A customer can make a payment either by card or by wallet or by cash or by Net Banking.

1. **Draw a Use Case Diagram.**

**Answer –**

A Use Case Diagram is a visual representation that illustrates the interactions between users (actors) and a system. It captures the functional requirements of a system, showing how different users engage with various use cases or specific functionalities within the system.

Use Case Diagrams provide a high-level overview of a system’s behaviour, making them useful for stakeholders, developers and analysts to understand how a system is intended to operate from the user’s perspective and how different processes relate to one another. They are crucial for defining system scope and requirements.

**Components of Use Case Diagram –**

**Actors –** Customer, Admin

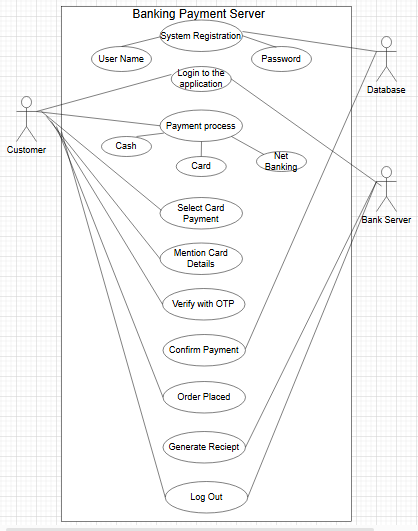
**Use Cases –** Browse Products, Add to Cart, Checkout, Manage Inventory (Admin)

**Relations –** The customer can browse products, add to cart and complete checkout.

The Admin can manage the inventory.

How to draw a Use Case Diagram –

* Identify actors – Determine who or what interacts with the system. These are actors. They can be users, other systems or external entities.
* Identify Use Cases – Identify the main functionalities or actions the system must perform. These are Use Cases. Each use case should represent a specific piece of functionality.
* Connect Actors and Use Cases – Draw lines (associations) between actors and the use cases they are involved in. This represents the interaction between actors and the system.
* Add system boundary – Draw a box around the actors and use cases to represent the system boundary. This defines the scope of the system.
* Define relationships – If certain use cases are related or if one use case is an extension of another, we can indicate these relationships with appropriate notations.
* Review and refine – Step back and review your diagram. Ensure that it accurately represents the interactions and relationships in the system. Refine as needed.
* Validate – Share your use case diagram with stakeholders and gather feedback. Ensure that if aligns with their understanding of the system’s functionality.



1. **Derive Boundary Classes, Controller Classes, Entity Classes**

**Answer –**

The analysis object model instantiates the Entity-Control-Boundary Pattern.

ECB is a simplification of the Model-View-Controller Pattern.

ECB partitions the system into three types of classes: Entities, Controls and Boundaries.

Boundary classes model the parts of the system that depend on its surroundings. Entity classes and control classes model the parts that are independent of the system’s surroundings.

* **Boundary Classes** – these classes represent the system’s interface with outside world, including user interfaces, network connections and other external systems. They handle user input, display information and interact with the controllers to execute actions.

|  |  |
| --- | --- |
| **Boundary Class – (Use Cases) Actors speak to the system (Authentication Information)**   * Combination of 1 actor and a use case 1 boundary class * Combination of 2 actors and a use case 2 boundary class * Combination of 3 actors and a use case 3 boundary class and so on   All the actors should be – (Primary Actors)  Primary actors – who initiate the use cases and interact with the system | **Customer registration**   * Customer Login * Bank server logs in * Customer logout * Bank server logs out |

* **Controller Classes** – These classes act as intermediaries, orchestration the execution of commands and coordinating interactions between the boundary and the entity classes. They handle user input from the boundary, process it and then interact with the entity classes to perform the necessary actions.

|  |  |
| --- | --- |
| **Controller Class – (Handles Users (Primary actors) input and processes the data**   * Use case will consider as Controller class system | **Registration Controller**   * Login Controller * Payment Controller * Credential Controller * Net Banking Controller * Logout Controller |

* **Entity Classes –** These classes represent the core data and business logic of the system,

Such as “Customer” Or “Order “. They are responsible for storing and managing the data related to the application.

|  |  |
| --- | --- |
| **Entity Class – All Actors**   * Each actor will be considered as an entity | * Customer * Bank Server * Cash * Card * Net Banking |

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

**Answer –**

In this three-tier architecture, the application tier handles the user interface, the Business Logic Layer manages the Business Logic and coordinates between the other tiers, the Data layer tier handles data storage and retrieval.

|  |  |
| --- | --- |
| Application Layer | Customer Registration  Customer Login  Bank Server Login |
| Business Logic Layer  (Primary actors associated with Boundary class) | Customer  Bank Server |
| Data Base Layer  (All the entity classes) – All actors | Customer  Bank Server  Cash  Card  Net Banking |

1. **Explain Domain Model for customer making payment through Net Banking.**

**Answer –**

Domain model is similar to the entity relationship model. The tables are connected to each other. In the below diagram, the customer table is connected to bank table, which is why the customer is able to make payment.

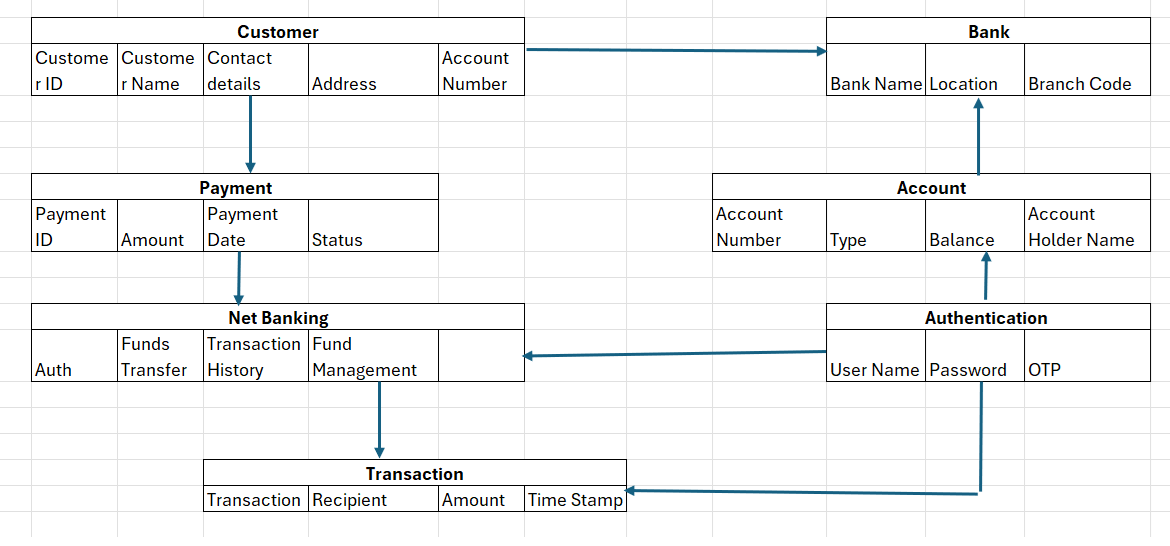
Customer table is also connected to payment table, because he should make the payment. Now the payment is done by net banking, so payment table is connected to net banking table.

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 authentication is to be performed there.

Also, the authentication table is connected to transaction table, because authentication will be done while transaction.

**Difference between ER diagram and domain model –**

|  |  |
| --- | --- |
| **ER Model –**   * Do not have attributes inside the box * It is a data modelling technique used in database design to represent table * Focuses on relationships required for storing and retrieving the data * Primarily used in database design | **Domain Model –**   * Do not have attributes mentioned inside the box * It is a conceptual model that represents real world entities * It focuses on capturing the behaviour of application * Used throughout the software development lifecycle |



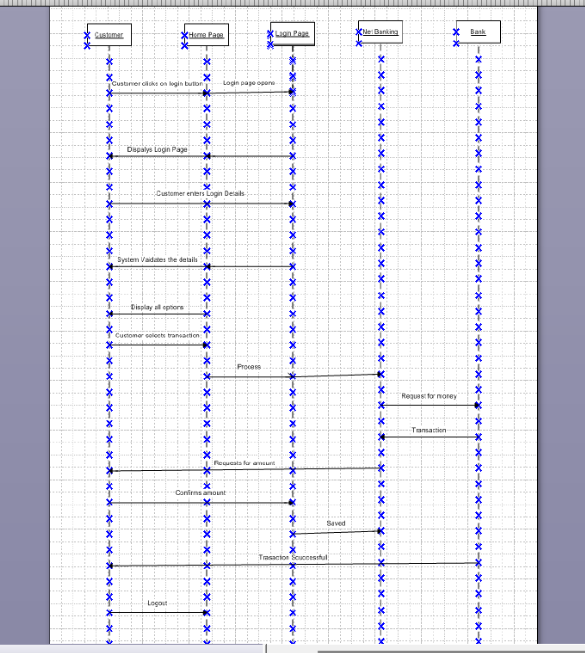
1. **Draw a sequence diagram for payment done by customer net banking**

**Answer –**

This diagram shows how the objects in the system interact and communicate with each other with time to achieve specific task. Developer will draw this.

It is used to show the ow of messages, events or actions between the objects of the system.

This diagram helps to visualize the behaviour of the system by showing the process in detail



1. **Explain conceptual model for this case**

**Answer –**

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

It serves as a foundation for designing the database schema, defining the application architecture and implementing the necessary functionalities within the system.

**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 and 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 and Privacy:** The bank should adapt 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 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



1. **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 is a design pattern where, the application is divided into 3 logical parts –

**Model, View and Controller**

**Each of these parts will have specific responsibility.**

**Model** – The Model represents the data and the business logic of the application. Model is responsible for multiple tasks like managing the application’s data, performing data validation, implementing business rules, and handling data access operations. Model does not depend on how the data is presented or how the user interacts with the application.

The model class is known about all the data that is needed to be displayed. This layer corresponds to the data-related logic that the user works with. It represents the data that is being transferred between View and Controller. It can add or retrieve the data from the database.

It responds to the controller’s request because the controller cannot interact with the database by itself.

The model interacts with the database and give the requested data. All the model classes are nothing but the entities. Model classes are represented as entity class.

**View** – The view is responsible for presenting the data to the user for handling the user interface.

The view can be a web page, a desktop application window, or any other form of user interface. It receives input from the user and passes it to the Controller for processing.

It represents the presentation of the application. View refers to the model.

It takes the data from Model and renders it in a way that is suitable for the user’s display or interaction.

For rendering the data, it uses query method. View does not depend upon application logic. View classes are represented as boundary class.

**Controller** – The controller acts as an intermediary between the Model and the View. It receives input from the user (via the View), processes the input by invoking the appropriate methods in the Model and then updates the View with the new data or state.

The Controller handles user interactions, interprets user input and translates it into instructions for the Model or the View. It coordinates the flow of data between the Model and the View, ensuring that they remain separated and independent of each other.

Whenever the user requests for anything, that request first goes to the controller. Controller works on the user’s request.

Takes input from the user/client. It interacts with the Model and View. Controller class represents as use case. Controller acts as a mediator between model and database.

Controller cannot directly get the data from the database. So, controller interacts with the model.

**Advantages of MVC –**

MVC has the feature of scalability, which in turn helps the growth of application. The components are easy to maintain. A model can be used by multiple views that provide reusability of code. By using MVC, the application becomes more manageable. As all the three layers are different and independent, they are maintained separately.

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



* Combination of one actor and one use case results in one boundary class
* Combination of two actors and one use case results in two boundary class
* Combination of three actors and one use case results in three boundary class

1. Use case will result in controller class
2. 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, Payment option view, Net Banking View, Bank Selection View, Credentials View, Payment amount View, Payment Confirmation View, Logout View.

**Controller Classes** – Login Controller, Payment Option Controller, Net Banking Controller, Bank Selection Controller, Credentials Controller, Payment Amount Controller, Payment Confirmation Controller, Logout Controller

**Guidelines to place classes in 3-tier architecture –**

**Presentation Layer** – This layer is nothing but a user interface. View is inside this layer.

**Tier Architecture** – has 3 layers

* Application Layer
* Business Logic Layer
* Database Layer

Data flowing from the 3 layers explains the 3-tier architecture where the information is passed. Here in elicitation techniques 3 tier architecture works as a medium where as a BA we get to know the information shared by the requester will it fit in application layer, business logic layer or database layer.

A diagram of a data layer

AI-generated content may be incorrect.

1. **Explain BA contributions in project (Waterfall Model – all stages)**

**Answer –**

Waterfall model is useful in the situation where the project requirements are well defined, and the project goals are clear.

* Waterfall model follows sequential approach.
* In this model each phase is completed entirely and then only moved to the next phase.
* Waterfall model relies on documentation to ensure that the project is well defined, and project team is working towards clear goals.
* Once that particular phase has been completed and once, we move to the next phase, we cannot go back to the previous phase to make changes.
* This model is suitable for the projects when requirements are clear.

**Requirement Gathering** –

First, the stakeholders are identified. In this phase, all the requirements are gathered from the stakeholders. BA and Project Manager participates in this phase. After completing this phase, BRD will be generated.

**Requirement Analysis** – The requirements are analysed to understand the scope of the project. Analysing means the BA will check all the requirements, if he founds convincing requirements then the BA will talk to the concerned stakeholder to clear it, remove the ambiguous requirements.

BA will prepare functional requirement. The document which contains the functional requirements is called (FRS) Functional Requirement Specification. Technical team will prepare non-functional requirement. The document which contains the non functional requirement is called (SSD) Supplementary Support Requirement. BA will combine FRS and SSD to form SRS (Software Requirement Specification). BA will prepare RTM by referring SRS.

**Design** –

After the requirements are cleared, Design phase starts. This has a detailed design document that outlines the software architecture, user interface and system components, ADD and solution document will be generated here. [High Level Design Document].

BA will collaborate with designers, architects and developers to translate requirements into system design. BA ensures that the design aligns with the documented requirements and addresses stakeholders’ needs.

**Development** –

The Development phase include implementation. It involves coding the software based on the design specifications. Programmers or developers are involved in this phase. Here BA acts as a mediator between the development team and the stakeholders. BA clarifies the requirements, check if the development is going on right track or not. BA also participates in scrum meetings.

**Testing** –

In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects. Testers are involved in this phase.

Test documents are generated here. BA works with the testing team to ensure that the solution meets the requirements. BA facilitates UAT. BA helps the users to know the functionality of the system and also helps them to use the system.

**Deployment** –

Once the software has been tested and approved, it is deployed to the production environment. BA ensures that there is smooth transition from development phase to the production phase.

**Implementation** –

This is the final stage of waterfall model. It involves running the code for the very first time in production phase. Release manager handles this phase.

BA will update documentation and requirements specifications to reflect changes in the system over time.

**Maintenance** – Running the code for second time in the production phase is called maintenance. This is done by support team.

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

**Answer –**

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

Conflict is an inevitable part of any workplace. So it is important to resolve it to promote learning and growth.

Conflict management is nothing but 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.

By identifying the conflicts efficiently, it will in turn be helpful to reduce negative impact and increase positive impact.

It is a process or skill to find creative ways to handle the disagreements. Thomas – Kilmann approach is widely used to recognize the approaches for conflict management.

A diagram of a company

AI-generated content may be incorrect.

**Y axis – assertiveness**

**X axis – co-cooperativeness**

**High Assertiveness and High Cooperativeness** – Collaboration – means working together to find solution

**High Assertiveness and Low Cooperativeness** – Competition – means defensive, that is standing for your individual beliefs and trying to win

**Low Assertiveness and High Cooperativeness** – Accommodation – stakeholder will prioritize their needs over others.

**Low Assertiveness and Low Cooperativeness** – Avoidance – means ignoring the conflict Assertiveness – the extent to which the person attempts to satisfy the other person’s concerns.

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

**Answer –**

Reasons for project failure are:

**Improper requirement gathering –**

If the requirements of the project are not gathered correctly, then this can lead to project failure.

**Lack of stakeholder involvement –**

A project can fail if the stakeholders are not participating in the process. The stakeholder’s input and feedback play a very important role in meeting 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. The team fails to identify the risks and do the risk mitigation, which can 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, the 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.

1. **List the challenges faced in projects for BA**

**Answer –**

Below are the challenges faced in project for BA –

* Lack of training
* Obtaining sign-off on the requirement
* Change management
* Co-ordination between developers and testers
* Conducting meetings
* Making sure status report is effective
* Driving clients for UAT completion
* Making sure that the project is going on right track and delivered as per the timelines without any issues
* Gathering clear and unambiguous requirements can be challenging
* Unable to understand what stakeholder with conflicting interest can be a difficult task for BA
* BA may face difficulties in understanding the requirements if the domain is not familiar to him
* Poor communication between stakeholder and BA can affect the process of gathering the information
* Technical complexity

1. **Write about Document Naming Standards**

**Answer –**

Document naming Standards are a set of guidelines for naming files and folders to ensure consistency, clarity and ease of organization within an organization or team.

A naming standard governs the format of file names in folders which have a naming standard applied. File naming standards helps you stay organized and makes it easier to identify your files.

Here’s the breakdown of key aspects of document standards as why they are important.

* Organization – They helped create a structure and easily navigable file system.
* Clarity – They make it easier to identify the purpose and content of documents.
* Collaboration – They ensure everyone in the team or organization uses the same naming conventions, facilitating easier file sharing and retrieval.
* Efficiency – They save time by making it easier to find and access the right documents.
* Version Control – They help track different versions of a document.

[ProjectID][Document Type]V[x]D[y].extension

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

1. **What are the Do’s and Don’t of a Business Analyst**

**Answer –**

Never say “NO” to the client

* There is no word called as “BY DEFAUL”
* Never imagine anything in terms of GUI
* Question everything in the world
* Go to the client with plain mind that is with no assumptions
* Listen to the client very carefully and after he is done, then ask questions
* Don’t interrupt the client
* Never try to give solutions to the client right away
* Try to concentrate only on important and required things
* Be like a lotus in mud – if a client comes with a fancy requirement, then talk to the project manager first
* Requirement hurried-project buried
* Never criticize the stakeholder. Always appreciate the stakeholder even for small efforts

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

**Answer –**

A Package is a grouping of related elements (like classes or other packages) for organization and to provide a namespace, while a subsystem is a larger self contained grouping of packages that represents a part of a larger system.

**Packages** –

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

Packages can be nested within other packages.

These are used as containers to organize elements.

It is very useful to represent system architecture.

* Purpose – Packages are used to organize and structure model elements, making it easier to manage and understand complex systems.
* Function – They act as containers for related classes, interfaces and other packages, or provide a name space to avoid naming conflicts.
* Analogy – Think of packages as folders in a file system, where you organize files (classes) into logical groups.
* UML – In UML (Unified Modelling Language), packages are a key element for modelling software architecture, allowing you to represent the structure and dependencies between different parts of a system.

Example – In a banking system, you might have packages for “Accounts,” “ Transactions," and “Security”.

**Subsystems** – 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.

* Purpose – Subsystems represent larger, functional units within a system, encapsulating related packages and their interactions.
* Function – They are used to break down a complex system into manageable parts, promoting modularity and useability.
* Analogy – Imagine a car: the engine, the wheels and the steering system are all subsystems of the entire car.
* UML – In UML, Subsystems are represented as packages with the stereotype “<<subsystem>>”.

Example – In an e-commerce system, you might have subsystems for “Order Management”, “Payment Processing,” and “ Inventory Management”.

1. **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 – firstName, lastNaame

In BA, camel-casing is used in requirements documentation.

In requirement documentation, BA often use camel-casing to name the entitieslike use case, features, user stories like validateCustomerDetails, calculateInterestRate, etc.

Business rules, which should be satisfied by the system use camel-casing.

While documenting business process or work flows, 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.

1. **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 grated 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 have the authority to make changes in certain areas of application where they have the access.

1. **What is Data Mapping**

**Answer –**

The database contains multiple tables in it.

There may come 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 migration 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 convert 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)

1. **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.

**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.