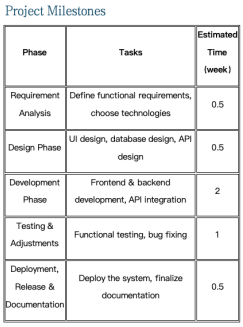
1 Personal Contribution

1.1 Project Framework Setup

Confirmation of Development Direction

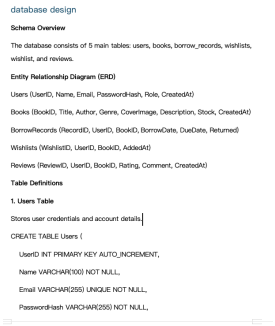
When most team members lacked a clear sense of direction, I proactively proposed the initial structure and approach for the project development, and drafted foundational documentation for group discussion and reference. I advocated for a "frontend-backend decoupled architecture with API integration" and provided preliminary suggestions on technology stack selection, API design, database schema, and testing strategy. This helped team members build a comprehensive development mindset and facilitated group discussions and iterative updates.

Outline



Project Milestones

Phase	Tasks	Estimated Time (week)
Requirement Analysis	Define functional requirements, choose technologies	0.5
Design Phase	UI design, database design, API design	0.5
Development Phase	Frontend & backend development, API integration	2
Testing & Adjustments	Functional testing, bug fixing	1
Deployment, Release & Documentation	Deploy the system, finalize documentation	0.5



database Initial Design

database design

Schema Overview

The database consists of 5 main tables: users, books, borrow_records, wishlists, wishlist, and reviews.

Entity Relationship Diagram (ERD)

Users (UserID, Name, Email, PasswordHash, Role, CreatedAt)

Books (BookID, Title, Author, Genre, CoverImage, Description, Stock, CreatedAt)

BorrowRecords (RecordID, UserID, BookID, BorrowDate, DueDate, Returned)

Wishlists (WishlistID, UserID, BookID, AddedAt)

Reviews (ReviewID, UserID, BookID, Rating, Comment, CreatedAt)

Table Definitions

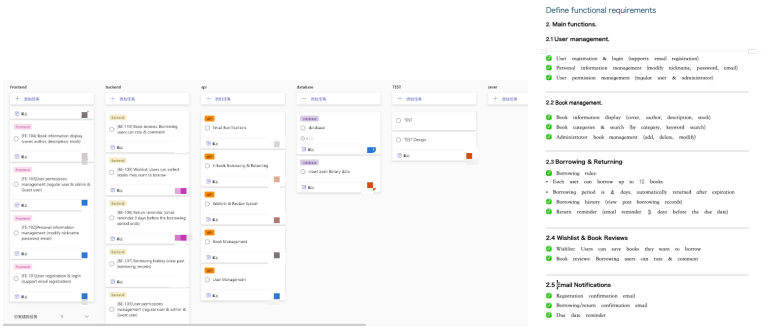
1. Users Table

Stores user credentials and account details.

```
CREATE TABLE Users (
    UserID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(100) NOT NULL,
    Email VARCHAR(255) UNIQUE NOT NULL,
    PasswordHash VARCHAR(255) NOT NULL,
```

Task Breakdown and Standardized Assignment

To enhance execution efficiency, I applied the SMART principles to decompose the requirements into 34 specific and actionable sub-tasks. I created a visual task board in Microsoft Teams to organize and assign tasks based on each team members' strengths and expertise.



Define functional requirements

2. Main functions.

2.1 User management.

- User registration & login (supports email registration)
- Personal information management (modify nickname, password, email)
- User permission management (regular user & administrator)

2.2 Book management.

- Book information display (cover, author, description, stock)
- Book categories & search (by category, keyword search)
- Administrator book management (add, delete, modify)

2.3 Borrowing & Returning

- Borrowing rules:
  - Each user can borrow up to 10 books
  - Borrowing period is 4 days, automatically returned after expiration
- Borrowing history (view past borrowing records)
- Return reminder (email reminder 3 days before the due date)

2.4 Wishlist & Book Reviews

- Wishlist: Users can save books they want to borrow
- Book reviews: Borrowing users can rate & comment

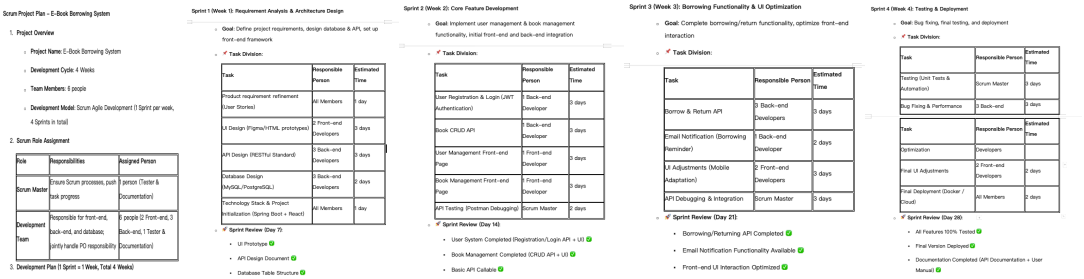
2.5 Email Notifications

- Registration confirmation email
- Borrowing/return confirmation email
- Due date reminder

Tasks Breakdown details

Project Planning

By introducing Scrum development methodology, I established sprint plans, daily stand-up meetings, and retrospective mechanisms to enhance the project's execution rhythm and improve team collaboration efficiency.



Scrum Project Plan – E-Book Borrowing System

1. Project Overview
   - Project Name: E-Book Borrowing System
   - Development Cycle: 4 Weeks
   - Team Members: 6 people
   - Development Model: Scrum Agile Development (1 Sprint per week, 4 Sprints in total)
2. Scrum Role Assignment

Role	Responsibilities	Assigned Person
Scrum Master	Ensure Scrum processes, push task progress	1 person (Tester & Documentation)
Development Team	Responsible for front-end, back-end, and database; jointly handle PO responsibility	6 people (2 Front-end, 3 Back-end, 1 Tester & Documentation)

3. Development Plan (1 Sprint = 1 Week, Total 4 Weeks)

Sprint 1 (Week 1): Requirement Analysis & Architecture Design

- Goal: Define project requirements, design database & API, set up front-end framework
- Task Division:

Task	Responsible Person	Estimated Time
Product requirement refinement (User Stories)	All Members	1 day
UI Design (Figma/HTML prototypes)	2 Front-end Developers	3 days
API Design (RESTful Standard)	3 Back-end Developers	3 days
Database Design (MySQL/PostgreSQL)	3 Back-end Developers	2 days
Technology Stack & Project Initialization (Spring Boot + React)	All Members	1 day

- Sprint Review (Day 7):
  - UI Prototype ✅
  - API Design Document ✅
  - Database Table Structure ✅

Sprint 2 (Week 2): Core Feature Development

- Goal: Implement user management & book management functionality, initial front-end and back-end integration
- Task Division:

Task	Responsible Person	Estimated Time
User Registration & Login (JWT Authentication)	1 Back-end Developer	3 days
Book CRUD API	1 Back-end Developer	3 days
User Management Front-end Page	1 Front-end Developer	5 days
Book Management Front-end Page	1 Front-end Developer	5 days
API Testing (Postman Debugging)	Scrum Master	2 days

- Sprint Review (Day 14)
  - User System Completed (Registration/Login API + UI) ✅
  - Book Management Completed (CRUD API + UI) ✅
  - Basic API Callable ✅

Sprint 3 (Week 3): Borrowing Functionality & UI Optimization

- Goal: Complete borrowing/return functionality, optimize front-end interaction
- Task Division:

Task	Responsible Person	Estimated Time
Borrow & Return API	3 Back-end Developers	3 days
Email Notification (Borrowing Reminder)	1 Back-end Developer	2 days
UI Adjustments (Mobile Adaptation)	2 Front-end Developers	3 days
API Debugging & Integration	Scrum Master	3 days

- Sprint Review (Day 21):
  - Borrowing/Returning API Completed ✅
  - Email Notification Functionality Available ✅
  - Front-end UI Interaction Optimized ✅

Sprint 4 (Week 4): Testing & Deployment

- Goal: Bug fixing, final testing, and deployment
- Task Division:

Task	Responsible Person	Estimated Time
Testing (Unit Tests & Automation)	Scrum Master	3 days
Bug Fixing & Performance	3 Back-end	3 days
Optimization	Developers	
Final UI Adjustments	2 Front-end Developers	2 days
Final Deployment (Docker / Cloud)	All Members	2 days

- Sprint Review (Day 28)
  - All Features 100% Tested ✅
  - Final Version Deployed ✅
  - Documentation Completed (API Documentation + User Manual) ✅

Introduce Version Control Concept

To prevent version conflicts in a multi-developer environment, I introduced the concept of team collaboration using GitHub, which helped avoid chaos in version management. I also explained the basic Git operations and collaboration workflows to the team. And until now, I think everyone in our group has mastered the correct use of GitHub as a team development version tool.

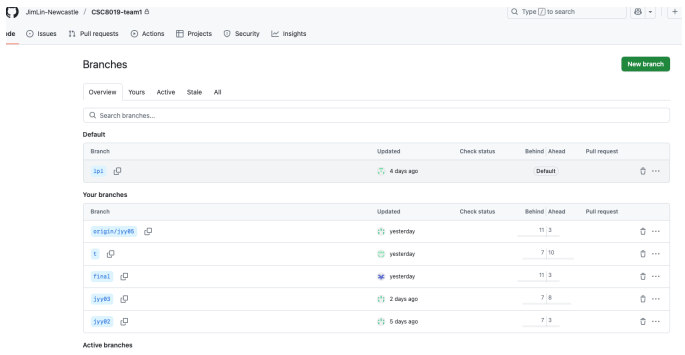


Figure GitHub branches example

1.2 Core Feature Development

Implementation of Book Management and Borrowing System

I was responsible for developing core backend modules, including book display, book search, administrator book management(add/edit/delete books), and email notification functionalities (Java). More details showed below:

Book Managemen

✅ Book information display (cover, author, description, stock)

✅ Book categories & search (search by category, keywords)

✅ Admin book management (add, delete, modify)
✅ Book information display (cover, author, description, stock)
✅ Book categories & search (search by category, keywords)
✅ Admin book management (add, delete, modify)

Email Notifications

✅ Registration confirmation email
✅ Borrowing/return confirmation email
✅ Expiration reminder

Database redesign /get data/create

Additionally, I not only created database but also was responsible for getting and inserting data. To get the data I completed the web scraping, data cleaning, and database import of approximately 200 real book records (Python + MySQL), with data sourced from Google.
Also, I create a '.sql' file to let others create a same one more easily.

Api connecting

In addition to backend development, I also participated in some of the interface docking work between the frontend and the backend, assisted in the implementation of some function buttons and API interaction logic on the page, and improved the smoothness of user operations and the overall consistency of the system.

Testing and Debugging

I wrote and conducted initial tests for key functional processes, and resolved multiple issues, including flaws in the borrowing logic and database schema design.

Team Collaboration

In the early phase of the project, I rapidly built and shared the basic backend architecture (Spring Boot), helping the team align on a unified technical path. I also assisted other team members in setting up their local environments, explained API-related concepts, and helped resolve onboarding challenges.

2. Three Things Done Well

Holistic Project Thinking & Process Optimization

I adopted a mindset combining both product and project management. In the early stages, I enhanced team execution efficiency by creating technical documentation, decomposing tasks, and implementing sprint planning mechanisms — effectively guiding the team from uncertainty to actionable execution, which can build our group confidence to start .

Efficient Implementation of Key Features

I quickly completed backend framework setup by using spring boot and core book borrowing module development, laying a solid technical foundation for the project's advancement.

Therefore, our back-end development progress is relatively faster, and we can have more time to help students in charge of front-end development debug the API in the later stage.

Proactive Communication & Strong Team Collaboration

I voluntarily handled non-functional tasks such as data collection, code refactoring, and backend deployment. I also proactively resolved API integration issues and task allocation conflicts, ensuring well-organized collaboration by considering the team's skill distribution.

For example, since the project was designed to be a local environment, all of us will need database, backend code, frontend code in our own computer. So I helped others to equip all these code one by one to help our group coding more efficiently.

What's more, I, as a member of team, always pay attention to the progress of each team member, find out early if someone is not making timely progress and think about corresponding alternative plans to ensure that the task can be completed within the specified time.

3. Three Issues and Resolutions

Ineffective Early Discussions & Lack of Direction

Issues: Due to limited practical experience, team members struggled to analyze tasks or define execution plans, leading to unproductive discussions.

Resolutions: I addressed this by proposing a clear technical architecture (more details in 1.1) and planning framework to help focus the team on tangible project outcomes. This practice proves that a clear technical route is more conducive to project advancement

Inconsistent Frontend-Backend API Formats

Issues: During integration, we one faced API could not work in frontend but do well in postman and no one could fix it out for a while.

Resolutions: We arranged time to learn online how to call the API and implement its functions, and finally successfully linked it after completing some front-end and back-end configuration items.

Lack of Transparency in Team Progress

Issues: Some team members couldn't attend every meeting.

Resolutions: Without blame, I suggested following up individually after meetings and mentally prepared to take on additional tasks if necessary, ensuring overall project continuity.

4. Areas for Improvement

Inconsistent Execution of Scrum Practices

Although the team understood the sprint concept, we didn't always strictly follow the planned framework. I lacked the discipline and emotional intelligence to enforce consistency without

affecting team morale. In the future, I aim to implement sprint planning more firmly and improve my leadership and interpersonal communication skills.

Indecisiveness & Lack of Assertiveness

I was overly democratic during design discussions, preferring team consensus over efficiency. For example, when the team didn't fully understand API concepts, I should have pushed forward with my design instead of waiting for group approval on every detail. Moving forward, I aim to strengthen confidence and lead with a "build first, adjust later" approach.

Insufficient Test Coverage

Testing focused primarily on main process flows, overlooking edge cases and exceptions. I plan to systematically study unit testing and integration testing to improve code robustness.

Limited Scope of Contribution

Although this was a group assignment, I had hoped to implement the entire project on my own to deepen my understanding of web application development and set up a personal blog. Also, I was wanted to push it to a VM sever and reconstruct a config file. However, I ended up only completing my assigned parts. I need to strengthen my execution resolve to achieve long-term personal technical goals.

5. Main Group Improvement

The key area for improvement is **execution discipline and rhythm control**. Although we created a plan early on, most of the real progress happened in the later stages (after a vague March and a long April holiday). As a result, we had to forgo several enhancements (e.g., UI design optimizations, multi-client support, Service Deployment, Sensitive Information abstract). In the future, I suggest strictly adhering to sprint mechanisms from the project outset to ensure consistent development momentum.