Description of the Use Case:

Traditional retail models are inefficient due to geographical boundaries, limited hours, and high costs. Customers face inconvenience, limited product variety, and lack of comparison shopping. **InstaCommerce** offers a virtual marketplace where customers shop from home. Merchants extend their reach beyond physical stores, overcoming limitations of location and time.

Users/Stakeholders: Customers, Merchants, Administrators and Developers

Key Functionalities:

Core Features:

1. **Product Catalog:** A vast array of products available for purchase.
2. **One-Click Purchase:** Streamlined checkout process for quick purchases.
3. **Order Tracking:** Real-time tracking of orders from placement to delivery.
4. **Customer Reviews:** User-generated reviews and ratings for products.
5. **Seller Marketplace:** Platform for third-party sellers to list and sell their products

User Interaction:

Users will interact with our web application, browsing products by categories and using various filters to find desired items efficiently. They can add products to their cart and proceed to a secure checkout process with multiple payment options. Sellers can manage their product listings and inventory through a dedicated dashboard. Integrated delivery services ensure smooth order processing and tracking. The responsive design ensures optimal user experience across devices.

Technical Highlights: **React JS + MongoDB + NodeJS + Express JS + MinIO**

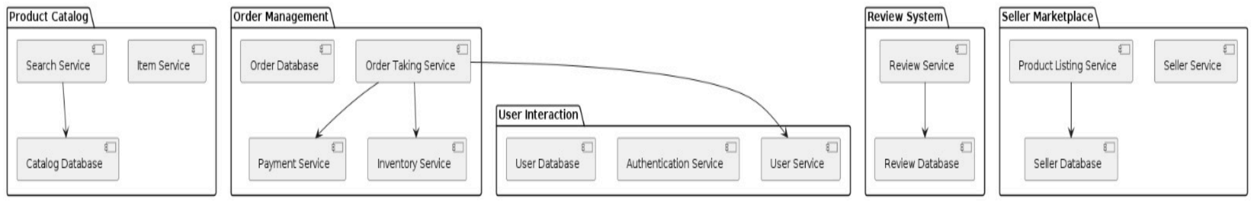
Design Patterns:

1. **Model-View-Controller (MVC)** separates data, user interface, and application logic for improved maintainability and scalability.
2. **Repository Pattern** abstracts data access, enhancing code maintainability and testability.
3. **Singleton Pattern** ensures consistency and efficiency by using single instances of specific components.

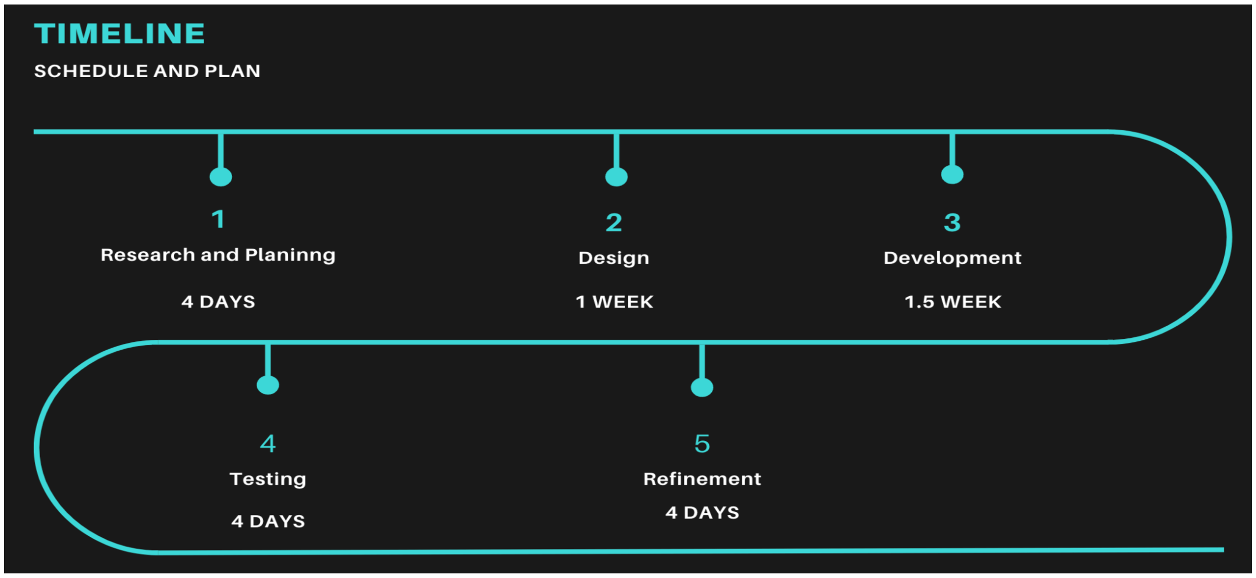
4. **Observer Pattern:** Implementing event-driven architecture for real-time updates and notifications, enhancing responsiveness and user experience.

Design Decisions:

1. **Database Choice:** Assess requirements and scalability for MongoDB (NoSQL) or SQL databases; MongoDB offers flexibility, while SQL provides strong consistency.
2. **Authentication Mechanism:** Choose between JWT or session-based authentication based on security and user experience.
3. **Error Handling:** Define robust error handling with informative messages and logging for debugging.
4. **Security Measures:** Employ input validation, encryption, and protection against SQL injection
5. **Testing Strategy:** Utilize Jest for unit tests ensuring reliability and correctness.



Expected Time to Build a Prototype:



Domain: E-Commerce