Software Requirements Specification

for

Last Roll

Version 1.0

Prepared by:

Jack Cook	**jwc602**	**jackcook99@tuta.com**
James Harden	**jah1479**	**james.harden.jr@live.com**
Tess Harris	**emh585**	**tessharris413@protonmail.com**
Grayson Smith	**gbs105**	**gbaileysmith@gmail.com**

Instructor: Nisha Pillai
Course: CSE 6214

September 10, 2025

1. Introduction

1.1 Purpose

Last Roll is an online marketplace for tabletop role-playing game dice. Users will be able to buy and sell dice sets and sort listings based on color, size, material, etc. easily through a simple tag filtering system with limited and definite metrics. The goal is to design and implement a creative and immersive user experience that's still simple and safe to use.

This SRS will describe site functionality as a whole. In short, there will be functionality for logging in and creating accounts, creating and deleting listings, buying and selling dice, and seeing past orders and receipts (both incoming and outgoing). An overall description of the product will be in Section 2, a list and description of major system features in Section 3, and further nonfunctional requirements in Section 4.

1.2 Document Conventions

The terms Customer and User (interchangeably) refer to both Buyers and Sellers of dice, while Admin or Administrator refers to accounts with administrative permissions over both and are not included in the class of User.

1.3 Intended Audience and Reading

This document is intended for developers, project managers, and administrative staff. We aim to maintain a seamless user experience that does not require knowledge of software development to understand this document.

An overall description of the product will be in Section 2, a list and description of major system features in Section 3, and further nonfunctional requirements in Section 4. It is recommended for any project managers, developers, and documentation writers to read the document in its entirety and use it as reference during the lifespan of the project. Any marketing staff, potential users, and testers should not need to read beyond Section 2: Overall Description.

1.4 Product Scope

The site will be able to host buy and sell orders for dice based on tagged specifications. This will facilitate distribution of dice to buyers and the processing of transactions resulting from this. The website owner will take (roughly) a 10% commission from each order placed on the website, and the site itself will keep track of seller payouts, order receipts, and fulfillment, creating an easy platform for new and potential sellers.

1.5 References

We will be referencing articles from this guide toward making our project accessible to all users: *https://consciousstyleguide.com/ability-disability/*.

2. Overall Description

2.1 Product Perspective

Last Roll is a self-contained e-commerce marketplace that allows users to buy and sell TTRPG dice. Buyers are provided with the means to add unique and interesting dice to their collections. Sellers are provided with a centralized location to show and sell their products to an interested audience. This website will be managed by our new company (Last Roll, LLC), which will serve as facilitator and moderator of transactions. While sites exist to purchase TTRPG dice, no site provides a marketplace with a focus on collaboration and user-to-user transactions. With this in mind, it should be stated that Last Roll does not replace an existing system.

The website's core functionality is distributed to several different software components:

- The frontend faces the userbase, and is accessible through standard web browsers. "Buyer" users can view and purchase listings posted by "seller" users. Sellers can manage their listings via a seller dashboard. Administrators can monitor site usage through an administrative dashboard.
- The backend application server organizes user profiles, processes orders, and updates listing information.
- The database subsystem stores user and listing information, such as a user's credentials or a listing's properties.

A few external services provide full functionality:

- The payment gateway allows buyers to securely pay for their items, and sellers to securely receive the profit from sale of their products.
- The shipping API generates shipping labels for sellers to use and allows buyers and sellers to track the shipping progress and delivery of their order..
- The email notification service provides updates to users, such as shipment notifications or updates to a particular listing. It additionally enables a second layer of authentication, to be used for security functions and password resets.
- An administrative dashboard allows user accounts and listings to be managed by Last Roll staff.

2.2 Product Functions

Buyer Functions

All users are "buyers" by default. Therefore, all users will be able to:

- Create and manage their user account.
- Browse for listings.
- Search for a listing based on descriptive tags, such as "Blue", "Resin", or "D20".
- View listing details, such as description, price, and availability.
- Add items to a "shopping cart" and update cart as needed.
- Place orders and complete checkout.
- Make secure payments via a payment gateway.
- View their order history.
- Track packages.
- Receive notifications through email.

Seller Functions

Users approved by Last Roll administrators to sell items will be given access to a seller dashboard. Here, users will be able to:

- Create listings, including a product photo and the appropriate descriptive tag.
- Modify or delete listings.
- View, update, fulfill, or cancel orders.
- Securely deposit their earnings to a connected bank account.

Administrator Functions
Last Roll administrators will be able to:

- Manage registered users.
- Approve Seller Accounts
- Remove listings (to prevent spam and fraud).
- Manage and edit tags on a product
- Monitor buyer, seller, and admin activity.

System Functions
The system will maintain confidentiality and accuracy by:

- Securely authenticating registered users.
- Updating listing database.
- Integrating with shipping APIs for label generation and tracking.
- Integrating with external payment systems for item purchases and seller deposits.
- Sending automated email notifications.
- Ensuring data backup and recovery.
- Enforcing security measures such as encryption and access control.

2.3 User Classes and Characteristics

Buyer *(Primary)*

- Role: purchases products, manages personal account.
- Frequency: occasional to frequent.
- Functions Used: account creation, login/logout, browsing catalog, cart management, checkout, payment, order tracking.
- Technical Expertise: basic Internet and shopping familiarity.
- Privilege Level: access to only their account data and order history.
- Importance: High

Seller *(Primary)*

- Role: sells products, manages personal account.
- Frequency: occasional to frequent.
- Functions Used: account creation, login/logout, listing creation, order management, deposit, shipping label generation.
- Technical Expertise: basic Internet and selling familiarity.
- Privilege Level: access to only their listings and outstanding orders.
- Importance: High

Guest *(Secondary)*

- Role: browses catalog without logging in.
- Frequency: occasional.
- Functions Used: view products.
- Technical Expertise: Minimal to none.

- Privilege Level: None
- Importance: Medium (helps attract new customers).

Administrator *(Primary)*
- Role: manages user accounts and listings.
- Frequency: daily.
- Functions Used: user account management, listing management.
- Technical Expertise: requires training.
- Privilege Level: full access to user accounts and listings.
- Importance: Critical

2.4 Operating Environment

The website's servers and services will run on an on-premises Ubuntu 24.04 LTS server (IP: 70.250.151.1) with sufficient CPU, memory, and storage to handle expected traffic. The HTTP server will use the Apache framework. The backend will be written in Python/Django, with a MySQL relational database system. The frontend will be primarily written in HTML/CSS, with some Java as deemed necessary for accessibility and aesthetics.

The payment gateway will emulate PayPal. The shipping API will emulate Easypost. Email notifications will be handled through SMTP.

End users will access the system through personal devices, including desktop computers, laptops, tablets, and smart phones. Although the user's device must be capable of running a modern Internet browser, the application is platform-independent.

2.5 Design and Implementation Constraints

We have not secured permission to operate yet, which limits our abilities to fully integrate the external services. We will emulate PayPal and Easypost's APIs to streamline deployment of these services. For now, users will be able load their accounts via gift certificates.

The development team's previous experience necessitates the use of specific programming environments to ensure a quality product is delivered on time. Therefore, no changes will be made to the framework unless absolutely necessary.

Our company is self-hosting the site, and we will be responsible for maintaining the software.

3. System Features

3.1 Buyer Account Features

The following features will be available for all user accounts. Every account is a buyer account by default.

3.1.1 Create a new buyer account. (HIGH)

1. User navigates to the top right of the “Home” page and clicks to open the navigation submenu.
2. In the submenu, user clicks the “Create Account” button.
3. Application server loads the “Account Creation” page.
4. User enters a valid desired username into the “Username” textbox.
    4.1. Valid username contains only capital and lowercase letters and digits 0-9.
        4.1.1. If user enters invalid username, an error message is displayed: “Username can only contain letters and numbers.”
    4.2. Application server queries database to check if desired username is already registered.
        4.2.1. If username is unique, a green check is displayed.
        4.2.2. If username is already registered, a red X is displayed with the error message “Username taken. Please choose another username.”
5. User enters a valid desired password into the “Password” text box.
    5.1. The password text is redacted.
    5.2. Valid password is at least 8 characters long and contains both a capital letter and a number.
        5.2.1. If user enters a password with an invalid length, an error message is displayed: “Password must be at least 8 characters long.”
        5.2.2. If user enters a password without a capital letter and a number, an error message is displayed: “Password must contain a capital letter and a number.”
6. User enters a valid email address into the “Email Address” text box.
    6.1. Valid email address follows convention [a-zA-z0-9]@[a-zA-z0-9].[a-z]
        6.1.1. If email address is invalid, an error message is displayed: “Invalid email address.”
7. User enters the same valid email address into the “Confirm Email Address” text box.
    7.1. If email address does not match the email address in the “Email Address” text box, an error message is displayed: “Email addresses do not match.”
8. User clicks the “Sign Up” button.
    8.1. Application server sends account credentials to an Account Staging database, where they are temporarily stored (~72hrs) until email confirmation.
    8.2. The application server generates and hosts a unique confirmation link.
    8.3. The email API sends a message containing the confirmation link to the entered email address.
    8.4. A message is displayed: “Please confirm your account through the link sent to your email within 72 hours.”
9. The user navigates to their personal email account, opens the email sent by the email API, and clicks the confirmation link.
    9.1. The application server records the date and time the account was verified.
    9.2. The verification date and time is appended to the account’s entry in the Account Staging database.
    9.3. The application server moves the full account credentials to the Accounts database.
    9.4. A message is displayed: “Thank you for confirming your account.”
10. The application server redirects the user to the “Log In” page.

3.1.2 Log into an existing buyer account. (HIGH)

1. User has an existing account (via 3.1.1).
2. User navigates to the top right of the “Home” page and clicks to open the navigation submenu.
3. User clicks the “Log In” button.

4. Application server loads the “Log In” page.
5. User enters their username into the “Username” field.
6. User enters their password into the “Password” field.
    6.1. The password text is redacted.
7. User clicks the “Log in” button.
8. Application server queries the Accounts database and checks if the provided password matches the stored password for the provided username.
    8.1. If passwords match, user is redirected to the “Home” page in the “logged in” state.
    8.2. If passwords do not match, an error message is displayed: “Invalid credentials.”

3.1.3 Log out. (HIGH)

1. User is logged in (via 3.1.2).
2. User navigates to the top right of the “Home” page and clicks to open the navigation submenu.
3. User clicks the “Log Out” button.
4. Application server sets the user’s status in the user database to “logged out”.
5. A message is displayed: “You have been logged out successfully.”
    5.1. The user is redirected to the “Home” page in the “logged out” state.

3.1.4 View account details. (MEDIUM)

1. User is logged in (via 3.1.2).
2. User navigates to the top right of the “Home” page and clicks to open the navigation submenu.
3. In the submenu, user clicks the “My Account” button.
4. Application server loads the “My Account” page.
5. The user’s username, date of account creation, and outstanding orders are displayed.

3.1.5 Delete account. (MEDIUM)

1. User views account details (via 3.1.4).
2. User clicks the “Delete Account” link.
3. Application server loads the “Delete Account” page.
4. User enters their password into the “Password” field.
    4.1. The password text is redacted.
5. User clicks the agreement check box, which is displayed with a message: “I agree to permanently delete my account, and I understand this action is irreversible.”
6. User clicks the “Delete my account” button.
    6.1. If agreement check box is not selected, an error message is displayed: “Please agree to the statement.”
    6.2. Application server queries Accounts database.
        6.2.1. If entered password does not match stored password, an error message is displayed: “Password is incorrect.”
        6.2.2. If passwords match, application server removes user account information from Accounts database.
7. A message is displayed in a pop-up window: “Your account has been deleted successfully.”
8. User is redirected to “Home” page in the “logged out” state.

3.1.x Browse for listings. (HIGH)

1. Priority medium This will allow users to browse the dice available for sale, using cards and pages.
2. Users can select a browse button that will display the first page of most recent listings
3. Users should be able to click from a list of commonly used tags to see dice matching these criteria.
4. This will require an HTTPS Server and a card and page results display system

3.1.x Search for listings based on tags. (HIGH)

1. Users need to be able to search for specific dice using tag systems for specific properties
2. The user will type the tags they want into the search box.
   2.1. The box will produce autofill results below the text box.
   2.2. Users can click autofill results to fill them in the box.
3. User will push the enter key or the go button once he has finished typing in tags.
4. System will run a query and return all results that match the criteria
5. System will then display results as cards and pages.
6. This will require a database for the available dice listings, as well as a tag based search engine. this will require a page and card based presentation of the search results and a HTTPS Server to host the actual website

3.1.x View item listing details. (HIGH)

1. Priority high This is so users can view listing details of dice they may want to purchase, giving them a detailed dossier of the dice they have selected. It should include characteristics of description price and availability.
2. User will click on a dice listing as displayed from section 3.4
3. System will redirect user to seperate page containing detailed information for the listing.
4. System will pull data from the dice database to populate the page.
5. This will require an HTTPS server and a dice listing database.

3.1.x Add item to cart. (HIGH)

1. Users should be able to add items to a shopping cart vector to accumulate the dice they wish to purchase. THe shopping cart should be persistent through cookies to allow it to be moved from page to page without clearing.
2. User will click an add to cart button on the card for a desired dice.
3. System will add the dice id to a shopping cart vector.
4. System will save this vector as cookies in users browser.
5. The system will verify that every id code is still valid each time the cart is viewed
6. This will require an HTTPS Server, Website cookies for the shopping cart the dice listing database and the shopping cart vector

3.1.x View cart.

1. Users will click on the cart icon
2. The system will redirect them to a page
3. The system will source the listing info for each id number recorded in the shopping cart cookie
4. The system will query the dice database for each id number and use the results to generate the listings on the page

3.1.x Remove item from cart.

1. The user will click small garbage icon next to an item in the cart,
2. The system will remove that item from the page and its ID from the shopping cart vector

3.1.x Check out. (HIGH)

1. User navigates to their cart (via 3.1.x).
2. User clicks the “Check Out” button.
3. Application server loads the “Check Out” page.
    3.1. Application server queries the Carts database for the contents of the user’s cart.
    3.2. Cart contents are displayed.
    3.3. Total cost of cart contents, plus appropriate tax, is calculated and displayed.
    3.4. Application server queries the Accounts database for saved address information.
        3.4.1. If user has a saved address, information is auto-filled into the appropriate fields.
4. User enters their first name in the “First Name” field.
5. User enters their last name in the “Last Name” field.
6. User enters their telephone number in the “Phone Number” field.
7. User enters the first line of a valid shipping address in the “Shipping Address Line 1” field.
    7.1. A valid shipping address begins with a number 0-9.
    7.2. User enters second line in shipping address in the “Shipping Address Line 2” field, if necessary.
    7.3. If user provides an invalid shipping address, or clicks away from field without entering an address, an error message is displayed: “Must provide a valid shipping address.”
8. User enters a valid city in the “Shipping Address City” field.
    8.1. A valid city contains only capital and lowercase letters.
    8.2. If user provides an invalid city, an error message is displayed: “Must provide a valid city.”
9. User chooses a state from the “Shipping Address State” dropdown menu.
10. If user’s shipping address is their billing address, user clicks “Use shipping address as billing address” checkbox.
    10.1. Otherwise, user enters their billing address into the “Billing Address Street Line 1”, “Billing Address Street Line 2” Billing Address City”, and “Billing Address State” fields as described in 3.1.7-3.1.9.
11. If desired, user checks “Save My Address Information” box.
    11.1. Upon submission, the application server hashes the address information and sends it to the Accounts database.
12. User enters their credit card number in the “Credit Card Number” field.
13. User enters a valid first name in the “First Name On Card” field.
    13.1. A valid name contains only capital and lowercase letters.
    13.2. If user provides invalid name, an error message is displayed: “Must provide a valid name.”
14. User enters a valid last name in the “Last Name On Card” field.
    14.1. A valid name contains only capital and lowercase letters.
    14.2. If user provides invalid name, an error message is displayed: “Must provide a valid name.”
15. User enters three-digit security code in the “Security Code” field.
    15.1. If security code is invalid, an error message is displayed: “Must provide a valid security code.”

16. User clicks the “Check Out” button.
17. If “Save Address Information” button is checked, application server hashes user address and adds it to the Accounts database.
18. Application server removes item listings from Listings database.
19. Application server queries payment gateway with appropriate payment information.
20. Application server adds order to Orders database.
21. Application server loads “Thank you for your order” page.
22. Email API sends purchase notification email to buyer.
23. Email API sends sale notification email to seller.

3.1.x View order history.

1. Priority medium users need to be able to view their order history so they don't try and make multiple orders for the same or similar dice by accident. This will also help them determine what dice they may still need to order
2. Users will click a button on their account screen
3. The system will redirect them to a list page that shows a list of their previous orders sourced from a orders database
4. users will select an order from the list to be brought to a detailed page of the order.
5. users should be able to click a button to be taken to a page with a list of detailed overviews of their previous orders that can be clicked on for more information.
6. This will require access to the dice and account databases, potentially its own database from previous orders.

3.1.x View tracking information.

1. Priority medium users need to be able to track the status of their packages to see when to be expecting them.
2. Users will click a button on the orders list page or the detailed order page for an order
3. The system will redirec them to a page containing the tracking status as queried through the API
4. users should be able to push a button on their previous orders to be taken to a page where they can see their order status.
5. this will require an HTTPS Server and access to the accounts and orders databases and the shipping API

3.2 Seller Account Features

3.2.x Create seller account.

11. User navigates to the top right of the homepage and clicks on the account button.
12. User clicks the “Create Account as seller” button.
13. Application server loads the account creation page.
14. User enters a valid desired username into the “Username” textbox.
    - 14.1. Valid username contains only capital and lowercase letters and digits 0-9.
        - 14.1.1. If user enters invalid username, an error message is displayed: “Username can only contain letters and numbers.”o
    - 14.2. Application server queries database to check if desired username is already registered.
        - 14.2.1. If username is unique, a green check is displayed.

14.2.2. If username is already registered, a red X is displayed with the error message “Username taken. Please choose another username.”

15. User enters a valid desired password into the “Password” text box.
    15.1. Valid password is at least 8 characters long and contains both a capital letter and a number.
        15.1.1. If user enters a password with an invalid length, an error message is displayed: “Password must be at least 8 characters long.”
        15.1.2. If user enters a password without a capital letter and a number, an error message is displayed: “Password must contain a capital letter and a number.”
16. User enters a valid email address into the “Email Address” text box.
    16.1. Valid email address follows convention [a-zA-z0-9]@[a-zA-z0-9].[a-z]
        16.1.1. If email address is invalid, an error message is displayed: “Invalid email address.”
17. User enters the same valid email address into the “Confirm Email Address” text box.
    17.1. If email address does not match the email address entered above, an error message is displayed: “Email addresses do not match.”
18. User clicks the “Sign Up” button.
    18.1. Application server sends account credentials to an Account Staging database, where they are temporarily stored (~72hrs) until email confirmation.
    18.2. The application server generates and hosts a unique confirmation link.
    18.3. The email API sends a message containing the confirmation link to the entered email address.
    18.4. A message is displayed: “Please confirm your account through the link sent to your email within 72 hours.”
19. The user navigates to their personal email account, opens the email sent by the email API, and clicks the confirmation link.
    19.1. The application server records the date and time the account was verified.
    19.2. The verification date and time is appended to the account’s entry in the Account Staging database and a flag is set for it to be approved by an admin.
    19.3. A message is displayed: “Thank you for confirming your account. An admin will verify your account soon”\
20. An admin must verify the new account from the admin dashboard.
21. The user is redirected to a login page.

3.2.x Log in.

1. User navigates to the top right of the homepage and clicks on the account button.
2. User clicks the “Sign In As Seller” button.
3. Application server loads the “Seller Log In” page.
4. User enters their username into the “Username” field.
5. User enters their password into the “Password” field.
    a. The password text is redacted with dots.
6. User clicks the “Log in as Seller” button.
7. Application server queries Accounts database and checks if the provided password matches the stored password for the provided username.
    a. If passwords match, user is redirected to the “Seller dashboard” page.
    b. If passwords do not match, an error message is displayed: “Invalid credentials.”

3.2.x Log out.

1. User navigates to the top right of the Seller dashboard and clicks on the account button.
2. User clicks logout.
3. Application server logs the user out.
4. The user is logged out.

3.2.x Create listing.

1. The user will click the create listing button from the seller dashboard
2. The system will redirect them to a page with a form to fill out contain an image upload box and several textboxes
3. The textboxes will be for name and attributes, one textbox will have an autofill for for tags.
4. The user will click the button saying “post listing”
5. The system will add the listing to the database with an unverified flag
6. The system will display a dialog box that says ‘thank you for posting the listing, an admin will verify it soon”
7. The system will redirect the user back to the seller dashboard

3.2.x Modify listing.

1. The user will click on the edit icon next to a particular listing that they wish to edit
2. The system will redirect them to the update listing form page, it is identical to the create listing page, with the exception of the header and labels
3. The system will query the database for the listing information and autofill that information into the form
4. The user will modify the data in the form boxes as needed.
5. The user will click the button that says “update listing”
6. The system will add the listing to the database with an unverified flag
7. The system will display a dialog box that says ‘thank you for posting the listing, an admin will verify it soon”
8. The system will redirect the user back to the seller dashboard

3.2.x Delete listing.

1. The seller will click the delete icon next to a listing they wish to delete.
2. The system will produce a confirmation box asking them to confirm that they wish to delete the listing with two buttons
    a. If the user clicks no, the confirmation box closes and nothing happens beyond that.
    b. If the user clicks yes, then the system removes the listing from the dice catalog database.

3.2.x Add deposit method.

1. The user will select “add deposit method” from the seller dashboard
2. The system will redirect them to portal page to use the Paypal OAuth API
3. The user will complete the Paypal OAuth
4. The system will redirect them back to the seller dashboard

3.2.x View order history.

1. The seller will click on the view order history button on the seller dashboard
2. The system will redirect the user to a list page
3. The system will query the order database for the sellers orders
4. The system will populate the list page with the queries results
5. The user will click on an order item in the list
6. The system will redirect them to a details page
7. The system will query the orders database and populate the page with the detailed information for the selected order

3.2.x View outstanding orders.

1. The user will click the filter drop down on the order history page.
2. The user will select only “show outstanding orders” and click filter
3. The system will requery the orders database and repopulate the list with only outstanding orders

3.2.x Print shipping label for orders.

1. From the order history page, or the order details page, the user will select the print icon next to the desired listing
2. The system will call the shipping API
3. The shipping API will generate a shipping label and return the rendered file to the system
4. The system will display the rendered file in the browsers print preview window.
5. The user will use the browsers print preview window to print the shipping label

3.2.x Mark order as shipped.

1. From the order history page, or the order details page, the user will select the complete icon next to the desired listing.
2. The system will update the flag in the orders database to indicate the order has been marked as shipped.

3.2.x Cancel outstanding order.

1. From the order history page, or the order details page, the user will select the delete icon next to the desired listing.
2. The system will produce a confirmation box asking them to confirm that they wish to delete the order with two buttons
    a. If the user clicks no, the confirmation box closes and nothing happens beyond that.
    b. If the user clicks yes, then the system removes the order from the orders database.

3.2.x View sales information.

1. From the sellers dashboard the user will click the view sales information button
2. The system will redirect them to the sales information page
3. The system will queries the orders database for information
4. The system will use this information to compute sales information
5. The system will display this information on the sales information page

3.3 Administrator Account Features

3.3.x Log in.

9. User navigates to the admin login page through dedicated URL
10. Application server loads the “Admin Log In” page.
11. User enters their username into the “Username” field.
12. User enters their password into the “Password” field.
    12.1. The password text is redacted with dots.
13. User clicks the “Log in” button.
14. Application server queries Accounts database and checks if the provided password matches the stored password for the provided username.
    14.1. If passwords match, user is redirected to the “Admin dashboard” page.
    14.2. If passwords do not match, an error message is displayed: “Invalid credentials.”

3.3.x Log out.

1. User navigates to the top right of the admin dashboard and clicks on the account button.
2. User clicks logout.
3. Application server logs the user out.
4. The user is logged out.

3.3.x Create new administrator account.

1. User navigates to the top right of the admin dashboard and clicks on the account button.
2. User clicks the “Create Account” button.
3. Application server loads the Admin account creation page.
4. User enters a valid desired username into the “Username” textbox.
    a. Valid username contains only capital and lowercase letters and digits 0-9.
        i. If user enters invalid username, an error message is displayed: “Username can only contain letters and numbers.”o
    b. Application server queries database to check if desired username is already registered.
        i. If username is unique, a green check is displayed.
        ii. If username is already registered, a red X is displayed with the error message “Username taken. Please choose another username.”
5. User enters a valid desired password into the “Password” text box.
    a. Valid password is at least 8 characters long and contains both a capital letter and a number.
        i. If user enters a password with an invalid length, an error message is displayed: “Password must be at least 8 characters long.”
        ii. If user enters a password without a capital letter and a number, an error message is displayed: “Password must contain a capital letter and a number.”
6. User enters a valid email address into the “Email Address” text box.
    a. Valid email address follows convention [a-zA-z0-9]@[a-zA-z0-9].[a-z]
        i. If email address is invalid, an error message is displayed: “Invalid email address.”
7. User enters the same valid email address into the “Confirm Email Address” text box.
    a. If email address does not match the email address entered above, an error message is displayed: “Email addresses do not match.”

8. User clicks the “Sign Up” button.
    a. Application server sends account credentials to an Account Staging database, where they are temporarily stored (~72hrs) until email confirmation.
    b. The application server generates and hosts a unique confirmation link.
    c. The email API sends a message containing the confirmation link to the entered email address.
    d. A message is displayed: “Please confirm your account through the link sent to your email within 72 hours.”
9. The user navigates to their personal email account, opens the email sent by the email API, and clicks the confirmation link.
    a. The application server records the date and time the account was verified.
    b. The verification date and time is appended to the account’s entry in the Account Staging database.
    c. The application server moves the full account credentials to the Accounts database with 2 unverified flags.
    d. A message is displayed: “Thank you for confirming your account. 2 other admins must approve this account before it can be used”
10. The user is redirected to the admin dashboard page.

3.3.x Refund users.

1. From the admin dashboard, the admin will select a user from a user list
2. The system will direct the admin to the detailed user info page
3. The system will query the account database for the user
4. The system will populate the page with the detailed user info
5. The user will select the edit icon next to the USD balance
6. The system reveal an additional text box below the USD balance
7. The admin will type in the refunded balance and click the submit button
8. The system will update the value in the database and refresh the page

3.3.x Modify listings.

1. From the admin dashboard select the desired listing from the list
2. The system will redirect the admin to the admin level listing details page
3. The system will populate the page with the information queried from the dice catalog database
4. The admin will click the edit icon
5. The system will redirect the admin to the edit listings page
6. The system will autofill the text boxes on the edit listing page
7. The admin will modify the details as needed and click submit
8. The system will update the listing details and redirect the admin to the admin level list

3.3.x Remove listings.

1. From the admin dashboard. The admin will click the delete icon on the desired listing, this button can also be pushed from the listing details page.
2. The system will produce a confirmation box asking them to confirm that they wish to delete the listing with two buttons
    a. If the user clicks no, the confirmation box closes and nothing happens beyond that.

    b. If the user clicks yes, then the system removes the listing from the dice catalog database.
3. The system will redirect the user to the admin dashboard

3.3.x Remove user accounts.

1. From the admin dashboard. The admin will click the delete icon on the desired user acount, this button can also be pushed from the account details page.
2. The system will produce a confirmation box asking them to confirm that they wish to delete the listing with two buttons
    a. If the user clicks no, the confirmation box closes and nothing happens beyond that.
    b. If the user clicks yes, then the system removes the listing from the accounts database.
3. The system will redirect the user to the admin dashboard

3.3.x Initiate password reset for users.

1. From the admin dashboard the admin will click the desired user
2. The system will redirect the admin to the user details page
3. The admin will click the password reset button on the user details page
4. The system will initiate the password reset proceedure for the user
5. The system will redirect the user to the admin dashboard

3.3.x View sales history and statistics.

1. From the admin dashboard the admin will select the button for sales history and statistics
2. The system will redirect them to the sales information page
3. The system will queries the orders database for information
4. The system will use this information to compute sales information
5. The system will display this information on the sales information page

3.3.x Verify new seller accounts.

3.1 From the admin dashboard the admin will click the verify users button
3.2 The system will redirect the user to the verify users page
3.3 The system will populate the page with the queried results
3.4 The admin will click one of two buttons
3.4.1 The approve button which will make the sellers account as authenticated
3.4.2 The deny button which will delete the sellers account

4. Other Nonfunctional Requirements

4.1 Performance Requirements

To ensure a satisfactory buyer and seller experience, we must adhere to specific performance requirements.

4.1.x The website will be robust to moderate traffic (~100 concurrent users) without experiencing outages.

4.1.x Should the website experience too much traffic, it will mitigate user frustration through use of graceful redirects.

4.1.x All pages will load in less than 15 seconds on a 25 Mbps wireless connection.

4.1.x Moderation actions, such as banning a user or removing a listing, will take no longer than 1 minute to occur.

4.1.x Customers will be notified via email of maintenance periods.

4.2 Safety Requirements

The website should have built in safety protections to make sure that the site can handle a variety of situations without having problems arise. A basic yet extremely important case is to make sure that the website won't allow a user to purchase something that they cannot afford. There also needs to be protections to make sure the seller has confirmed and approved the price they're listing something for. Not having something like that in place could lead to errors and a negative seller experience which could hurt the website's reputation. The website obviously also needs to make sure it follows all laws and regulations so that there won't be any legal issues. Site Admins need the ability to flag, approve/deny, and remove any listing that might potentially violate rules. The website also needs a proper filter so that inappropriate comments or links are not allowed. Review botting or botting in general should also be prevented by the website. In conclusion, the website needs to have safety kept in mind when being developed.

4.3 Security Requirements

The website needs to follow secure software practices to make sure that it is safe and secure. There needs to be ample testing to ensure that there are no bypasses that allow malicious users to get past regular site features and protocols. Almost all text inputs need to be sanitized to make sure that any malicious code or anything that shouldn't be submitted is submitted. The website needs protections in place to make sure that accounts stay safe and do not have any vulnerabilities. In general, following secure software principals are a great way to make sure that the website is safe from any malicious threats.

4.4 Software Quality Attributes

1. Availability: We want the platform to be available at all times.
2. Flexibility: We want to make sure the platform can handle all sorts of situations.
3. Maintainability: We want to make sure that once complete the website will be easy to maintain.
4. Reliable: We want the website to be functional at all times.
5. Useability: We want to make sure the website is easy to use since a variety of people are going to be using it.

5. 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.>

TTRPG - tabletop role playing game

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.>