Software Requirements Specification

for

ThriftU

Version 1.1

Prepared by:

Group Name: Group 4

Lucas Cox	**902-793-387**	**Coxlucas2022@gmail.com**
Moustapha Niang	**904-678-803**	**mn979@msstate.edu**
Gunner Beck	**903-865-239**	**gjb144@msstate.edu**
Brenden Grant	**123-456-789**	**bjg357@msstate.edu**

CSE 4214 Section 03
TA: Tirian Judy
Prof: Dr. Nisha Pillai

1. Introduction

1.1 Purpose

The software requirements specification for ThriftU is intended to define the entire ThriftU system which is a college-themed thrifting application which allows students to sell and buy items on campus. This document will establish the system's functionality such as creating and managing user accounts, listing items for sale, purchasing items, browsing items, user messaging, and transaction management.

1.2 Document Conventions

This document uses Times New Roman font throughout the entire document. Section titles are bold, size 18 font. Subsection headings are bold, size 14 font. Section descriptions are in plain text, size 12 font. Every requirement statement will have its own priority level, with those priority levels being high, medium, and low.

1.3 Intended Audience and Reading

This document is intended for stakeholders and users of ThriftU which includes the users, the customer, and developer. This document serves as an outline for the technical requirements of the system. Section two details the project functionality and implementation details. Section three is an overview of the requirements for the application, specifically the functional and non-functional requirements.

1.4 Product Scope

ThriftU is a college-themed thrifting application which allows students to sell and exchange items via the ThriftU marketplace. ThriftU will encourage the reuse of items that would otherwise be discarded while also connecting students with similar needs and wants on campus. ThriftU will help to reduce waste, allow students to generate an income they otherwise may not have, and strengthen campus community by connecting students.

1.5 References

- Prograham Cracker SRS Document; Mutarevic et al.
- Vercel documentation: https://vercel.com/docs
- Firebase documentation: https://firebase.google.com/docs/guides
- React.js documentation: https://react.dev/learn/creating-a-react-app
- Next.js documentation: https://vercel.com/docs/frameworks/full-stack/nextjs
- Diagram creation: https://app.diagrams.net/
- Logo designing: https://design.com
- Color pallet: https://coolors.co/6a0a0a-4f1515-a8a8a8-7e7c7c-dac8ae

2. Overall Description

2.1 Product Perspective

ThriftU is a new, self-contained product. There are not any pre-existing frameworks that this product will build upon, so it must be built from the ground up for both the front-end and back-end development. The front-end will be a webpage that is user accessible and allows the user to indirectly communicate with the database. The back-end will handle the database connection to the front-end and store, modify, retrieve, and update the database.

The front-end and back-end components of ThriftU will be used and interacted with by the users of the website. The front-end web application will show the user a page thar displays a selection of items for sale by sellers that includes filter and search functionality, and admin users having a style of page with admin privileges that displays pending seller account requests and pending seller listing requests.

The back-end components of ThriftU will be a NoSQL database through Firebase that allows the storage and usage of the data in the websites. The backend will provide a backbone for database connection with the front-end and allowing management of the data and website as a whole.

2.2 Product Functions

- All users will be able to create and log into their account.
- All users will be able to modify and view account information.
- All users will be able to view a notification panel, which notifies them on specific events related to their user group.
- A buyer will be able to browse listings offered by sellers
- A buyer will be able to search for listings by name, or through tag-based descriptors.
- A buyer will be able to sort listings by condition, date listed, name, and price.
- A buyer will be able to view, add/remove the items in their cart.
- A buyer will be able to go through an order confirmation process, where they will checkout the items in their cart.
- A buyer will be able to view current and past order history.
- A buyer will be able to request a refund and cancel a pending order.
- A buyer will receive alerts when a listing they have bid on has received a higher bid from another buyer, if a bid they have made has won, or if a certain product they want is listed or back in stock.
- A seller will be able to view a seller home panel that lists information on pending orders, information on each order listing, profit information.
- An admin will be able to view, accept, and deny pending seller authorization requests.
- An admin will be able to remove or modify seller listings.
- An admin will be able to ban seller accounts.

2.3 User Classes and Characteristics

The product will have three separate user groups, a buyer group, who will be the primary end user of the platform; a seller group, who will be selling and updating listings for the buyer group; and a admin group, who will be in charge of managing sellers and the platform itself.

The Buyer is the primary end user of the web app. The platform is expected to have a large amount of people within this group. Buyers will be able to search and filter products, compare products, conduct purchases, and return products from the sellers. This group is the most important to satisfy, as they are important in generating revenue for the client.

The Seller group is the secondary end-user group of the platform. They will be selling products and receiving payments from buyers. The sellers will be able to create product listings, modify listing details, track their sales, and receive payments from sales.

The Admin group will oversee the operations of the platform with the highest privilege level. Admins will be able to approve or block seller accounts, approve or remove product listings, monitor user actions such as sales or price changes, see a data summary of all sales done on the platform, and access a dashboard that contains these features. This group is of secondary importance because it allows the client to maintain the platform's quality.

2.4 Operating Environment

ThriftU will be a web service that is accessed on the internet. There is no required hardware platform besides a device that is capable of running a modern web browser. The web application will also run on any desktop or mobile operating system, including Windows, Mac, Linux, Android, and iOS.
The front-end of the platform will be deployed using Vercel; which will also house the API and serverless functions needed to communicate with the backend services.
A graphic of this structure is shown in Appendix A, Figure 2.

2.4.1 Design:

- Language: JavaScript, HTML, CSS
- Frameworks: Node.js, Next.js
- Database: NoSQL (Firebase)
- Server: Vercel

2.4.2 Constraints:

- User must authenticate with Firebase Authentication Service at login
- User must have a modern web browser
- Developers must use NoSQL database provided by Firebase, Firebase is not compatible with SQL databases.
- There isn't much room for changes in the programming languages due to the framework systems.

3. System Features

3.1 Login and Sign-Up Page

3.1.1 Description and Priority

The user should be able to login or sign-up with a ThriftU account on pages dedicated to login and sign-up. Once the user is logged in, they should be navigated to a home landing page. This is a High Priority objective and should be completed with the utmost urgency.

Priority component Ratings (1-9): Benefit: 9, Cost: 3, Risk: 3

3.1.2 Stimulus / Response Sequences

Precondition: On login page and has account

1. User fills out email box
2. User fills out password box
3. User clicks login
4. Firebase Auth validates that email and password are correct
4. User is brought to the Store Page while logged in

Precondition: On login page and does not have account

1. User clicks Sign-Up
2. User fills out box for email
3. User fills out box for first name
4. User fills out box for last name
5. User fills out box for password
6. User fills out box for password conformation
7. User checks box to request seller account
    - 7.1 User clicks create account
    - 7.2 Admin receives a seller account request notification
    - 7.3 Account is created in the backend
8. User doesn't check request seller account box
    - 8.1 User clicks create account
    - 8.2 Account is created in the backend
9. The user is brought to the Store Page while logged in

3.1.3 Functional Requirements

SIGNUP: Allow user to create account
LOGIN: Allow user to login
AUTH: Backend login validation
SELLER-REQ: Allow user to request a seller account
READ: Pull information about items from the database
WRITE: Push information to the database

3.2 Store Page

3.2.1 Description and Priority

The user should be able to see a display of listings and have access to a search system to filter through listings. The user should be able to navigate easily to other pages through a set of UI systems and buttons. Non-Seller users should be able to add the listed items to their cart. This is a High Priority objective and should be completed with the utmost urgency.

Priority component Ratings (1-9): Benefit: 8, Cost: 4, Risk: 3

3.2.2 Stimulus / Response Sequences

Precondition: On store page and is logged in.

1. The user views some recent item listings
2. The user clicks on a category tab
3. The user clicks on an item listing
    3.1 The backend receives the relevant information about the item
    3.2 The website displays the collected information
4. The user clicks “add to cart”
    4.1. The backend adds that item listing to the user’s cart
5. The user clicks on the cart icon
6. The user is redirected to the cart page

Precondition: On store page and is not logged in

1. The user views some recent item listings
2. The user clicks on a category tab
3. The website displays item listings
4. The user clicks on an item listing
    4.1 The backend receives the relevant information about the item
    4.2 The website displays the collected information
5. The user clicks the “add to cart” button
    5.1 The backend sees that the user does not have an account
    5.2 The user receives a pop-up message saying, “Please login to add items to cart”
6. The user closes the pop-up message
7. The user clicks the login button
8. The user is taken to the login/sign-up page

3.2.3 Functional Requirements

ADD-CART: Add to cart
VIEW-LIST: View product listings
PAGE-NAV: Move between different pages
SEARCH-TAG: View items with common tags
READ: Pull information about items from the database
WRITE: Push information to the database

3.3 Cart Page

3.3.1 Description and Priority

The user should be able to view the items in their cart, and they should be able to modify the quantity or remove items from the cart. The user should be presented with a popup requesting required information for the checkout process. The user should be able to maneuver through other pages through an intuitive UI system. This is a Medium Priority objective and should be completed soon after the High Priority objectives.

Priority component Ratings (1-9): Benefit: 6, Cost: 4, Risk: 5

3.3.2 Stimulus / Response Sequences

Precondition: On cart page and logged in with items in cart.

1. The user sees all the items in their cart
2. The user selects one of the items in the cart
    - 3.1 The backend receives the relevant information about the item
    - 3.2 The website displays the collected information
4. The user deselects the item
5. The user modifies the quantity of an item in cart
    - 6.1 The backend updates the quantity of the item in that cart in the database
    - 6.2 The backend updates the cart price displayed on the website
7. The user removes an item from their cart
    - 7.1 The backend removes the instance of that item in the cart in the database
    - 7.2 The backend updates the cart price displayed on the website
8. The user selects checkout
9. The user inputs their payment information
10. The user inputs their address information
11. The user selects Review Order
12. The user is taken to the order review page

3.3.3 Functional Requirements

MOD-CART: Modify item quantity in the cart
REM-CART: Remove items from the cart
CHECKOUT: Initiate the checkout process
VIEW-CART: View items in the cart of a specific user
READ: Pull information about items from the database
WRITE: Push information to the database

3.4 Search Page

3.4.1 Description and Priority

The user should be presented with a selection of items that adhere to search and sorting parameters that the user defines: Tags, conditions, date listed, name, or price. The user should be able to add the listed items to their cart. The user should be able to maneuver through other pages through an intuitive UI system. This is a Medium Priority objective and should be completed immediately after the High Priority objectives.

Priority component Ratings (1-9): Benefit: 7, Cost: 5, Risk: 3

3.4.2 Stimulus / Response Sequences

Precondition: Buyer is on the search page and enters a valid keyword
1. Buyer enters a keyword into the search bar
2. Buyer clicks the "Search" button
3. The backend searches the database for items matching the keyword used
4. Search results found and are then displayed such as item name, item price, description of item, and a picture of the item

Precondition: Buyer is on the search page and applies sorting filters
1. Buyer selects a filter option such as price, date listed, condition, and/or tags
2. The system filters results based on the sorting category selected by the buyer
3. The item listings are then refreshed

Precondition: Buyer is on the search page and adds an item listing to their cart
1. Buyer clicks "Add to Cart" on an item listing
2. The system sends the item to the cart module
3. The cart module processes the request (reference Cart Page for full details)

3.4.3 Functional Requirements

SEARCH: Allow buyers and guests to search for items by keywords or tags
DISPLAY-SEARCH: Display search results
ERROR: Display a message if no results match the keyword
FILTER: Allow buyers and guests to filter results by the listed sorting categories
DISPLAY-SEARCH: Refresh and update item listings when filters or keywords are applied
LOGGED-IN: Allow only logged in buyers to add items from search results to their cart
ADD-CART Add items to cart
CREATE-PROMPT: Deny guest users from adding items to cart and instead prompt the guest to log in or create an account
READ: Pull information about items from the database
WRITE: Push information to the database

3.5 Admin Control Panel

3.5.1 Description and Priority

This should only be accessible to users with admin permission. The admin should be able to view pending seller authorization requests, pending item posting requests, user activity, and active seller listings. The admin should also be able to modify seller listings, ban sellers, and accept/deny any pending requests. The user should be able to maneuver through other pages through an intuitive UI system. This is a High Priority objective and should be completed with the utmost urgency.

Priority component Ratings (1-9): Benefit: 8, Cost: 6, Risk: 6

3.5.2 Stimulus / Response Sequences

Precondition: Admin accesses Control Panel
1. Admin attempts to open the Admin Control Panel
2. System verifies the admin credentials
- If the admin credentials are valid, the Admin Control Panel opens
- If the admin credentials are invalid, access is denied and an error message is displayed

Precondition: Admin reviews a pending seller request
1. Admin selects “Pending Seller Requests”
2. System displays a list of pending seller requests
2. Admin click “Approve” or “Deny” on a request
3. System notifies the user of their seller status

Precondition: Admin bans a seller
1. Admin selects a seller’s account
2. Admin clicks “Ban Seller”
3. A confirmation message is displayed asking if the admin is sure they want to ban the seller’s account
- If the admin selects “No” when prompted to confirm the ban on the seller’s account the system cancels the action and the seller’s account remains intact
- If the admin selects “Yes” when prompted to confirm the ban on the seller’s account the seller account will become deactivated

3.5.3 Functional Requirements

ADMIN-VALID: Only users with an admin account can access the Admin Control Panel
DISPLAY-REQ: Display pending seller requests and allow admins to approve/deny

BAN: Allow admins to ban sellers and deactivate their accounts

READ: Pull information about items from the database
WRITE: Push information to the database

3.6 Notification Panel

3.6.1 Description and Priority

This will be available to all users, but it will differ in content between them. The user should be able to see all notifications that are tied to that user: Admins will see the amount of pending requests, Sellers will see approvals or denials of their listings and pending purchases, buyers will see any involuntary changes to their cart. The users should be able to clear the notification by marking them as seen. The user should be able to maneuver through other pages through an intuitive UI system. This is a Low Priority objective and should be completed after all other objectives are met.

Priority component Ratings (1-9): Benefit: 5, Cost: 3, Risk: 1

3.6.2 Stimulus / Response Sequences

Precondition: User views the Notification Panel
1. User navigates to the Notification Panel
2. All notifications are displayed depending on the user's role
- Admin: seller pending requests
- Seller: item listing purchases, item listing taken down by admin
- Buyer: refund updates, confirmation of purchase
3. If no notifications exist, a message will be displayed stating that there are no notifications

3.6.3 Functional Requirements

DISPLAY-NOTIF: Display notifications depending upon the user's role
DISPLAY-NOTIF: Display the appropriate message when a user has no notifications
PAGE-NAV: Integrate the Notification Panel with other pages
READ: Pull information about items from the database
WRITE: Push information to the database

3.7 Seller Hub Page

3.7.1 Description and Priority

Only users with seller or admin permissions should be able to view this page. The user should be able to see all of their active listings and all of their listings pending review. The user should also be able to see pending payments and orders for individual listings. The user should be able to access the shipping data for the individual who placed the order, so the seller can ship the orders to the buyer. The user should be able to maneuver through other pages through an intuitive UI system. This is a High Priority objective and should be completed after the other High Priority objectives have been completed.

Priority component Ratings (1-9): Benefit: 8, Cost: 6, Risk: 5

3.7.2 Stimulus / Response Sequences

Prerequisite: The user is signed in as a seller and is validated.

1. The user selects "Profile" from the navigation bar.
2. The seller's profile data (store name, description, products, ratings, etc.) is retrieved by the backend.
3. The Seller Hub is displayed by the system.

Modifying Profile Information
4. The user selects Edit Profile.
5. User-updated fields (such as contact details, store name, and description)
6. The user selects "Save."
7. The database's seller profile is updated and validated by the backend.
8. The system displays the changed profile and verifies success.

Including a Product
9. The user selects "Add Product."
10. The user enters the product's name, description, cost, quantity, and pictures.
11. The user selects Submit.
12. The new product is added to the seller inventory after the backend verifies the inputs.
13. The product appears in the profile listing when the system displays confirmation.

3.7.3 Functional Requirements

SELLER-VALID: Permit the seller to access their profile data
MOD-ITEM: Permit seller to amend store information
ADD-ITEM: Permit the vendor to add item listings
ITEM-VERIFY: Backend verification of product modifications and seller profiles
READ: Pull information about items from the database
WRITE: Push information to the database

3.8 Account Management Page

3.8.1 Description and Priority

The user should be able to view and edit all of their account information and view their current and past orders. The user should be able to see the content of their past orders and request a refund if the order is still within the refund window. The user should be able to maneuver through other pages through an intuitive UI system. This is a Medium Priority objective and should be completed soon after the High Priority objectives.

Priority component Ratings (1-9): Benefit: 5, Cost: 3, Risk: 6

3.8.2 Stimulus / Response Sequences

Prerequisite: The user is signed in.
1. The "Account Settings" button is clicked.
2. Information about stored accounts is retrieved by the backend.
3. The Account Management Page is displayed by the system.

Changing Your Password or Email
4. The user selects "Edit" next to the password or email.
5. The user inputs new data
6. The user selects "Save."
    6.1 The backend verifies the input
    6.2 Account record updates in the backend
    6.3 A success message is displayed by the system.

Address Updating
7. The user selects "Edit Address."
8. The user enters their new address.
9. The user selects "Save."
    9.1 Address format is verified by the backend.
    9.2 The backend stores the most recent address
    9.3 The system updates with the new address.

Modifying the Payment Method
10. The "Payment Options" button is clicked.
11. The user modifies the payment method
12. The user selects "Save."
    12.1 The backend verifies the technique
    12.2 Database updates in the backend
    12.3 Success is confirmed by the system

3.8.3 Functional Requirements

MOD-ACCOUNT: Permit the user to update their account information
MOD-ADDRESS: Permit the user to amend their shipping details
MOD-PAYMENT: Permit the user to modify their payment options
VERIFY-MOD: Verification of account updates in the backend
READ: Pull information about items from the database
WRITE: Push information to the database

3.9 Order Confirmation Page

3.9.1 Description and Priority

The user should be able to review their order before final checkout. All the information regarding checkout should be seen and editable: Cart, payment method, mailing address, and overall cost. The user should be able to maneuver through other pages through an intuitive UI system. This is a Low Priority objective and should be completed after the Medium Priority objectives are met.

Priority component Ratings (1-9): Benefit: 5, Cost: 4, Risk: 4

3.9.2 Stimulus / Response Sequences

Prerequisite: The user adds things to their cart and checks them out.
1. The user selects "Checkout."
2. Total cost (products + shipping + tax) is calculated by the backend.
3. The order review page is displayed by the system.

Verifying the Order
4. The user chooses the payment method and shipment address.
5. The "Place Order" button is clicked.
    - 5.1 Payment is verified by the backend
    - 5.2 Transaction processing in the backend
    - 5.3 Order records are generated by the backend.
    - 5.4 Inventory is updated by the backend
    - 5.5 The backend email confirmation.

Confirmation Displaying
6. The Order Confirmation Page is displayed by the system with**:**
    - 6.1 The order number
    - 6.2 A receipt with items
    - 6.3 The anticipated date of delivery

3.9.3 Functional Requirements

ORDER: User order confirmation.
CREATE-ORDER: Create a new order
MOD-ITEM: Update inventory
DISPLAY-ORDER: Show the details of the order confirmation
READ: Pull information about items from the database
WRITE: Push information to the database

4. Other Nonfunctional Requirements

4.1 Performance Requirements

- This website needs a secure connection to the internet and an up-to-date web browser to allow for a secure and fast connection to the ThriftU website to function at full capacity.
- All page changes and search functions should be done in less than 2 seconds.
- All database updates and information changes should take less than 5 seconds to update.
- All charges in website window scaling should take less than 5 seconds.

4.2 Safety Requirements

Access to the database used in production will be restricted to authorized users. Database backup will not be available for use, since that is not included within the Firebase Free Tier. It is possible that a bad actor can gain unauthorized access to information and gain access to PII. To prevent this, there will strict programming and database rules to only allow user data to be read on a need-to-know basis.

It is likely that outages from one of the services used by the product will make the product unusable, thus resulting in a loss of profit. To prevent this, the product will be built on reliable services managed by reputable companies, including Vercel and Google.

4.3 Security Requirements

4.3.1 Account Security

All users will be required to register an account and login using an email address; duplicate email addresses will be prevented to protect against abuse. Passwords for user accounts will follow the industry recommended password requirements including: a length of at least 8 characters, at least one uppercase and one lowercase letter, and at least 1 number or special character. Furthermore, logins will be handled by the Firebase Authentication service.

4.3.2 Storage Security

Password storage will be hashed and salted by Google before storage in the Firebase Authentication database. Since the platform uses Cloud Firestore for the NoSQL database, it will already be encrypted by Google.

4.3.3 System Security

To be in compliance with industry standards, API keys and other environment variables will also be protected from public access. Along with this, user inputs will be sanitized to prevent NoSQL injection attacks.

4.4 Software Quality Attributes

4.4.1 Availability

- The Website should be available and responsive, unless the user loses access to the internet or the web browser used by the user is down.

4.4.1 Dependability

- The product should not differ so much across different scenarios as to be unpredictable.
- The product should be reliable if the user is performing normal functions, the product does not break, cause a data leak, or become inaccessible.

4.4.1 Usability

- The product should be easy and intuitive to use with no complex systems requiring any knowledge of programming or systems for use.
- The product should be effective and efficient at its specified purpose, and it should fill its internet niche while being unique in its design.

4.4.1 Flexibility

- The product should be able to support a wide variety of web browsers: Chrome, Firefox, Opera, Safari, etc.
- The product should be able to support multiple different operating systems.
- The product should be able to host many concurrent users and web traffic without losing responsiveness.

5. Other Requirements

- The database must be able to pull and update large amounts of information efferently
- The database is going to be in NoSQL, but still must be easily accessible through code
- The users should be able to easily read and understand all the information on the website
- The database should be flexible and scalable to fit a growing consumer base
- The website should be accessible through the main internet and not just through private connections or specific/complex links
- The website's domain must be easy to find and easy to remember
- The website should stick to the same color pallet and scheme throughout
- The users should only be able to access what they are permitted to access, even if they type in a specific domain related to a page that they do not have access to.
- There should be a page dedicated to access denied and 404 errors

6. Appendix A: Glossary Terms

API - Communication between various software components is made possible by APIs, or application programming interfaces.

Buyer - A person who shops on ThriftU and buys things there.

Seller - An individual who posts goods for sale on ThriftU.

Admin - A user with specific rights to oversee accounts, listings, and sellers.

GUI - Graphical User Interface, or GUI, is the visual interface that users use.

UI – User interface; anything that the user can directly interact with.

Firebase - ThriftU's database, hosting, and authentication are provided by this Google-hosted backend service.

NoSQL - A database type that uses collections or documents rather than tables to store data, such as Firebase Firestore.

SRS - The document known as the Software Requirements Specification (SRS) outlines both functional and non-functional requirements.

7. Appendix B: Analysis Models

Figure 1:

Figure 2:

Figure 3:

Appendix C: To Be Determined List

1. Stands for maximum number of concurrent users supported (TBD).
2. Gateways for payments that are supported (TBD).
3. predetermined reporting/flagging thresholds prior to administrative action (TBD).
4. Officially supported mobile operating system versions (TBD).
5. Processing time for refunds (TBD).
6. Mobile device support (TBD).
7. Allow buyer-seller communication via direct messaging (TBD).