National University of Computer & Emerging Sciences
Islamabad Campus

Borrow Box: A Peer-to-Peer Product Rental Platform.

Deliverable 2
Software Engineering
Section: E,D
Submission Date: March 23,2025

Group Members:

22i-2362 Fazal Zaman
22i-2371 Heer
22i-2340 M.Fawaz

1. Introduction

1.1 Purpose

This document provides a detailed Software Requirements Specification (SRS) for the **Borrow Box: A Peer-to-Peer Product Rental Platform**. It serves as a reference for developers, stakeholders, and end-users to understand the system's functionalities, constraints, and overall design.

1.2 Scope

Borrow Box is designed to facilitate seamless and efficient rental management for users and product owners. It enables users to rent, view, and manage products digitally while providing an organized and user-friendly platform for product owners to list and manage their inventory.

The system will provide:

- Secure user authentication and profile management.
- A structured platform for listing rental items.
- An efficient rental request and approval system.
- Secure payment processing for rental transactions.
- A feedback mechanism for users to review rentals.
- Notification alerts for rental updates and availability.

2. Functional Requirements

2.1 User Authentication & Management

- Users must be able to register using a valid email and password.
- Users can update their profile details and password.
- Admins can manage and deactivate user accounts.

2.2 Product Listing & Management

- Owners must be able to add, edit, and remove their rental items.
- Each product must have a title, description, price, availability status, and category.
- Users can report incorrect product details.

2.3 Rental Request & Approval

- Users can request to rent a product by specifying the rental duration.
- Owners can approve or reject rental requests.
- Approved rentals are marked as "Active" until completion.
- Renters can cancel a request before approval.

2.4 Report Generation & Analytics

- Admins should be able to generate detailed rental reports.
- Reports should include rental history, payment transactions, and user activity logs.
- Users should be able to download their rental invoices.
- Reports should support filtering by date range, user, and product category.
- The system should allow exporting reports in PDF, CSV, and Excel formats.

2.5 Feedback & Reviews

- Renters can leave feedback on completed rentals.
- Owners can respond to user reviews.
- Reviews should have a rating system (1-5 stars).

2.6 Search & Filtering

- Users should be able to search for rental items by name, category, price range, and availability.
- Advanced filters should be available for sorting by highest-rated, lowest price, etc.

2.7 Rental History Tracking

- Users should have access to their rental history, including past payments and completed transactions.
- Admins should be able to generate rental reports.

2.8 Admin Dashboard & System Management

- Admins should be able to monitor active rentals and user statistics.
- System logs should be maintained for auditing purposes.

2.9 Dispute Resolution

- Users should be able to report rental issues (e.g., damaged product, fraudulent listings).
- Admins should be able to review and resolve disputes.

3. Non-Functional Requirements

3.1 Product Requirements

- The system should load pages within **2 seconds** on a standard internet connection.
- It must maintain an **uptime of 99.9%**.
- All transactions must be encrypted using **SSL/TLS**.
- The system should have a **modular architecture** to support easy updates and maintenance.
- **Scalability**: The system should efficiently handle a growing number of users and transactions.
- **Error Handling**: Clear error messages should be displayed for invalid actions.
- **Logging & Monitoring**: The system should maintain logs for debugging and security audits.

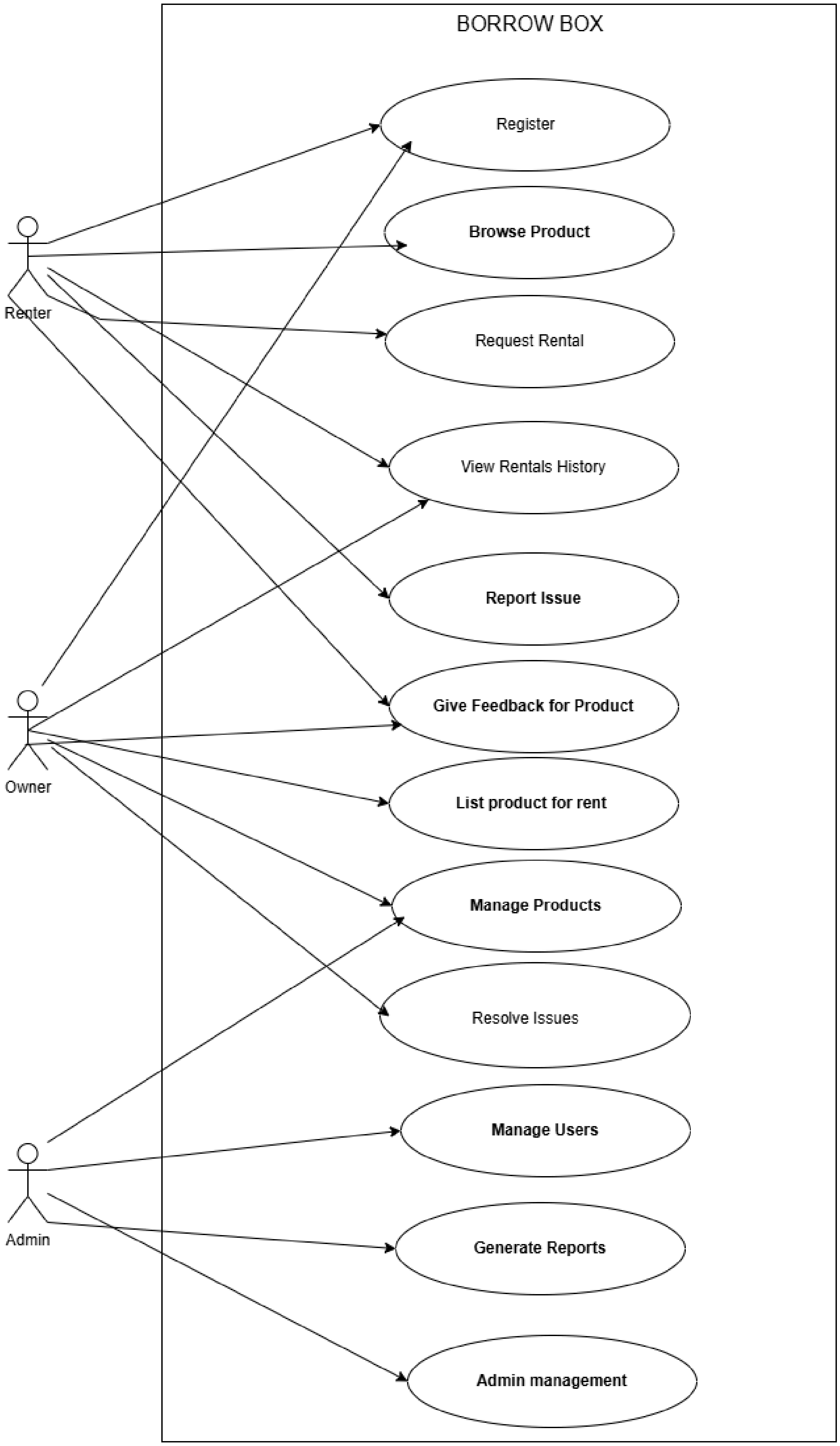
3.2 Organizational Requirements

- The system must follow **Agile development methodologies**.
- It should be built using **Spring Boot for backend** and **React.js/Vue.js for frontend**.
- The database should be optimized for **high-volume queries**.
- The system should support **multi-language localization** to cater to a global user base.
- It should provide **role-based access control (RBAC)** to restrict access to different system functionalities.

3.3 External Requirements

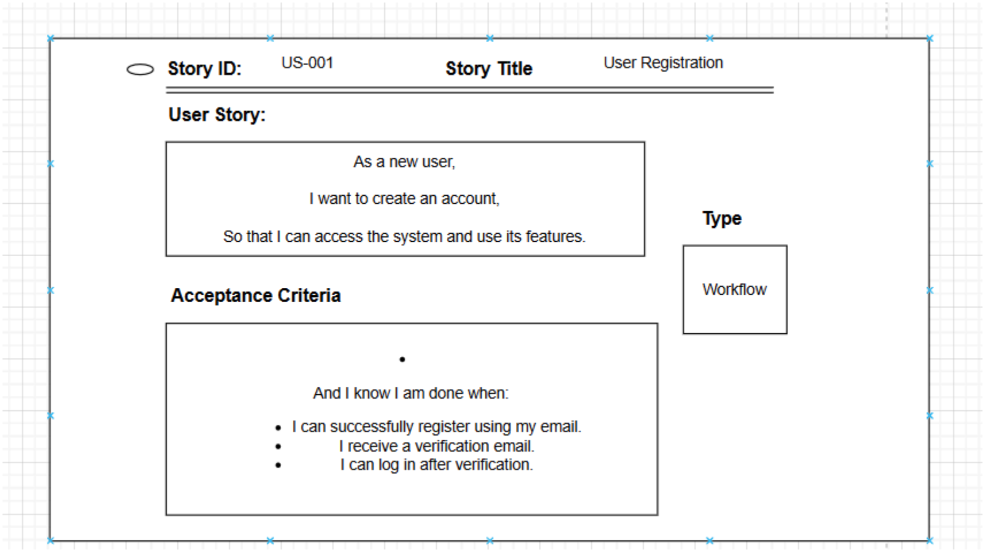
- **Security Compliance**: The system must comply with **GDPR** for user data protection.
- **Cross-Browser Compatibility**: The UI should work on all major browsers.
- **Third-Party API Support**: The system should allow integration with third-party services for extended functionality.
- **Backup & Disaster Recovery**: Data backups should be taken at regular intervals to prevent data loss.

4. Use case Diagram



5. User Stories.

User Story 1: Register



Story ID: US-001 Story Title User Registration

User Story:

As a new user,

I want to create an account,

So that I can access the system and use its features.

Type

Workflow

Acceptance Criteria

-

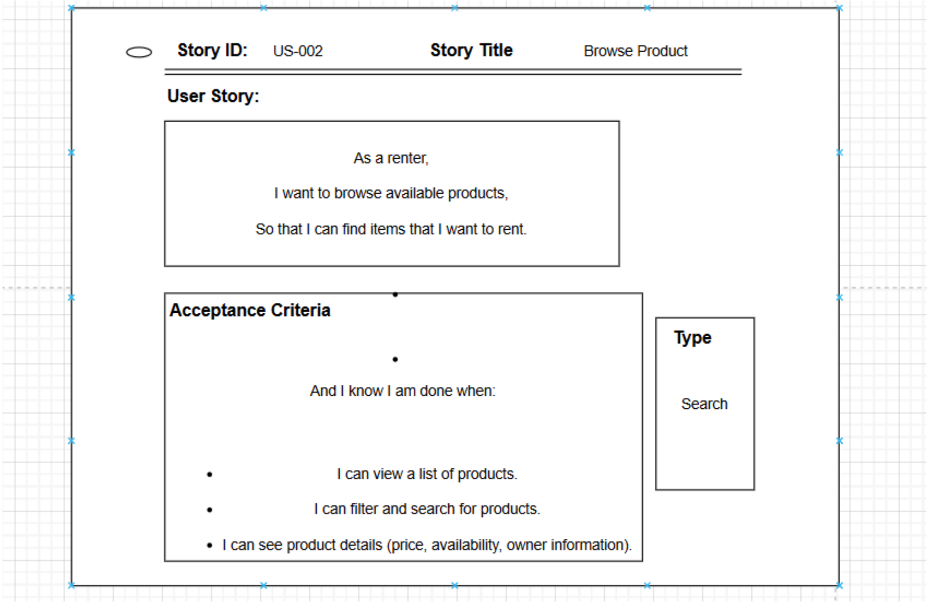
And I know I am done when:

- I can successfully register using my email.
- I receive a verification email.
- I can log in after verification.

User Story 1: Register

- **Preconditions:**
  - The user does not have an existing account.
  - The system is online and operational.
- **Postconditions:**
  - A new user account is created.
  - The user receives a confirmation email.
  - The user can log in to the system.

User Story 2: Browse Product



Story ID: US-002 Story Title Browse Product

User Story:

As a renter,

I want to browse available products,

So that I can find items that I want to rent.

Acceptance Criteria

-

-

And I know I am done when:

- I can view a list of products.
- I can filter and search for products.
- I can see product details (price, availability, owner information).

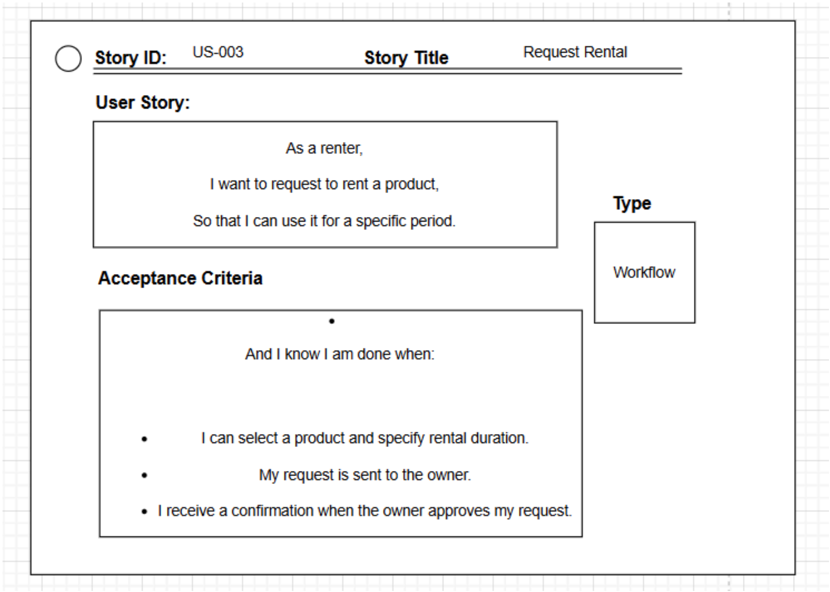
Type

Search

User Story 2: Browse Product

- **Preconditions:**
  - The user is logged in.
  - There are products available for rent in the system.
- **Postconditions:**
  - The user can view the list of products.
  - The system displays product details based on the user's search or filters.

User Story 3: Request Rental



○ **Story ID:** US-003 **Story Title** Request Rental

User Story:

As a renter,

I want to request to rent a product,

So that I can use it for a specific period.

Type

Workflow

Acceptance Criteria

•

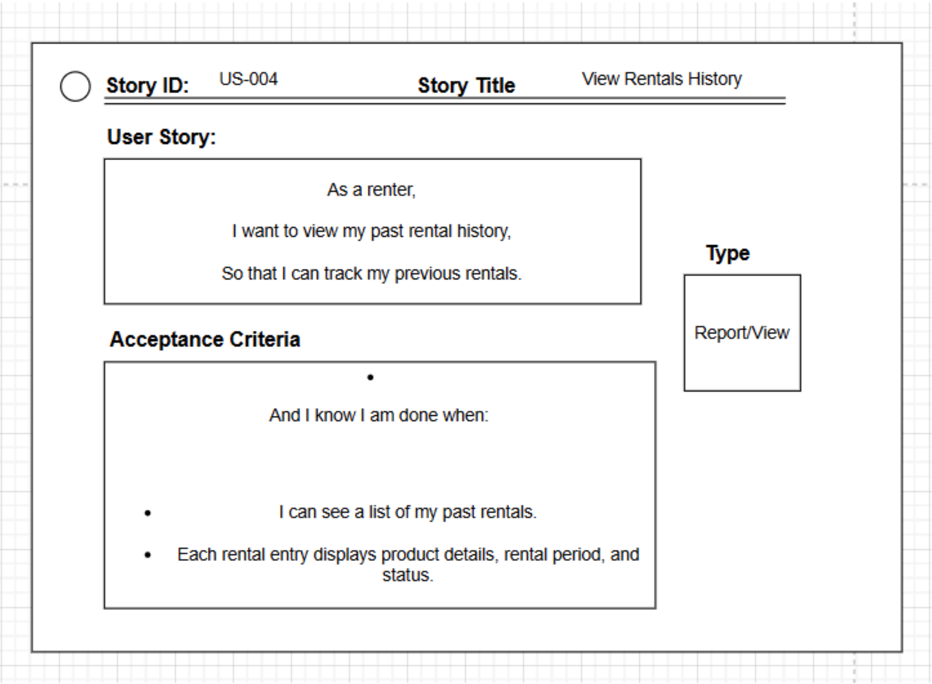
And I know I am done when:

- I can select a product and specify rental duration.
- My request is sent to the owner.
- I receive a confirmation when the owner approves my request.

User Story 3: Request Rental

- **Preconditions:**
  - The user is logged in.
  - The requested product is available for rent.
- **Postconditions:**
  - The rental request is sent to the product owner.
  - The user receives a confirmation or rejection notification.

User Story 4: View Rentals History



○ **Story ID:** US-004 **Story Title** View Rentals History

User Story:

As a renter,

I want to view my past rental history,

So that I can track my previous rentals.

Type

Report/View

Acceptance Criteria

-

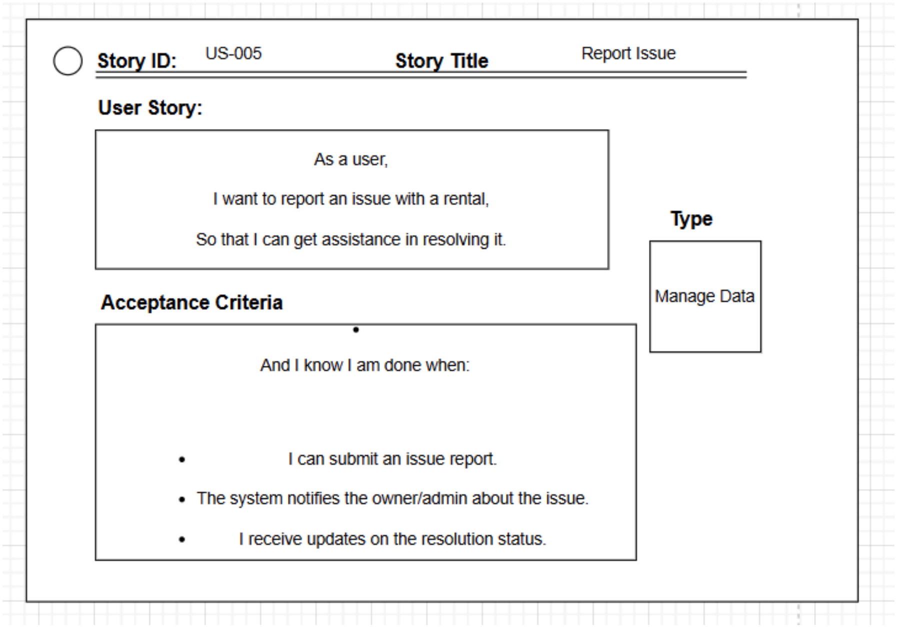
And I know I am done when:

- I can see a list of my past rentals.
- Each rental entry displays product details, rental period, and status.

User Story 4: View Rentals History

- **Preconditions:**
  - The user is logged in.
  - The user has rented products in the past.
- **Postconditions:**
  - The system displays the user's rental history.

User Story 5: Report Issue



○ **Story ID:** US-005 **Story Title** Report Issue

User Story:

As a user,

I want to report an issue with a rental,

So that I can get assistance in resolving it.

Type

Manage Data

Acceptance Criteria

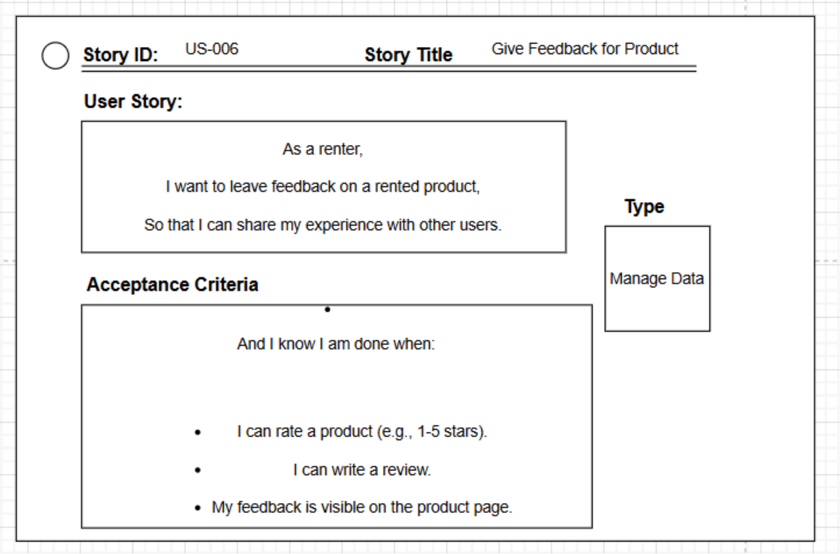
And I know I am done when:

- I can submit an issue report.
- The system notifies the owner/admin about the issue.
- I receive updates on the resolution status.

User Story 5: Report Issue

- **Preconditions:**
  - The user is logged in.
  - The user has an issue with a rental or transaction.
- **Postconditions:**
  - The issue report is submitted.
  - The admin or owner receives the report.
  - The user is notified about the status of the issue

User Story 6: Give Feedback for Product



○ Story ID: US-006 Story Title Give Feedback for Product

User Story:

As a renter,

I want to leave feedback on a rented product,

So that I can share my experience with other users.

Type

Manage Data

Acceptance Criteria

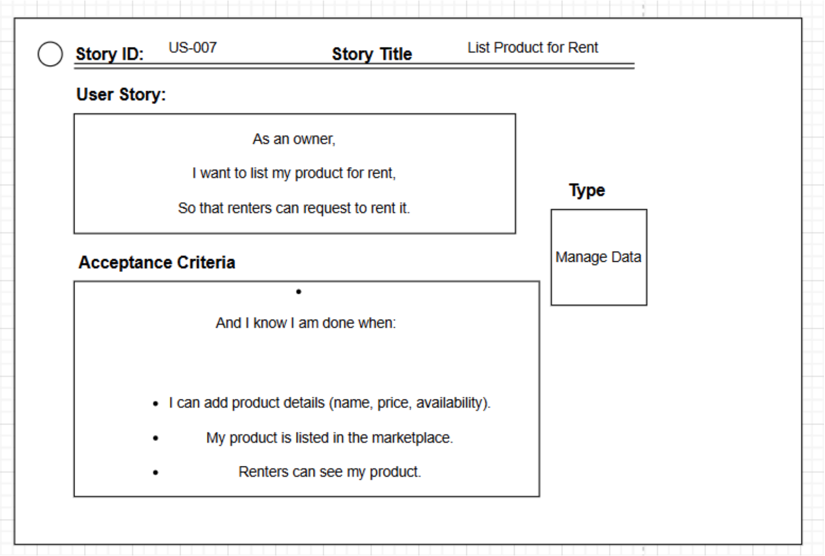
And I know I am done when:

- I can rate a product (e.g., 1-5 stars).
- I can write a review.
- My feedback is visible on the product page.

User Story 6: Give Feedback for Product

- **Preconditions:**
  - The user is logged in.
  - The user has completed a rental for the product.
- **Postconditions:**
  - The feedback is stored in the system.
  - Other users can view the feedback on the product page.

User Story 7: List Product for Rent



○ Story ID: US-007 Story Title List Product for Rent

User Story:

As an owner,

I want to list my product for rent,

So that renters can request to rent it.

Type

Manage Data

Acceptance Criteria

And I know I am done when:

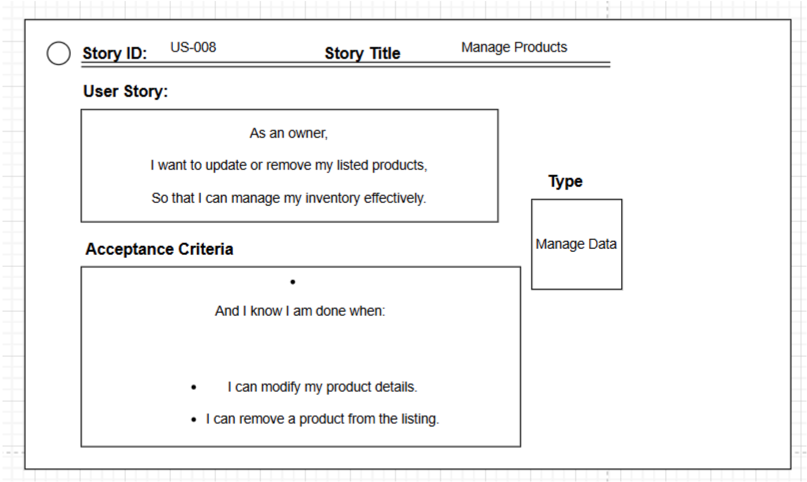
- I can add product details (name, price, availability).
- My product is listed in the marketplace.
- Renters can see my product.

User Story 7: List Product for Rent

- **Preconditions:**
    - The owner is logged in.
    - The owner has a product they want to rent out.

- **Postconditions:**
    - The product is listed in the system.
    - Renters can see and request the product.

User Story 8: Manage Products



○ **Story ID:** US-008 **Story Title** Manage Products

User Story:

As an owner,

I want to update or remove my listed products,

So that I can manage my inventory effectively.

Type

Manage Data

Acceptance Criteria

-

And I know I am done when:

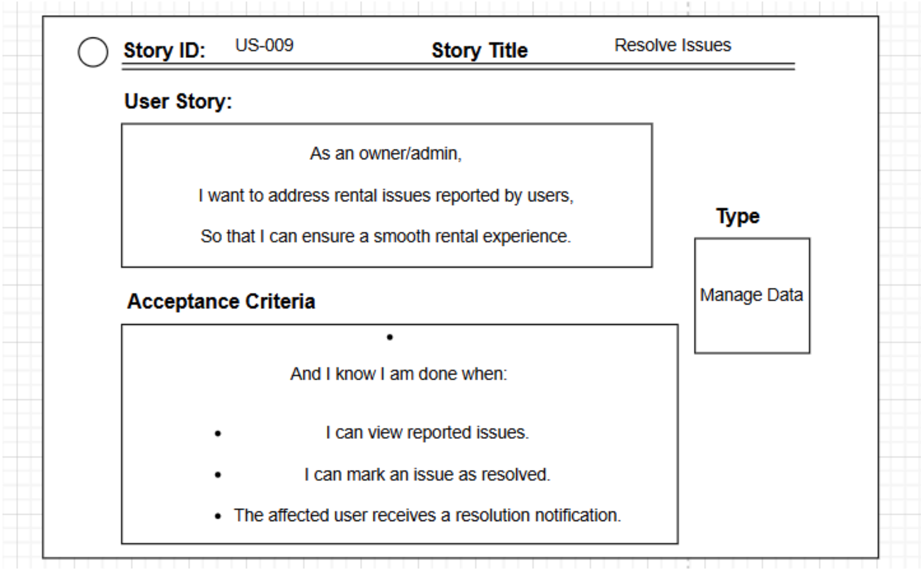
- I can modify my product details.
- I can remove a product from the listing.

User Story 8: Manage Products

- **Preconditions:**
    - The owner is logged in.
    - The owner has at least one listed product.

- **Postconditions:**
  - The product details are updated or deleted based on the owner’s action.

User Story 9: Resolve Issues



○ **Story ID:** US-009 **Story Title** Resolve Issues

User Story:

As an owner/admin,

I want to address rental issues reported by users,

So that I can ensure a smooth rental experience.

Type

Manage Data

Acceptance Criteria

-

And I know I am done when:

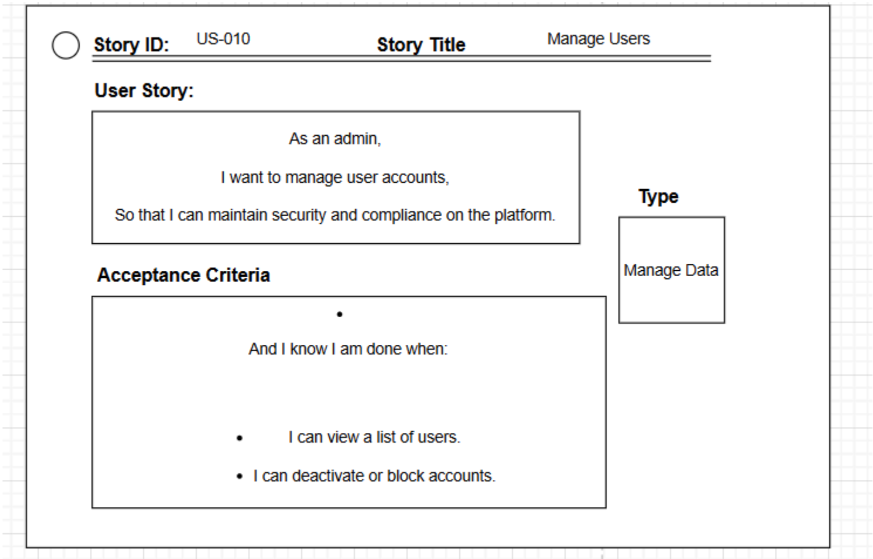
- I can view reported issues.
- I can mark an issue as resolved.
- The affected user receives a resolution notification.

User Story 9: Resolve Issues

- **Preconditions:**
  - The admin/owner is logged in.
  - There is at least one unresolved issue reported by a user.

- **Postconditions:**
  - The issue is marked as resolved or updated with status changes.
  - The affected user is notified.

User Story 10: Manage Users



○ **Story ID:** US-010 **Story Title** Manage Users

User Story:

As an admin,

I want to manage user accounts,

So that I can maintain security and compliance on the platform.

Type

Manage Data

Acceptance Criteria

•

And I know I am done when:

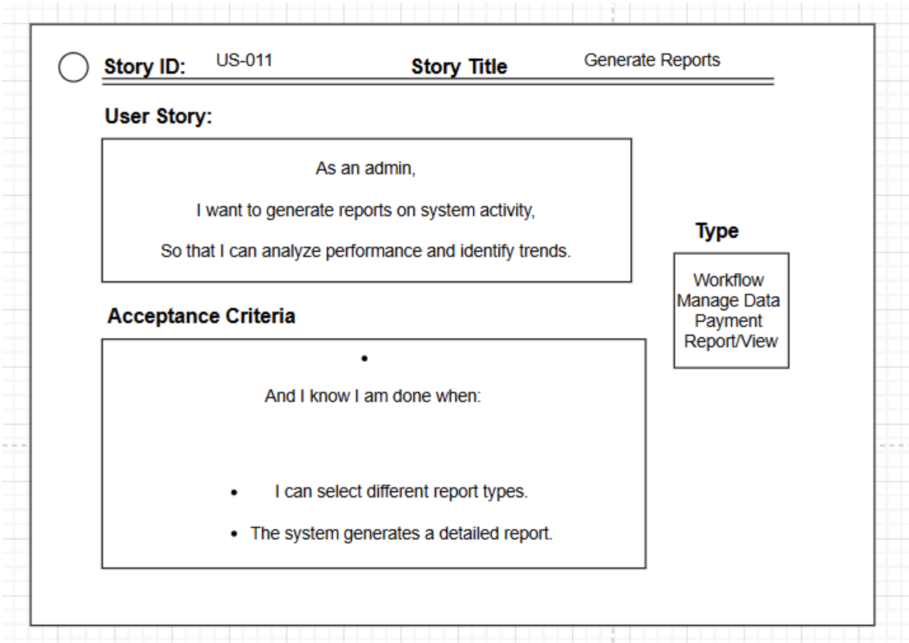
- I can view a list of users.
- I can deactivate or block accounts.

User Story 10: Manage Users

- **Preconditions:**
    - The admin is logged in.
    - There are registered users in the system.

- **Postconditions:**
    - The admin can view, block, or deactivate user accounts.
    - The affected user’s account status is updated.

User Story 11: Generate Reports



○ Story ID: US-011 Story Title Generate Reports

User Story:

As an admin,

I want to generate reports on system activity,

So that I can analyze performance and identify trends.

Acceptance Criteria

-

And I know I am done when:

- I can select different report types.
- The system generates a detailed report.

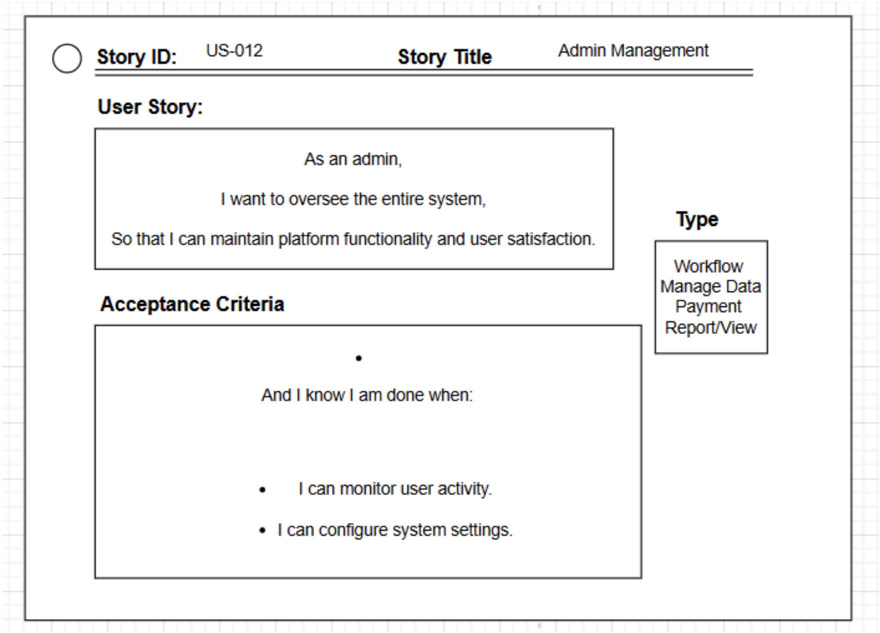
Type

Workflow
Manage Data
Payment
Report/View

User Story 11: Generate Reports

- **Preconditions:**
  - The admin is logged in.
  - There is relevant data available for reporting.
- **Postconditions:**
  - The system generates and displays the requested report.

User Story 12: Admin Management



○ Story ID: US-012 Story Title Admin Management

User Story:

As an admin,

I want to oversee the entire system,

So that I can maintain platform functionality and user satisfaction.

Acceptance Criteria

-

And I know I am done when:

- I can monitor user activity.
- I can configure system settings.

Type

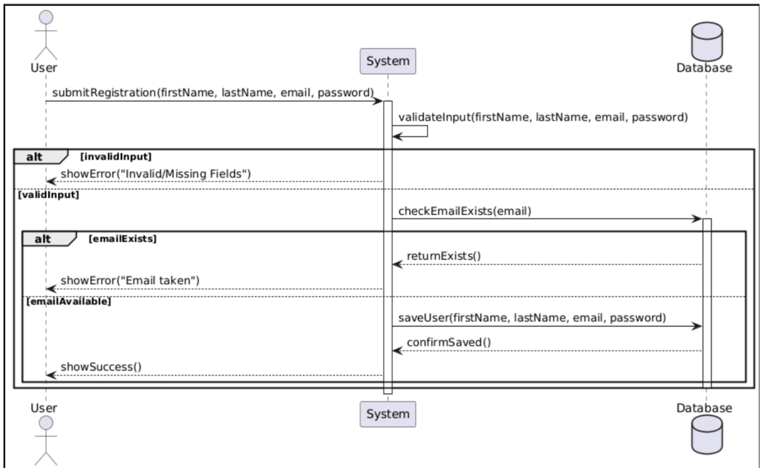
Workflow
Manage Data
Payment
Report/View

User Story 12: Admin Management

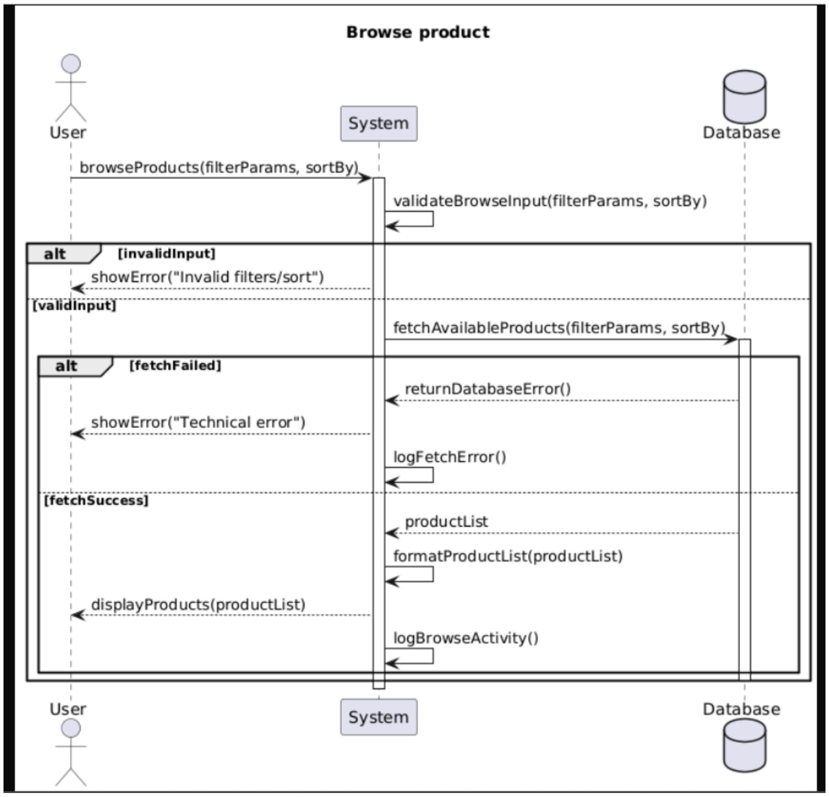
- **Preconditions:**
    - The admin is logged in.
    - The system is functioning correctly.
- **Postconditions:**
    - The admin has performed actions like monitoring, managing settings, or handling issues.

6. Three main SD’s

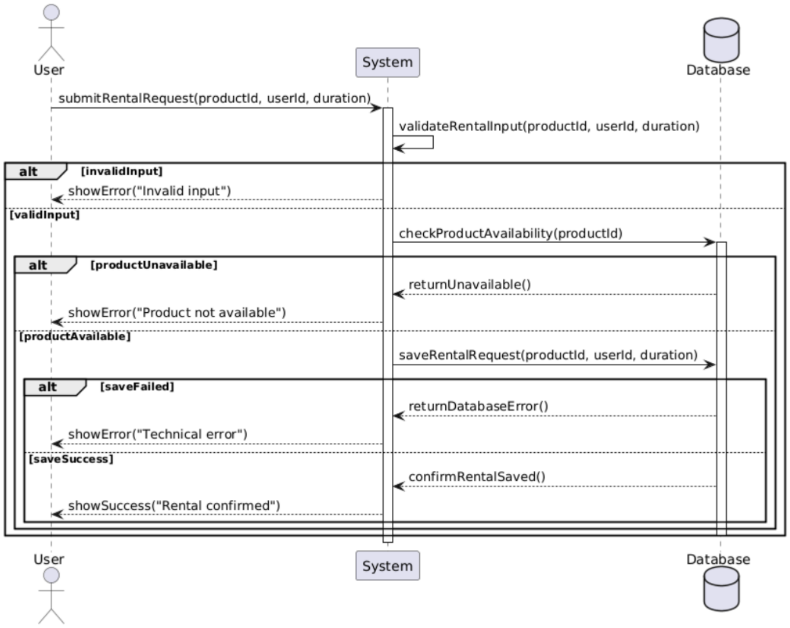
SD for Registration:


User
System
Database
submitRegistration(firstName, lastName, email, password)
validateInput(firstName, lastName, email, password)
alt
[invalidInput]
showError("Invalid/Missing Fields")
[validInput]
checkEmailExists(email)
alt
[emailExists]
returnExists()
showError("Email taken")
[emailAvailable]
saveUser(firstName, lastName, email, password)
confirmSaved()
showSuccess()
User
System
Database

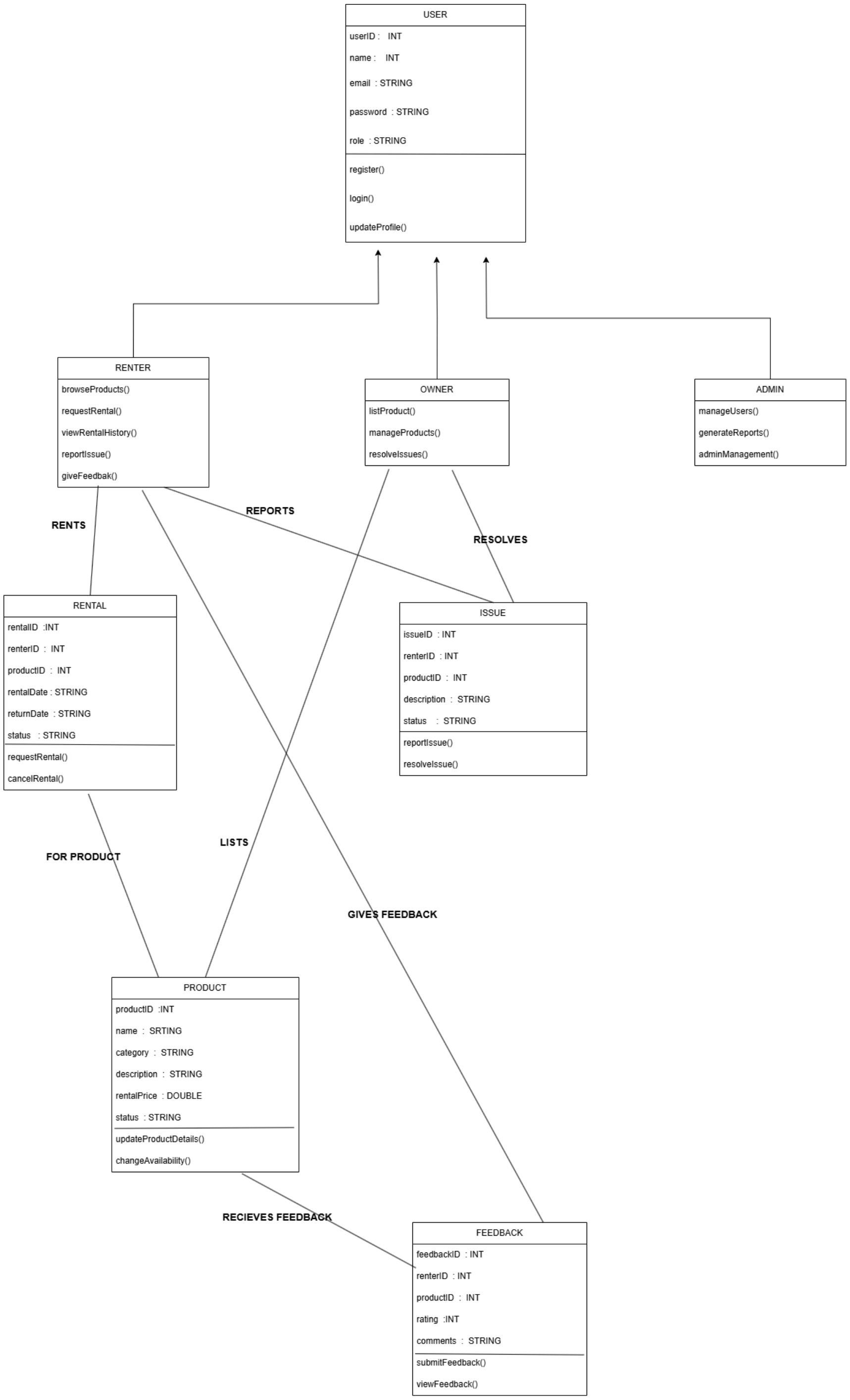
SD for Browse products:


Browse product
User
System
Database
browseProducts(filterParams, sortBy)
validateBrowseInput(filterParams, sortBy)
alt
[invalidInput]
showError("Invalid filters/sort")
[validInput]
fetchAvailableProducts(filterParams, sortBy)
alt
[fetchFailed]
returnDatabaseError()
showError("Technical error")
logFetchError()
[fetchSuccess]
productList
formatProductList(productList)
displayProducts(productList)
logBrowseActivity()
User
System
Database

SD for Rental request:


User
System
Database
submitRentalRequest(productId, userId, duration)
validateRentalInput(productId, userId, duration)
alt
[invalidInput]
showError("Invalid input")
[validInput]
checkProductAvailability(productId)
alt
[productUnavailable]
returnUnavailable()
showError("Product not available")
[productAvailable]
saveRentalRequest(productId, userId, duration)
alt
[saveFailed]
returnDatabaseError()
showError("Technical error")
[saveSuccess]
confirmRentalSaved()
showSuccess("Rental confirmed")
User
System
Database

7. Class Diagram:


USER
userID : INT
name : INT
email : STRING
password : STRING
role : STRING
register()
login()
updateProfile()
RENTER
browseProducts()
requestRental()
viewRentalHistory()
reportIssue()
giveFeedbak()
OWNER
listProduct()
manageProducts()
resolveIssues()
ADMIN
manageUsers()
generateReports()
adminManagement()
RENTS
REPORTS
RESOLVES
RENTAL
rentalID :INT
renterID : INT
productID : INT
rentalDate : STRING
returnDate : STRING
status : STRING
requestRental()
cancelRental()
ISSUE
issueID : INT
renterID : INT
productID : INT
description : STRING
status : STRING
reportIssue()
resolveIssue()
FOR PRODUCT
LISTS
GIVES FEEDBACK
PRODUCT
productID :INT
name : SRTING
category : STRING
description : STRING
rentalPrice : DOUBLE
status : STRING
updateProductDetails()
changeAvailability()
RECIEVES FEEDBACK
FEEDBACK
feedbackID : INT
renterID : INT
productID : INT
rating :INT
comments : STRING
submitFeedback()
viewFeedback()