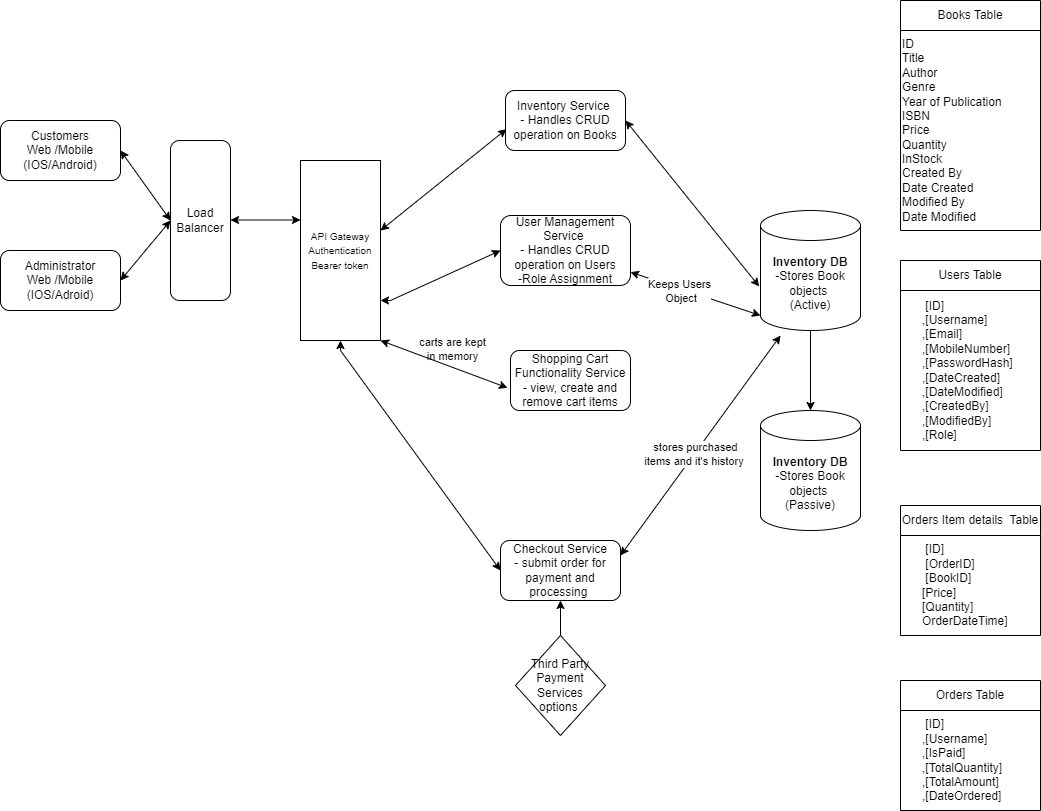
**High Level Design Document**



# HIGH LEVEL DESIGN

APIs for Read/Write scenarios

* User APIs: createUser, authenticateUser, getUserDetails, updateUserDetails
* Inventory APIs: addInventory, update Inventory, search Inventory, get Inventory
* Order APIs: submitOrder, viewPurchaseHistory
* Cart APIs: viewCart, create/updateCart, removeItemsfromCart

Database schema

* Users, Books, Orders, OrderItems each with appropriate fields primary and foreign key as required.

High level design for Read/Write scenario

* Micro services architecture, with services for Users, Books, Orders, and Payments.
* **Web Server**: This component serves as the entry point for all incoming HTTP requests. It's responsible for routing requests to the appropriate controllers or endpoints.
* **Database Cluster** (Primary and Secondary Nodes): The database cluster consists of multiple nodes distributed across different machines for high availability and fault tolerance. The primary node handles read and write operations, while the secondary nodes replicate data asynchronously from the primary node for backup and failover purposes.
* **Models** (DTOs, Entities): These are data transfer objects (DTOs) and entity classes that represent the structure of data in the application. They may include classes for mapping database entities and DTOs for transferring data between layers.
* **ORM Framework**: Object-Relational Mapping (ORM) frameworks like Entity Framework Core can be used to facilitate interaction with the database. They provide an abstraction layer that allows developers to work with database entities using object-oriented principles.
* **Data Access Layer**: This layer encapsulates the logic for accessing and manipulating data in the database. It typically includes repositories or data access objects (DAOs) that abstract away the details of database operations.
* **Services**: Encapsulate business logic and interact with the data access layer to perform operations on the database. Contains the core logic and rules of the application. It governs how data is processed and ensures consistency and integrity.
* **Web API Controllers**: Responsible for handling incoming HTTP requests, processing them, and returning appropriate responses. They serve as the interface between the eBook store applications and the backend services.
* **API Endpoints**: Expose specific functionalities of the back end application to external ebook store (client) through HTTP endpoints. Clients can interact with these endpoints to perform CRUD (Create, Read, Update, Delete) operations on resources such as books, users and orders in the ebook store application.