

Chapter 3(System Design)

3.1 → Overall System Design

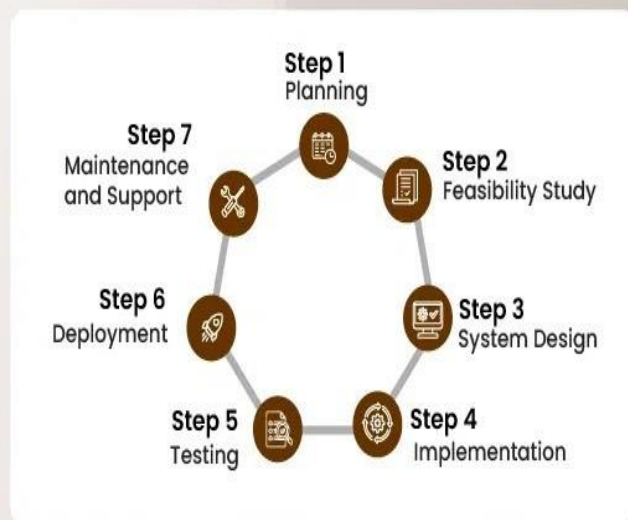
3.2 → Data Dictionary

3.3 → Input/output Design

3.1 System Design

- **Systems Design** is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements.
- It involves translating user requirements into a detailed blueprint that guides the implementation phase.
- The goal is to create a well-organized and efficient structure that meets the intended purpose while considering factors like Scalability maintainability, and performance.

What is System Design?



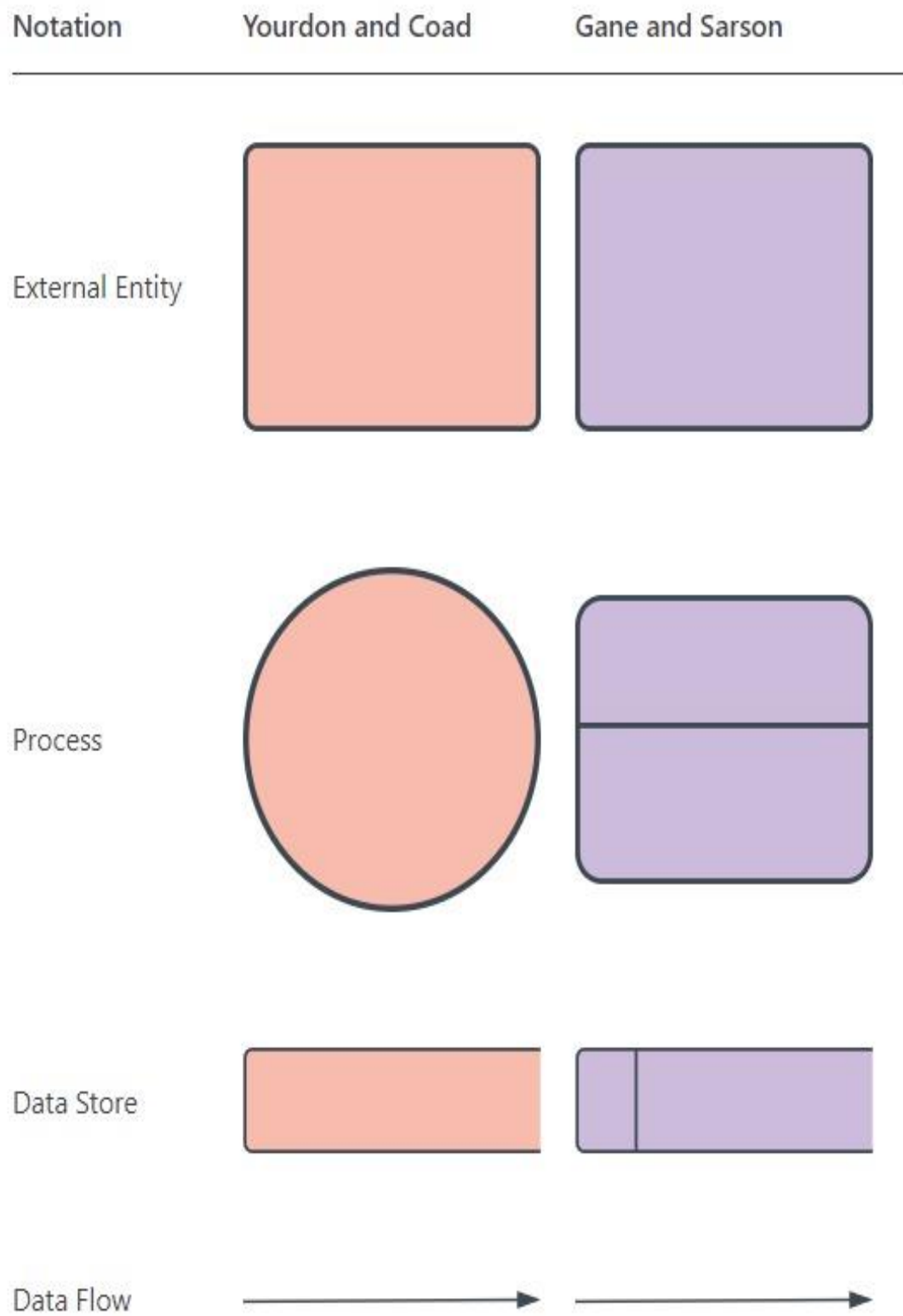
Why Learn System Design?

- In any development process, be it Software or any other tech, the most important stage is Design.

- Systems Design not only is a vital step in the development of the system but also provides the backbone to handle exceptional scenarios because it represents the business logic of software.

❖ **Data Flow Diagram:**

- Data Flow Diagram (DFD) represents the flow of data within information systems.
- Data Flow Diagrams (DFD) provide a graphical representation of the data flow of a system that can be understood by both technical and non-technical users.
- The models enable software engineers, customers, and users to work together effectively during the analysis and specification of requirements.
- The Data Flow Diagram (DFD) belongs to structured-analysis modeling tools.
- Data Flow diagrams are very popular because they help us to visualize the major steps and data involved in software-system processes.
- A Data flow diagram (DFD) maps out the flow of information for any process or system.
- It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.
- The two main types of notation used for data flow diagrams are Yourdon-cord and gane-Sarson.



External Entity:

- An external entity is a source or destination of a data flow. Only those entities which originate or receive data are represented on a data flow diagram. The symbol used is a rectangular box.

Process:

- A process shows a transformation or manipulation of data flow within the system. The symbol used is an oval shape.

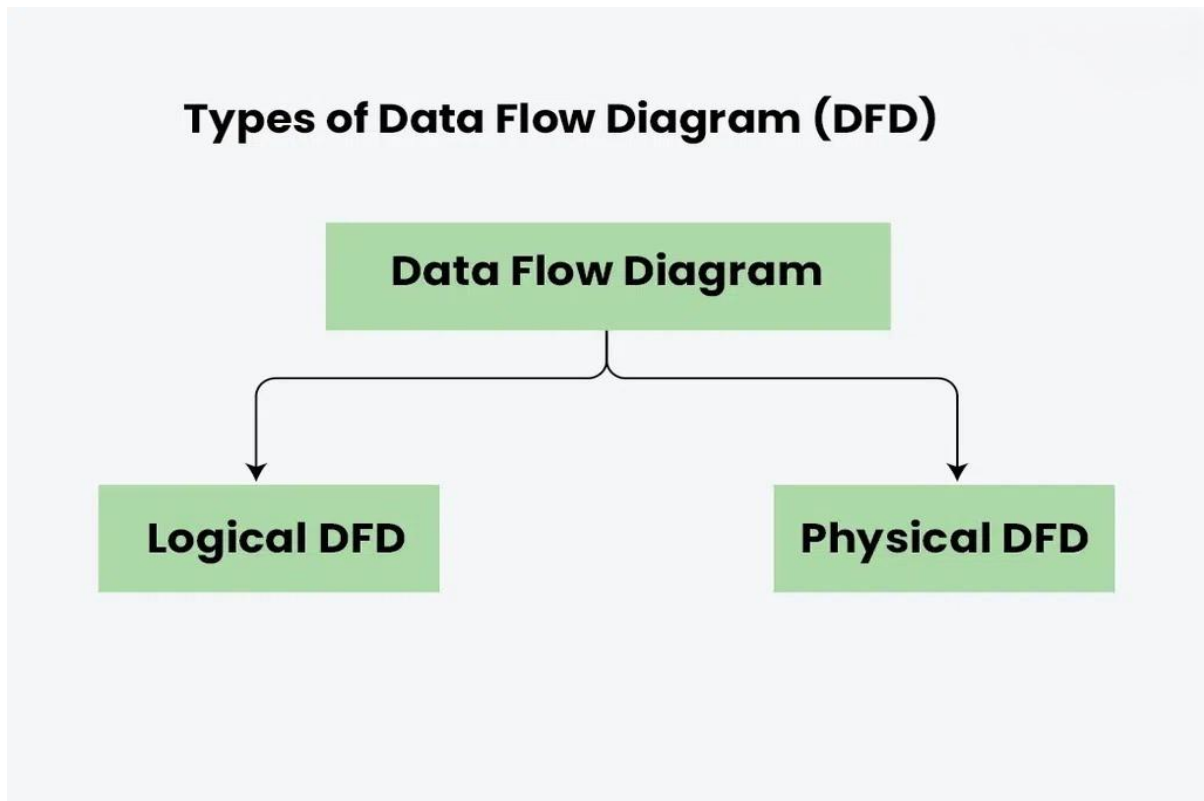
Data Store:

- A data store does not generate any operations but simply holds data for later access. Data stores could consist of files held long term or a batch of documents stored briefly while they wait to be processed. Input flows to a data store include information or operations that change the stored data. Output flows would be data retrieved from the store.

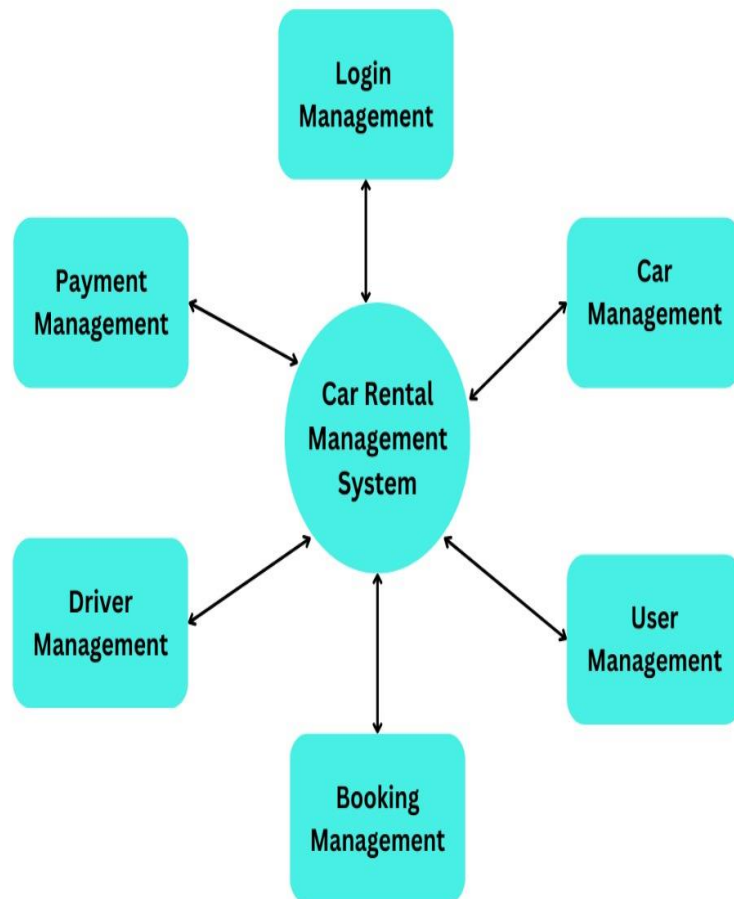
Data Flow:

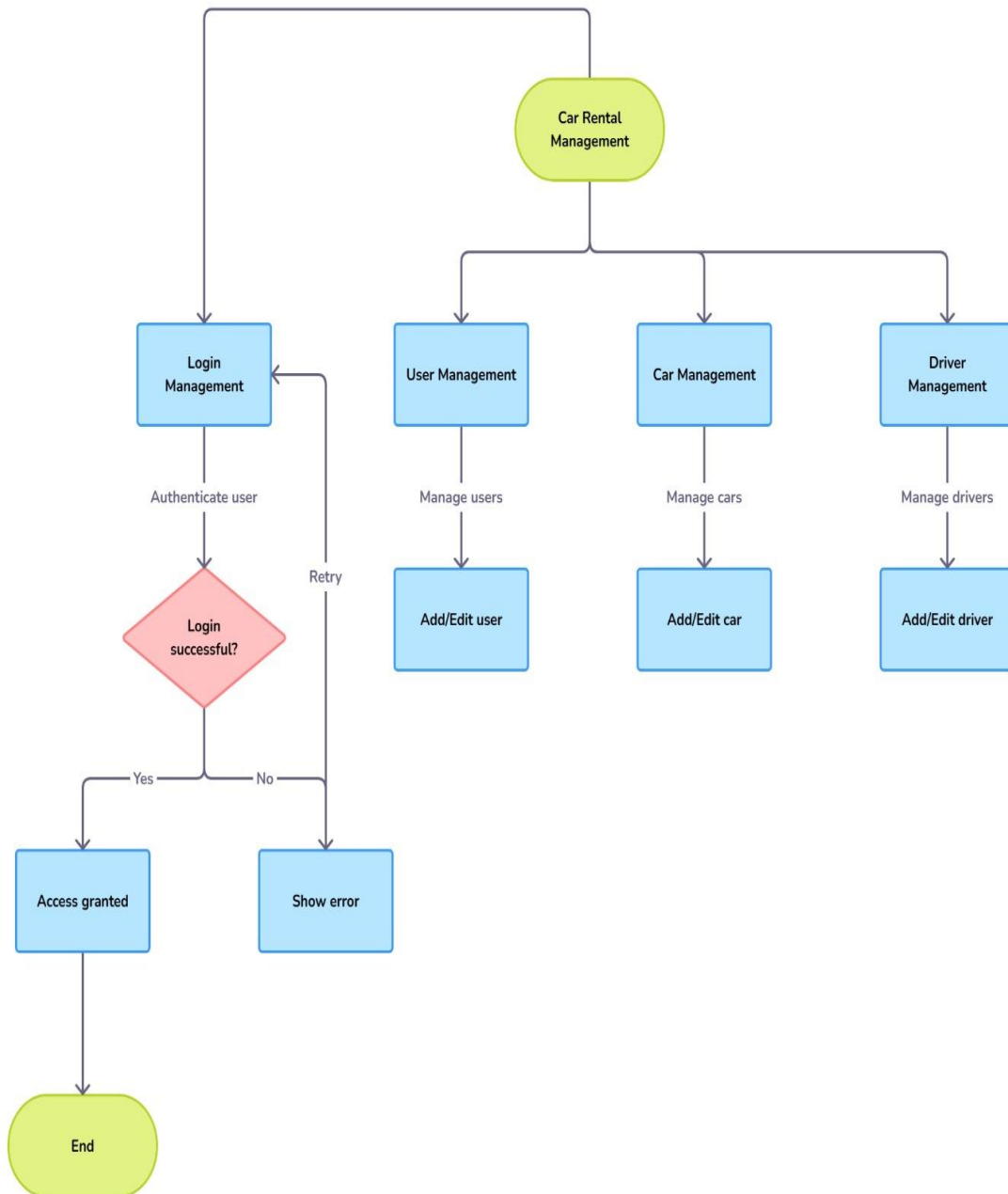
- Movement of data between external entities, processes and data stores is represented with an arrow symbol, which indicates the direction of flow. This data could be electronic, written or verbal. Input and output data flows are labeled based on the type of data or its associated process or data store, and this name is written alongside the arrow.

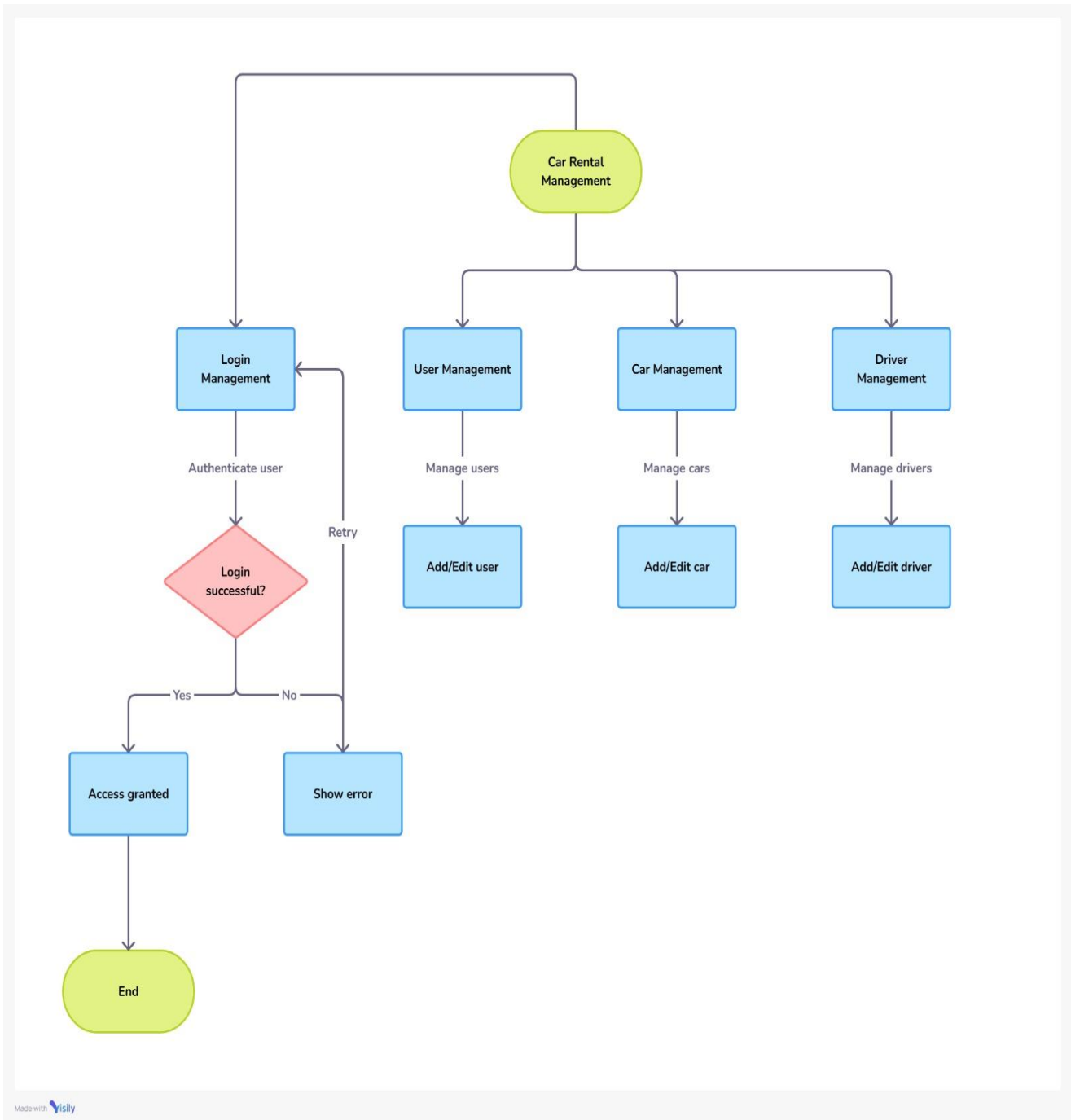
❖ DFD Types:



1. **Logical DFD:** Logical data flow diagram mainly focuses on the system process. We focus on the high-level processes and data flow without delving into the specific implementation details. Logical DFD is used in various organizations for the smooth running of system. Like in a Banking software system, it is used to describe how data is moved from one entity to another.
2. **Physical DFD:** Physical data flow diagram shows how the data flow is actually implemented in the system. In the Physical Data Flow Diagram (DFD), we include additional details such as data storage, data transmission, and specific technology or system components. Physical DFD is more specific and close to implementation.







❖ UML Diagram:

- Unified Modelling Language (UML) is a general-purpose modelling language. The main aim of UML is to define a standard way to visualize the way a system has been designed.
- It is quite similar to blueprints used in other fields of engineering. UML is not a programming language, it is rather a visual language.
- Unified Modelling Language (UML) is a standardized visual modeling language that is a versatile, flexible, and user-friendly method for visualizing a system's design.
- Software system artifacts can be specified, visualized, built, and documented with the use of UML.
- We use UML diagrams to show the behaviour and structure of a system. UML helps software engineers, businessmen, and system architects with modelling, design, and analysis.
- UML (Unified Modelling Language) was developed between 1994 and 1996 by Grady Booch, Ivar Jacobson, and James Rumbaugh while working at Rational Software.

Unified Modeling Language (UML) Diagrams



- It was officially adopted as a standard by the Object Management Group (OMG) in 1997.


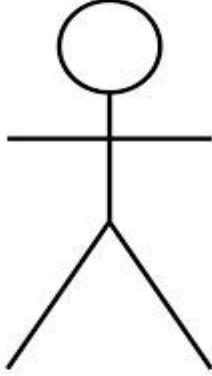




➤ There are various kinds of methods in software design. They are follows:

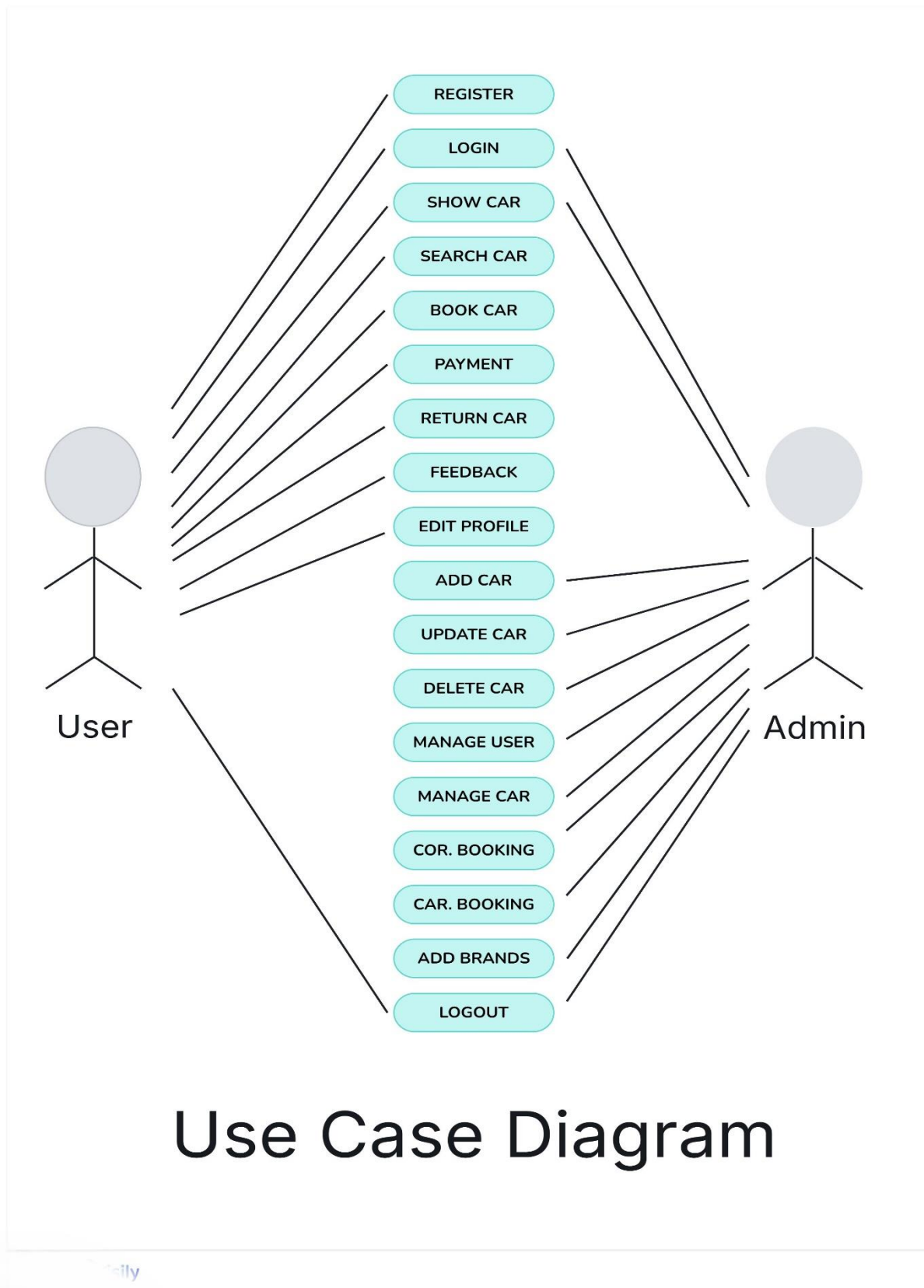
- Use Case Diagram
- Activity Diagram
- Class diagram

Use Case Diagram:

- A Use Case Diagram in **Unified Modelling Language (UML)** 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
- When you need to gather and clarify user requirements, use case diagrams help visualize how different users interact with the system.
- If you're working with diverse groups, including non-technical stakeholders, these diagrams provide a clear and simple way to convey system functionality.
- During the system design phase, use case diagrams help outline user interactions and plan features, ensuring that the design aligns with user needs.
- When defining what is included in the system versus what is external, use case diagrams help clarify these boundaries.

➤ Following element are used in Use Case Diagram:

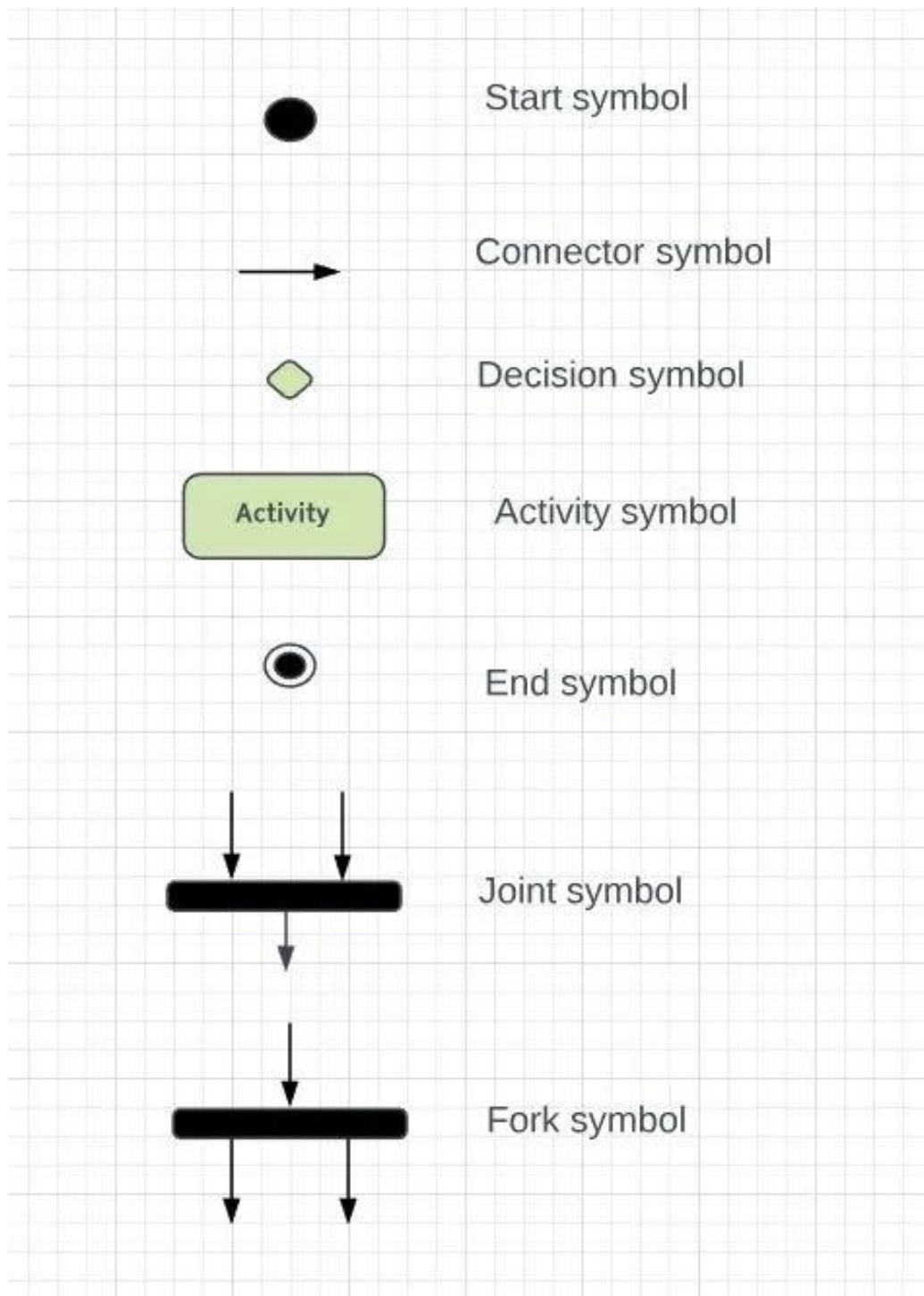
Use Case	
Actor	
Communication	
Include	
Exclude	
System boundary	

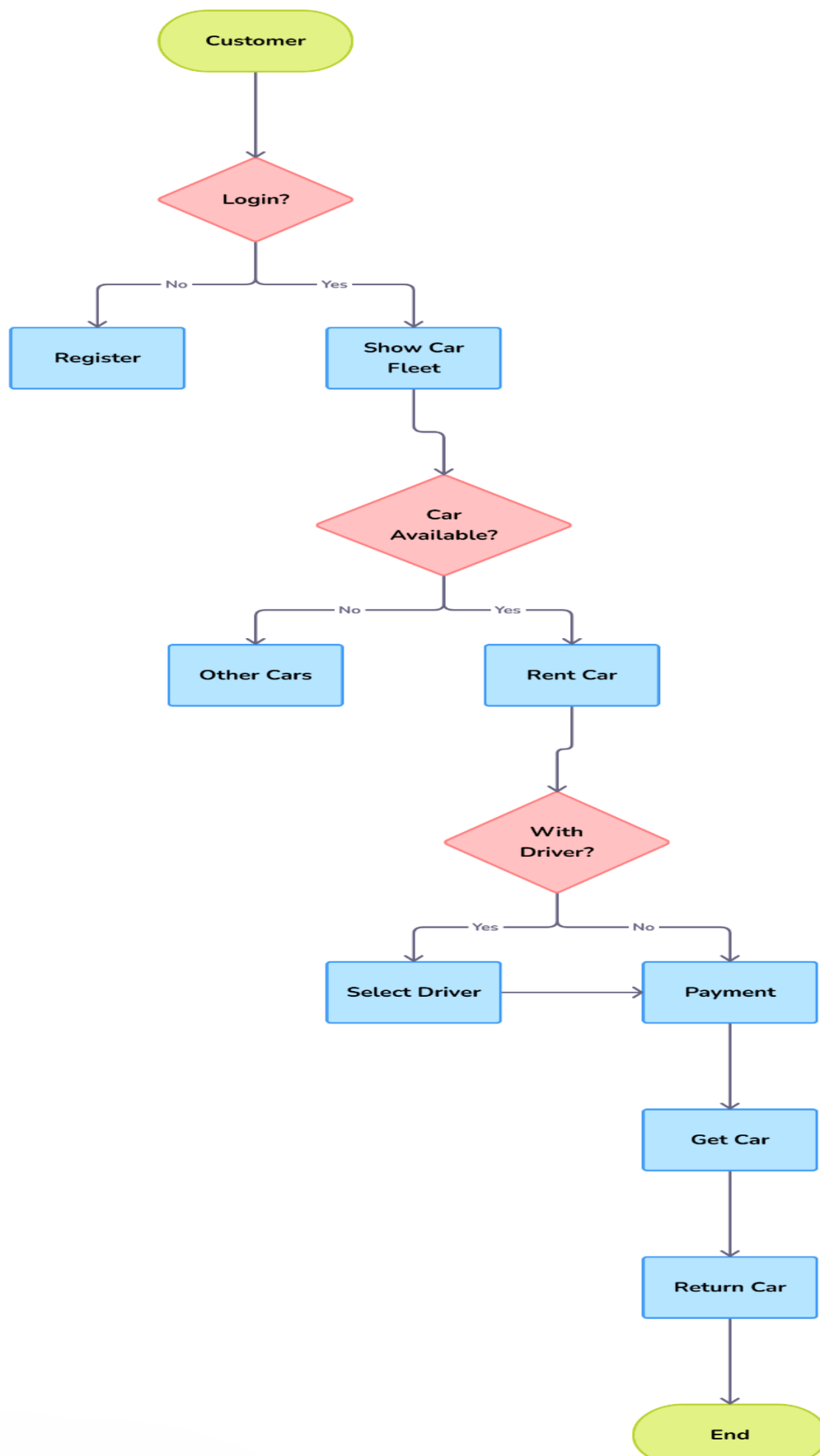


Activity Diagram:

- Activity diagrams show the steps involved in how a system works, helping us understand the flow of control.
- They display the order in which activities happen and whether they occur one after the other (sequential) or at the same time (concurrent).
- These diagrams help explain what triggers certain actions or events in a system.
- An activity diagram starts from an initial point and ends at a final point, showing different decision paths along the way.
- They are often used in business and process modelling to show how a system behaves over time.
- Activity diagrams are useful in several scenarios, especially when you need to visually represent the flow of processes or behaviours in a system.
- **Activity diagram** is essentially an advanced version of flow chart that modelling the flow from one activity to another activity.
- Activity diagram is a special kind of state chart diagram that show the flow of system.

- Following element are used in Activity Diagram:



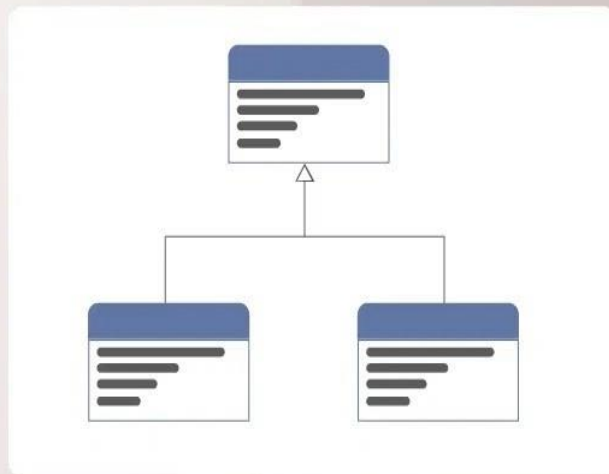


Class Diagram:

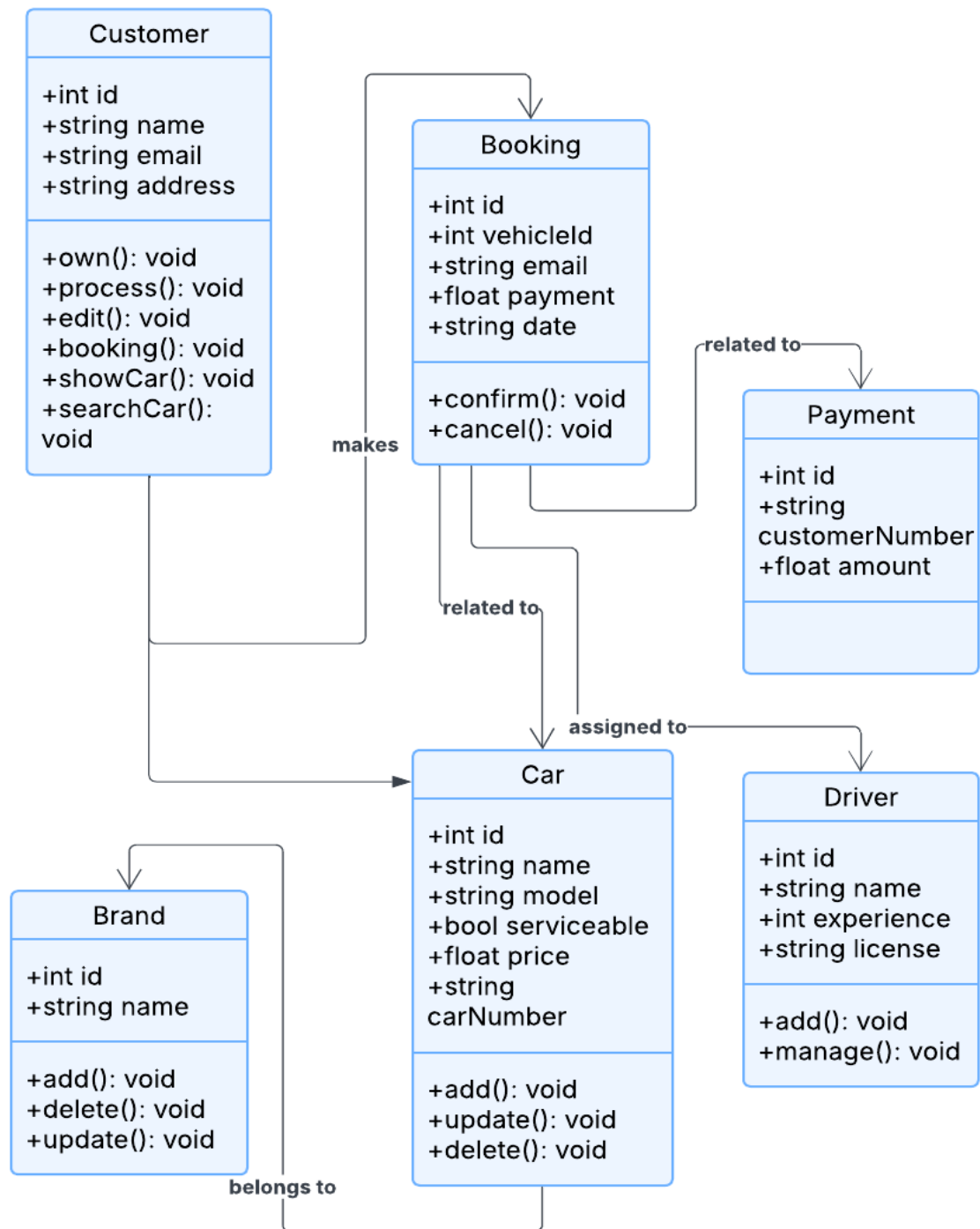
- A UML class diagram is a visual tool that represents the structure of a system by showing its classes, attributes, methods, and the relationships between them.
- It helps everyone involved in a project like developers and designers understand how the system is organized and how its components interact.
- Class diagrams are a type of UML diagram used in software engineering to visually represent the structure and relationships of classes within a system i.e. used to construct and visualize object-oriented systems.

Class Diagram

Unified Modeling Language (UML)



- Class diagrams provide a high-level overview of a system's design, helping to communicate and document the structure of the software.
- They are a fundamental tool in object-oriented design and play a crucial role in the software development lifecycle.



Class Diagram

3.2 Data Dictionary

- A Data Dictionary comprises two words i.e. Data which simply means information being collected through some sources and Dictionary means where this information is available.
- A Data Dictionary can be defined as a collection of information on all data elements or contents of databases such as data types, and text descriptions of the system.
- It makes it easier for users and analysts to use data as well as understand and have common knowledge about inputs, outputs, components of a database, and intermediate calculations.
- Physical information about the tables such as where they are stored and how.
- Table constraints such as primary key attributes, foreign key information etc.