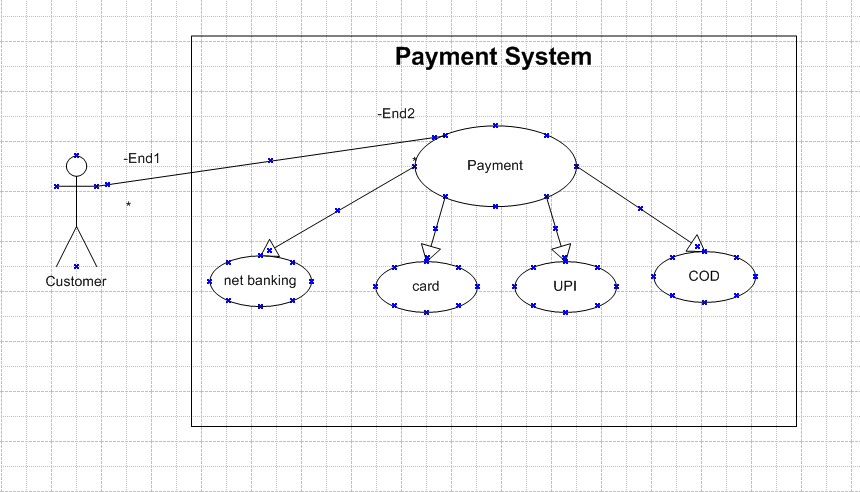
**Q1. Draw a Use Case Diagram**

**Answer:**

****

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

**Answer:**

|  |  |
| --- | --- |
| **Boundary Classes** | Payment Option |
| Netbanking Payment option |
| Card Payment Option |
| **Controller Classes** | Payment Controller |
| Netbanking Payment Controller |
| Card Payment Controller |
| **Entity Classes** | Payment |
| Customer |
| Card |

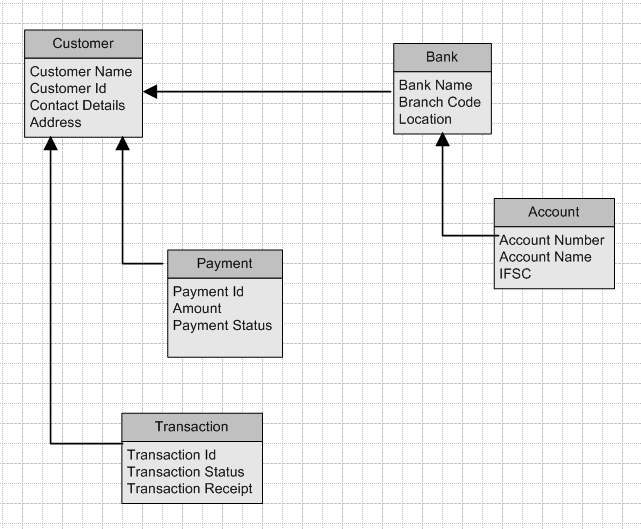
**Q3. Place these classes on a three tier Architecture.**

**Answer:**

|  |  |
| --- | --- |
| **Application Layer** | |
| **Boundary Classes** | Payment Option |
| Netbanking Payment option |
| Card Payment Option |
| **Business Logic Layer** | |
| **Controller Classes** | Payment Controller |
| Netbanking Payment Controller |
| Card Payment Controller |
| **Data Layer** | |
| **Entity Classes** | Payment |
| Customer |
| Card |

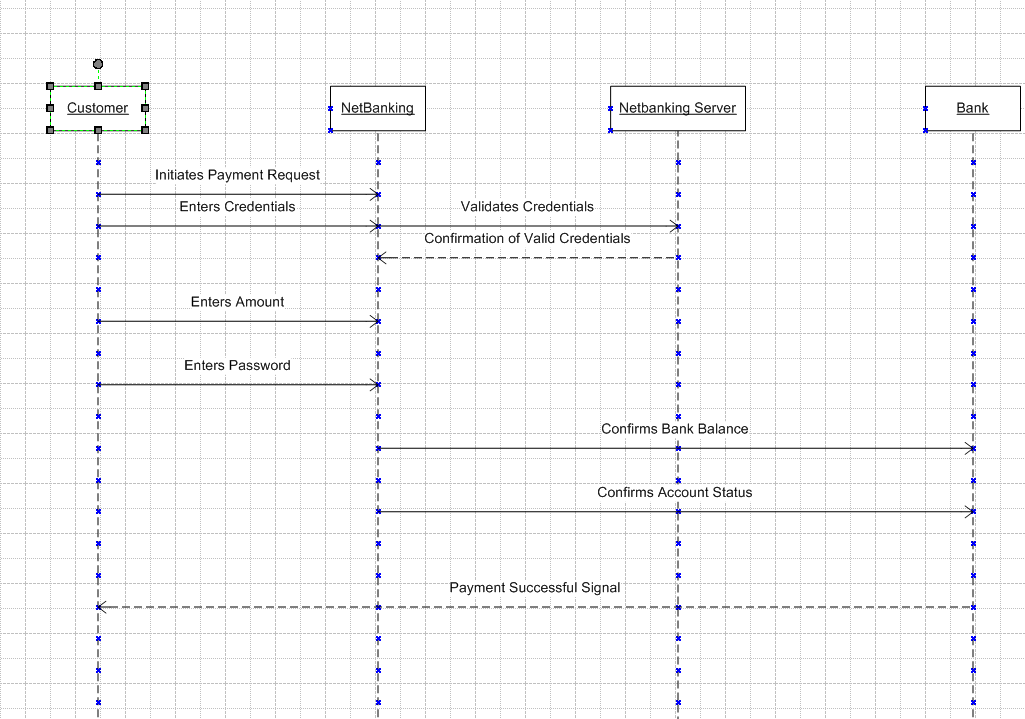
**Q4. Explain Domain Model for Customer making payment through Net Banking**

**Answer:**

****

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

**Answer:**

****

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

**Answer:**

-Conceptual model for payment through net banking outlines some key entities processes and interaction between them.

-It helps to visualise the overall structure for the flow of payment.

-The involved entities in this process are

* Customers
* Bank
* Merchants
* payment gateways

**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:**

To identify classes from use case diagrams we can apply MVC rules.

**1. Model:**

* The model class knows about all the data that need to be displayed.
* It is 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 business rules that govern access to and updates this data.
* All model classes are represented as entity classes.

**2. View:**

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

**3. Controller:**

* When user sends a request for something then it always go through controller.
* The controller is responsible for intercepting the requests from view and passes it to the model for appropriate action.
* After action has been taken on the data, the controller is responsible for directing the appropriate view to the user.
* In GUIs, the views and controllers often work very close together.
* Controller class is working based on user’s command.
* It understands command given by user through boundary class.

MVC Architecture Rules:

1. Combination of one actor and 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 result in three boundary classes.
4. Use case will result in a controller class.
5. Each actor will result in one entity class.

MVC Classes in 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 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:**

|  |  |
| --- | --- |
| **Pre Project** | -Enterprise analysis like SWOT analysis, Gap analysis Feasibility study, Root cause analysis  -Risk analysis -Stake Holders Analysis |
| **Requirements Gathering** | -Identify the requirements from stakeholder and document them -Preparation of BRD -Formation of prototypes -Sorting of gathered requirements -Privatization of requirements -Validation of requirements |
| **Requirements Analysis** | -Drawing UML diagrams -Prepare FRD from business requirements -Prepare SSD |
| **Design** | -From use case diagrams preparation of test case -Communicates with client on design and solution document -Updating the RTM |
| **Development** | -Organise JAD sessions -Updating the RTM |
| **Testing** | -Preparation of test cases from use cases -Perform high level testing -Preparations of client for UIT Updating the RTM -Taking sign off from client on project acceptance form |
| **Implementation and Deployment** | -Forwarding of RTM to client or PM for attachment to project closure document |

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

**Answer:**

-Thomas- Killmann technique is a conflict management framework. It identifies conflict management styles based on 2 dimensions one is Assertiveness (try to fulfil own needs) and second is Cooperation (try to fulfil other’s needs).  
-In this technique we take Assertiveness on the Y-axis and Co-operation on the X-axis.  
-Five options in this conflict management technique are 1. Competing 2. Avoiding 3. Accommodating 4. Collaborating 5. Compromising.

**1.Competing –**

-High assertiveness and less co-operation  
-It is a win-lose approach.

**2.Collaborating –**

-High assertiveness and high co-operation.  
-It is a win-win approach.

**3.Accommodating-**

-Less assertiveness and high co-operation.  
-It is a win-lose approach where one party gives in.

**4.Compromising-**

-Moderate assertiveness and moderate cooperation.  
-It is a give and take approach.

**5.Avoiding-**

-Less cooperativeness and less assertiveness.  
-It is a lose-lose approach.

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

**Answer:**

There are many Reasons for project failure, some of them are mentioned here-

-Lack of executive support  
-Lack of training to resources  
-Lack of executive support  
-Continuous change in requirements  
-Conflicts between developers and testers  
-Continuous change in management with respect to cost and timelines  
-Conflict between different people and different teams  
-Improper of requirement gathering  
-Lack of requirement analysis  
-Lack of user involvement

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

**Answer:**

Challenges faced by BA in the project are-

-Lack of training of resources  
-Continuous change in requirements  
-Change of management with respect to cost and timelines  
-Conducting meetings  
-Collaboration between developers and testers  
-Making sure status reporting is effective  
-Driving clients for UAT completion  
-People management that is coordinating between different people and different teams  
-Over all making sure project health is in good shape  
-Making sure the product is delivered as per the timelines without any issue

**Q12. Write about Document Naming Standards**

**Answer:**

All documents will be named using some standards like

[ProjectID][DocumentType]V[x]D[y].ext

Example-

PQ786BRDV1D3.dox

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

**Answer:**

**Do's as a BA**:

-Go to a client with a plane mind.  
Go to the client with no assumptions.  
-Listen carefully and completely until the client is done.  
-Ask your queries when the client is done speaking.  
-Maximum try to extract the leads to solution from the client.  
-Question the existence of existence.  
-Consult SME for clarification in requirements.

**Don'ts as a BA:**

-Never say no to the client.  
-There is no word called 'BY DEFAULT'.  
-Do not interrupt the client when he is giving you the problem.  
-Don't be watched away by I don't functionalities.  
-Don't imagine solutions on screen basis.  
-Never try to give solutions to clients straight away with your previous experience and assumptions.

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

**Answer:**

|  |  |  |
| --- | --- | --- |
| Sr. No. | Packages | Sub-Systems |
| 1. | Collection of components which are not re-usable in nature | Collection of components which are reusable in nature |
| 2. | Packages cannot contain multiple sub-systems | Sub-system may contain multiple packages |
| 3. | Packages are smaller and used to organise code in a system | Sub-systems are large and may consist of multiple packages |
| 4. | Application development Companies work on Packages | Product development Companies work on Sub-systems |
| 5. | Packages help to manage large codebases by grouping similar elements. | Sub-systems are used in architectural design to breakdown a system into manageable units. |

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

**Answer:**

-Camel case is way of writing.

-In camel casing initial word is small alphabet and from second word onwards first alphabet is capital and rest are small.

-Camel casing is used to write method names.

-Method names are represented as camel casing.

-One object sends message to perform an operation to other object and receiving object performs an operation. These messages are sent by methods and method names are represented as camel casing.