What is an SRS ?

- SRS is the official statement of what the system developers should implement.
- SRS is a complete description of the behavior of the system to be developed.
- SRS should include both a definition of user requirements and a specification of the system requirements.
- The SRS fully describes what the software will do and how it will be expected to perform.

What is the purpose of SRS

- The SRS precisely defines the software product that will be built.
- SRS used to know all the requirements for the software development and thus that will help in designing the software.
- It provides feedback to the customer.

Characterstics of an SRS:

1. Correctness: The SRS accurately represents the requirements of the software system.
2. Completeness: The SRS includes all necessary functional and non-functional requirements.
3. Consistency: The SRS avoids conflicts or contradictions within the document.
4. Unambiguous: The SRS is clear and free from ambiguity or vagueness.
5. Verifiable: The requirements in the SRS are testable and measurable.
6. Modifiable: The SRS can accommodate changes and updates to requirements.
7. Traceable: The SRS establishes links between requirements and related artifacts.
8. Understandable: The SRS is written in language that is easily understood by all stakeholders.

9. Feasible: The requirements in the SRS are achievable within project constraints.
10. Stakeholder-focused: The SRS addresses the needs and expectations of stakeholders.

Components of an SRS:

1. Introduction: Provides an overview of the document, including its purpose, intended audience, and any references or related documents.
2. Scope: Defines the boundaries of the software system, specifying what functionalities and features are included and excluded.
3. Functional Requirements: Describes the specific functionalities and behavior of the software system, outlining the inputs, outputs, and actions expected from the system in response to different user interactions.
4. Non-functional Requirements: Specifies the qualities and characteristics of the software system, including performance, reliability, usability, security, and other constraints.
5. User Requirements: Outlines the needs and expectations of the users, describing their goals, tasks, and interactions with the software system.
6. System Requirements: Defines the technical requirements of the software system, including hardware, software, network dependencies, performance requirements, compatibility, and any external interfaces.
7. Use Cases: Provides detailed descriptions of typical user scenarios or interactions with the software system, including the sequence of actions and expected system behavior.
8. Data Requirements: Describes the data elements, structures, formats, and relationships that the software system will handle, including data inputs, outputs, storage, and processing requirements.
9. Assumptions and Constraints: States any assumptions made during the requirements gathering process and any constraints that may impact the design or implementation

of the software system.

10. Risks and Dependencies: Identifies any potential risks or dependencies associated with the development, deployment, or operation of the software system.
11. Glossary: Includes a list of key terms, acronyms, and definitions used within the document to ensure a common understanding among stakeholders.