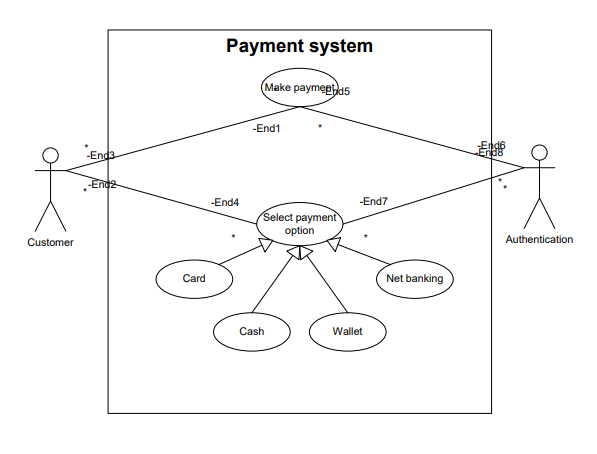
Project 3 part 1

Q 1) a customer can make a payment either by card or by wallet or by net banking, draw a use case diagram.



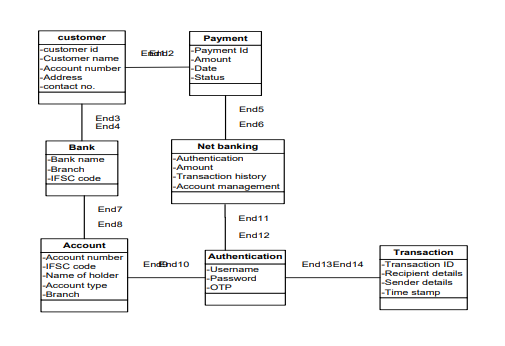
Q 2) derive boundary class, controller class and entity class

|  |  |  |
| --- | --- | --- |
| **Boundary class** | **Controller class** | **Entity class** |
| Payment option boundary  Card payment boundary  Wallet payment boundary  Cash payment boundary  Net banking payment boundary | Payment initiate controller  Card payment controller  Wallet payment controller  Cash payment controller  Net banking controller | Customer  Authentication server  Card  Wallet  Bank account |

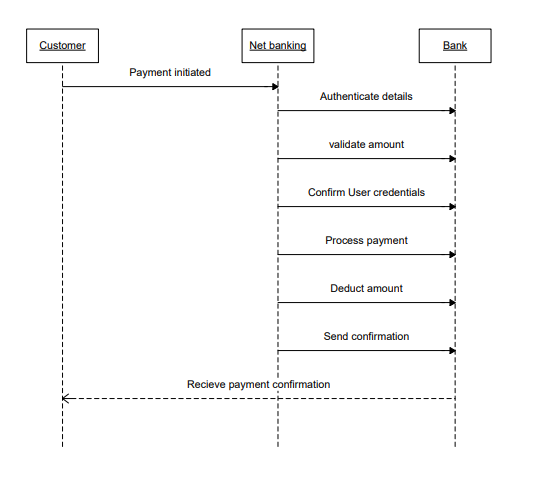
Q 3) place above classes in three tier architecture

|  |  |  |
| --- | --- | --- |
| **Presentation Layer** | **Business Logic Layer** | **Data Layer** |
| Payment option boundary  Card payment boundary  Cash payment boundary  Wallet payment boundary  Net banking boundary | Payment initiate controller  Card payment controller  Wallet payment controller  Cash payment controller  Net banking controller | Customer  Authentication server  Bank account  Wallet  card |

Q 4) explain domain model relationship



Q 5) sequence diagram



Q 6)

Conceptual Model for this case

Conceptual model is nothing but the relationship between different entities of the model. In this model we understands how a relationship between a customer ,bank and net banking services works.

Here is the customer who is opting for the payment through net banking must have an account in any particular bank which offer net banking service this is the relationship among these three entities.

So when a customer select a net banking the bank receives the notification to validate the details and process the payment. After authenticating the details and user credentials the bank processes the payment and sends the confirmation to the customer and that’s how the net banking payment is made.

Q 7)

MVC or Model, View, Controller is a software architectural pattern commonly used for developing user interfaces. It divides an application into three interrelated components.

1. **Model:** this represent the data and business logic of the application. It directly manages the data, logic and rules of the application.
2. **View**: this is the user interface of the application. It display the data from the model to the user and sends user commands to the controller.
3. **Controller**: this act as a intermediary between model and view. It processes user input, interact with the model and update the view accordingly.

Following rules need to be consider to derive classes from the use case diagram.

* Identify the actors determine the actors involved in the use case, each actor may correspond to controller or set of controllers.
* Define use case each use case can lead to the creation of model class that encapsulate the data and behavior relevant to that use case.
* Establish relationship identify relationships between use cases and model classes. This helps in defining how data flows between the model and view.
* Separation of concerns ensure that the responsibilities of the model view and controller are clearly defined and separated. The model should not directly interact with the view.

3 tier typically consists of presentation, business logic and data accesses layers, the following guidelines can be applied

1. Presentation Layer: this layer corresponds view in the MVC. Classes in this should handle user interface elements and user interactions. They should communicate with controller to send user inputs and receive updates.
2. Business logic layer: this layer corresponds to the controller and part of the model MVC. Classes should contain business rules and logic of the application. They should process inputs from the presentation layer and interact with data access layer to retrieve or manipulate data.
3. Data Access layer: this layer corresponds to the model in MVC. Classes in this layer should handle the data storage and retrieval, interacting with data bases or other data source. They should provide a clean interface for the business logic layer to access data.

By following above guidelines we can effectively implement MVC architecture within 3 tier architecture ensuring well structure and maintainable application.

Q 8)

In the waterfall model which is linear and sequential approach to software development, business analyst plays a crucial rule at each stage of the project.

1. **Requirement Analysis**

Stakeholders engagement: BA engage with stakeholders to gather requirements ensuring that all perspective are cancelled.

Requirement documentation: they document functional and non-functional requirements in a clear and concise manner often using technique like use case diagram, user stories and requirement specifications.

Feasibility Study: BA assess the feasibility of the project in terms of cost, time and technology providing insights that guide decision making.

1. **System Design**

Design Specifications: BA collaborate with system architects and developers to translate requirements into design specifications ensuring that design aligns with business needs.

Prototyping: they may create prototypes or wireframes to visualize the system, helping stakeholders understand how final product will function.

Translate business requirements into functional and non-functional requirements.

Ensure that all use cases, workflows and processes are accurately represented in the design.

1. **Implementation(Development)**

The Business Analyst supports the testing team by ensuring that the system is validated against the original business requirement.

Review test cases to ensure they cover all business requirements.

Facilitate user acceptance testing(UAT) by preparing and test scripts and scenarios.

Assist in validating that the systems functions as expected from business perspective.

Manage any change request if discrepancies between the requirements and implementation arise.

1. **Deployment**

The Business Analyst ensures the system is ensure the system is deployed as per business needs and all necessary stakeholders expectations are met.

Coordinate with the business team and technical teams to ensure the deployment plan is smooth and causes minimal disruptions.

Support end user training and ensure users are familiar with the systems functionalities.

Conduct final validation to confirm that the system meets the agreed requirements and objectives.

1. **Maintenance**

The Business Analyst remains involved post deployment to address any changes, enhancements or issues that may arise.

Manage change requests, enhancements or fixes that come after the system is alive.

Conduct reviews and gather feedback from users to identify areas for improvement.

Ensure that the systems continues to meet the business objectives and provide value.

Collaborate with the development and support teams to address any ongoing issues or maintenance needs.

Throughout the waterfall model the Business Analyst plays an integral role by ensuring that the business requirements are clearly defined, accurately communicated and fulfilled at every stage. They act as a bridge between the business stakeholders and the technical team, ensuring the smooth communication, understanding and alignment on the project goals.

Q 9)

**Conflict Management** refers to the process of identifying and addressing conflicts in a constructive manner to prevent in escalation and find a resolution that satisfy all parties involved. It understands the root cause of the conflict, ensuring the open communication and applying strategies to resolve the disagreement in a way that minimizes the negative impact while fostering cooperation and maintaining relationship.

Thomos-Killmann is widely considered approach for the field of conflict management which identify the five key strategies based on two dimension that is assertiveness and cooperativeness

* Competing(high assertiveness and low cooperativeness)

It is used when the person uses competitive approach when they pursue their own goal aggressively with little regards for the needs and opinions of others. This approach is assertive and uncooperative.

* In situation where the quick and decisive action is necessary.
* When unpopular decision need to be enforced such as during emergencies.
* When defending against a person trying to take advantage of non confrontational attitude.
* Collaborating(high assertiveness and high cooperativeness)

This approach seeks win win solution by fully addressing the concerns of both parties. It involves working together to find mutually beneficial solution.

* When the issue is important to both parties and they are willing to invest time and effort to find the solution.
* When both parties perspectives are important and consensus is needed.
* In situations where maintaining relationship crucial and creative problem solving can be used.
* Compromising (moderate assertiveness, moderate cooperativeness)

Compromising seeks middle ground where both parties give up something in order to reach an agreement. This approach is moderate.

* When the goals are moderately important but not a worth long, drawn out resolution process.
* When both parties have equal power and are committed to resolving the issue quickly.
* When time is limited and expedient solution is needed.
* Avoiding(low assertiveness and low cooperativeness)

In this approach persons sidesteps or postpones the dealing with the conflict altogether. This can be done by withdrawing from the situation or avoiding the issue.

There are two types of individual assertive and cooperative. The one who are concerned for their own needs are called assertive and where as the ones concerned for others need are called cooperative.



Q 10) **Reasons for Project Failure**

1. Improper requirement gathering.
2. Continuous change requirement.
3. Lack of user requirement.
4. Lack of executive support.
5. Unrealistic expectations.
6. Improper planning.

Q 11) Challenges faced by Business Analyst

1. Lack of training
2. Obtaining sign off on requirements
3. Change management with respect to cost and timeline.
4. Coordination between testers and developers
5. Conducting meetings
6. Making sure status report is effective.
7. Driving clients for UAT completion.
8. People management (coordinating with different people and different teams)
9. Overall making sure project health is good shape and delivered as per the timeline without any issue.

Q 12) Document Naming Standards and file naming convention

All documents should be named using some documents standard like following

[project ID][document type]V[x]D[y].ext

Ex- PQ786BRDV1D2.docx

Q 13) Do’s and Don’ts for a business analyst

1. Never say NO to client.
2. Never imagine anything in terms of GUI.
3. There is no word called as “by default”
4. Consult a SME for clarifications for the requirement
5. Every problem of client is unique.
6. Go to client with plain mind with no assumptions.
7. Do not interrupt the client, let the client complete and then ask queries if any.
8. Try to extract the leads to the solution from the client itself.
9. Never try to give solutions to the client straight away with your previous experience.
10. Try to concentrate on the important and truly required requirements.
11. Don’t be washed away by the add on functionalities or don’t even imagine solution on screen basis.

Q 14)

|  |  |
| --- | --- |
| **Packages** | **Subsystems** |
| Packages are primarily used for organizing and grouping related elements like classes interfaces and design artifacts | Subsystem are more complex and represent a logical grouping of multiple packages. They typically encapsulate a specific functionality or domain within a system. |
| Packages are typically smaller and more granular tham subsystem. They are often used to organize code at more local level such as within a specific calss or module. | Subsystem are larger and more encompassing than packages. They represent a higher level of abstraction and may span multiple modules or components of a system. |
| Packages can have diferent level of visibility such as public , private or protected. This determines whether the elements within the package can be assessed from outside the package or from within other packages. | Subsystem typically have well defined interface that specifies which components are visible to other subsystems and how they can interact. This helps to maintain modularity and to control the flow of information between different parts of the system. |

Q 15)

**Camel casing** is also known as camel case and upper camel case, is a naming convention for variables classes and functions , other identifiers in programming languages including C, C++, Javascript.

Some of the benefits of camel casing are

**Readability-** camel casing makes identifiers more readable by visually separating words and making it easier to distinguish between them.

**Consistency**- camel casing is widely used naming convention, so it helps to ensure consistency across different code bases and projects.

**Type – safety**- camel casing can help to improve type safety in some programming languagesuch as java and C#.

Q 16) development server and accesses a business analyst has?

**A development server** is type of server specifically designed for developing and testing software applications. It provides runtime environment along with all the necessary hardware and software tools, for developers to code, debug and test their applications before deploying it to production environment.

These servers are isolated from production environment to prevent changes made during the development phase. Development phase are typically accessible locally within the development environment, allowing developers to work directly with the server without relying external connections.

Business Analyst has

Read only access, Collaborative access and limited configuration access.

Q 17) Data Mapping

**Data mapping** is the process of establishing a relationship or connection between data elements in two or more data source or data formats.

The purpose of data mapping is to ensure that the data can be accurately and effectively transferred, converted or transformed between different systems, databases and formats.

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

Q 18)

**API is the abbreviation of Application Data Interface**, which is set of definitions and protocols for building and integrating application software.

Following steps need to consider for date conversion using API conversion.

1. **Implement a date format conversion endpoint**: create a dedicated API endpoint within your application that accepts date strings in mm-dd-yyyy format. This endpoint will be responsible for converting the income date to the dd-mm-yyyy format used by your application.
2. **Use a date parsing library**: utilize a date parsing library such as java time or dateutil in java to parse the incoming date string in dd-mm-yyyy format. These libraries provide function to extract individual date components (date, month, year) from a date string.
3. **Convert the parsed date**: once the date components are extracted , reconstruct the date string in the dd-mm-yyyy format using the extracted components.
4. **Return the converted date**: return the converted date string as the response from the API endpoint.
5. **Integrate with the external endpoint**: modify the communication between your application and the external application to direct the incoming date data to the date point conversion API endpoint. This will ensure that the date data is transformed to the correct format before it enters your application processing pipeline.