Software Requirements Specification

for

Bookstagram

Version 1.0 approved

Prepared by <author>

<organization>

<date created>

Table of Contents

Revision History

Name	**Date**	**Reason For Changes**	**Version**

1. Introduction

Bookstagram is an innovative platform that allows you to upload and read books. It creates an interactive and engaging reading experience by incorporating advanced features like eye-tracking technology that monitors user behaviour. If a user's eyes remain closed for more than ten seconds, the system will automatically close the book to ensure an active reading experience. Furthermore, the platform monitors users' reading progress, maintaining streaks and rewarding them with achievements to promote consistent reading habits.

1.1 Purpose

The purpose of Bookstagram is to create a seamless and engaging digital reading experience by integrating technology that enhances user interaction. The platform aims to encourage reading habits by providing features such as eye-tracking to maintain focus, streak tracking to motivate users, and a book-uploading feature to support a diverse range of reading materials.

This project is intended for students, educators, and book enthusiasts who want an interactive reading platform that not only provides access to books but also fosters a disciplined and immersive reading environment. By leveraging advanced analytics and tracking, Bookstagram helps users monitor their reading progress and develop consistent reading habits.

This SRS document outlines the functional and non-functional requirements for the Bookstagram system. It defines the scope of the project, ensuring that all stakeholders, including developers, designers, and testers, have a clear understanding of the system's objectives. The document will cover all essential subsystems, including user authentication, book management, reading analytics, and achievement tracking, ensuring a comprehensive development plan for the Bookstagram platform.

1.2 Document Conventions

This document follows standard typographical conventions to maintain clarity and consistency:

- **Bold Text** is used to highlight important terms and headings, making them stand out for easy reference and quick navigation.

- *Italicized Text* is used for emphasis on key concepts, drawing attention to important details that require special consideration or further elaboration.

- Monospace font is used for system commands, code snippets, or filenames, ensuring that any technical elements are clearly distinguished from regular text and easily identifiable by developers and testers.

- Numbered lists are used to outline structured steps or requirements, providing a logical and sequential flow that enhances readability and comprehension of processes.

- Bullet points are used for general lists and key highlights, making it easier to scan and grasp important pieces of information quickly and efficiently.

- Proper indentation and spacing are maintained throughout the document to improve overall readability and ensure a professional presentation of information.

- Headings and subheadings follow a hierarchical structure, allowing users to navigate the document seamlessly and locate relevant sections with ease.

- Tables and diagrams may be included where necessary to represent data, workflows, or system architecture visually, further improving clarity and reducing textual complexity.

1.3 Intended Audience and Reading Suggestions

This document is intended for various stakeholders involved in the development, deployment, and usage of the Bookstagram platform. The primary audience includes:

1. **Developers:** Responsible for implementing the platform's features based on the specified requirements.
2. **Project Mentor:** Overseeing the progress and ensuring that the development aligns with the outlined objectives and timeline.
3. **Marketing Staff:** Understanding the key features and benefits to strategize promotional efforts effectively.
4. **End Users (Readers & Uploaders):** Individuals who will engage with the platform, upload books, and track their reading progress.
5. **Testers:** Ensuring the platform meets quality standards and functions as expected before deployment.
6. **Documentation Writers:** Responsible for maintaining user guides, FAQs, and other support documents.

1.4 Product Scope

- ### Reading Suggestions:

To facilitate efficient navigation, readers are encouraged to follow the sequence below based on their role:

- **General Overview:** Readers new to the project should start with the **Introduction** and **Purpose** sections to gain a high-level understanding of Bookstagram.
- **Technical Readers (Developers & Testers):** Should focus on the **Functional Requirements**, **System Architecture**, and **Use Case Scenarios** sections to understand how the platform should be implemented and tested.
- **Project Managers & Stakeholders:** Should review the **Scope**, **Features**, and **Milestones** sections to track progress and ensure alignment with business objectives.
- **Marketing & End Users:** May find value in the **User Experience**, **Interface Design**, and **Achievement System** sections, which detail engagement and usability aspects.

This structured reading approach ensures that each stakeholder can efficiently access the most relevant information based on their responsibilities and interests.

1.5 References

1. Chen, C.-M., Wang, J.-Y., & Lin, Y.-C. (2019). A visual interactive reading system based on eye tracking technology to improve digital reading performance. Journal Name (if available).

2.Z-Library. (n.d.). Z-Library: The world's largest ebook library. Retrieved [Date], from https://z-lib.id

3.Murata, N., Miyamoto, D., Togano, T., & Fukuchi, T. (2017). Evaluating silent reading performance with an eye tracking system in patients with glaucoma. PLOS ONE. https://doi.org/10.1371/journal.pone.0170230

4.Project Gutenberg. (n.d.). A digital library offering over 60,000 free eBooks, including public domain books available in various formats like PDF and EPUB. Retrieved from https://www.gutenberg.org/

5.Internet Archive. (n.d.). A non-profit digital library providing free access to millions of books, audio files, videos, and m*ore, including downloadable PDFs. Retrieved from https://archive.org/*

2. Overall Description

2.1 Product Perspective

Bookstagram is a self-contained digital library and reading platform that allows users to **upload, download, and read books for free**. It is designed to create an interactive and engaging reading experience by integrating **real-time eye-tracking**, **gamification through reading streaks, and collaborative content sharing**.

Bookstagram is not a direct replacement for any existing system but rather an innovative enhancement in the e-learning and digital reading space. It provides a **user-friendly interface** for **students, book enthusiasts, and researchers** to easily access and share books while encouraging consistent reading habits.

- **System Context & External Interfaces**

Bookstagram operates as a **web-based or mobile** application that interacts with:

1) **Users (Readers & Uploaders)** – for book uploads, downloads, and reading sessions.
2) **Eye-Tracking Module** – to detect eye movement and automatically pause reading if the user's eyes are closed for more than 10 seconds.
3) **Streak & Achievement System** – to encourage consistent reading habits.

2.2 Product Functions

- ○ Bookstagram is designed to enhance the reading experience by integrating user engagement, accessibility, and security through various functional modules. Each function plays a crucial role in ensuring smooth operation, user satisfaction, and system efficiency.

- **1. User Management**

- ○ A robust user management system ensures authentication, authorization, and personalization. By implementing role-based access control (RBAC) and a structured database schema, Bookstagram allows seamless user interactions while maintaining security.

- ○ **Functions:**

- User Registration & Login: Supports multiple authentication methods (Email, Google, etc.) to enhance accessibility.
- Profile Management: Tracks user activity, reading history, and achievements for a personalized experience.

- **2. Book Management**

- ○ Digital libraries follow information retrieval principles, where efficient indexing, metadata tagging, and structured storage improve searchability and accessibility. Data normalization techniques help manage books efficiently within the database.

- ○ **Functions:**

- Upload Books: Users can contribute books in PDF format, with metadata like author, genre, and summary.
- Download Books: Ensures high-speed, secure access to books for offline reading.
- Search & Filter: Implements Boolean search and keyword-based filtering to optimize book discovery.

- **3. Reading System**

- ○ Human-computer interaction (HCI) principles dictate that digital reading systems should minimize distractions, enhance usability, and optimize cognitive load. Features like bookmarking, eye-tracking, and session resumption enhance user retention.

- ○ **Functions:**

- In-app Book Reader: Provides an interactive interface with adjustable fonts, themes, and reading modes.
- Eye-Tracking Feature: Uses computer vision and gaze-tracking algorithms to detect when a user’s eyes are closed for more than 10 seconds, automatically pausing the book.
- Bookmarking & Resume: Implements session persistence to allow users to resume from where they left off.

- **4. Gamification & Engagement**

  - Self-Determination Theory (SDT) states that motivation in learning is enhanced by autonomy, competence, and relatedness. Gamification techniques such as progress tracking, rewards, and leaderboards encourage user participation.

  - **Functions:**

- Reading Streaks & Achievements: Encourages users to build consistent reading habits through daily goals.
- Leaderboard: Creates a competitive environment where users can track their progress.
- Daily Reading Goals & Progress Tracker: Uses behavioral reinforcement models to sustain user engagement.

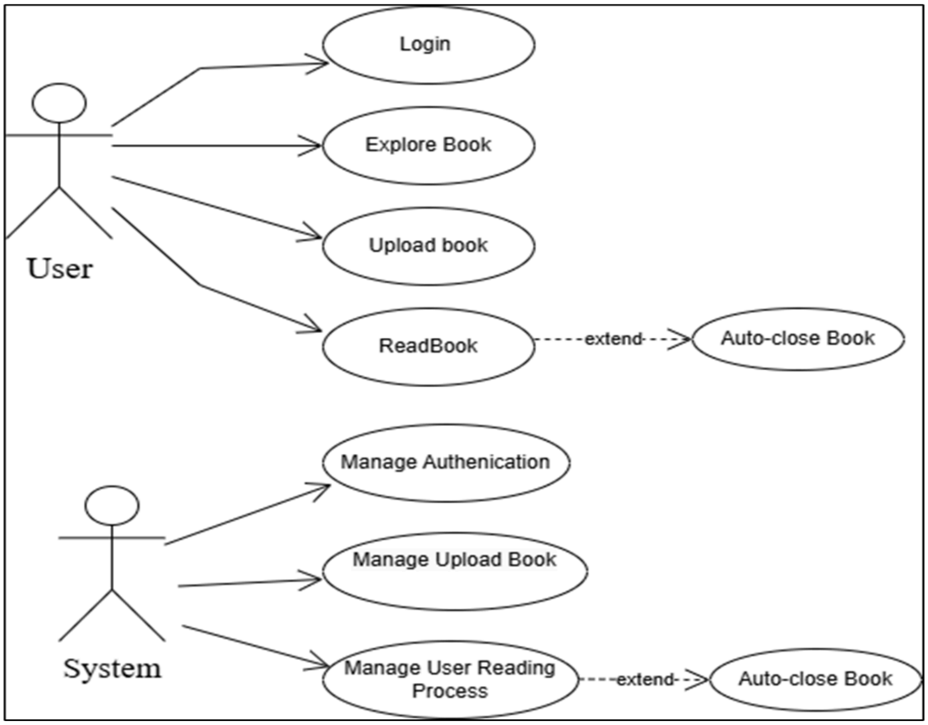
- **5. Security & Moderation**

  - A multi-layered security model protects user data, prevents abuse, and ensures regulatory compliance. Cryptographic encryption, content moderation AI, and access control lists (ACLs) safeguard user interactions.

  - **Functions:**

- **Data Encryption & Secure Cloud Storage**: Implements AES encryption and cloud-based storage solutions to protect user data and books from unauthorized access.

Use Case Diagram: A Use Case Diagram illustrating the interactions between **User** and **System** will be added here.



2.3 User Classes and Characteristics

Bookstagram caters to the following user classes:

1. **Readers**:
   - **Frequency of Use**: Daily or frequent users who access the platform to read books.
   - **Technical Expertise**: Basic to intermediate; familiar with web applications.
   - **Security/Privilege Levels**: Standard user access with permissions to read, download, and bookmark books.
   - **Educational Level**: Varied, from students to professionals.
   - **Experience**: May range from casual readers to avid book enthusiasts.
2. **Uploaders**:
   - **Frequency of Use**: Occasional users who upload books to the platform.
   - **Technical Expertise**: Intermediate; familiar with file uploads and metadata entry.
   - **Security/Privilege Levels**: Elevated access to upload and manage books, subject to moderation.
   - **Educational Level**: Typically educators, researchers, or book enthusiasts.
   - **Experience**: Likely to have experience with digital libraries or content sharing platforms.

2.4 Operating Environment

Bookstagram will operate in the following environments:

- ***1.* Hardware Platform**

- **Servers:** Cloud-based servers (e.g., AWS, Google Cloud) for hosting the application and storing books.
- **Client Devices:** Modern desktops, laptops, tablets, and smartphones with front-facing cameras for eye-tracking functionality.

- ***2.* Minimum Hardware Requirements**

- **Client Devices:**
  - Minimum 2GB RAM, dual-core processor
  - Front-facing camera for eye-tracking feature
- **Servers:**
  - Scalable cloud infrastructure
  - Sufficient storage and processing power to handle concurrent users

- ***3.* Software Platform**

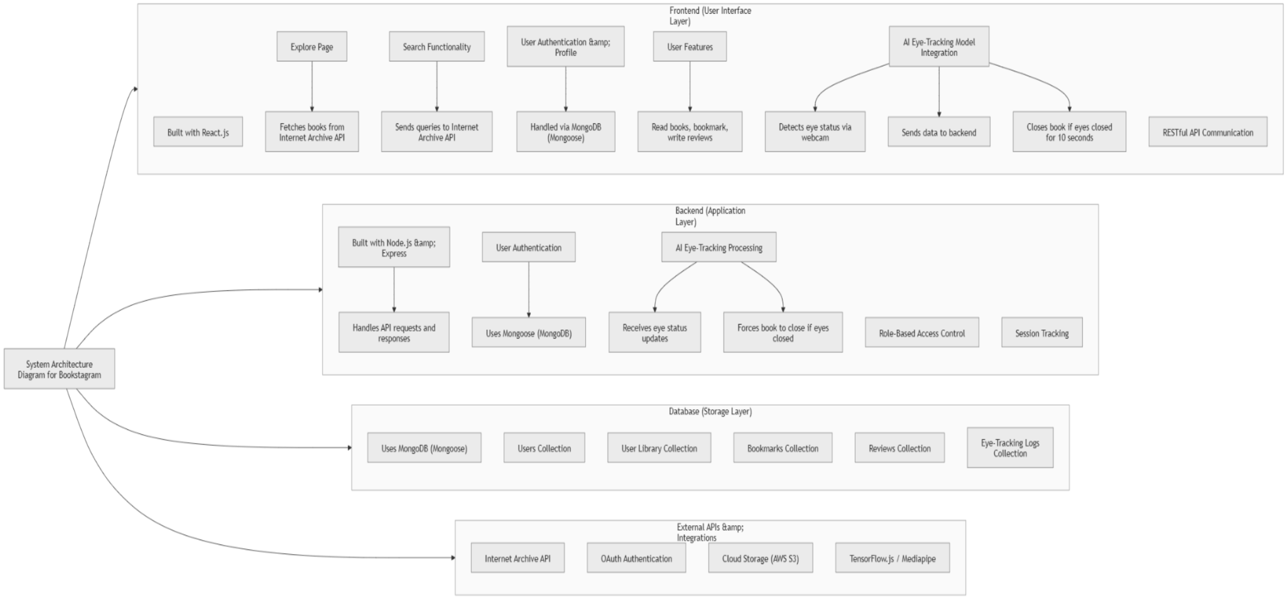
- **Operating Systems:** Windows 10/11, macOS, Linux, Android, and iOS
- **Web Browsers:** Google Chrome, Mozilla Firefox, Microsoft Edge, Safari (latest versions)
- **Mobile Platforms:** Android 8.0+ and iOS 12.0+ for mobile app compatibility

- ***4.* Other Software Components**

- **Database:** MongoDB (Mongoose) for storing user data and book metadata
- **Cloud Storage:** AWS S3 or Google Cloud Storage for hosting uploaded books
- **Third-Party Libraries:**
  - ***OpenCV*** *for eye-tracking*
  - ***React.js*** *for frontend*
  - ***Flask/Node.js*** *for backend*

- ***5. System Architecture Diagram***

The system architecture is structured into **Frontend (User Interface Layer), Backend (Application Layer), Database (Storage Layer), and External Integrations.**


System Architecture Diagram for Bookstagram
Frontend (User Interface Layer)
Explore Page
Search Functionality
User Authentication &amp; Profile
User Features
AI Eye-Tracking Model Integration
Built with React.js
Fetches books from Internet Archive API
Sends queries to Internet Archive API
Handled via MongoDB (Mongoose)
Read books, bookmark, write reviews
Detects eye status via webcam
Sends data to backend
Closes book if eyes closed for 10 seconds
RESTful API Communication
Backend (Application Layer)
Built with Node.js &amp; Express
User Authentication
AI Eye-Tracking Processing
Handles API requests and responses
Uses Mongoose (MongoDB)
Receives eye status updates
Forces book to close if eyes closed
Role-Based Access Control
Session Tracking
Database (Storage Layer)
Uses MongoDB (Mongoose)
Users Collection
User Library Collection
Bookmarks Collection
Reviews Collection
Eye-Tracking Logs Collection
External APIs &amp; Integrations
Internet Archive API
OAuth Authentication
Cloud Storage (AWS S3)
TensorFlow.js / Mediapipe

2.5 Design and Implementation Constraints

Copyright & Intellectual Property Regulations: Since Bookstagram allows book uploads, strict compliance with copyright laws (such as the **Copyright Act of India, 1957, and DMCA**) is necessary to avoid unauthorized content sharing.

Data Protection Laws: The system must comply with data privacy regulations such as I**ndia's IT Act (2000) & GDPR** ensuring user data is handled securely.

Content Moderation Policies: Uploaded books must be reviewed for inappropriate, plagiarized, or copyrighted content, requiring an automated moderation system or manual review processes.

■

■

■

■ **Technical Constraints**

- **Hardware Limitations:**

  The **eye-tracking feature** requires a device with a **front camera**, which may limit compatibility on older hardware.

  Mobile devices with low **RAM and processing power** may struggle with **real-time tracking and book rendering**.

- **Storage & Performance Requirements:**

Hosting large volumes of books requires a **scalable cloud storage solution** (e.g., **AWS S3, Google Cloud Storage**).

- **Technology Stack:**

  Backend: **Flask / Node.js (JavaScript)**

  Frontend: **React.js**

  Database: **MongoDB**

  Cloud Storage: **AWS / Google Cloud**

  AI for Eye Tracking: **OpenCV**

- **Third-Party Integrations:**
  - **Google OAuth & Social Login** for seamless user authentication.
  - **Payment Gateway (if needed for premium features or donations).**

3. Operational Constraints

- **Concurrent Users & Server Load:**

  The system must **support multiple concurrent users** without performance degradation.

- **Interfaces to Other Applications:**

  Bookstagram may need to **integrate with external libraries**, such as **Google Books API** for metadata enrichment.

- **Parallel Operations:**

  he platform should support **simultaneous book uploads, downloads, and reading sessions** without impacting performance.

  Asynchronous processing will be required for resource-heavy tasks.

4. Security Considerations

- **User Authentication & Authorization:**

  Implement **OAuth 2.0 and JWT-based authentication** to prevent unauthorized access.

  Role-based access control (RBAC) to differentiate between readers, uploaders, and admins.

- **Data Encryption:**

Books, user credentials, and reading history must be stored using **AES-256 encryption** for security.

- **Protection Against Cyber Threats:**

Prevent **DDoS attacks, SQL injection, and XSS** using security best practices.

- **5. Design and Maintenance Constraints**

- **Programming Standards & Code Maintainability:**

The system should adhere to **PEP 8 (Python) and ESLint (JavaScript)** coding standards.

Use **MVC (Model-View-Controller) architecture** for modular design and easy maintainability.

- **Long-Term Maintenance & Scalability:**

Ensure **version control (Git, GitHub)** and CI/CD pipelines for smooth updates.

Use **microservices architecture** if scalability demands grow in the future.

2.6 User Documentation

The following user documentation components will be provided along with **Bookstagram** to assist users in understanding and effectively using the platform:

- **Documentation Components**

User Manual (PDF & Web Format)

- A step-by-step guide covering account creation, book uploads, reading features, and troubleshooting.
- Available for download as a **PDF** and accessible on the website under the “Help” section.

Tutorial Videos (YouTube & Embedded on Website)

- Short **video tutorials** demonstrating core functionalities like uploading books, managing reading streaks, and enabling eye-tracking.
- Hosted on **YouTube**

2.7 Assumptions and Dependencies

Assumptions

User Access to Internet – It is assumed that users will have a stable internet connection to access and upload books. Offline functionality is not a primary focus in the initial release.

Device Compatibility – Bookstagram will primarily be designed for web-based access, assuming users have modern browsers (Chrome, Firefox, Edge, Safari) with JavaScript enabled. A mobile-friendly version may be considered later.

User-Generated Content Compliance – It is assumed that users will upload only legal and copyright-free books or materials for personal or educational purposes. The system may include a reporting mechanism for violations.

Eye-Tracking Hardware & Support – The eye-tracking feature assumes users have a webcam enabled on their devices for functionality. It also assumes that facial recognition APIs or libraries used will function accurately across different lighting conditions and face structures.
Scalability & Initial User Base – It is assumed that the system will start with a moderate number of users and scale up as required. Initial server capacity and database structure will be designed accordingly.

Free Access Model – Bookstagram will operate under a free access model where users can upload and read books at no cost, assuming that storage and operational costs will be manageable through available resources or sponsorship.

Dependencies

Database Management System – Bookstagram will rely on a database solution (e.g., PostgreSQL, MongoDB, or Firebase Firestore) to store user data and uploaded books. Any change in database service policies may require adjustments.

External Libraries & Frameworks: The system will use third-party libraries (e.g., React.js for frontend, Flask/Django for backend). Any discontinuation or major updates may require refactoring.

3. External Interface Requirements

3.1 User Interfaces

<Describe the logical characteristics of each interface between the software product and the 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 (e.g., help) that will

appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

● 3.2 Hardware Interfaces

Bookstagram will interact with the following hardware components:

1.Front-Facing Camera:

- The eye-tracking feature requires a front-facing camera to monitor user engagement.
- The system will use **OpenCV** to process real-time video feed and detect eye movements.

2.Client Devices:

- The platform will support desktops, laptops, tablets, and smartphones with modern browsers and front-facing cameras.
- Minimum hardware requirements: 2GB RAM, dual-core processor, and a front-facing camera.

3.Communication Protocols:

- The system will use **HTTP/HTTPS** for communication between the client and server.

Data transfer rates will be optimized to ensure smooth book rendering and eye-tracking functionality.

○ 3.3 Software Interfaces

Bookstagram will integrate with the following software components:

1. **Database**:
   - **MongoDB** or **PostgreSQL** will be used to store user data, book metadata, and reading progress.
   - Data sharing mechanism: RESTful APIs will be used to communicate between the frontend and backend.
2. **Third-Party Libraries**:
   - **OpenCV**: For eye-tracking functionality.
   - **React.js**: For the frontend user interface.
   - **Flask/Node.js**: For the backend server.
3. **Cloud Storage**:
   - **AWS S3** or **Google Cloud Storage** will be used to store uploaded books.
   - Data transfer protocols: Secure FTP (SFTP) for uploading and downloading books.
4. **Authentication**:
   - **Google OAuth** will be used for user authentication.
   - **JWT (JSON Web Tokens)** will be used for session management.

- ### 3.4 Communications Interfaces
  - Bookstagram will use the following communication protocols and standards:

1. **HTTP/HTTPS:**
   - All communication between the client and server will be encrypted using HTTPS.
   - Data transfer rates will be optimized to ensure fast loading times for books and eye-tracking data.
2. **WebSocket:**
   - Real-time communication for eye-tracking data and reading progress updates.
3. **Security:**
   - Secure authentication: OAuth 2.0 and JWT will be used for secure user access.

   ○

4. System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1 System Feature 1

<Don't really say "System Feature 1." State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:

REQ-2:

4.2 System Feature 2 (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 that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

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 issues that affect the product. Define any security or privacy certifications that must be satisfied.>

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

5.5 Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>