

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

Example - Product development companies work on Sub Systems.

Difference between Packages and Sub Systems are described below -

Aspects	Package	Sub-system
Definition	A collection of related functionalities or components bundled together for a specific purpose.	A self-contained unit within a larger system, consisting of multiple components or packages.
Scope	Focused on a specific, narrow functionality.	Covers broader and more integrated business functions.
Complexity	Generally simpler and less complex.	Larger and more complex due to integration of multiple components.
purpose	Provides specific features or services.	Represents a significant part of a larger system's functionality.
Size	Smaller in size and scope.	Larger, often consisting of several packages or modules.
Components	May consist of a single module or component.	Comprised of multiple components, potentially including packages.
Integration	Typically needs to be integrated with other packages or systems.	Operates as part of a larger system, often with complex interdependencies.
Example	Payment processing module. Reporting package.	Inventory Management System, CRM System.

Q15 What is camel-casing and explain where it will be used
Ans 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 letters.

Ex-firstName, lastName

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 validateCustomer Details, calculateInterestRate, 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.

16Q Illustrate Development server and what are the accesses does business analyst has?
Ans A development server refers to a dedicated environment that is used during the software development process.

It provides platform for the developers and the testers to build, test, develop and debug the application.

The accesses a BA has are

ReadOnly- BA's 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.

Limited Access-

Depending upon the project needs, the BA's will be granted limited access to the specific modules in. the application.

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

Q17 What is Data Mapping
Ans 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 covert 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).

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
Ans
API stands for Application Programming Interface.

It is a software intermediary that allows the two applications to communicate with each other.

It is the set of rules, protocols and tools that define how different software application should interact with each other.

API allows sharing of only necessary information and keeps the internal system details hidden, which helps the system security.

For the above scenario,

Establish API communication-set up API communication between your application and other application to exchange data.

Do Data formatting-while sending the data from one application to other, convert the date format from dd-mm-yyyy to mm-dd-yyyy.

While receiving the data from other application, parse the data and extract the date, month and year and re-arrange them accordingly.

Perform Data Validation and ensure that the converted date remains in a valid format.