Summary of Documents

1. Architectural Patterns

- MVC: Data, user interface, and control logic are separated. Allows multiple views and interactions.
- Layered Architecture: Organizes the system into layers to separate functionalities.
- Repository: Uses a central data store to share data among subsystems.
- Client-Server: Servers provide services, and clients request services.
- Pipe and Filter: Composed of filters that process data and pipes that transfer data.
- Broker: Acts as a mediator in distributed systems to coordinate communication.

2. Design Concepts

- Modularity: Dividing the system into small, independent modules.
- Abstraction: Extracting important information and hiding details.
- Encapsulation: Bundling data and functions into a single unit and hiding it from outside.
- Implementation Stage: Minimizing dependencies between modules and defining clear interfaces.
- Testing: Ensuring the system works as specified in the design phase.

3. Software Evolution

- Evolution: Continuously improving software to meet user needs and technological changes.
- Legacy Systems: Maintaining and updating old systems to meet modern requirements.
- Evolution Processes: Activities like adding new features, improving performance, and fixing bugs.

4. Testing Types

- Unit Testing: Verifying the functionality of individual modules or components.
- Integration Testing: Testing the interaction between combined modules.
- System Testing: Ensuring the complete system meets the requirements.
- Release Testing: Verifying that a specific release meets the required performance and reliability.
- User Testing: Actual users test the system to discover issues.