Software Requirements Specification (SRS)

# KitHub – Campus Used Equipment Library

## 1. Product Perspective

KitHub will be a standalone, web-based application developed specifically for campus use. It is not part of a larger system, but is intended to serve as a reusable, modular tool for equipment loaning in university environments. It will be hosted on GitHub and designed for future extensibility.

## 2. Product Functions

The system’s primary functions include:

• User registration and login

• Equipment browsing with search and filter options

• Borrowing request, approval, and return

• Notifications for due dates and administrator decisions

• Administrator dashboard to manage equipment and users

## 3. User Attributes

• Primary Users: College students, ages 18-30, who are familiar with basic online tools.

• Secondary Users: Administrators, faculty, or club leaders who will manage product approvals and inventory.

## 4. Restrictions

• The system must be web-based and compatible with modern browsers (Chrome, Firefox, Edge).

• Development will be done using HTML/CSS/JavaScript and Python.

• Backend will use MySQL for data storage.

• Planned to be ready for use by August 2025

• Only English language interface will be provided in MVP.

## 5. System Features (Use Case-Based)

### Use Case 1: Register and Log In

• Main Actor: Student

• Goal: To securely register and access the system

• Preconditions: The user is not logged in

• Main Flow:

1. User opens the application.  
 2. Selects “Register” or “Login” from the home page.  
 3. Provides credentials (email, password, etc.).  
 4. System validates and creates/logs in the user.

• Postconditions: User is logged into the system and can access the dashboard.

### Use Case 2: Browse and Search Equipment

• Main Actor: Student

• Goal: To view available equipment and filter by type or availability

• Preconditions: User is logged in

• Main Flow:

1. User navigates to the equipment page.  
 2. Uses search bar or filters (e.g., category, availability).  
 3. System displays matching equipment items.

• Postconditions: The user sees relevant available equipment with current status.

### Use Case 3: Borrow Equipment

• Main Actor: Student

• Goal: To request borrowing of a selected item

• Preconditions: The user is logged in; equipment is available

• Main Flow:

1. User selects an item from the catalog.  
 2. Clicks “Borrow” and fills in borrow request form.  
 3. System sends the request to admin for approval.  
 4. Admin approves/rejects the request.

• Postconditions: If approved, item status becomes “borrowed”; else, remains “available”.

### Use Case 4: Admin Manages Inventory

• Main Actor: Administrator

• Goal: To manage equipment items (add, edit, remove)

• Preconditions: Admin is logged in

• Main Flow:

1. Admin accesses the dashboard.  
 2. Selects “Add Equipment” or edits existing items.  
 3. Enters or updates item details.  
 4. Saves changes.

• Postconditions: Equipment list reflects the latest changes.

## 6. Non-Functional Requirements

### Usability

The system should be easy to understand and operate, even for users with limited technical experience. Design a clean, intuitive interface. Actions such as browsing, borrowing, and managing items should be accessible within 2-3 clicks.

### Performance

The application should respond to user actions within 2 seconds under normal usage and network conditions. Database queries and page loads should be optimized to minimize latency.

### Portability

KitHub should be compatible with all modern web browsers (Google Chrome, Mozilla Firefox, Microsoft Edge). The interface should be responsive and work correctly on desktop, laptop, and tablet devices.

### Reliability

The system should maintain functional stability during common usage scenarios. System downtime should be minimized and the failure rate during operations (e.g. borrow approval, sign-in, item management) should be below 5%. Error cases should be handled gracefully with user-friendly messages.

### Maintainability

Codebase should follow modular and well-documented structure for ease of updates and long-term maintenance.

## 7. External Interface Requirements

No external interfaces are implemented in the MVP version. Future versions may consider integration with university entrance systems or equipment tracking technologies (e.g. QR, RFID).

## 8. User Interfaces

• Login/Register Page: Allows users to create an account or log in.

• Student Dashboard: Displays available equipment, request history, and current loan status.

• Equipment Detail Page: Displays item properties, availability, and request button.

• Borrow Request Form: Allows students to submit requested loan times and notes.

• Admin Panel: Allows administrators to manage equipment inventory, review requests, and track returns.

• Notification Panel or Banner: Displays return due dates and request status updates.

## 9. Software Interfaces

KitHub is a modular web application and uses the following software components:  
  
• Front End → Back End Communication: Via internal REST APIs using HTTP. All communication will follow a RESTful structure with JSON payloads.  
  
• Database Layer: MySQL used to store users, items, and transaction records.  
  
The first version does not use third-party system integration, but the structure is designed to allow easy expansion in the future.

## 10. Task Matrix for This Document

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Responsible Member** |
| Functional Requirements | |  | | --- | |  |  |  | | --- | | Defined system features, user flows, and use cases | | Betül Biçer |
| Non-Functional Specs | Wrote about usability, performance, reliability, maintainability | Aylin Barutçu |
| UI & Interface Section | Described key user and system interfaces | Mehmet Karatekin |
| Final Review | Proofreading, format consistency, and submission organization | İlbey Efe Taşabatlı |