Here's a detailed outline for a Software Design Document (SDD) for a hospital appointment system:

1. *Introduction*
   - Purpose of the document.
   - Scope of the hospital appointment system.
   - Objectives and benefits.

2. *System Overview*
   - Brief description of the hospital appointment system.
   - Key features and functionality.

3. *Stakeholders*
   - Identification of stakeholders (e.g., patients,receptionist , hospital administrators).
   - Description of their roles and responsibilities.

4. *System Architecture*
   - High-level architectural overview.
   - Components of the system and their interactions.
   - Deployment architecture.

5. *Functional Requirements*
   - Use cases or user stories detailing system functionality.
   - Description of each use case, including preconditions, post-conditions, and main flow.
   - Supplementary requirements (e.g., non-functional requirements).

6. *Data Design*
   - Entity-Relationship Diagram (ERD) showing the data model.
   - Description of each entity and its attributes.
   - Database schema if applicable.

7. *User Interface Design*
   - Mockups or wireframes of user interfaces.
   - Description of user interactions with the system.

8. *System Modules*
   - Detailed description of each system module.
   - Class diagrams, sequence diagrams, or activity diagrams illustrating module behavior.
   - Interfaces between modules.

9. *Security Design*
   - Description of security measures (e.g., authentication, authorization, data encryption).
   - Threat modeling and mitigation strategies.

10. *Testing Strategy*
    - Overview of the testing approach (e.g., unit testing, integration testing, system testing).
    - Test scenarios and test cases for each functional requirement.
    - Tools and techniques used for testing.

11. *Deployment Plan*
    - Description of deployment environment (e.g., hardware, software).
    - Deployment procedures and scripts.
    - Rollout strategy.

12. *Maintenance and Support*
    - Procedures for system maintenance and updates.
    - Support procedures for handling user issues and system failures.

13. *Dependencies*

- External dependencies (e.g., third-party libraries, APIs).

- Internal dependencies between system components.

14. *Assumptions and Constraints*

- Assumptions made during system design and development.

- Constraints that may impact system implementation.

15. *Glossary*

- Definitions of terms and acronyms used throughout the document.

16. *References*

- Any external references or documents used in the development process.

17. *Appendices*

- Additional information such as sample data, supplementary diagrams, or technical specifications.

Non functional

. 1-Performance: Specifies how the system should perform in terms of response time, throughput, and resource utilization under certain conditions. For example, the system should be able to handle a certain number of concurrent users without significant degradation in performance.

2-Scalability: Describes the system's ability to handle increasing amounts of work by adding resources. It includes horizontal scalability (adding more machines) and vertical scalability (upgrading existing machines).

3-Reliability: Defines the system's ability to function correctly and consistently under different conditions for a specified period. This includes measures such as fault tolerance, availability, and mean time between failures (MTBF).

4-Availability: Specifies the percentage of time that the system should be operational. It often includes requirements for redundancy, failover mechanisms, and maintenance downtime.

5-Security: Addresses the system's ability to protect data, resources, and functionality from unauthorized access, modification, or destruction. This includes authentication, authorization, encryption, and auditability.

6-Maintainability: Describes how easy it is to maintain and evolve the system over time. This includes factors such as modularity, documentation, coding standards, and ease of debugging.

7-Usability: Refers to how user-friendly and intuitive the system is for its intended users. It includes factors such as user interface design, accessibility, and error handling.

8-Compatibility: Specifies how well the system interoperates with other systems, platforms, or software versions. This includes compatibility with different browsers, operating systems, and third-party integrations.

9-Portability: Describes the ease with which the system can be transferred from one environment to another. This includes considerations for hardware, software, and configuration dependencies.

Regulatory Compliance: Ensures that the system complies with relevant laws, regulations, and industry standards. This may include requirements for data privacy, data retention, and audit trails.