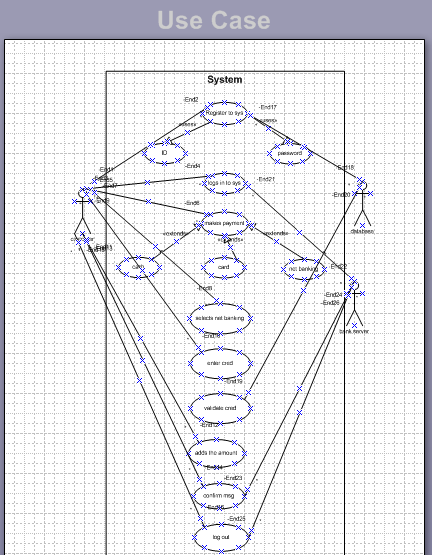
**CAPSTONE PROJECT PREP3 PART 1:**

Case study 1:

A customer can make a payment wither by card or by cash or by net banking.

**Q1. Draw a Use Case Diagram?**

**Answer:**

****

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

**Answer:**

**Boundary class: (All use cases)**

Combination of 1 actor and a use case results in one boundary class.

Ex: customer registration, customer login, bank server login.

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

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

Ex: customer logout, bank server logout.

And those actors should be primary actors. Primary actors mean the actors who initiate the use case and interact with the system.

**Controller class: (handles user input and process the data)**

Use case will results in controller class.

Ex: Registration controller, login controller, payment controller, credentials controller, net banking controller, email controller, log out controller.

**Entity class: (All actors)**

Each actor will be considered as one entity.

Ex: customer, bank server, cash, card, net banking.

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

Answer:

|  |  |
| --- | --- |
| 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 entity classes | Customer  Bank server  Cash  Card  Net banking |

**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 remaining Boundary Classes, place them in Business Logic Layer else place them in Application Layer.

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

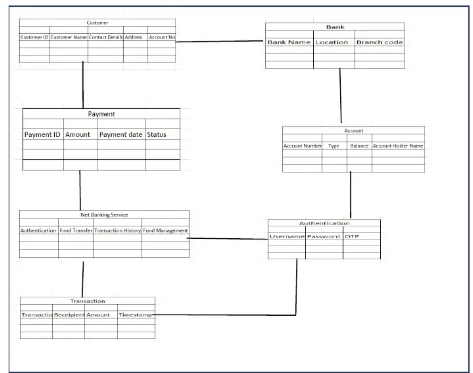
**Answer:** A Domain model is a conceptual representation that defines the structure, relationships, and behaviours of entities within a specific problem domain.

It is similar to the entity relationship model.

In the below diagram,

The customer table is connected to bank table, where the customer is able to make the 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 the authentication is to be performed there. Also the authentication table is connected to transaction table, where authentication will be done while transaction.



Difference between ER diagram and domain model:

ER MODEL: do not have attributes inside the box

Domain model: have attributes mentioned inside the box

ER model: data modelling technique used in data base to represent tables.

Domain model: it is a conceptual model that represents real world entries.

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.

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

Answer: A Sequence diagram is used primarily to show the interactions between classes in the sequential order in which those interactions occur. A sequence diagram can map a scenario by a use case in step by step detail to define how classes collaborate to achieve your application’s goal.

Developer draw the sequence diagram

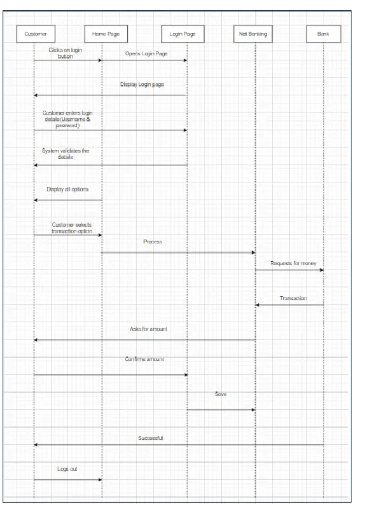
**Lifeline:** Lifeline indicates the life of an class (here the Class)

**Camel Casing:** entire first word will be in lowercase and subsequent words first letter should be in Upper Case. There will be no gap in between words. Example: getEmpId(); turn Left And Slow Down();

**Return message:** This will always flow towards the Controller; this is just a message and NOT a method.

**Note:** Method always flows in timeline. Never Method will flow backward in timeline.

**Focus of Control:** It shows the life of method.



**Q6. Explain Conceptual Model for this Case?**

**Answer:**

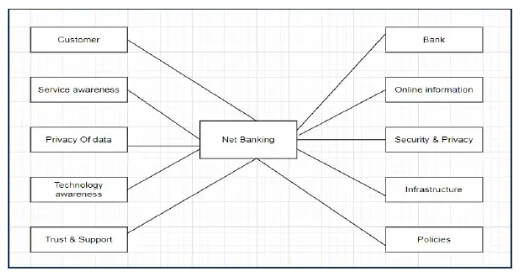
A conceptual model is a high level representation of a system that helps in understanding, visualizing, and communicating the essential aspects of a domain.

The conceptual model helps in understanding the key concepts, their relationships, and the overall structure of the net banking payment system.

It provides the clear and simplified view of the domain, making it easier to understand.

Key elements of conceptual model:

1. Entities- customer, product, order & payment.
2. Attributes- customer id, name, email, phone number.
3. Relationships- for example, a customer places an order.



**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 is a design pattern where, the application is divided in to 3 logical parts- Model, View and Controller.

To identify classes from use case diagram we apply MVC rules on each use case to derive classes.

Model:

The model represents the data and the business logic of the application. The model class knows all the data that needs to be displayed. It is the model who is aware about all the operations that can be applied to transform that class. It only represents the data of an application. The model represents enterprise data and the business rules that govern access to and updates of this data.

All model classes are represented as Entity classes.



View: The View is responsible for presenting the data to the user for handling the user interface. The view represents the presentation of the application. The View class refers to the model. It uses the query methods of the model to obtain the contents and renders it. The view is not dependent on the application logic. It remains the same if there is any modification in the business logic. View class is the data required by the query and is represented as Boundary class or form class.



Controller:

The controller acts as an intermediary between the Model and the View. Whenever the user sends a request for something then it always go through the controller. The controller is responsible for intercepting the requests from view and passes it to the model for the appropriate action. After the action has been taken on the data, the controller is responsible for directing the appropriate view to the user. In GUIs, the views and the controllers often work very closely together.



**MVC Architecture Rules:**

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 classes

3. Combination of Three Actors and a use case results in Three Boundary classes and so on....

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

4. Use case will result in a controller class

5. Each Actor will result in one entity class

Example:

Applying MVC rules we get the classes as

Employee login boundary class

Login controller class

Employee entity class

MVC Classes in 3 tier guidelines:



**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 remaining Boundary Classes, place them in Business Logic Layer else place them in Application Layer

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

Answer:

Waterfall model is a traditional model. Waterfall model follows a structured approach with each phase having specific deliverables.

At the end of each phase a review takes place to determine if the project is running fine.

Waterfall model works well for smaller projects where requirements are very well understood.



Deployment: moving code from development environment to production.

Implementation: running the code for very first time in production

After implementation, maintenance stage starts, and support team will take care.

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

**Answer:**

Conflict can occur due to various reasons, such as differences in goals, values, personalities, resources, or communication breakdowns.

Conflict is an inevitable part of any work place, so it is important to promote learning and growth.

Conflict management is 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.

Conflict Management -Thomas Kilmann Technique:

X Axis-Co-operation, Y Axis-Assertiveness

5 Options of Conflict Management

* Competing
* Avoiding
* Accommodating
* Collaborating
* Compromising

**5 Steps to Conflict Management:**

* Identify Conflict
* Discuss the details
* Agree with root problem
* Check for the every possible solution for the conflicts.
* Negotiate the solution to avoid the future conflicts.

**Q10. List down the reasons for project failure?**

Answer:

**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

**Q11. List the Challenges faced in projects for BA?**

**Answer:**

* Obtaining sign-off on requirement
* Change Management-with respect to cost and timelines
* Coordination between developers & testers
* Conducting meetings
* Driving client for UAT completion
* People Management (coordinating with different people and different teams)
* Lack of training
* Gathering clear and unambiguous requirement can be challenging
* Unable to understand what stakeholder is trying to convey
* Scope creep- change in requirement or scope of the project during project life cycle can lead to scope creep.
* BA may face difficulties in understanding the requirements if the domain in not familiar to him
* Poor communication between stakeholder and BA can affect the process of gathering the information.
* Technical complexity

**Q12. Write about Document Naming Standards?**

**Answer:**

Document Naming Standards:

All documents will be named using some standards

like [Project ID][Document Type]V[X]D[Y].extension

Example-[PQ777FRDV1D1.docx] or [PQ777FRD1.1docx]

IT Company standards:

Some of the standards that IT companies adopt are:

CMMI (Capability Maturity Model Integration)

IEEE (Institute of Electrical & Electronics Engineering)

ISO

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

**Answer:**

* Never say NO to client
* Never imagine anything in terms of GUI
* There is NO word called as “By Default”
* Consult an SME for clarifications in Requirements
* Try to concentrate only on important and required things.
* Question everything in the world.
* Go to the client with plain mind i.e. with no assumptions.
* Never try to give solutions to the client right away.
* Be like a lotus in mud- if a client comes with fancy requirement, and then talk to the project manager first.
* Requirement hurried-project buried.
* Never criticize the stakeholder and always appreciate the stakeholder even for small efforts.

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

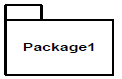
**Answer:**

**Packages:**  Package is defined as a collection of components which are not reusable in nature.

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

Packages can be nested within other packages.

Used as containers to organize elements and is very useful to represent system architecture.



**Sub systems:**

Collection of components which are reusable in nature is called sub systems.

Note: product development companies work on sub systems and application development companies work on packages.

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.

**Q15. 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 letters. 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.

**Camel Casing:** entire first word will be in lowercase and subsequent words first letter should be in Upper Case. There will be no gap in between words. Example: get EmpId(); turn Left And SlowDown();

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

Answer:

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-

Read Only- BA’s may be granted with the read only 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 has the authority to make changes in certain areas of application where they have the access.

**Q17. What is Data Mapping?**

**Answer:**

The database contains multiple tables in it.

There may be 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?**

**Answer:**

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

