Software Requirement Specification

Problem Statement: A few sentences to describe the problem you are trying to solve, i.e., justify why this software is needed.

- Due to expensive shipping fees, many people are unable to complete international deliveries cheaply and quickly. Our international shipping app allows users to combine their individual packages into one order so they can split and reduce their combined shipping fees. Other shipping/shopping or budgeting services do not account for this specific case, and do not have features nor the intent to cover them, so our platform is intended to bridge the disconnect between these two types of apps.
- Usually, tracking and managing a number of small international orders requires a disproportionate amount of time and effort, so we wanted to provide a solution that makes the process cheaper, automatic, and stress-free.

Potential Clients: Who are influenced by this problem and would benefit from the proposed solution? (i.e. the potential users)

- As a small business owner, I want to be able to lower the cost of shipping in order to provide higher-quality and cheaper service to my customers.
- As an international student, I want to buy products at a low price from my homeland if I am homesick or missing a specific item only available there.
- As a frugal person, I want to buy a small number of products affordably in a bundle with other users to maintain a low shipping cost and save money.
- As a time-sensitive or busy person, I want to locate shipping companies and ship my items quickly and conveniently, in order to avoid having to spend time or money to use international shipping.

Proposed Solution: Write a few sentences that describe how a software solution will solve the problem described above.

- Since we believe that domestic shipping fees are relatively low in comparison, the most cost-inducing part of the supply chain is international shipping. Therefore, users can use our service to combine orders of individual packages, and simultaneously reduce and divide the shipping costs between them.
- Once multiple users want to receive their orders, we calculate the total weights of their items for the total international shipping fee, and divide by the number of users, the more users participate, the higher discount they receive. The app would also weigh the shipping fee contributions fairly (e.g. users with heavier items pay more in comparison). To connect users, the app would also find orders that have similar weights, and users that live closer together to also reduce the domestic shipping fee.

(How could we manage each user's order? System organized or the manager? How?)

Functional Requirements: List the (functional) requirements that software needs to have in order to solve the problem stated above. List these in role-goal-benefit format. It is useful to try to group the requirements into those that are essential (must have), and those which are non-essential (but nice to have).

Must have

1. Interactive notebook to keep track of orders
2. Underlying order management infrastructure, to calculate/minimize costs and find available shipping routes
3. The UI automates the customers' order weights and allows them to input the address, payment, and contact so users will get notified when payment deadline approaches.
4. User registration and authentication, which allows users to participate in the shipping community
5. User profile management, which allows users to manage their profile such as shipping address, payment methods, and preferences, etc.

6. Payment management / authentication

Nice to have

1. System gathers coupons provided by the different stores to use at checkout
    a. Integrating with existing shopping sites
2. Automated customer service (e.g. chatbot)
3. Suggested routes and shipping methods
4. Matching warehouses, international shipping carriers with users

Non-functional Requirements:

1. Availability: The system should be available 24/7, with planned downtime notified 24 hours in advance.
2. Response Time: Data transformation time should be limited to 10 seconds per image if applicable, and general database data operations should have a response time of less than 2 seconds.
3. User-Friendly: A typical user should be able to use all system functions within 20 minutes of self-exploration.
4. Failure Rate: The failure occurrence rate from the system server should not exceed 1 in 50 requests from users.
5. Scalability: The system should support up to 1000 users simultaneously.

Software Architecture & Technology Stack: Will this be a Web/desktop/mobile (all, or some other kind of) application? Would it conform to specific software architecture? What programming languages, frameworks, databases, …, will be used to develop and deploy the software?

1. Type: A web-based application accessible through standard web browsers.
2. Architecture: A client-server architecture, where the frontend (client) communicates with the backend server.
3. Programming Languages: JavaScript, HTML, and CSS for frontend, and Express.js for backend.
4. Frameworks:
    a. Frontend: React.js
    b. Backend: Express.js
5. Database: MongoDB
6. Hosting/Deployment:

Similar Apps: List a few similar applications to the one you are developing. Don't be eager to conclude no similar app exists! There is always something similar to what you are building! Finding those will help you to better specify your project. You must be prepared to explain how your app is different from the existing ones.

1. Amazon: it is the most commonly used shipping app for buyers in the U.S., it provides a variety of goods and multiple delivery options (time), but it does not have features like our app that allow the users to combine their orders to cut the cost.

There are many international shopping/shipping websites and budget sharing apps, but few that actually do both specifically for the purpose of lowering shipping costs like our app idea is proposing.

- International shopping/shipping
- Online shops are specifically built to be shopping sites first, and shipping sites second, which makes sense, but this leaves a gap in functionality that can be improved upon
- Many shops also focus on one specific type of product, that may be available on another site. Hence, the motivation to “share” shipping fees
- All of the following apps are for shopping, but none have budget-sharing features to split shipping costs
    1. Weee!: very high shipping fees

2. Yesasia.com: may not have the intended products available
3. Amazon.com: has a large and efficient network, but it has limited stocks (hence why this is not a viable alternative for many products)
4. ShipRush: finds the cheapest shipping option for an order, but is targeted for domestic use
5. DHGate.com: may not have the intended products available
6. AliExpress: may not have the intended products available

- Budget sharing or budget-tracking apps
- Are purely for budget management and sharing, but are not related to shipping at all
- None of the following are specifically tailored to split shipping fees or place international orders
    1. Honeydue
    2. Splitwise
    3. Zelle
    4. Venmo