**Capstone Project 3**

A customer can make a payment either by Card or by Wallet or by Cash or by Net banking.

**Question 1:**

Draw a Use Case Diagram

**Answer:**

****

**Question 2:**

Derive Boundary Classes, Controller classes, Entity Classes.

**Answer:**

**Boundary Class**:

* Boundary Class acts as an interface between the system and the external actors.
* All the use cases are represented as Boundary classes.

Example: PaymentModeBoundary, LoginBoundary, OrderConfirmationBoundary

**Controller Class**:

* Controller Class acts as an intermediate between Boundary and Entity Classes.
* Receives input from Boundary classes, processes it and updates it to Entity Classes.

Example: PaymentInitiatedController, CardPaymentController, WalletPaymentController, DatabaseMangementController, NetbankingController

**Entity Class**:

* Represents a real-world object or concept.
* Contains Data and Behavior of that object.
* All the actors are represented as entity classes

 Example: CustomerEntityclass, AppDBEntityclass, FCHEntityclass, OBSEntityclass, UPIAPPEntityclass

**Question 3:**

Place these classes on a three tier Architecture

**Answer:**

Application Layer: Consists of Boundary classes

* PaymentModeBoundary
* CardpaymentBoundary
* WalletpaymentBoundary
* CashpaymentBoundary
* NetbankingBoundary
* LoginBoundary
* LoginBoundary
* CancelOrderBoundary
* ReturnBoundary
* OrderConfirmationBoundary
* SmsBoundary
* EmailBoundary

Business Logic Layer: Consists of Controller Classes

* PaymentInitiatedController
* CardPaymentController
* WalletPaymentController
* DatabaseMangementController,
* NetbankingPaymentController

Database Layer: Consists of Entity Classes

* CustomerEntityclass,
* AppDBEntityclass
* FCHEntityclass
* OBSEntityclass

UPIAPPEntityclass

**Question 4:**

Explain Domain Model for Customer making payment through Net Banking

**Answer:**

****

**Question 5:**

Draw a sequence diagram for payment done by Customer Net Banking

**Answer:**

****

**Question 6:**

Explain Conceptual Model for this Case

**Answer:**

Conceptual Model: A conceptual model is a high-level, abstract representation of a system, process, or idea. It's a simplified, diagrammatic representation of the key concepts, relationships, and rules without getting into technical details.

A conceptual model might include:

* **Entities**: Customer, Account, Payment, Transaction
* **Relationships**: Customer has Accounts, Account receives Payments, Payment initiates Transaction
* **Attributes**: Customer name, Account number, Payment amount, Transaction status
* **Rules**: A customer can have multiple accounts, a payment is associated with one transaction

The relationships between these entities can be described as follows:

1. **Customer**: This node represents the customers or users of net banking services.
2. **Service awareness**: Customers should be aware of the available net banking services and their features.
3. **Privacy of data:** The importance/significance of this node is to protect the privacy and confidentiality of customer data in the context of net banking.
4. **Technology awareness**: The significance of this node is that customers should be aware and comfortable with the underlying technology used in net banking services.
5. **Trust & Support**: This node indicates that the bank provides such good services that it will help to enhance the customer’s trust.
6. **Bank:** This node represents a service provider responsible for offering net banking services.
7. **Online information**: This aspect highlights the importance of providing accurate and up-to-date online information about net banking services to customers.
8. **Security & Privacy**: The bank should adapt to the security policies which will help the customers to keep their data related to their transaction secure and private.
9. **Infrastructure:** This component suggests that the underlying technological infrastructure, including hardware and software systems, plays an important role in enabling net banking services.
10. **Policies:** This node represents the various policies and regulations that govern the implementation and operation of net banking services, ensuring compliance and customer protection.

**Question 7:**

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 (Model-View-Controller) architecture is a software design pattern that separates an application into three interconnected components:

1. Model: Represents the data and business logic of the application. It manages the data and performs operations on it.

2. View: Responsible for rendering the user interface (UI) and displaying data to the user.

3. Controller: Acts as an intermediary between the Model and View, handling user input, updating the Model, and selecting the appropriate View to render.

**Rules to derive the classes from use case diagram-**

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

The combination of two actors and one use case results in two boundary classes.

The combination of three actors and one use case results in three boundary classes.

2. Use case will result in controller class.

3. Each actor will result in one entity class.

**Consider the example of Online shopping application with the following use case:**

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

View Classes-Login View, PaymentOptionView, NetBankingView, BankSelectionView, CredentialsView, PaymentAmountView, PaymentConfirmationView, LogoutView

Controller Classes-LoginController, PaymentOptionController, NetBankingController, BankSelectionController, CredentialsController, PaymentAmountController, PaymentContirmationController, LogoutController

**Question 8:**

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

**Answer:**

In a Waterfall model project, a Business Analyst (BA) plays a crucial role in the following stages:

1. Requirements Gathering:

* Elicit requirements from stakeholders through interviews, surveys, and workshops.
* Document requirements in Business Requirements Documents (BRDs) or Functional Requirements Documents (FRDs).
* Ensure requirements are clear, complete, and unambiguous.

2. Analysis:

* Analyze requirements to identify business needs and solutions.
* Develop use cases, user stories, and process flows.
* Identify and document business rules and constraints.

3. Design:

* Collaborate with architects and designers to ensure design aligns with business requirements.
* Provide input on user interface and user experience design.

4. Implementation:

* Support developers in understanding business requirements.
* Clarify requirements and answer questions.

5. Testing:

* Develop test cases and test scripts based on business requirements.
* Participate in testing to ensure requirements are met.

6. Deployment:

* Support the operations team in deploying the system.
* Ensure business stakeholders are trained and ready for the new system.

**Question 9:**

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

**Answer:**

**Conflict management** refers to the process of identifying, addressing, and resolving conflicts in a constructive manner to minimize their negative impact and maximize their potential benefits.

The Thomas-Kilmann Conflict Mode Instrument (TKI) is a widely used framework for conflict management. It identifies five modes or strategies for handling conflicts, based on two dimensions:

* **Assertiveness** (attempting to satisfy one's own needs and concerns)
* **Cooperativeness** (attempting to satisfy the other person's needs and concerns)

Here are the five modes:

1. **Competing** (Assertive, Uncooperative): Focus on winning, achieving one's own goals, and satisfying one's own needs, often at the expense of others.

2. **Collaborating** (Assertive, Cooperative): Work together to find a mutually beneficial solution, satisfying both parties' needs and concerns.

3. **Compromising** (Moderately Assertive, Moderately Cooperative): Find a middle ground, sacrificing some of one's own needs and concerns to reach a mutually acceptable solution.

4. **Avoiding** (Unassertive, Uncooperative): Sidestep or withdraw from the conflict, neglecting one's own needs and concerns.

5. **Accommodating** (Unassertive, Cooperative): Prioritize the other person's needs and concerns, sacrificing one's own needs and goals

**Question 10:**

List the reasons for project failure?

**Answer:**

Here are some common reasons for project failure:

* Poor planning and inadequate scope definition.
* Unrealistic expectations and overcommitting
* Inadequate resource allocation (time, money, personnel)
* Lack of clear communication and stakeholder management
* Ineffective project management and leadership
* Poor team dynamics and lack of collaboration
* Inadequate training and skills development
* Unrealistic timelines and deadlines
* Scope creep and uncontrolled changes
* Inadequate testing and quality assurance
* Lack of stakeholder support
* Inadequate monitoring and control
* Failure to adapt to change and evolving requirements
* Inadequate documentation
* Unrealistic budgeting and cost management
* Lack of accountability and ownership

**Question 11:**

List the Challenges faced in projects for BA?

**Answer:**

Here are some common challenges faced by Business Analysts (BAs) in projects:

* **Requirements gathering and management**: Eliciting and documenting accurate requirements from stakeholders.
* **Stakeholder management**: Managing diverse stakeholders with conflicting interests and expectations.
* **Communication**: Effectively communicating technical and business information to various audiences.
* **Scope creep**: Managing changes to project scope and ensuring they align with business objectives.
* **Technical complexity**: Understanding and analyzing complex technical systems and data.
* **Solution** **evaluation**: Assessing and recommending suitable solutions to meet business needs.
* **Project management**: Collaborating with project managers to ensure project deliverables are met.
* **Change management**: Helping stakeholders adapt to changes introduced by the project.
* **Team collaboration**: Working effectively with cross-functional teams, including IT and stakeholders.
* **Documentation and reporting**: Creating and maintaining accurate project documentation and reports.
* **Stakeholders buy-in**: Gaining support and commitment from stakeholders for project outcomes.
* **Continuous learning**: Staying up to date with industry trends, tools, and methodologies.

**Question 12:**

Write about Document Naming Standards?

**Answer:**

Document Naming Standards refer to a set of rules and guidelines for naming and organizing documents in a consistent and logical manner. These standards ensure that documents are easily identifiable, accessible, and manageable throughout their lifecycle.

Document naming syntax:

ProjectID-Document Name-v []-D []. Ext

Example:

ABC123-BRD1-V1.0-D1.ext

* ABC123 is Project ID
* BRD1 is Document Name
* V1.0 is the Version Number
* D1 is the Document Number

**Question 13:**

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

**Answer:**

Here are some key Do's and Don'ts for a Business Analyst:

DO’s:

* **Ask questions**: Clarify requirements and assumptions to ensure understanding.
* **Listen actively**: Pay attention to stakeholders' needs and concerns.
* **Document thoroughly**: Record requirements, decisions, and changes.
* **Analyze thoroughly**: Evaluate data, processes, and systems to identify opportunities.
* **Communicate clearly**: Explain complex concepts in simple terms.
* **Set realistic goals and timelines**.
* **Collaborate**: Work with stakeholders, IT, and project teams.
* **Stay organized**: Prioritize tasks, manage time, and meet deadlines.
* **Continuously learn**: Update skills, knowledge, and industry trends.
* **Be adaptable**: Adjust to changing requirements and project scope.

DON'T:

* Assume requirements
* Make technical decisions alone
* Overcommit.
* Ignore stakeholders
* Skip documentation
* Miss deadlines
* Forget to validate

**Question 14:**

Write the difference between packages and sub-systems?

**Answer:**

Packages and sub-systems are both used to organize and structure complex systems, but they serve different purposes and have distinct characteristics.

|  |  |
| --- | --- |
| Packages | Sub Systems |
| A package is a logical grouping of related components | A sub-system is a physical grouping of components. |
| Packages focus on reusability and functional cohesion | Sub-systems focus on achieving a specific system-level goal. |
| Packages are typically smaller in scope | Sub-systems encompass a broader range of components. |
| Packages are designed to be self-contained | Sub-systems often rely on external components. |
| Packages are designed to be reusable | Sub-systems are designed for a specific system or project. |
| Components within a package are related by functionality | Components within a sub-system are related by their contribution to the overall system's functionality. |

**Question 15:**

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

**Answer:**

Camel-casing is a naming convention in which the first letter of each word in a compound word is capitalized, except for the first word, which is in lowercase. It is also known as camelCase or camel notation.

Example: helloWorld, camelCasing, thisIsAnExample

Camel-casing is commonly used in:

* Programming languages
* API design
* Database design
* File naming
* URLs

Camel-casing is useful because it:

* Improves readability by separating words
* Reduces errors by avoiding spaces and special characters
* Enhances code consistency and maintainability

**Question 16:**

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

**Answer:**

A Development Server is a server environment where software developers and Business Analysts (BAs) can create, test, and refine applications, systems, or solutions.

Accesses that Business Analysts typically have on a Development Server:

* Read-only access
* Write access
* Testing and validation
* Debugging
* Collaboration with stakeholders and developers
* Version control (ex: Git)
* Data analysis
* Reporting and visualization
* API access
* Documentation

**Question 17:**

What is Data Mapping?

**Answer:**

Data Mapping is the process of creating a visual representation of the relationships between different data elements, systems, and processes. It explains how data flows, transforms, and interacts.

Benefits of Data Mapping:

* Improved data quality
* Enhanced data governance
* Increased data integration
* Better data-driven decision-making
* Reduced data redundancy and errors

Data Mapping serves several purposes:

* Data Integration
* Data Migration
* Data Warehousing
* Data Governance
* Business Intelligence

**Question 18:**

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:**

An API (Application Programming Interface) is a set of defined rules that enable different applications, systems, or services to communicate with each other. It allows for the exchange of data, functionality, or services between systems.

In the case of your application, which accepts data from another application in the US with a different date format (mm-dd-yyyy), API integration can be used to:

1. **Receive data**: Use the API to receive data from the US application, including dates in the format mm-dd-yyyy.

2. **Transform data**: Use data transformation techniques to convert the received dates from mm-dd-yyyy to your application's format, dd-mm-yyyy.

3. **Validate data**: Validate the transformed data to ensure it meets your application's requirements and standards.

4. **Store data**: Store the transformed and validated data in your application's database or storage system.

5. **Return data**: If necessary, return data to the US application or other systems, using the API to ensure seamless communication.

Benefits of API integration in this scenario:

* Seamless data exchange
* Automated data transformation
* Improved data accuracy
* Reduced manual data entry

By using API integration, your application can efficiently receive, transform, and utilize data from the US application, ensuring data consistency and accuracy.