SRS Template & Question Bank

Software Requirements Specification for [Project Name]

Version [number] - [Date]

Prepared by [Your Name/Your Team's Name]

Table of Contents

1. Introduction

1.1 Purpose

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Project Scope

1.5 References

2. Overall Description

2.1 Product Perspective

2.2 Product Features

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

3. System Features

3.1 System Feature 1 (and so on for each feature)

3.1.1 Description and Priority

3.1.2 Stimulus/Response Sequences

3.1.3 Functional Requirements

4. External Interface Requirements

4.1 User Interfaces

4.2 Hardware Interfaces

4.3 Software Interfaces

4.4 Communications Interfaces

5. Other Nonfunctional Requirements

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

5.5 Business Rules
6. Other Requirements
6.1 Database Requirements
6.2 Compliance Requirements
6.3 Licensing and Installation
Appendix A: Glossary
Appendix B: Analysis Models
Appendix C: To Be Determined List
Questions Bank for SRS

Software Requirements Specification for [Project Name]

Version [number] - [Date]

Prepared by [Your Name/Your Team's Name]

Table of Contents

1. Introduction
   1.1 Purpose
   1.2 Document Conventions
   1.3 Intended Audience and Reading Suggestions
   1.4 Project Scope
   1.5 References
2. Overall Description
   2.1 Product Perspective
   2.2 Product Features
   2.3 User Classes and Characteristics
   2.4 Operating Environment
   2.5 Design and Implementation Constraints
   2.6 User Documentation
   2.7 Assumptions and Dependencies

3. System Features
   3.1 System Feature 1
   3.1.1 Description and Priority
   3.1.2 Stimulus/Response Sequences
   3.1.3 Functional Requirements
   3.2 [Other System Features]
4. External Interface Requirements
   4.1 User Interfaces
   4.2 Hardware Interfaces
   4.3 Software Interfaces
   4.4 Communications Interfaces
5. Other Nonfunctional Requirements
   5.1 Performance Requirements
   5.2 Safety Requirements
   5.3 Security Requirements
   5.4 Software Quality Attributes
   5.5 Business Rules
6. Other Requirements
   6.1 Database Requirements
   6.2 Compliance Requirements
   6.3 Licensing and Installation

Appendix A: Glossary
Appendix B: Analysis Models
Appendix C: To Be Determined List

---

1. Introduction

1.1 Purpose

[Describe the purpose of this SRS and its intended audience.]

1.2 Document Conventions

[Describe any standards or typographical conventions used.]

1.3 Intended Audience and Reading Suggestions

[Identify the various audiences for the document and suggest how they should use it.]

1.4 Project Scope

[A brief description of the software application, including its functionality and matters it addresses or solves.]

1.5 References

[List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document.]

2. Overall Description

2.1 Product Perspective

[Describe the context and origin of the product being specified in this SRS. Include a system block diagram if appropriate.]

2.2 Product Features

[Summarize the major features of the product.]

2.3 User Classes and Characteristics

[Describe the different types of users who will use this product.]

2.4 Operating Environment

[Describe the environment in which the software will operate: hardware, operating system, other required software, network connectivity, etc.]

2.5 Design and Implementation Constraints

[Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (size, processing power, disk space, network bandwidth limitations); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operation; audit functions; control functions; language requirements; communications protocols; security considerations; design conventions or programming standards (e.g., if the customer's organization will be responsible for maintaining the delivered software).]

2.6 User Documentation

[List the user documentation components (such as user manuals, online help, and tutorials) that will be delivered along with the software.]

2.7 Assumptions and Dependencies

[List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. Also list any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).]

3. System Features

3.1 System Feature 1 (and so on for each feature)

3.1.1 Description and Priority

[Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority.]

3.1.2 Stimulus/Response Sequences

[List the sequences of user actions and system responses that stimulate the behavior defined for this feature. This is essentially a step-by-step account of how the system will behave in response to a user action under certain conditions.]

3.1.3 Functional Requirements

[Itemize the detailed requirements associated with this feature. These are often written as “shall” statements.]

4. External Interface Requirements

4.1 User Interfaces

[Describe the logical characteristics of each interface between the software product and its users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (like cancel, save, exit), colors, fonts, etc.]

4.2 Hardware Interfaces

[Describe the logical and physical characteristics of each interface between the software and the hardware components of the system. This might include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.]

4.3 Software Interfaces

[Describe the connections between this software and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components.]

4.4 Communications Interfaces

[Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on.]

5. Other Nonfunctional Requirements

5.1 Performance Requirements

[If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.]

5.2 Safety Requirements

[Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations containing safety requirements that affect the product.]

5.3 Security Requirements

[Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security requirements that affect the product. Specify any auditing or logging requirements.]

5.4 Software Quality Attributes

[Address other quality characteristics of the system including accessibility, scalability, maintainability, reliability, and portability.]

5.5 Business Rules

[Define any business rules that affect the product.]

6. Other Requirements

6.1 Database Requirements

[Describe any requirements for database systems, including requirements for database integrity, robustness, backup, and data migration.]

6.2 Compliance Requirements

[Describe any compliance requirements, including compliance to standards, regulations, and other legal issues.]

6.3 Licensing and Installation

[Requirements for licensing and installation of the software.]

Appendix A: Glossary

[Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations.]

Appendix B: Analysis Models

[Optionally include any analysis models used, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.]

Appendix C: To Be Determined List

[Collect a list of items that are not yet decided or that are pending further discussion.]

Questions Bank for SRS

1. **Describe System Requirements Specification (SRS) document.**

   The System Requirements Specification (SRS) document is a critical piece of documentation in software development. It serves as a detailed description of the software system to be developed, providing a comprehensive outline of the functional and non-functional requirements. The SRS document typically includes the following components:

   1. **Introduction**: This section provides an overview of the document, including its purpose, scope, definitions, acronyms, abbreviations, and references. It sets the stage for the rest of the document.
   2. **Overall Description**: This part offers a high-level view of the application, including its general factors and constraints. It describes the general factors that affect the product and its requirements, giving a background for what will be outlined in greater detail in later sections.
   3. **Functional Requirements**: This section is key to the SRS, detailing the functionalities the software must offer. It describes what the system should do, how it should react in particular situations, and what it should not do. This section is often written in collaboration with stakeholders to ensure all necessary features are captured.
   4. **Non-Functional Requirements**: Also known as quality attributes, this part of the document specifies the non-functional aspects of the software, such as performance, security, usability, and scalability. These requirements do not deal directly with the specific behaviors of the system but describe the environmental conditions under which the system must remain effective or qualities that the system must have.
   5. **System Models**: This can include various types of diagrams or models, like use case diagrams, data flow diagrams, or entity-relationship diagrams, which help to visualize and specify the software system's structure and behavior.

6. **User Interface Design**: Though not always included, this section outlines the look and feel of the software's user interface. It may include wireframes or mockups of screens, indicating how users will interact with the software.
7. **Constraints**: This part outlines any constraints on the system, such as hardware limitations, interfaces with other applications, or specific technologies that must be used or avoided.
8. **Assumptions and Dependencies**: Any assumptions that have been made during the drafting of the document and the dependencies that the project might have on external factors are documented here.
9. **Appendices**: Often used for supporting information that is not critical to the understanding of the document but may be helpful, like a glossary of terms or referenced documents.
10. **Approval and Sign-off**: This is typically the final section, where those responsible for the product's development officially approve the SRS, signifying that they agree with and understand what is expected.

An effective SRS is clear, concise, and unambiguous, ensuring that all stakeholders have a common understanding of what the software will achieve and providing a solid foundation for the development process.

2. **How many phases are there in Brainstorming?**

Brainstorming is a collaborative technique used to generate creative ideas and solutions. It typically involves a group of people and is used in various contexts, including business, education, and software development. The process of brainstorming is divided into three distinct phases: Preparation, Execution, and Follow-up. Each phase plays a crucial role in ensuring the effectiveness of the brainstorming session.

1. **Preparation Phase:**
    - **Objective Setting:** The first step is to clearly define the purpose of the brainstorming session. This involves setting specific goals and objectives. What issue or problem is the brainstorming intended to address?

- **Participant Selection:** Choose participants with diverse backgrounds and perspectives to enrich the brainstorming process. The ideal group size is typically between 5-10 people.
- **Creating an Agenda:** Outline the structure of the session, including time allocations for each part of the process.
- **Logistics Arrangement:** Organize the necessary materials (whiteboards, markers, post-it notes, etc.) and set up the physical or virtual space for the session.
- **Setting Ground Rules:** Establish guidelines to create a safe and open environment. Common rules include encouraging wild ideas, withholding criticism, and building on others' ideas.

2. **Execution Phase:**
    - **Idea Generation:** This is the core of the brainstorming session where participants freely express their ideas. Techniques such as mind mapping, free writing, or round-robin brainstorming can be used.
    - **Encouraging Participation:** Ensure that all participants have the opportunity to contribute. Facilitators play a key role in encouraging quieter members to share their ideas.
    - **Idea Documentation:** Record all ideas without judgment. This can be done on a whiteboard, through note-taking, or using digital tools in the case of virtual brainstorming.
    - **Time Management:** Keep track of time to ensure that the session remains focused and productive.
3. **Follow-up Phase:**
    - **Idea Review and Categorization:** After the session, review and categorize the ideas generated. Group similar ideas and eliminate duplicates.
    - **Evaluation and Selection:** Assess the ideas based on predefined criteria relevant to the session's goals. This might involve voting, discussion, or further analysis.
    - **Action Plan Development:** Develop an action plan for the selected ideas. Determine the next steps, assign responsibilities, and set timelines.

    - **Feedback and Reflection:** Gather feedback from participants about the brainstorming process and reflect on what worked well and what could be improved for future sessions.

   Each phase is essential to the overall success of the brainstorming session. The preparation phase sets the stage, the execution phase is where ideas are generated, and the follow-up phase involves evaluating and implementing the best ideas. Effective facilitation and adherence to the phases can significantly enhance the productivity and creativity of brainstorming sessions.

3. **What is a Software Requirements Specification (SRS)?**
   - A document that describes the software system to be developed, outlining functional and non-functional requirements and user interactions.
4. **What does SRS stand for in a project context?**
   - Software Requirements Specification, a document detailing the expected behavior of a software system.
5. **Define System Requirements Specification (SyRS).**
   - A structured collection of information that embodies the requirements of a system.
6. **Describe SRS in Software Engineering.**
   - A comprehensive description of how the system is expected to perform, signed off at the end of the requirements engineering phase.
7. **What are Software Requirements?**
   - A discipline within software engineering that deals with establishing the needs of stakeholders to be solved by software.
8. **What are User Interface Requirements?**
   - Requirements for a software or hybrid system's user interface, focusing on ease of operation, brief response, efficient error handling, and consistent user interface.
9. **Explain the concept of Requirement Gathering.**
   - The practice of collecting system requirements from users, customers, and other stakeholders.

10. **Why is an SRS document important?**
    - It acts as a contract between the client and software vendor, helps in costing and pricing, provides input for detailed design, and manages client expectations.
11. **What are the contents of an effective SRS document?**
    - Project scope, functional requirements, requirement analysis models, external interface requirements, and non-functional requirements.
12. **Differentiate between Functional and Non-Functional Requirements.**
    - Functional requirements define features and functionality within and from the software system. Non-functional requirements are implicit characteristics like security, performance, and interoperability.
13. **What is Prototyping in software development?**
    - Building a user interface without detailed functionality to interpret the features of the intended software product.
14. **What is Requirements Analysis?**
    - The process of identifying user expectations for a new or modified product, focusing on quantifiable, relevant, and detailed features or functional specifications.
15. **Discuss Software Requirement Validations.**
    - The process of verifying that requirements stated in the SRS document are practical, valid, complete, and testable.
16. **What are the features of an SRS?**
    - User requirements in natural language, technical requirements in structured language, design descriptions, format of forms and GUI screen prints, and notations for DFDs.
17. **What is Reliability in the context of software requirements?**
    - A characteristic of SRS where the requirements are expected from the system itself and are correct, clear, viable, complete, flexible, consistent, and accurate.
18. **Are MS Office, Photoshop, and Figma examples of application software?**

- Yes, they are examples of application software.

19. **What is Requirement Engineering?**
    - The process of gathering software requirements from the client, analyzing, and documenting them.
20. **What is a Feasibility Study in the context of software development?**
    - A feasibility study in software development assesses the practicality and viability of a proposed project, considering factors like technical feasibility, economic viability, legal requirements, and operational feasibility.
21. **Explain the concept of Software Requirement Validation.**
    - Software Requirement Validation is the process of ensuring that the requirements stated in the software requirements specification (SRS) document are feasible, valid, complete, and testable. It aims to identify and correct ambiguities, incompleteness, or errors.
22. **Describe the role of a Business Analyst in software development.**
    - A Business Analyst in software development plays a critical role in bridging the gap between business needs and the solutions offered by IT. They gather, analyze, and document business requirements, help in guiding the development process, and ensure that the final product meets business needs.
23. **What is the purpose of a Software Development Life Cycle (SDLC)?**
    - The Software Development Life Cycle (SDLC) provides a structured process for planning, creating, testing, and deploying an information system. The main purpose of the SDLC is to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.
24. **How do Agile methodologies influence software requirements?**
    - Agile methodologies emphasize flexible, iterative development and collaboration. They impact software requirements by promoting adaptive planning, evolutionary development, early delivery, and continual improvement, allowing for rapid and flexible response to changes in requirements.
25. **Discuss the importance of Software Maintenance.**

- Software maintenance is crucial as it involves modifying and updating software application after delivery to correct faults, improve performance, and adapt to a changed environment. It ensures the software continues to function reliably and efficiently in the new or changed environment.

26. **What is the importance of Software Analysis?**
    - Software analysis is important as it involves understanding and documenting software requirements and specifications. It helps in identifying potential problems early in the development process, ensuring that the final product meets the user's needs and works efficiently.
27. **Explain Software Design in the context of software requirements.**
    - Software design in the context of software requirements involves the process of transforming requirements into a suitable form that helps programmers in software coding and implementation. It is about creating a blueprint for the development of the software.
28. **What are the challenges in Organizing Requirements?**
    - Organizing requirements involves challenges like dealing with ambiguous or conflicting requirements, prioritizing a large number of requirements, managing changes in requirements, and ensuring stakeholder agreement and understanding.
29. **Write Down Product Documentation for a Food Delivery App**

▼ Answer

1. Introduction

1.1 Purpose

This document provides a comprehensive overview of the Food Delivery App, designed to connect customers with a wide range of restaurants and food outlets for home delivery services.

1.2 Scope

The Food Delivery App facilitates the browsing, ordering, and delivery of food from local restaurants to customers' doorsteps. It covers user account management, restaurant listings, order placement and tracking, payment processing, and feedback mechanisms.

2. User Interface

2.1 Customer Interface

- **Home Screen:** Displays featured restaurants, promotions, and a search bar.
- **Restaurant List:** Shows available restaurants and filters (cuisine, ratings, etc.).
- **Food Menu:** Lists food items with descriptions, prices, and add-to-cart option.
- **Cart:** Review selected items, quantities, and total price before checkout.
- **Order Tracking:** Real-time updates on order preparation and delivery status.

2.2 Restaurant Interface

- **Dashboard:** Overview of incoming orders, performance metrics, and notifications.
- **Menu Management:** Add or update menu items, prices, and descriptions.
- **Order Management:** View and manage incoming and ongoing orders.

2.3 Delivery Partner Interface

- **Delivery Requests:** List of available delivery jobs with details on location and pay.
- **Navigation:** Integrated GPS functionality for route planning.
- **Earnings Tracker:** Overview of completed deliveries and earnings.

3. Features

3.1 Customer Features

- **Account Registration and Login**
- **Location-Based Restaurant Search**
- **Order Customization**
- **Secure Payment Gateway**
- **Order History and Reordering**
- **Ratings and Reviews**

3.2 Restaurant Features

- **Restaurant Profile Management**
- **Sales Analytics and Reporting**
- **Customer Feedback Review**
- **Promotion and Discount Management**

3.3 Delivery Partner Features

- **Flexible Work Schedule**
- **In-app Earnings Withdrawal**
- **Delivery Route Optimization**
- **Performance Analytics**

4. Technical Specifications

4.1 System Requirements

- **Mobile Application:** Compatible with iOS and Android devices.
- **Web Application:** Accessible on modern web browsers.

4.2 Security

- **Data Encryption:** Secure data transmission and storage.
- **User Authentication:** Multi-factor authentication for user accounts.
- **Payment Security:** PCI DSS compliant payment processing.

5. Installation and Setup

5.1 Customer App Installation

Instructions on downloading and installing the app on various platforms.

5.2 Restaurant and Delivery Partner Setup

Guidance on setting up accounts, profiles, and necessary operational details.

6. User Guide

6.1 How to Place an Order

Step-by-step instructions on browsing menus, adding items to cart, and checkout process.

6.2 Managing Your Account

Details on updating user profiles, payment methods, and viewing order history.

7. Support and Maintenance

7.1 Customer Support

Information on how to reach customer service for assistance and inquiries.

7.2 App Updates

Details on regular app updates for feature enhancements and security improvements.

8. Terms and Conditions

8.1 User Agreement

Legal agreements and terms of use for customers, restaurants, and delivery partners.

8.2 Privacy Policy

Information on data collection, usage, and protection policies.

30. **Write Down Product Documentation for a Ride Booking App**
    a. Same like above Food Develivery App