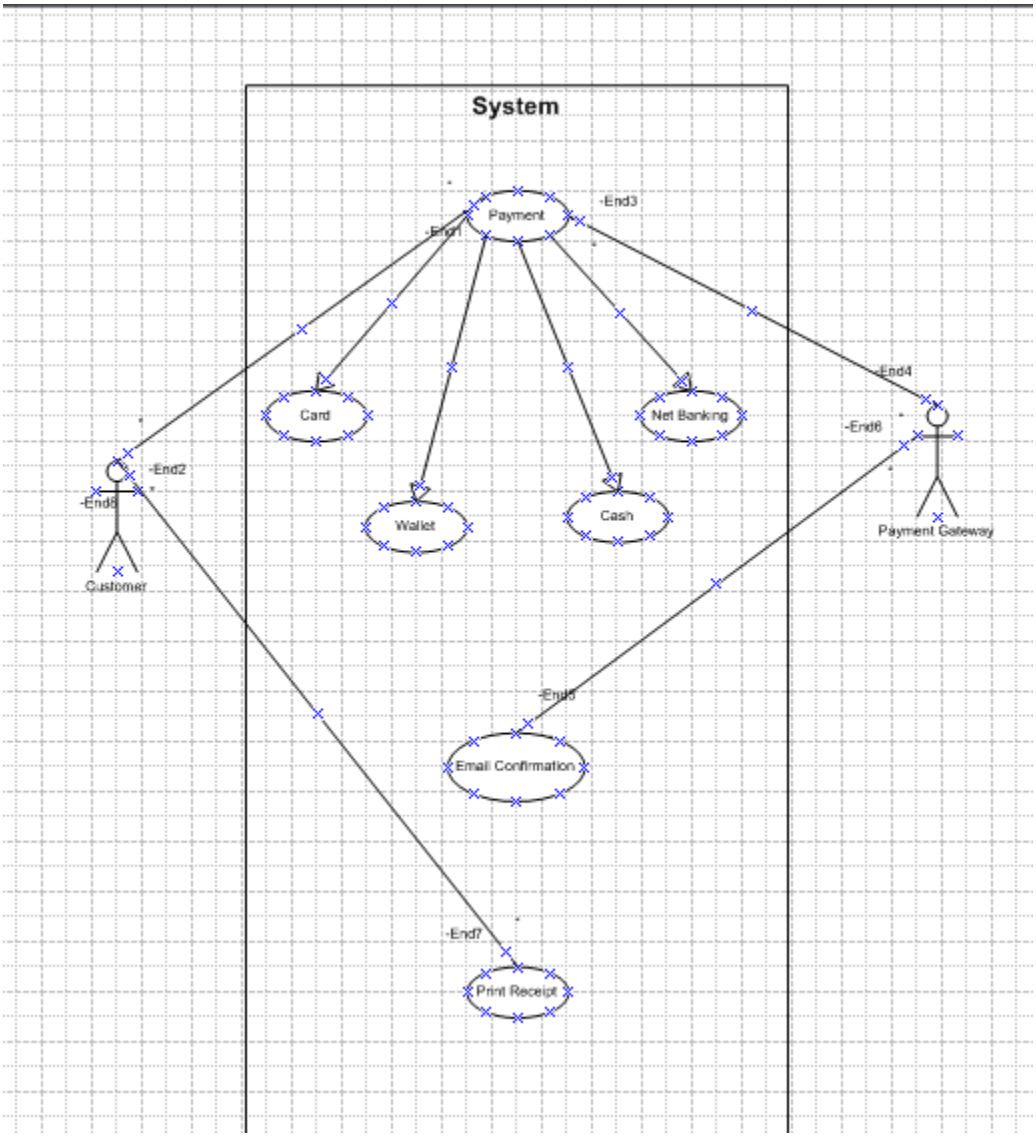
Q1. Draw a Use Case Diagram



Q2: Derive Boundary Classes, Controller classes, Entity Classes

Boundary class:

The Boundary class is a class that is the boundary of the system and other system or user (which is actor in the use case diagram).

The following are the features of the Boundary class.

1. This class is easier to change to than the Entity and Controller class.
2. The attribute of this class and screen layout are defined at the basic design.

3. In a class diagram, there are cases where the stereotype (<<boundary>>) is added.
4. In a class diagram, there are cases that are shown by the following icon

Controller classes:

The followings are the feature of the Controller class.

This class has few attributes.

In a class diagram, there are cases that the stereotype (<<control>>) is added.
This class is a class to achieve use cases in the Use case diagram.
In a class diagram, there are cases that are shown by the following icon

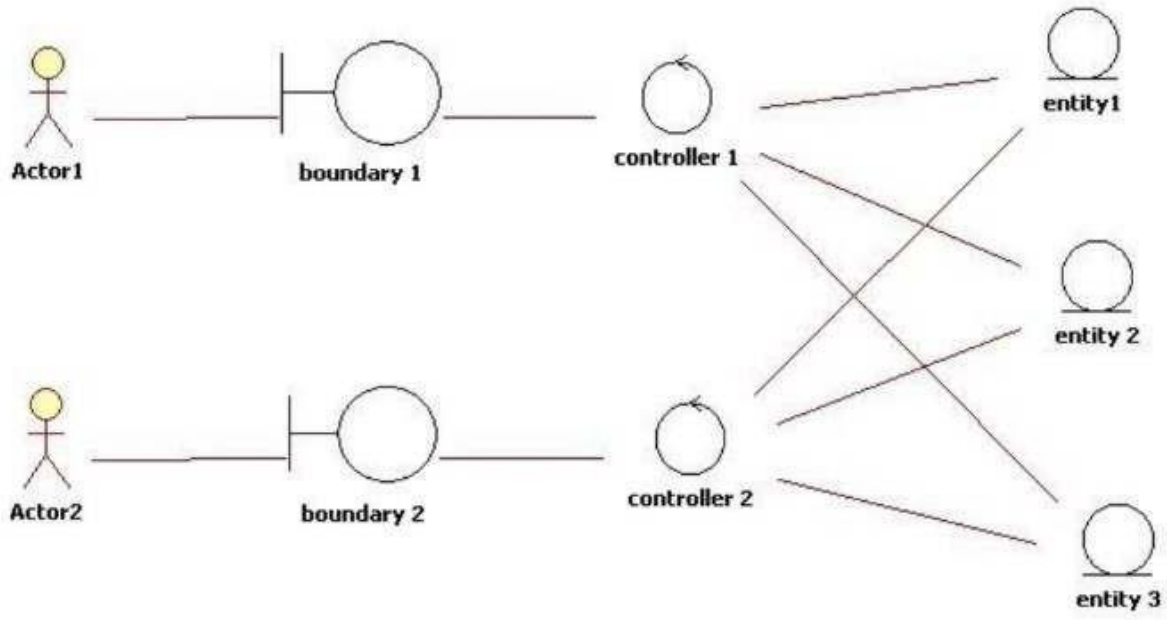
Entity Classes:

The Entity class is a class that has data. The "E" of the ER diagram means "Entity".

The following are the features of the Entity class:

There are many cases that these objects of this class are perpetuated in DB.
The extraction of the class is like ER diagram2.
This class is related to the DOA (Data-oriented approach).
The module cohesion of this class is high and is not easy to be changed.
In a class diagram, there are cases that the stereotype (<<entity>>) is added.
In a class diagram, there are cases that are shown by the following icon

Q3. Place these classes on a three tier Architecture. -



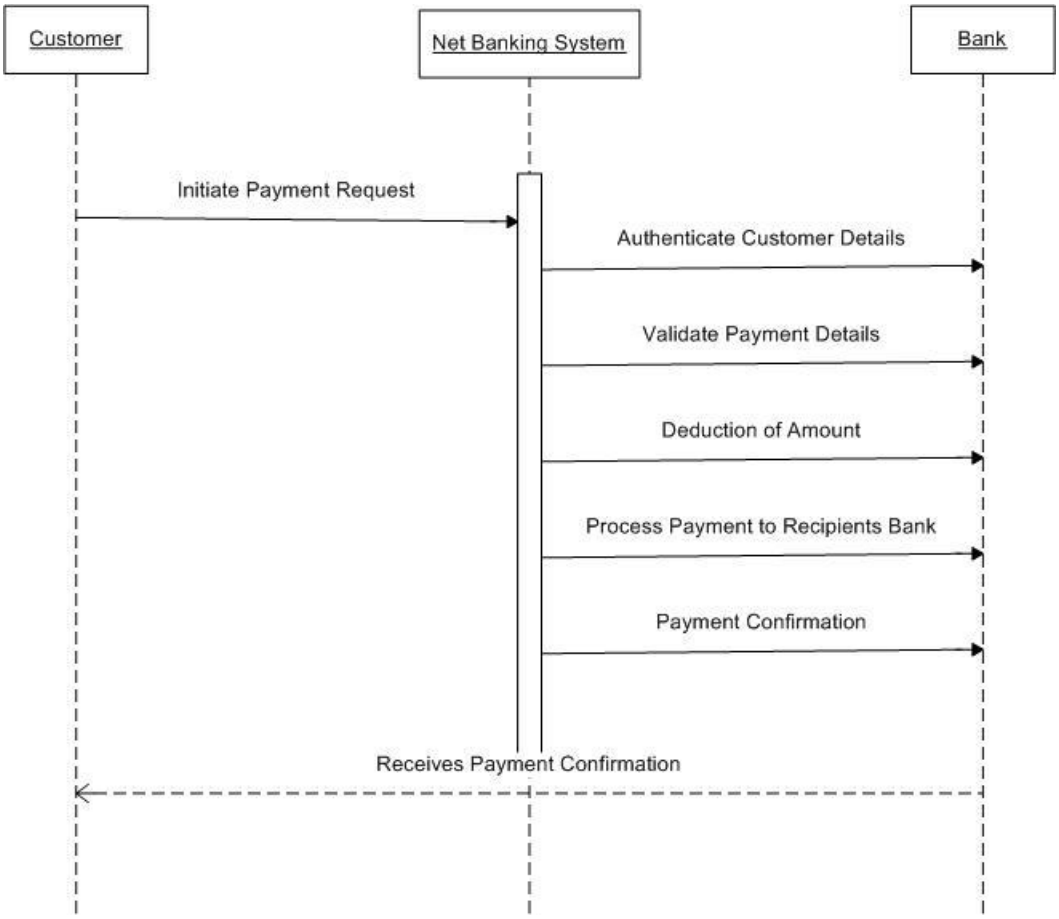
Q4: Explain Domain Model for Customer making payment through Net Banking

A domain model is a conceptual model of the domain that incorporates both behavior and data. It is a system of abstractions that describes selected aspects of a sphere of knowledge, influence, or activity. The model then can be used to resolve problems related to that domain.

The domain model is represented of meaningful real-world concepts pertinent to the domain that need to be modeled in software. The concepts include data involved in business and rules the business uses in relation to that data.

A domain model leverages natural language of the domain. It generally uses the vocabulary of the domain, thus allowing a representation of the model to be communicated to non-technical stakeholders. It should not refer to any technical implementations such as database or software components that are being designed

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



Q6: 6. Explain Conceptual Model for this Case

Conceptual modeling is a representation of the business model we have. There are entities, and their relationships among them based on which this is created. This is a Representation of a system which will be created, and this is not a language used to communicate with technical team.

• Entities

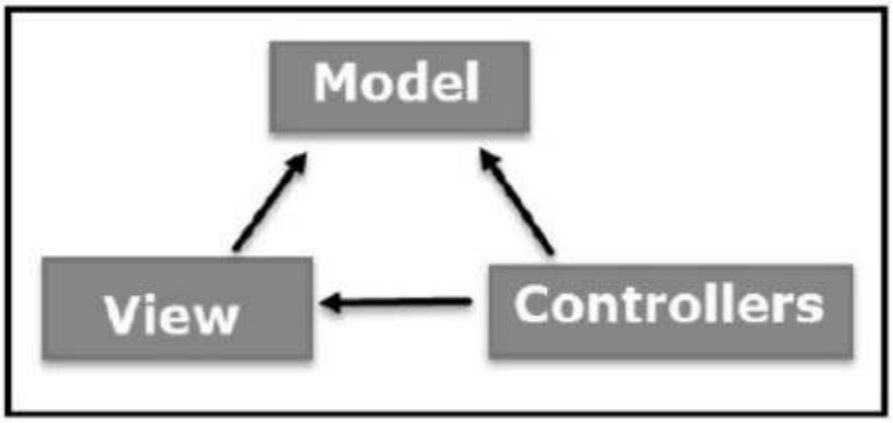
In our case study, since customer is initiating online payment through net banking following entities will be there. Bank, Customer, Net banking system of a Bank, Final beneficiary (Customer's customer).

• Relationships

Customer will be initiating payment with multiple beneficiaries, so it will be One to Many Relationships. Bank System will interact with Customer, so it will be one to one relationship. Bank will be facilitating payment for multiple customers; hence it will be One to May Relationship. Based on above Entities and Relationships we will be creating a Entity Relationship diagram, which will be used as a representation to take sign off from the client

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

The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development framework to create scalable and extensible project



Model:

It represents the business layer of application. It is an object to carry the data that can also

contain the logic to update controller if data is changed. All Model Classes are represented as Entity Classes.

View:

It represents the presentation layer of application. It is used to visualize the data that the, model contains. All View classes are represented as Boundary classes/Form Class. Actor Speaks to the system via boundary class and vice versa.

Controller:

It works as a data transmitter between the model and view. It is used to manage the flow of application, i.e., data flow in the model object and to update the view whenever data is changed. Controller class is working based on the user's command. Understands the command/request given by the actor/user through boundary/form class

MVC Architecture Rules:

- Combination of One actor and a use case results in one boundary class.
- Combination of Two actors and a use case results in two boundary class.
- Combination of Three actors and a use case results in three boundary class.

Note: Only one primary actor is to be considered with a use case.

- Use case will result in a controller class.
- Each actor will result in one entity class.

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

- Place all entity classes in DB layer
- Place Primary actor associated Boundary class in Application Layer
- Place Controller class in application layer.
- 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)

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.

Business Analyst will verify if the product is delivered as per the requirements, and it is meeting the business need. Maintenance: Once the implementation is done the team has to give support by installing patches, handling change requests, etc

Stages in Waterfall Model are as follows:

- Requirement Gathering and Analysis
- Designing
- Coding
- Testing
- Deployment
- Maintenance

Requirement Gathering and Analysis:

This is the initial stage of the project where an involvement of the BA is. BA is responsible for preparing BRD document (Business Requirement Document)

Artifacts: Functional Specification Document
Business Requirement Document

Designing:

In this phase the architect will start designing the system based on the
Business analyst inputs and requirement documents. The BA helps him to clear the doubts about the requirements.

Artifacts: Design Documents

UML diagrams get ready in this phase

Coding:

This phase is quite lengthy as the core development starts in this phase. Developer starts product development based on the requirement document prepared by the BA. Developer may ask questions to BA regarding the requirement, and he needs to answer the questions as and when required.

Artifacts: Code

Testing:

After coding, the testing phase will start, in this phase BA helps the testing team to understand the requirements so that they will build proper functional test cases. BA has to review whether the test cases covering the whole functionality.

Artifacts: Test Cases and test results

Deployment:

Once the code is developed and tested, it is ready to deploy in the production environment. The BA will verify the product is delivered as per the requirements and it is meeting the business needs.

Artifacts: Implementation Review document

Maintenance:

Once the implementation is done the team has to give support by installing patches, Handling change requests etc.

A BA is the person who knows every nook and corner of the project. So, every change request has to be reviewed by him and based on his inputs and reports the team will respond.

Artifacts: User Satisfaction review and change request review.

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

Conflict management is the use of techniques to resolve disagreements or control the level of discord. Conflict occurs whenever people disagree. The disagreement could be over their perceptions, ideas, values, motivations, or desires. Researchers Kenneth Thomas and Ralph Kilmann developed a model for resolving conflicts. This model is known as the Thomas-Kilmann model.

This model is based on two dimensions of conflict management: Assertiveness and Empathy. Based on these two dimensions, there are five conflict resolution strategies: Competing, Avoiding, Accommodating, Collaborating and Compromising

Competing:

Competing, the first Thomas-Kilmann conflict mode is assertive and non- cooperative. It refers to addressing only one's own concerns at the cost of the concerns of the other. It is a power-oriented mode—one uses whatever power dynamic seems appropriate to get a favorable outcome for one self. An individual's ability to debate, their position in the hierarchy, or their financial power matters the most.

Competing is defensive—it strictly means standing up for your individual beliefs and simply trying to win

Accommodating:

According to the Thomas-Kilmann model, the Accommodating mode is both accepting and cooperative. It is the opposite of competing. While accommodating, the individual in question neglects their own problems or beliefs to address the problems of the other party. The element of self-sacrifice is highlighted in this mode. Accommodating typically involves selfless understanding, generosity and or charity.

At times, accommodating would require you to follow the other person's orders when you would not like to do so, or submit to the other's perspective or decisions

Avoiding:

In the Thomas-Kilmann model, avoiding is both unassertive and uncooperative. The individual wants to address neither their own problems nor the problems of others. This ultimately means that they do not want to engage in the conflict at all. Avoiding might be seen at times as a diplomatic move involving bypassing or ignoring the issue. It could also involve putting off the issue until the time is favorable, or simply stepping back from an uncomfortable or hazardous situation

Collaborating:

Collaborating is the most beneficial outcome in the Thomas-Kilmann conflict model is both assertive and cooperative. This mode is the opposite of avoiding. Collaborating includes a voluntary effort to work alongside the opposition to find a perfect solution that wholly addresses the collective problem. Collaborating involves deep-diving into an issue to locate the critical demands of the concerned individuals or parties. Collaborating between two or more people might take the form of a quest to understand the 'why' of the disagreement. It involves striving to look for creative answers to interpersonal issues and enriching yourself from the other person's insights

Compromising:

The last outcome in the Thomas-Kilmann conflict model falls on the average point on both the assertiveness and cooperativeness scales. The goal here is to find a mutually acceptable and robust solution that, in some ways, satisfies both individuals. It comes midway between competing and accommodating. It addresses an issue more directly than avoiding but falls short of investigating it with as much depth and rigor as collaborating. In certain situations, compromising might involve seeking middle- ground solutions, providing concessions, or looking for a quick solution that provides some way forward from the impasse.

Q10. List down the reasons for project failure

1. Poor planning
    - Although sometimes overlooked in importance, lack of planning can make a project fail.
    - Having a successful project depends on properly defining in detail the scope, the time frame, and each member's role. This way, you'll have a route laid out to follow.
2. Inconsistently defined resources
    - Let's be clear: planning shouldn't be limited to agendas, meetings, and responsibilities. It should also include human, intellectual, financial, or structural resources. If these are not consistently determined, deadlines can't be met, which can jeopardize the project's conclusion.
3. Unclear objectives
    - Project objectives should be clearly defined, so as time goes by, you'll know if you're doing what's right or not. Remember that choosing measurable goals helps you better visualize your progress and helps you see how close you are to achieving your results.
4. Lack of detail control
    - Monitoring is essential for successful projects, even knowing the details of many projects simultaneously can be very challenging.
    - As a result, it's important to know how your project is going, if it is on schedule and if the budget is under control. This way, if there are any divergences from the initial plan, you can still correct them.
5. Lack of transparency
    - It's essential that everyone involved in the projects have complete project visibility so that it doesn't fail – not only the project manager, but other team members too.
    - This includes clear communication, good document management, and transparency about tasks' status, all of which can be achieved with centralized, all-digital files.

6. Lack of communication
    - Communication is the key to good project management. Without the right tools and processes to allow interaction among team members and the project manager from the beginning, efficient communication can seldom be achieved.
7. Change of direction
    - Among the ways projects fail, a very common one is scope creep. This concept refers to changes requested when the project has already started which had not been planned before. This is very common when projects are not appropriately documented and defined beforehand.
8. Unrealistic expectations
    - When you want to do something fast, with a limited budget, and a reduced team, it can really make your project fail. You should be realistic when it comes to your teams' capabilities, deadlines, and the resources available – only then can you obtain the results you want.
9. Lack of monitoring
    - Providing a schedule to the team is not enough for a project to be successful. You should also make sure everything goes as planned. This means having frequent progress checks or meetings, as well as making adaptations, when necessary, is essential.
10. Unrealistic due dates
    - Planning complex tasks for short due dates is definitely one of the causes for project failure. It is vitally important to carefully consider how long each project phase will take, in addition to extra time for unexpected events. This is the only way to develop a quality project.
11. Poorly assigned roles
    - When each team member receives their responsibilities clearly, they will know what, when, and how to perform their activities without someone needing to constantly ask for it.

Q11. List the Challenges faced in projects for BA

A BA is responsible for multiple tasks at the same time. From managing the projects, maintaining client relationships, interacting with stakeholders, and managing project deadlines, Business Analysts got a lot on their plate. Below are the challenges faced by business analyst.

- Lack of training
- Obtaining Sign-off on requirement
- Coordination between developers and testers
- Conducting meetings
- Preparing effective status reports which satisfies all the project stakeholders.
- Driving client for UAT completion
- People Management (Coordination between different people and different teams)

Business analysts (BAs) face many challenges in projects, including:

- **Poor communication**

Effective communication is essential for a project's success.

- **Conflicts with stakeholders**

Stakeholders may have conflicting expectations of what the project will deliver.

- **Stakeholder design**

Stakeholders may want to dictate how the system should work instead of providing details about what it should do.

- **Changing needs**

Nonalignment of goals among stakeholders can lead to changing needs and priorities.

- **Inadequate resources**

A lack of resources can manifest as inadequate staffing, limited budget, insufficient equipment, or a lack of required expertise.

- **Insufficient knowledge of user needs**

A BA may not understand user needs due to unavailability of key stakeholders or time or resource constraints.

- **No accountability**

Failure to continuously monitor and communicate project milestones in real time, and budget performance, dilutes project accountability and responsibility.

- **Resistance to change**

Staff members may oppose change due to fatigue.

Q12. Write about Document Naming Standards

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. 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: YYMMDD
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

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

As a Business Analyst we have to follow certain Rules and Regulations, which help us in improving our productivity as well as quality of work. Following are the Do's and Don'ts of Business Analyst.

- Never Say No to client: While client is saying something, we must never interrupt and say NO to client. As client is expecting us to listen to him and provide the solution to his problems.
  When client is explaining his problem or giving requirements, listen carefully and try to understand what he/ she is trying to explain, and never say "No" to client affront, because here client is explaining his problem and he expects some solution from us.

- Never Imagine anything in terms of GUI: We must not imagine the requirements by just seeing the graphical representation ask the right question to client and get clarity on the requirements. I.E. As login page may appear the same for all the websites but Functionality is different.
  For example: If you want to login to any website we need to enter correct user id and password to login the page. Here user id and password is common, but password length and validations differ from website to website based on the client requirement.

- Question Everything: We must not feel bad to ask questions and should get clarity from client. We can ask the questions till we get clarity from the client. Sometimes clients may not tell us the entire requirement until we probe them.
  Example: Client will say I need login page. But here you need to ask multiple questions to client to get clarity. Let us see some sample questions here.
  - What are the validations required,
  - Terms and conditions are required or not.
  - And when this button should be disabled or enabled.
  - Which type of error message should be shown on the screen if user enters wrong password or user id?
  - Password length should be how much and all.

- Consult an SME for clarifications in requirements: If the requirement is not clear and you need more clarity on the requirement, then we can discuss with SME. And ensure to

document the requirements what you discussed with SME and get approval from solution owner. And explain to him what you understand by discussing with the SME.

- Every problem of client is unique.
  Every problem of Client is unique, so talk to the client with a open mind with no assumptions from your previous experience.

- Do not interrupt the client, when he/she is giving you the problem.
  Listen very carefully and completely to the client as well as to the end user and then ask question, don't interrupt them in between.

- Maximum try to extract the leads to solution from the client itself.

- Never try to give solutions to client straight away with your previous experience and assumptions.

- Should not be hurry.
  Should not gather the requirements in hurry, conduct a meeting in a convenient time and take your own time to understand the requirement or gather the requirements. Because if you are in a hurry to capture the requirement then there is a chance to misunderstand the requirement, it may lead to project failure. As a Business Analyst you should be have open mind when you are gathering requirements

- BA should focus on "what" and "when" to develop rather than focus on "how" to develop.
  As a Business Analyst our responsibility is to understand what to deliver and when to deliver the project, how to develop is the responsibility of development team or development manager. We need not to concentrate on this part and need not to worry. Always have a prior discussion with your project manager and sponsor before conducting a meeting

- Should not miss any requirement
  Make sure that you have gathered all the requirements from the stakeholder for your project, missing out any information can results to unwanted redo the work as well as delay projects and increase cost.

- Should know what the Scope of the Project is.
  Sometimes non functional requirements of client are not feasible because of budget or time constraint, so it's always better to liaison with your PM to find out what is out of scope so that all will be in the same page and avoid misunderstanding.

There is no Word called as "By Default": In every requirement we have some Unique specifications, hence we cannot consider anything as By Default. Also, we should never Imagine something as by default by listening to client's requirements, we must either acknowledge it or probe it further.

Q14. Write the difference between packages and sub-systems

A package is a collection of headers and source files that provide related functionality.

A subsystem is a collection of one or more packages. For example, the Foundation subsystem contains packages such as Layout, MVC, Events, Properties, Image, and Print.

Package: In UML models and Object Oriented Analysis, A package is a organised group of elements. It can be termed as the UML mechanism for grouping things. It may contain many structural things like classes, components and other packages in it.

It can be used to:

- Group semantically related elements.
- Define a semantic boundary in the model.
- Provide units for parallel working and configuration management.
- It is used to provide encapsulated namespace within which all names must be unique.

Subsystem: In UML models, subsystems are a type of stereotyped components that represent independent, behavioral units of a system. They are widely used in class, component, and use case diagrams to represent large-scale components that are to be modelled.

An entire system can be modelled by a hierarchy of subsystems. the behavior of each subsystem can be defined by specifying the interfaces and operations that support interfaces in accordance with subsystems.

Key differences between Package and Subsystems:

Package	**Subsystem**
A package can be termed as a container which tends to organize, group elements present in the system into a more manageable unit.	A subsystem is a stereotyped components which represents individual behavioral units in a system hierarchy.
Package can be termed as a collection of components which are not reusable in nature.	Subsystems can be termed as a collection of components which are reusable in nature
Application development companies work on packages.	Product development companies work on subsystems.
Package is represented as a rectangle with tab in upper left corner. the rectangle contains name of the package had icon	Subsystems is displayed as a rectangle that contains the name of the subsystem and icon along with<<subsystem>> keyword
Packages are smaller and more focused in scope.	Subsystems are larger and encompass multiple packages or modules
They manage dependencies at class and component level	They manage dependencies at a higher level, defining boundaries and interfaces between different parts of the system.

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

Camel case is a writing style where each word in a phrase is capitalized except for the first word, and there are no spaces or punctuation. It's used in many places, including:

- Programming and web development: Camel case is a common naming convention for variables, functions, and other elements. It's popular because it's easy to read and understand.
- Company names: Some companies use camel case in their names or for their products and systems, such as FedEx, PlayStation, PayPal, MasterCard, and PowerPoint.
- Acronyms and abbreviations: Camel case is used in many acronyms and abbreviations, such as PhD.

The formal name for camel case is “medial capitals”. It was first used in 1813 by Swedish chemist Jacob Berzelius to make it easier to write and identify chemicals.

CamelCase is a way to separate the words in a phrase by making the first letter of each word capitalized and not using spaces. It is commonly used in web URLs, programming and computer naming conventions. It is named after camels because the capital letters resemble the humps on a camel's back.

The “camel case” is the practice of writing sentences so that each word or abbreviation in the middle of the sentence begins with a capital letter, without spaces or punctuation. For example: “iPad” and “eBay” The camel case is normally used for variable names in computer programming. Some programming styles prefer uppercase and lowercase letters with the first letter capitalized, others do not. The Camel case is distinct from the Title Case (or Capitalized Case), which capitalizes all words, but maintains the spaces between them.

The format indicates the first word starting with either case, then the following words having an initial uppercase letter. Common examples include YouTube, PowerPoint, HarperCollins, FedEx, iPhone, eBay, and LaGuardia. Camel case is often used as a naming convention in computer programming.

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

A Development server is a type of server that is designed to facilitate the development and testing of programs, websites, software, or applications for software programmers. It provides a runtime environment, as well as all hardware/software utilities that are essential to program debugging and development.

As a Business Analyst we have to access various Tools and Software's.

- ➢ Tools We will be Accessing:

  • Requirements management
  • Project management
  • Enterprise resource planning (ERP)
  • Modeling / Diagramming
  • Wireframing
  • Collaboration / Communication
  • Customer relationship management (CRM)
  • Data visualization

- ➢ Business Analyst Software:
  - Microsoft Office
  - Wrike (Project Management Tool)
  - Oracle NetSuite (Used for ERP)
  - Pencil (Modeling & Diagramming Tool)
  - Trello (Communication tool for stakeholders)
  - Microsoft Visio, Tableau , Power BI

Q17. What is Data Mapping

Data mapping is the process of matching fields from one database to another. It's the first step to facilitate data migration, data integration, and other data management tasks.

Data mapping bridges the differences between two systems, or data models, so that when data is moved from a source, it is accurate and usable at the destination.

Following are the steps of Data mapping.

• Define: Defining the data to be moved, including tables, fields with each table, and the format of the fields after it is moved. For integration Frequency is also defined.

• Map the Data: Matching Source fields to destination fields.

• Transformation: If Transformation is needed, formulas or rule is coded.

• Test: Using Test System, sample data is tested from Source data to run the transfer to see how it works and adjust as necessary.

• Deploy: Once transfer of data is done, we can schedule a migration or integration to go live Event.

- Maintain and Update: For ongoing integration, data mapping and updating is needed as source data is always changed. Hence maintenance and updating is needed.

Here are some things to know about data mapping:

- **Data mapping techniques**: There are several techniques for data mapping, including manual, spreadsheet-based, code-based, and automated:
    - **Manual data mapping**: Involves connecting data sources and documenting the process with code. This method is customized to specific needs, but it's labor-intensive and error-prone.
    - **Automated data mapping**: A tool that handles all aspects of the data mapping process. This is a good option if you don't have access to a coder.
- **Data transformation**: The process of converting data into a more readable form. It's often used in conjunction with data mapping.
- **Data profiling tools**: Used to discover and understand the data in its original format.

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.

API stands for Application Programming Interface, which is a set of rules and protocols that allow different software applications to communicate with each other. APIs can be used to integrate data, services, and capabilities from other applications.

Here are some steps you can take to integrate an API into an app:

1. Hire an API integration developer
2. Create a project within the API provider system
3. Receive the API key and authorization token
4. Integrate the API framework for the app
5. Use API request instances and methods

Application Programming Interface (API) is a software-to-software interface. APIs provide a secure and standardized way for applications to work with each other and deliver the information or functionality requested without user intervention.

An API, or application programming interface, is a set of defined rules that enable different applications to communicate with each other. It acts as an intermediary layer that processes data transfers between systems, letting companies open their application data and functionality to external third-party developers, business partners, and internal departments within their companies.