Theory

Part A: Data Flow Diagram (DFD)

A Data Flow Diagram (DFD) represents how data flows through a system. It focuses on what data enters the system, how it is processed, and where it is stored or sent, without showing implementation details.

DFDs mainly consist of:

- **External Entities** – sources or destinations of data (e.g., User)

- **Processes** – operations that transform data

- **Data Stores** – places where data is stored

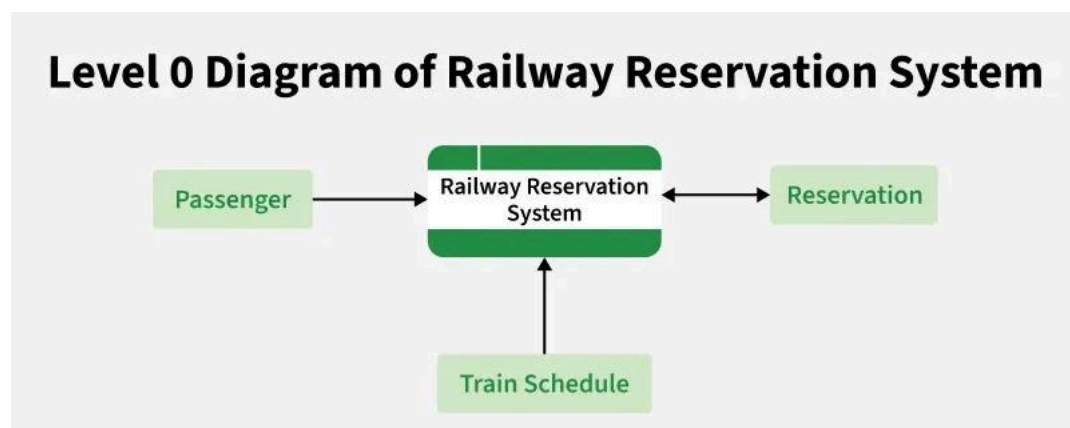- **Data Flows** – movement of data between components

DFDs are hierarchical:

- **Level 0 DFD (Context Diagram):** Shows the entire system as a single process.

- **Level 1 DFD:** Breaks the main process into sub-processes for more detail.

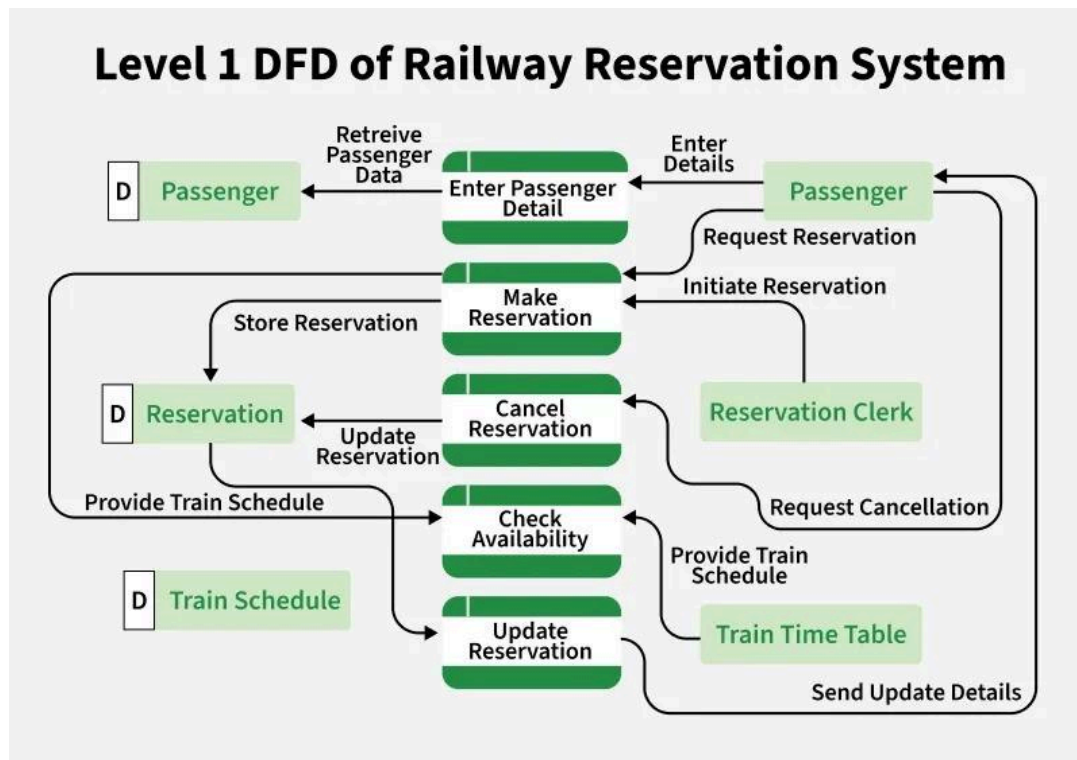### Example: Railway Reservation System

Level 0 DFD: Shows the reservation system as one process interacting with :

- Passenger (external entity)
- Payment system
- Database (data store)

Level 1 DFD: Breaks the system into subprocesses such as:

● Search Train
● Book Ticket
● Cancel Ticket
● Process Payment

Part B: Structural Aspects of Software (UML Diagrams)

UML (Unified Modeling Language) diagram is used to model the structure and design of software systems. Unlike DFDs, UML focuses on how the system is organized internally.
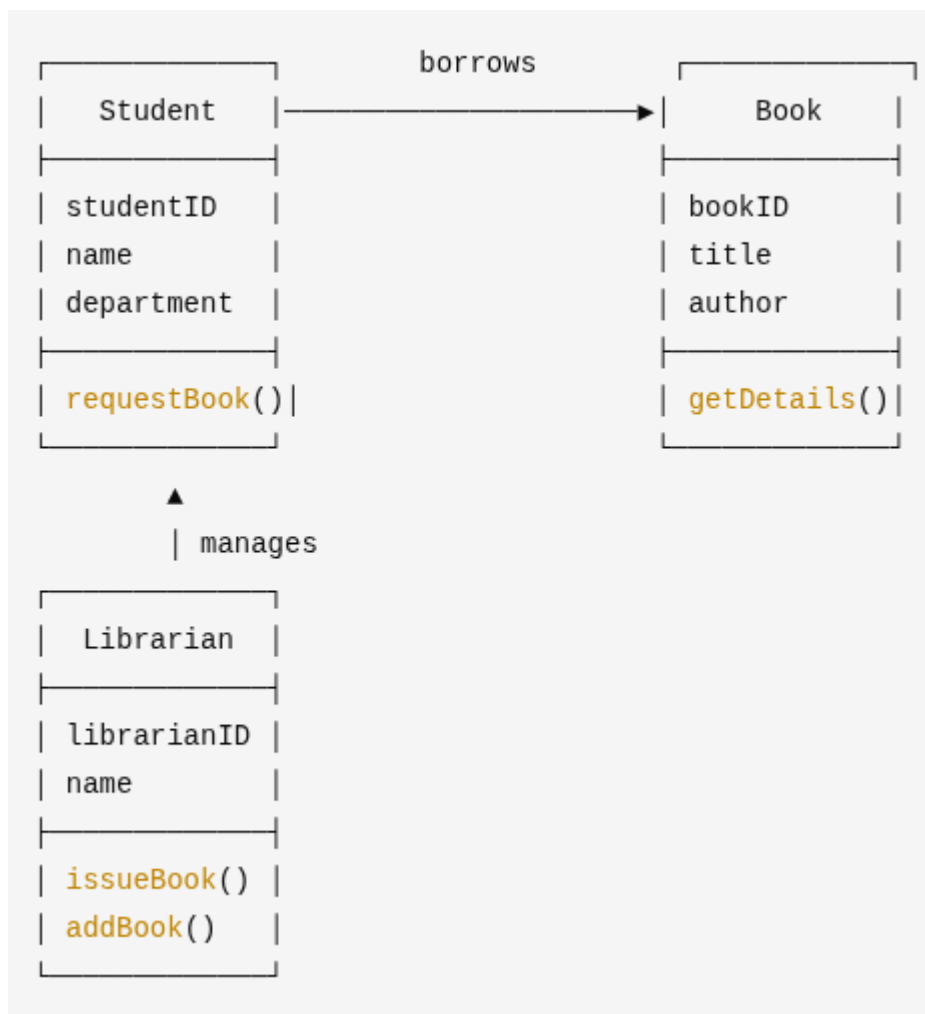
Structural modeling mainly uses:

● **Class Diagram** – shows classes, attributes, methods, and relationships

● **Object Diagram** – shows instances of classes at a specific time

UML helps in understanding:

- System architecture

- Relationships between components

- Software design before implementation

Example: Library Management System (Class Diagram)


```
                          borrows
 ┌─────────────┐                      ┌─────────────┐
 │   Student   │────────────────────►│    Book     │
 ├─────────────┤                      ├─────────────┤
 │ studentID   │                      │ bookID      │
 │ name        │                      │ title       │
 │ department  │                      │ author      │
 ├─────────────┤                      ├─────────────┤
 │ requestBook()│                     │ getDetails()│
 └─────────────┘                      └─────────────┘

         ▲
         │ manages
 ┌─────────────┐
 │  Librarian  │
 ├─────────────┤
 │ librarianID │
 │ name        │
 ├─────────────┤
 │ issueBook() │
 │ addBook()   │
 └─────────────┘
```

This diagram shows static structure and relationships among system components.