**SDA ASSIGNMENT 1**

**Laiba Fatima 22k-5195**

**Part 1: UML Importance**

**The importance of UML in software engineering.**

**Explanation of UML and its purpose.**

UML or Unified Modeling Language is a visual language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. It is not a programming language.

It is used to sketch out our systems, used to communicate information and help in understanding the system/software better. We use UML to describe the problem domain, describe the activities that occur, and eventually describe the software classes

**Benefits of using UML in software development.**

-UML Diagrams make understanding of software easier by offering visual representations, it is also easier to analyze and modify.

-It helps in communication among software developers

-They serve as documentation artefacts, which is very important in software engineering. They show system requirements and the design decisions.

-It supports Analysis (requirement gathering) and supports Design (class diagram, sequence diagram etc.)

-It enhances collaboration for a more efficient design process

**How UML helps in communication among stakeholders.**

It serves as a common language between software developers(stakeholders) to collaborate/ communicate their ideas. It can be understood by both technical and non-technical stakeholders because visual representations are much easier to understand than technical jargon of words, so everyone has a shared understanding of the system, and this reduces miscommunication

**Different types of UML diagrams and their roles in the software development lifecycle.**

Two types: Behavioral and Structural

Behavioral Diagrams:

* Use Case Diagram: contains system functionalities, capturing actors, use cases, and their relationships.
* Activity Diagram: shows the flow of activities within a system containing decisions and actions
* Sequence Diagram: shows interaction between objects, highlighting the messages exchanged in order.
* Communication Diagram: shows interactions between objects in terms of sequenced messages.
* State Machine Diagram: represents the behavior of individual objects showing states and transitions.
* Timing Diagram: shows behavior of objects highlighting the timing constraints and sequences of events.
* Interaction Overview Diagram: provides overview of flow of control between interactions.

Structural Diagrams:

* Class Diagram: shows structure of a system, showing classes, attributes, methods, and their relationships.
* Composite Structure Diagram: depicts internal structure of a class, showing how parts are interlinked to form a larger structure.
* Deployment Diagram: shows deployment of software components across hardware nodes or environments.
* Package Diagram: shows dependencies between packages or modules
* Profile Diagram: defines extensions and constraints to UML elements.
* Object Diagram: shows instances of classes and their relationships at a particular point in time.
* Component Diagram: shows components of a system and their dependencies.

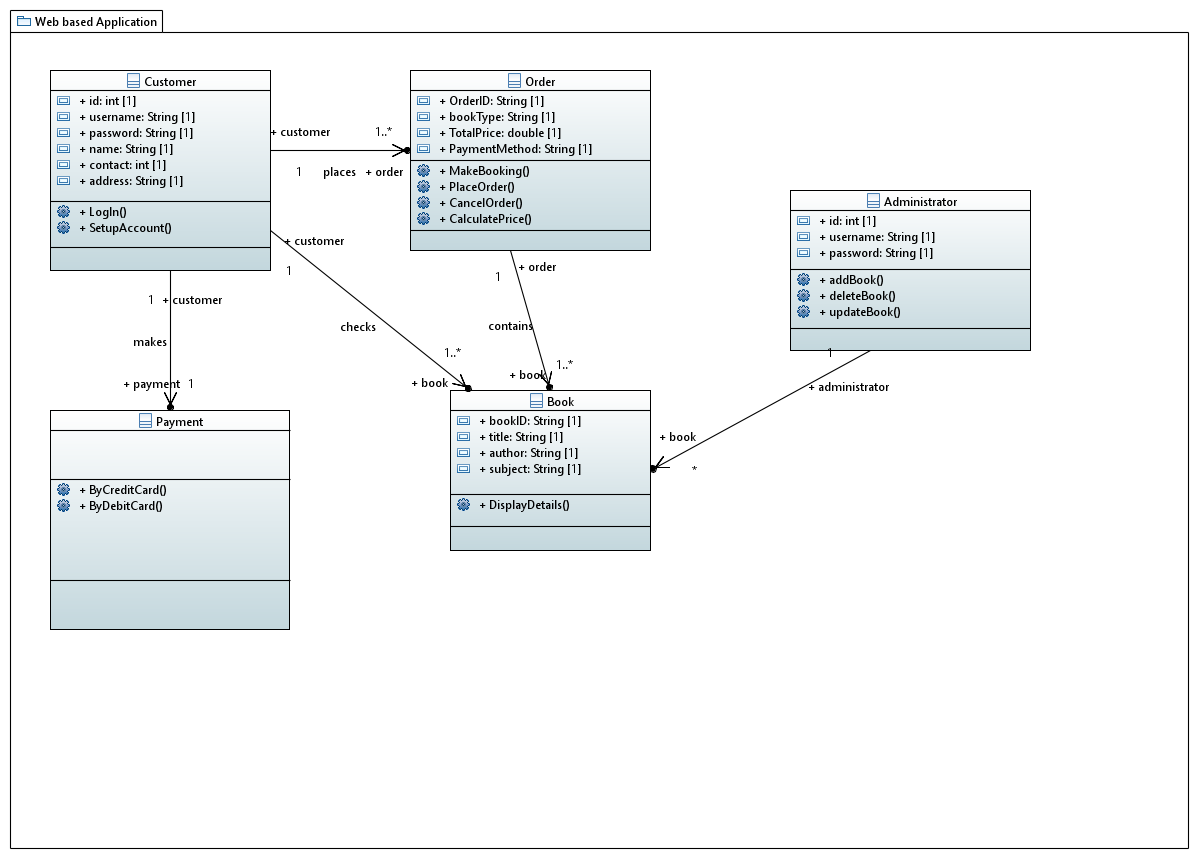
**Importance of UML in achieving better design and maintainability of software systems.**

UML provides visualization; making system easier to understand, communication; by providing common language, analysis; understanding system requirements, design; understanding system’s structures and interactions. It also encourages abstraction by hiding implementation details through class diagrams; this simplifies system complexity. UML acts as a documentation artefact and keeps stakeholders up to date.

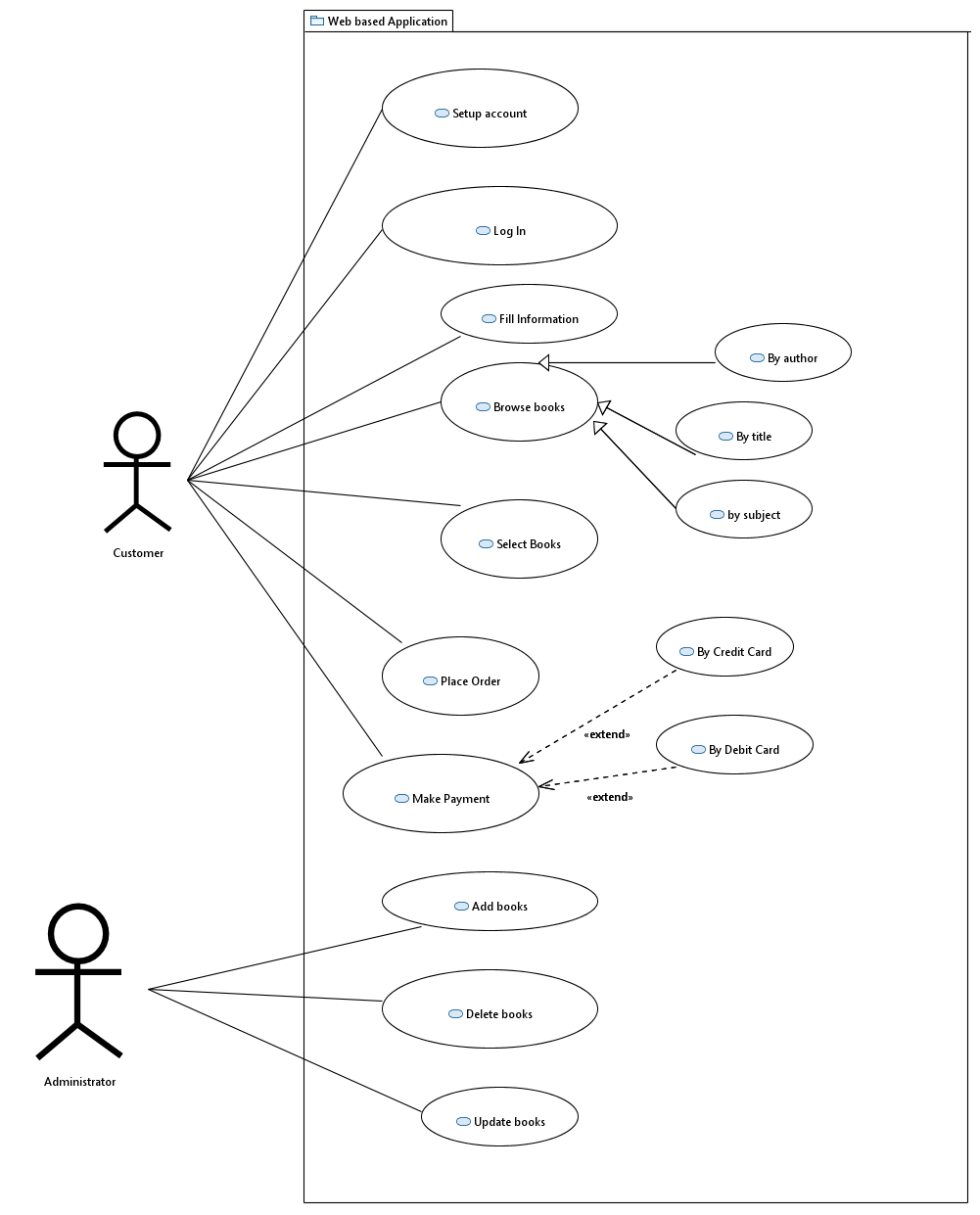
UML is a blueprint for understanding system’s structure, behavior and interactions. By incorporating UML throughout thesoftware development process, teams can create robust, maintainable, and adaptable software systems that meet stakeholders' needs effectively.

**Part 2: Class diagram and use-case model**

**Q1: Class Diagram**



**Q2: Use-Case Diagram**



**Q3: Modified Use-Case Diagram**

Another requirement is the integration of a shipping and delivery management system. This feature allows users to select their preferred shipping method, enter shipping details, and track the delivery status of their orders. The addition of user receiving notifications that their order has been shipped.

The Class Diagram would include another class of Shipping/Delivery with the appropriate properties of shipment details. And the use case would include the customer to select shipment method (standard or express) and the customer to track their order, and receive timely notifications