**CAPSTONE PROJECT SRS**

**(SOFTWARE REQUIREMENT SPECIFICATIONS)**

**PROJECT TITLE – FLYSHARE**

**GROUP NUMBER – 6**

TEAM MEMBERS

Siddharth Kumar - 11452

Tharun Balisetty - 11462

Abhishek Polamarasetty - 11478  
Sravya Pothuraju - 11485

**CHAPTER – 1**

**INTRODUCTION**

* 1. **PURPOSE**

The purpose of this Software Requirements Specification (SRS) document is to provide a comprehensive technical specification covering all aspects of the project (FlyShare) including functional requirements, system architecture, database needs, performance parameters, quality attributes, software components and hardware infrastructure. This SRS document serves as a reference guide for developers, testers and stakeholders, facilitating communication and outlining the intended functionalities, technologies and development methodologies to be employed for the platform. Overall, it aims to specify the requirements, architectures and techniques necessary to successfully build, test and deploy the complete project (FlyShare).

**1.2 SCOPE**

**a) Software Product(s) to be Produced:**

The software products within the scope of Flyshare encompass:

* **Flyshare Web Application:** The primary platform accessible via web browsers for users to engage in luggage-sharing activities.
* **Backend Systems:** Comprising authentication, database management, chat functionalities, and external integrations necessary for the application's operations.

**b) What the Software Product(s) Will Do:**

* **Facilitate Luggage Sharing:** Flyshare will enable users to share luggage space by connecting with co-passengers, thereby potentially avoiding excess baggage fees.
* **Manage User Authentication:** Ensuring secure user authentication, account management, and profile settings.
* **Enable Real-time Communication:** Providing a chat interface for users to negotiate luggage-sharing arrangements securely.

**What the Software Product(s) Will Not Do:**

* Flyshare will not handle airline ticketing, flight bookings, or baggage handling logistics directly. It will not serve as a substitute for official airline baggage policies or procedures.

**c) Application of the Software and Relevant Goals:**

**Flyshare's application aims to:**

* **Enhance Travel Experience:** By reducing excess baggage costs and fostering collaborative interactions among travellers.
* **Promote Cost Efficiency:** By enabling users to coordinate luggage sharing, leading to cost savings for participants.
* **Facilitate Seamless Communication:** Providing a platform for users to communicate and organize luggage-sharing activities effectively.

**1.3 DEFINITION, ACRONYMS & ABBREVATIONS**

* **Flyshare:** The name of the web application developed for facilitating luggage-sharing among airline passengers.
* **PNR (Passenger Name Record):** A unique identifier used by airlines to manage reservations and store passenger information.
* **Backend:** The server-side part of the application responsible for business logic, data processing, and database management.
* **Frontend:** The client-side part of the application visible to users, facilitating interactions and displaying information.
* **PNR -** Passenger Name Record
* **AWS -** Amazon Web Services
* **HTML -** hyper Text Markup Language
* **CSS -** Cascading Stylesheets
* **JS –** JavaScript

**1.4 REFERENCES**

* IEEE Recommended Practice for Software Requirements Specifications -

<http://www.math.uaa.alaska.edu/~afkjm/cs401/IEEE830.pdf>

* Sample Mobile Application Reference -   
  <https://xd.adobe.com/view/b9b46ecd-8ecb-466d-9786-ffca352352f6-ece7/screen/821e91de-bbc6-49cb-b4dd-ae8b2cdf2aaa/>
* Baggage allowance - <https://www.yatra.com/baggage-allowance.html>  
   - <https://www.goindigo.in/baggage/excess-baggage.html>

**1.5 OVERVIEW**

The Software Requirements Specification (SRS) encompasses a comprehensive outline detailing the functional and non-functional requirements necessary for the development of the Flyshare application. This document provides a detailed breakdown of the system's objectives, functionalities, constraints, and specifications crucial for understanding the application's scope and requirements.

**Organization of the SRS:**

The SRS is structured into several sections to facilitate a clear and organized presentation of information:

**Introduction:** Provides an overview of the project, including its purpose, scope, definitions, and intended audience.

**Overall Description:** Encompasses a high-level description of the system, its functions, user classes, constraints, assumptions, and dependencies.

**Specific Requirements:** Details functional requirements, database requirements, performance requirements, software quality attributes, and software and hardware prerequisites.

**CHAPTER - 2**

**OVERALL DESCRIPTION**

**2.1. PRODUCT PERSPECTIVE  
Industry Context:** Describes the broader travel industry, highlighting the challenges faced by passengers dealing with excess baggage fees and the need for a cost-effective luggage-sharing solution.

**Role in the Travel Ecosystem:** Explains Flyshare's role as an innovative platform providing a solution to excess baggage costs by facilitating communication between passengers. It clarifies how Flyshare addresses a specific need within the travel ecosystem.

**Interaction with Other Systems:** Details any interactions or integrations Flyshare has with other systems or services. For instance, it might mention interaction with different websites for PNR verification and the use of Gmail for notifications.

**User Interaction:** Describes how passengers engage with Flyshare, emphasizing the platform's functionalities and how it serves as a communication medium between co-passengers.  
  
**2.2 PRODUCT FUNCTIONS**

**PNR Verification:** Users can input their Passenger Name Record (PNR) and flight details for verification, ensuring valid access to the platform.

**Real-Time Chat Functionality:** Provides a secure communication platform for users to interact, negotiate luggage-sharing arrangements, discuss terms, and exchange information in real-time.

**Luggage Description and Image Upload:** Allows users to describe their extra luggage and upload images for better visualization, aiding potential co-passengers in understanding the additional baggage.

**Notifications via Gmail:** Sends timely updates and notifications to users' Gmail accounts regarding important events, such as new chat messages, successful luggage-sharing arrangements.

**2.3 USER CLASSES AND CHARACTERISTICS  
  
Passengers Seeking Luggage Sharing:**

* **Role:** These passengers are the primary audience of Flyshare, seeking a solution to manage excess baggage fees during their flights.
* **Characteristics:** They have PNRs, flight details, and additional luggage, and they engage with the platform to connect with potential co-passengers for luggage-sharing arrangements.

**Developers/Administrators:**

* **Role:** Individuals responsible for building, maintaining, or overseeing the Flyshare platform.
* **Characteristics:** They possess technical knowledge, manage platform functionalities, ensure system reliability, and handle administrative tasks.

**Travel Experience:**

Flyshare caters to a spectrum of users with varied travel experiences. It accommodates both frequent flyers and individuals embarking on their first flights, facilitating luggage-sharing opportunities irrespective of user’s familiarity with air travel procedures.

**2.4 DESIGN AND IMPLEMENTATION CONSTRAINTS**

**Scalability Constraints:** The application's ability to handle increased user traffic or data volume over time without compromising performance or user experience.

**Third-Party Dependencies:** Reliance on external services, APIs, or third-party integrations that might be subject to their limitations, downtimes, or changes in functionality.

**User Interface Constraints:** Design limitations or challenges in creating a simplified user interface for various devices, screen sizes, and accessibility requirements.

**2.5 ASSUMPTIONS AND DEPENDENCIES**

**ASSUMPTIONS:**

* **User Behaviour:** Assumptions about how users will interact with the platform, such as their willingness to engage in luggage-sharing discussions or their preference for certain communication methods.
* **Data Accuracy:** Assuming the accuracy and authenticity of the information provided by users, like the correctness of PNR details and uploaded luggage images.
* **Hardware and Browser Compatibility:** The application assumes compatibility with standard computing devices and web browsers commonly used by passengers. Any significant changes in hardware specifications or browser behaviour may require adjustments to ensure seamless user experience.

**DEPENDENCIES:**

* **Collaboration Dependencies:** Dependency on effective collaboration between users for luggage-sharing arrangements, relying on their willingness to engage in discussions and agreements.
* **Integration Dependencies:** Dependencies on integrating various modules or components within the Flyshare platform, ensuring seamless interaction between different features (e.g., chat, image upload).
* **Testing Dependencies:** Reliance on comprehensive testing environments and procedures to ensure the platform's functionality, security, and performance, which may require specific testing tools or environments.
* **External Services:** Reliance on external services (such as Gmail for notifications) or APIs for functionality, assuming their continuous availability and compatibility.

**CHAPTER – 3  
SPECIFIC REQUIREMENTS**

**3.1 FUNCTIONAL REQUIREMENTS**

**PNR Verification:**

* **Requirement:** Users should have access to an external link for PNR verification.
* **Functionality:** Provide users with a link or a means to access an external service (e.g., airline's official website) for PNR verification. The Flyshare platform facilitates the accessibility of this external verification source rather than performing the verification internally.

**Real-Time Chat Functionality:**

* **Requirement:** Enable users to communicate securely within the platform.
* **Functionality:** Provide a chat interface allowing users to discuss luggage-sharing arrangements, negotiate terms, and exchange information in real-time.

**Luggage Description and Image Upload:**

* **Requirement:** Users should describe their extra luggage and upload images.
* **Functionality:** Allow passengers to describe their additional luggage and provide images for visualization, aiding potential co-passengers in understanding the extra baggage.

**Notifications via Gmail:**

* **Requirement:** Users should receive updates on their Gmail accounts.
* **Functionality:** Implement a notification system that sends timely updates and notifications to users' Gmail accounts regarding significant events on the platform, such as new chat message or verified PNRs.

**3.2 DATABASE REQUIREMENTS**

**Data Storage:** The database should efficiently store user-provided information, including PNR details, flight information, user profiles, chat messages, and uploaded images of extra luggage.

**Data Retrieval and Verification:** Enable quick retrieval and verification of PNR details entered by users for the external verification process. The database should facilitate access to this information as needed.

**Scalability:** The database structure should support potential scalability needs as the user base grows, ensuring the system can handle increased data volume without compromising performance.

**Backup and Recovery:** Implement mechanisms for regular data backup and a robust recovery system to prevent data loss in case of system failures or unforeseen circumstances.

**3.3 PERFORMANCE REQUIREMENTS  
a) Scalability of Terminals:**

The application should exhibit scalability to support a considerable number of terminals concurrently accessing the system. The exact number shall be determined during system testing.

**b) Concurrent User Support:**

Flyshare aims to accommodate a substantial number of simultaneous users interacting with the platform concurrently. The exact concurrent user capacity will be established during load testing.

**c) Information Handling Capacity:**

The system is designed to manage and process diverse types of passenger data, including luggage descriptions and images, ensuring scalable storage capacity to meet operational demands.

**3.4 SOFTWARE QUALITY ATTRIBUTES**   
**Reliability:** The system's ability to perform consistently and accurately under varying conditions, ensuring data integrity, user authentication, and reliable functionality throughout user interactions.

**Security:** Ensuring robust security measures to protect user data, implementing encryption, secure authentication methods, and access control to prevent unauthorized access or data breaches.

**Maintainability:** Designing the system in a way that allows for easy maintenance, updates, and enhancements without disrupting the platform's functionality or usability.

**3.5 SOFTWARE & HARDWARE REQUIREMENTS**

**SOFTWARE REQUIREMENTS -**

**Frontend Development:**

* **HTML, CSS, JavaScript:** For creating the user interface and interactivity within the Flyshare application.

**Backend Development:**

* **Django:** Utilized as the primary backend framework to handle server-side logic and API integrations.

**Database Management System:**

* **MongoDB:** It is used for user authentication and for flexible storage of luggage-related information. It’s document-oriented structure allows for quick retrieval and storage of unstructured data, making it a suitable choice for managing real-time chat data.

**Hosting and Deployment:**

* **Cloud Hosting:** Utilizing cloud-based services (such as AWS) for hosting and deploying the Flyshare application.

**Version Control:**

* **Git:** Employed for version control, enabling collaborative development, tracking changes, and managing codebase versions.

**Continuous Integration/Continuous Deployment (CI/CD):**

* **Jenkins:** Integrated into the development workflow to automate the CI/CD pipeline for efficient development, testing, and deployment processes.

### HARDWARE REQUIREMENTS -

**1. Development Machines:**

* **Hardware:** Standard development machines with sufficient RAM (8GB or more), modern processors, and ample storage.
* **Details:** Developers' machines should meet the hardware requirements for efficient coding, testing, and running development servers.

**2. Build and Deployment Servers:**

* **Hardware:** Servers with sufficient resources for building and deploying the application.
* **Details:** These servers should have the capacity to handle the build process, automated testing, and deployment tasks efficiently.

**3. Database Server:**

* **Hardware:** Dedicated server or cloud-based instance with adequate resources for hosting the MongoDB database.
* **Details:** The database server should be configured to handle the expected data load and concurrent connections.

**4. AWS Infrastructure:**

* **Hardware:** Cloud-based infrastructure on AWS (e.g., EC2 instances).
* **Details:** AWS resources should be provisioned based on scalability and performance requirements. The infrastructure should support load balancing and automatic scaling.