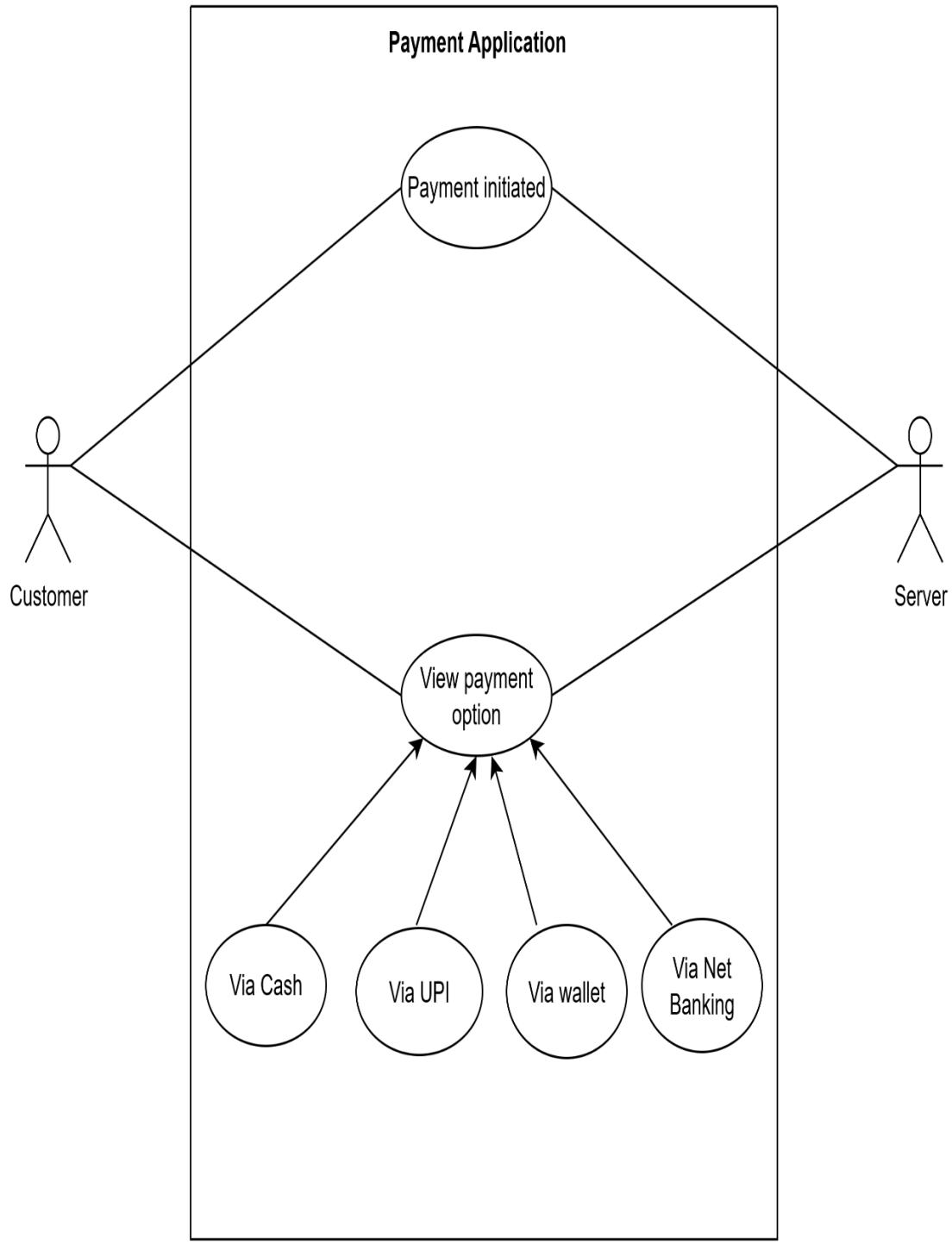
Capstone Project 3 – Part: 1

Question:1 - Draw use case diagram



Question 2: Derive Boundary Classes, Controller classes, Entity Classes.

Boundary Classes:

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 followings are the feature of the Boundary class.

1. This class is easier to be changed than the Entity and Control class.
2. The attribute of this class and screen layout are defined at the basic design.
3. In a class diagram, there are cases that the stereotype (<<boundary>>) is added.
4. In a class diagram, there are cases that is shown by the following icon.

Controller classes:

The followings are the feature of the Control class.

1. This class has a few attributes.
2. In a class diagram, there are cases that the stereotype (<<control>>) is added.
3. This class is a class to achieves use cases in the Use case diagram.
4. In a class diagram, there are cases that is shown by the following icon.

The Entity classes:

The Entity class is a class that has data.

The "E" of the ER diagram means "Entity" too, if you know the ER diagram, you easily understand.

The followings are the feature of the Entity class.

1.There are many cases that these objects of this class are perpetuated [1] in the DB.

1. The extraction of the class is like ER diagram [2].

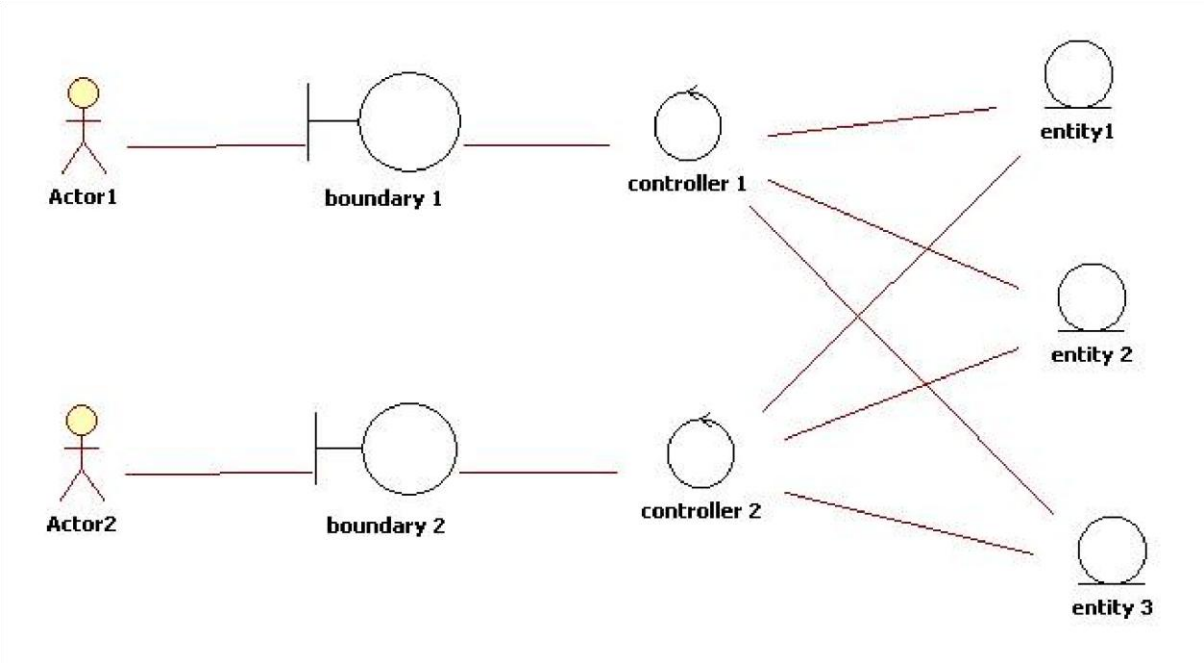
2. This class is related to the DOA (Data-oriented approach) [2].
3. The module cohesion of this class is high [3], and is not easy to be changed.
4. In a class diagram, there are cases that the stereotype (<<entity>>) is added.
5. In a class diagram, there are cases that is shown by the following icon.

Question 3: Place these classes on a three tier Architecture.

Application layer – customer payment boundary class, payment by net-banking controller class, payment by cash controller class, payment by e wallet controller class, payment by card controller class.

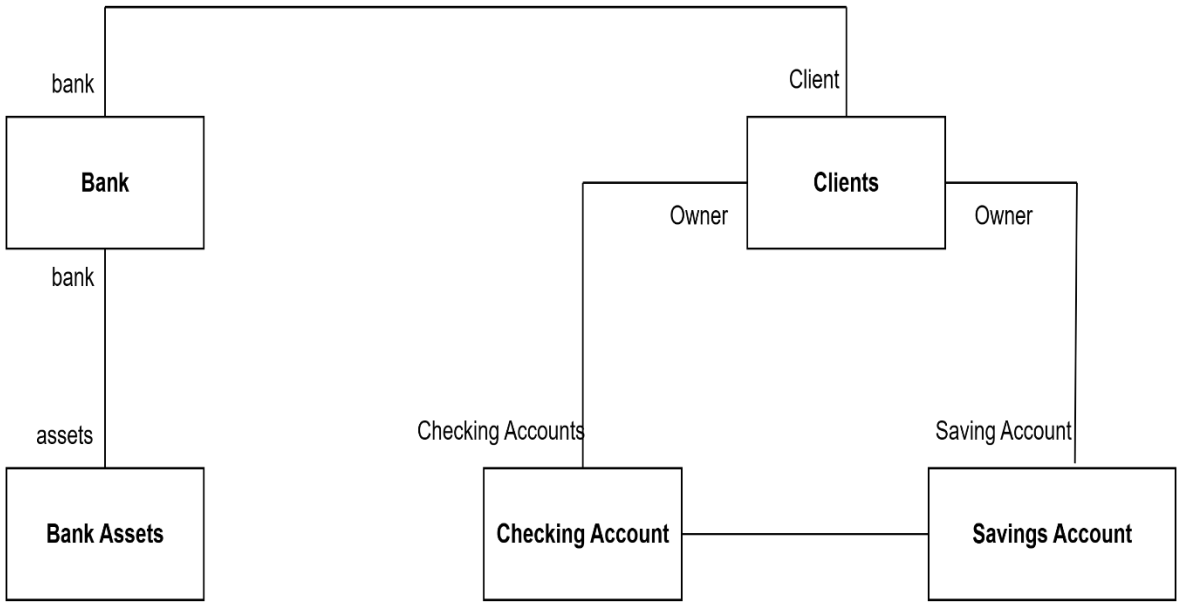
Business layer- merchant bank boundary class, payment gateway boundary class.

Database layer- e-wallet entity class, net banking entity class, cash entity class, card entity class.

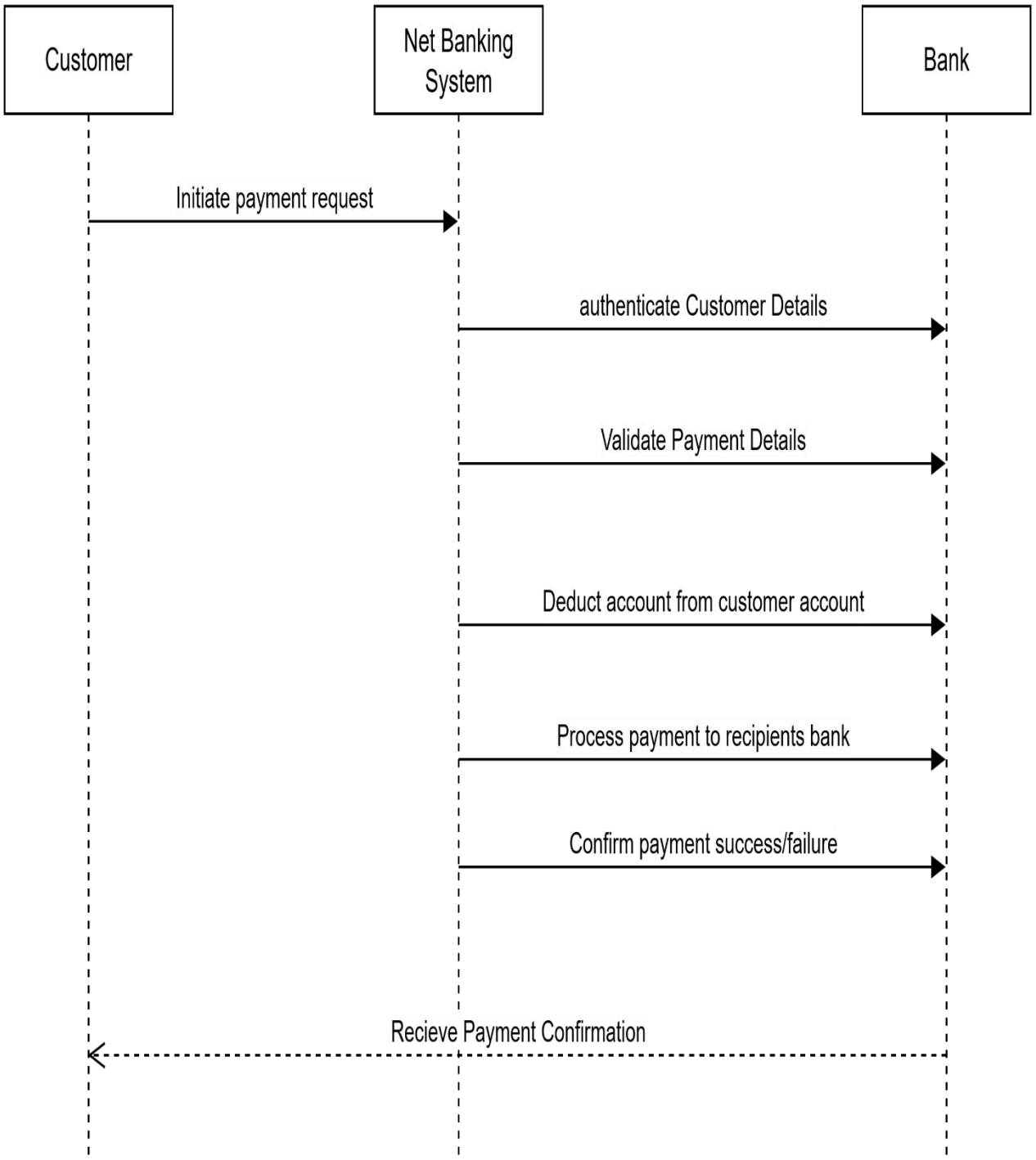


Question 4: Explain Domain Model for Customer making payment through Net Banking.

The domain model is an object model of the domain that incorporates both behavior and data. It is a way to describe and model real-world entities and the relationships between them which collectively describe the problem domain space. In UML, a class diagram is used to represent the domain model.

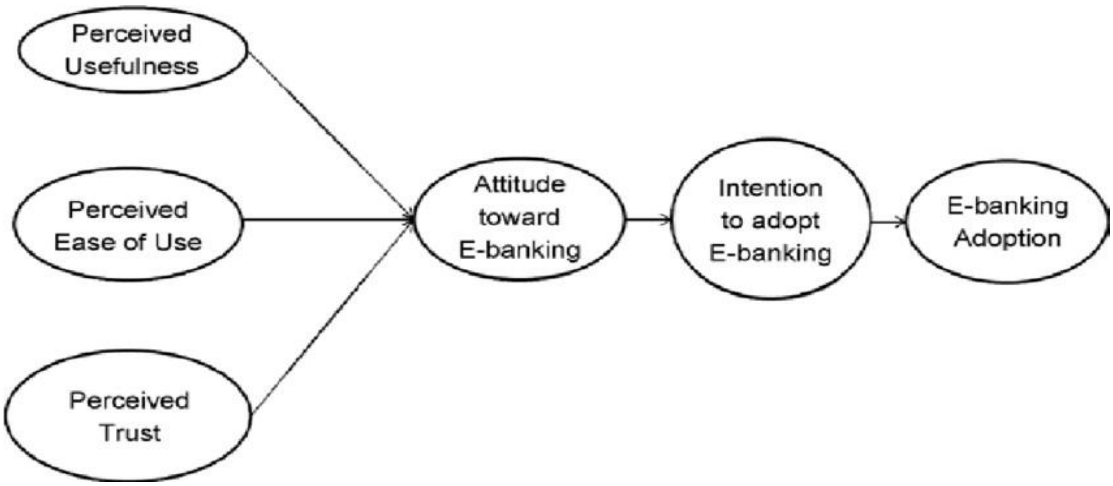


Question 5. Draw a sequence diagram for payment done by Customer Net Banking



Question6: Explain Conceptual Model for this Case.

The conceptual model is the model of an application that designers want users to understand. By using the software and perhaps reading its documentation, users build a model in their minds of how it works. Its primary objective is to convey the fundamental principles and basic functionality of the system which it represents. It is an abstract, psychological representation of how tasks should be carried out.



Question 7: What is MVC architecture? Explain MVC rules to derive classes from use case diagram and guideline to place classes in 3-tier architecture.

The Model-View-Controller (MVC) is a well-known design pattern in the web development field. It is way to organize our code. It specifies that a program or application shall consist of data model, presentation information and control information. The MVC pattern needs all these components to be separated as different objects.

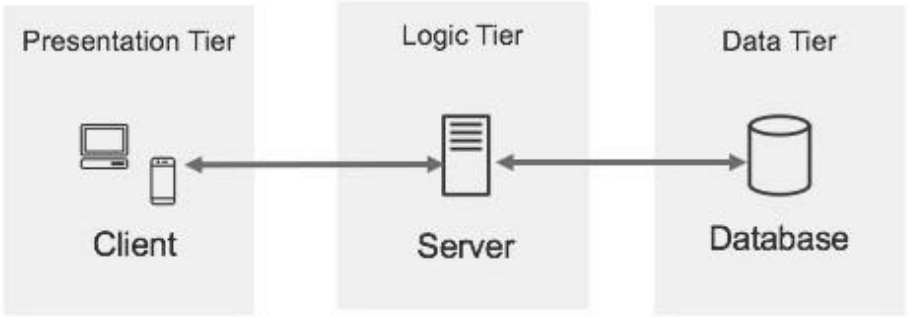
- **Model:** It represents the business layer of application. It is an object to carry the data that can also contain the logic to update controller if data is changed.
- **View:** It represents the presentation layer of application. It is used to visualize the data that the model contains.
- **Controller:** It works on both 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.

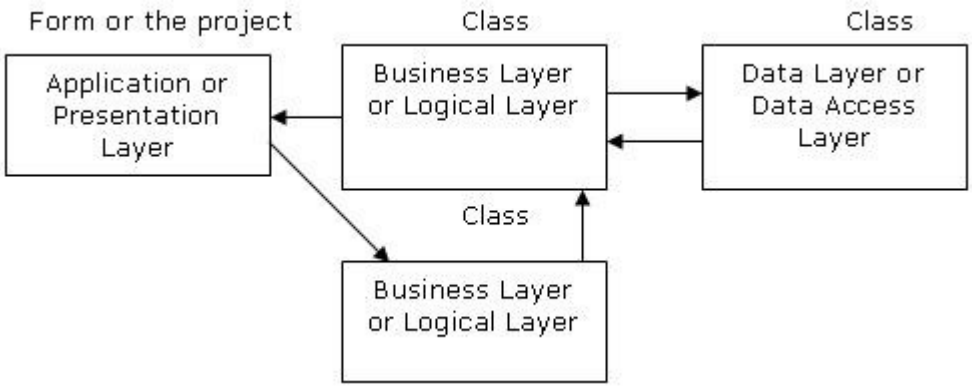
Advantages of MVC Architecture

The advantages of MVC architecture are as follows:

- MVC has the feature of scalability that in turn helps the growth of application.
- The components are easy to maintain because there is less dependency.
- A model can be reused by multiple views that provides reusability of code.
- The developers can work with the three layers (Model, View, and Controller) simultaneously.
- Using MVC, the application becomes more understandable.
- Using MVC, each layer is maintained separately therefore we do not require to deal with massive code.
- The extending and testing of application are easier.

Three-tier architecture, which separates applications into three logical and physical computing tiers, is the predominant software architecture for traditional client-server applications.





Question 8: **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.

The business analyst will verify 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.

1. Requirement Gathering and Analysis
2. Designing
3. Coding
4. Testing
5. Deployment
6. Maintenance

- **Requirement Gathering and Analysis:**

This is the initial stage of the project where is an involvement of the BA. 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 and UML diagrams get ready in this phase.

- **Coding:**

This phase is quite lengthy as the core development starts in this phase. Developer start 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 his 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 changes 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.

Question 9: **What is conflict management? Explain using Thomas – Kilmann technique.**

In the 1970s, researchers Kenneth Thomas and Ralph Kilmann developed a model for conflict resolution. It was called the Thomas-Kilmann model after them. Under this model, the term 'conflict' is described as the condition in which people's concerns can't be compared with the others. If two or more people or groups care about things that are contradictory to each other, then the outcome is conflict.

This model describes the two core dimensions while choosing a code of conduct in a situation of conflict: 'assertiveness' and 'cooperativeness'. Assertiveness is the extent to which you try to solve and resolve for your preferred outcomes. Think of this as the factor on the Y-Axis of a graph. On the other hand, Cooperativeness is the level to which you try to resolve the other party's problems. This is the factor on the X-Axis of the graph. Thomas-Kilmann's Five Modes for Handling Conflicts

From the correlation of these two and the scale of implementation, Thomas-Kilmann gave us the following five modes for handling the presented conflicts:

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 oneself. 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, 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 neither address 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, the most beneficial outcome in the Thomas-Kilmann conflict model. is both assertive and cooperative. This mode is the complete 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 the 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.

The Thomas-Kilmann Model is based on two dimensions: assertiveness and empathy. There are 5 conflict resolution strategies: Compete, Avoid, Accommodate, Collaborate and Compromise. Each strategy has its benefits and disadvantages. Choose the appropriate one according to the situation.

Question 10: **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

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 co Unrealistic due dates 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.

Question 11: **List the Challenges faced in projects for BA**

A BA is responsible for multiple tasks at the same time. From handling the projects, maintaining client relationships, interacting with stakeholders, and managing project deadlines, Business Analysts got a lot on their plate. Read below to find out the challenges faced by business analysts and a possible solution to them

1. Lack of Domain Knowledge

A Business Analyst needs to collaborate with the business users to understand the requirements. Domain knowledge plays a vital role in having a clear and complete understanding of the requirements.

It is challenging for Business Analysts to be assigned to a wide variety of projects as learning new domains needs time and energy.

Possible solution: Whenever you are assigned a new project, sit with the responsible person and understand the project requirements. Take notes whenever necessary and understand them thoroughly. It is challenging to learn new domains sometimes, but you must make mistakes.

Hence, go on a loop until you make a very bit of your knowledge count on your fingertips. It will help you while implementing and processing the outcome of the project.

2. Lack of up-to-Date Process

The success of a project does not happen overnight. First, much effort and mental exhaustion are poured in to bring results. Following this, the lion's share is the up-to-date process of maintaining and evolving the project. The biggest challenge is the lack of up-to-date techniques and documentation. In most cases, the Project Documentation is incomplete, which hampers productivity.

Possible solution: Testing a system is the most remarkable technique to learn about an existing project. It may seem odd, but it has been used for a long time. To further understand the flow, request a demo from a staff member or SME. Afterward, conduct extensive testing.

3. Changing Business Needs or Requirements

Business stakeholders frequently request revisions to requirements even after they have been finalized and approved, as experienced by Business Analysts.

It might happen more once, even for the exact requirement, making it one of the most frequent issues. These adjustments could have an impact on the Business Analysis effort as well as the total project effort, cost, and schedule.

Possible solution: A change in the implementation cycle might impact the delivery process even if there are approaches that, like Agile, accept change. Business Analysts and other essential stakeholders must therefore determine how the difference may be implemented in the best way.

4. Inadequate Stakeholder Involvement

One of the essential success criteria for every project is stakeholder involvement. You might encounter any of the following as a Business Analyst:

Lack of crucial stakeholders: If this occurs, there will be multiple problems since they will not be up to date on discussions about the most recent requirements. Either they won't be able to express their ideas, or they will subsequently propose revisions.

Stakeholders' Lack of Cooperation: Occasionally, you may encounter one or more stakeholders who are unwilling to cooperate. It could cause delays, sign-off problems, and even approval problems.

Possible solution: Business analysts may record the requirement discussions, particularly significant decisions made, and distribute them to all stakeholders in the meeting minutes. Before the scheduled requirements sessions, they may ask everyone who wasn't present to review the points. This will reduce the likelihood of miscommunication and reopening requirements items that have already been closed.

5. Unrealistic Timelines

As a Business Analyst, you may find yourself in a problematic situation where timelines might be the concern. In that case, pressure is created, which might hamper your work. In that case, understand how to tackle such a situation while maintaining the quality of the work.

Possible solution: Sales Team may be forced to accept a difficult situation for tactical reasons. As a Business Analyst, you cannot change the terms of the agreement, but you can evaluate its effects and inform management of the probable costs and losses. You have the option of starting over. Unrealistic Expectations from stakeholders are widespread. It's crucial to manage these expectations balanced without permanently damaging the relationships.

6. Technical Skills

When it comes to Business Analysts, it's a myth that they don't require technical skills. On the contrary, most of them are champions in coding, know how to maintain business processes, and have a knack for technically undertaking the requirements. Moreover, Business Analysts are involved in every step of the product development cycle; hence, they must understand the technical and functional side of the business as well.

Possible solution: Working with multiple clients, customers, and stakeholders is not easy. It requires a lot of skills to put in to bring the best results. Therefore, develop your skills over time. Whenever you are available, read, take courses and understand the technicality of the Project and the business. This will help you in developing better Project documents and will help in multiple ways.

7. Professionalism

Business analysts are one of the most underappreciated, underpaid, and ignored members of the IT world. They frequently serve as the binding agent between a project's technical and business aspects. They are the one who contributes to the development of the project plan and who supports

the project from beginning to end. They will collaborate with developers to ensure the project is constructed following the most current standards and satisfies the business' expectations.

8. Managing Communication

When you communicate effectively, you aid developers in understanding the needs, limits, and requirements of the business. You contribute to the development of solutions that benefit the client as well as the company. You guarantee the work is completed on schedule and to the required standards. But communicating the point is difficult. It involves a variety of abilities and trade secrets.

Possible solution: Soft skills are part of better work opportunities and personality. Try to communicate your views clearly and confidently to your team so they can understand them easily. It will help incur the communication gap between the team. While intersecting with the stakeholders, try to break the idea into pointers and explain the leads to them.

9. Conflict with Users

Sometimes, you might find yourself in a situation where you cannot understand the user's complaint. It happens during the product release stage and might come as rude feedback. Even conflict between stakeholders and business analysts may arise when a team suggests a new strategy pertinent to the existing business process.

10. Mindset

Business analysts must be prepared to deal with various difficulties throughout their work, from limitations of the technologies they employ to push back from stakeholders and other team members. But how one approaches their task can significantly alter if they are ready for the most typical obstacles.

Question 12: Write about Document Naming Standards.

1. **Keep file names short, but meaningful.**

Correct - /…/Orientation/20181105SchdlVlntrs.pdf

Incorrect -

The_schedule_and_volunteers_for_Orientation_Nov_18.pdf

2. **Avoid unnecessary repetition and redundancy in file names and folder names/file paths.**

Correct - /…/Doe/Events/Kids Sibs/20181105BnceHsRsrvtn.pdf

Incorrect - /

…/Doe/Events/KidsNSibs/20181105KidsNSibsBounceHouseReservati on.pdf

3. **The most preferred is title case (File Name).** Less preferable are, no separation (filename), underscores (file name), dashes (filename), or spaces (File Name).

Correct/Preferred – PSYCSyllabus.docx

Incorrect/Not Preferred – PSYC_syllabus.docx, psych syllabus.docx

4. **When including a number, use leading zeros to ensure files sort properly, i.e., "001, 002…101" instead of "1, 2… 101".**

Correct – (In alphanumeric sort order) Image01.jpg, Image02.jpg, Image03.jpg, Image10.jpg, Image11.jpg, Image20.jpg

Incorrect – (In alphanumeric sort order) image1.jpg, image10.jpg, image11.jpg,

5. **Date format should be YYYYMMDD (or YYMMDD) so years of files sort in chronological order.**

Correct - 2018FAPSYC100SmithTest01V02.docx,

2018FAPSYC100SmithSyllabusV03.docx

Incorrect - test psychology smith Fall 18.docx, smith psych 100 syllabus Fall 2018.docx

6. **When including a personal name in a file name give the family name first followed by the initials.**

Correct - DoeJL20180421.jpg

Incorrect -John-L-Doe20180421.jpg

7. **Avoid using common words such as "draft" or "letter" at the start of file names. Avoid using common words such as "draft" or "letter" at the start of file names.**

Correct – SyllabusV02Draft.docx, SyllabusV03Final.docx,

TestV01Draft.docx, TestV04Final.docx

Incorrect - DraftSyllabusV02.docx, DraftTestV01.docx,

FinalSyllabusV03.docx, FinalTestV04.docx

8. **Order the elements in a file name in the most appropriate way to retrieve the record**.

Correct -

/…/PlanningCttee/

20040630Agenda.rtf

20040630Minutes.rtf

20050120Agenda.rtf

20050120Minutes.rtf

/…/Events/

GardenParty20040630.rtf

ProcurementAward20040905.rtf

WeddingDinner20030304.rtf

Incorrect -

/…/SausageCttee/

Agenda1Feb2005.rtf

Agenda20Jan2005.rtf

Minutes1Feb2005.rtf

Minutes20Jan2005.rtf

/…/Events/

20030304WeddingDinner.rtf

20040630GardenParty.rtf

20040905ProcurementAward.rtf

9. The file names of records relating to recurring events should include the date and a description of the event, except where the inclusion of either of these elements would be incompatible with rule 2.

Correct - KidsNSibs20181012.docx, KidsNSibs20191016.pdf,

Orientation20180810.pptx

Incorrect – SibsWeekend.docx, WeekendWithTheKids.docx, 20180810.pptx

10. The file names of correspondence should include the name of the correspondent, an indication of the subject, the date of the correspondence and whether it is incoming or outgoing correspondence, except where the inclusion of any of these elements would be incompatible with rule 2.

Correct – /…/Returns/DoeJL20180815rcvd.txt

Incorrect – LetterFromJohnDoeReReturnAug18.txt

11. The version number of a record should be indicated in its file name by the inclusion of 'V' followed by the version number and, where applicable, 'Draft'.

Correct – SyllabusV02Draft.docx, SyllabusV03Final.docx,

TestV01Draft.docx, TestV04Final.docx

Incorrect - DraftSyllabusV02.docx, DraftTestV01.docx,

FinalSyllabusV03.docx, FinalTestV04.docx

12. Avoid using special characters, i.e., ~! @ # $ % ^ & * ( ) ` ; < > ? , [ ] { } ' "

Correct - GardenParty20040630.rtf

Incorrect – "Picnic & Garden Party, June 30, 2004".pdf

Question :13 - What are the Do's and Don'ts of a Business analyst

(Do's of a Business Analyst)

1. Understand Business Needs Clearly

- Engage stakeholders early and often.
- Ask the *right* questions to uncover the *real* problem.
- Clarify goals, constraints, and success criteria.

2. Document Requirements Precisely

- Use clear, unambiguous language.
- Maintain traceability from business needs → functional requirements → test cases.
- Use diagrams (e.g., UML, BPMN) to enhance understanding.

3. Communicate Effectively

- Bridge the gap between business and technical teams.
- Adapt communication style to your audience (executives, developers, users).
- Facilitate meetings, workshops, and demos with clarity.

4. Stay Involved Throughout the Project Lifecycle

- Support the development and testing teams with clarification.
- Validate if the delivered solution meets the business requirements.
- Help manage scope changes with proper impact analysis.

5. Continuously Improve

- Gather feedback on your work.
- Learn domain knowledge relevant to your project.
- Keep updating your skills (tools like JIRA, Confluence, SQL, Agile, etc.).

(Don’ts of a Business Analyst)

1. Don’t Make Assumptions

- Never assume you know what the stakeholder wants—*ask*.
- Always validate information before proceeding.

2. Don’t Write Vague Requirements

- Avoid generic phrases like “user-friendly” or “fast”.
- Be specific, measurable, and testable.

3. Don’t Ignore Stakeholders

- Avoid engaging only with one or two voices.
- Ensure you gather input from all relevant users and departments.

4. Don’t Overstep into Project Management

- Don’t try to manage timelines or resources unless assigned that responsibility.
- Stay focused on analysis and requirements unless your role explicitly combines duties.

5. Don’t Resist Change

- Be open to requirement changes with proper impact assessment.
- Don’t become attached to earlier versions of requirements.

Question :14 - Write the difference between packages and sub-systems.

Packages:

A Packages is a grouping and organizing element in which other elements reside, which must be uniquely named. In the UML, packages are used in a manner similar to the way directories and folders in an operating system group and organize files. For example, the project management system may be decomposed into a collection of classes organized into packages as follows:

Sub-Systems:

Recall that a system is an organized collection of elements that may be recursively decomposed into smaller subsystems and eventually into non decomposable primitive elements. For example, the project management system may be decomposed into the following:

A user interface subsystem responsible for providing a user interface through which users may interact with the system

A business processing subsystem responsible for implementing business functionality

A data subsystem responsible for implementing data storage functionality.

While a package simply groups elements, a subsystem groups elements that together provide services such that other elements may access only those services and none of the elements themselves. A subsystem is shown as a package marked with the subsystem keyword.

Feature	Package	Subsystem
Definition	A package is a grouping of related classes, interfaces, or components.	A subsystem is a logical grouping of elements that provides a specific behaviour or service.
Purpose	Used to organize and manage the structure of a model.	Used to define a logical unit of functionality in the system.
Focus	Focuses on organization and modularity.	Focuses on functionality and behaviour.
Contents	Contains classes, interfaces, other packages.	Contains packages, classes, components, interfaces, and diagrams.

Feature	Package	Subsystem
Dependency	Often used to show dependency between model elements.	Can have interfaces to show how it interacts with other subsystems.
Diagram Representation	Shown as a tabbed folder symbol in UML diagrams.	Shown as a package-like symbol with an "S" for subsystem.
Example Use	Grouping all utility classes into a utils package.	A Payment Processing subsystem handling all payment logic.
Visibility Control	Supports import and access visibility for elements.	Defines provided/required interfaces for integration.

Question :15 - What is camel-casing and explain where it will be used.

Camel-casing (also called *camelCase*) is a naming convention where:

- Multiple words are joined together without spaces.
- The first word starts with a lowercase letter.
- Every subsequent word starts with an uppercase letter (like a camel's humps)

Camel-case is a naming convention for writing file or object names using compounded or joined words with at least of those words beginning in a capital letter.

Camel-case is used in programming language to name different files and functions without violating the naming laws of the underlying language.

Camel-case is also known as medial capitals and Pascal case.

The term camel-case is derived from its appearance, which can resemble a camel's back. It is used in many programming languages that doesn't allow spaces in file names. Camel-case enables the creation of names that are more unique and have more meaning for the developer.

For example, file names big ball, Big Ball and big Ball can be read much more easily than big-ball

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.

Question 16: Illustrate Development server and what are the accesses does business analyst has?

A development server is an environment used by the development team to build, test, and debug the application before it is moved to staging or production.

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 run-time environment, as well as all hardware/software utilities that are essential to program debugging and development.

A development server is the core tier in a software development environment, where software developers test code directly. It is comprised of the essential hardware, software and other components used to deploy and test the software under development, including bulk storage, development platform tools and utilities, network access and a high-end processor. Upon testing completion, the application is moved either to a staging server or production/live server.

Business Analyst has the visualizing access in development server. BA has the access to all the functional servers and not to the technical servers.

Question 17: **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.

Data mapping has been a common business function for some time, but as the amount of data and sources increase, the process of data mapping has become more complex, requiring automated tools to make it feasible for large data sets.

Data mapping is an essential part of many data management processes. If not properly mapped, data may become corrupted as it moves to its destination. Quality in data mapping is key in getting the most out of your data in data migrations, integrations, transformations, and in populating a data warehouse.

Data mapping is an essential part of ensuring that in the process of moving data from a source to a destination, data accuracy is maintained. Good data mapping ensures good data quality in the data warehouse.

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.

An API, is Application Programming Interface, 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.