School of Engineering and Informatics

UNIVERSITY OF SUSSEX

975G5: ESRS Project

Title	ESRS Project Specification
Candidate Number & Names	Tom Naccarato - 277822 Jawad Nasar Shah - 276267 Rohan Kadam - 276280 Aman Thapa Magar - 277822 Mya Lwin - 284318 Robert Ryan - 260029 Moldir Zhumakhazhi - 284496
Module Leader	Prof. Hsi-Ming Ho
Course Title	Msc Advanced Computer Science
Date Submitted	2/16/2024

Table of Contents

Table of Figures

Project Specification

Project Title

Bingle - A book lending web application

Project Deadlines

Specification Document - 16 February 2023 @ 16:00

Final codebase - 14 May 2023 @ 16:00

Group Organisation

Role	Name
Project Leader	Rohan
Overseer/ Infra Setup	Aman
Development (Frontend/Backend)	Tom, Rohan, Rob, Aman, Moldir, Mya
Database	Jawad
QA & Documentation	Mya, Moldir
Development (Frontend/Backend)	Tom, Rohan, Rob, Aman, Moldir, Mya

Customer/Stakeholder

Module Convenor	Prof. Hsi-Ming Ho	hsi-ming.ho@sussex.ac.uk

1 Project Brief

Create a web application for mobile and desktop devices, Bingle is a simple and user-friendly application created to help people share and borrow books within a community. Users can connect with fellow book lovers, explore new reads, and share their favorite titles. The app consists of features like discovering books, creating virtual bookshelves, and securely lending. Whether you're looking to borrow the latest bestseller or share a beloved classic, this app encourages sharing knowledge and building relationships through a mutual love of reading.

2 Project Scalability and Reliability

2.1 System Design Concept

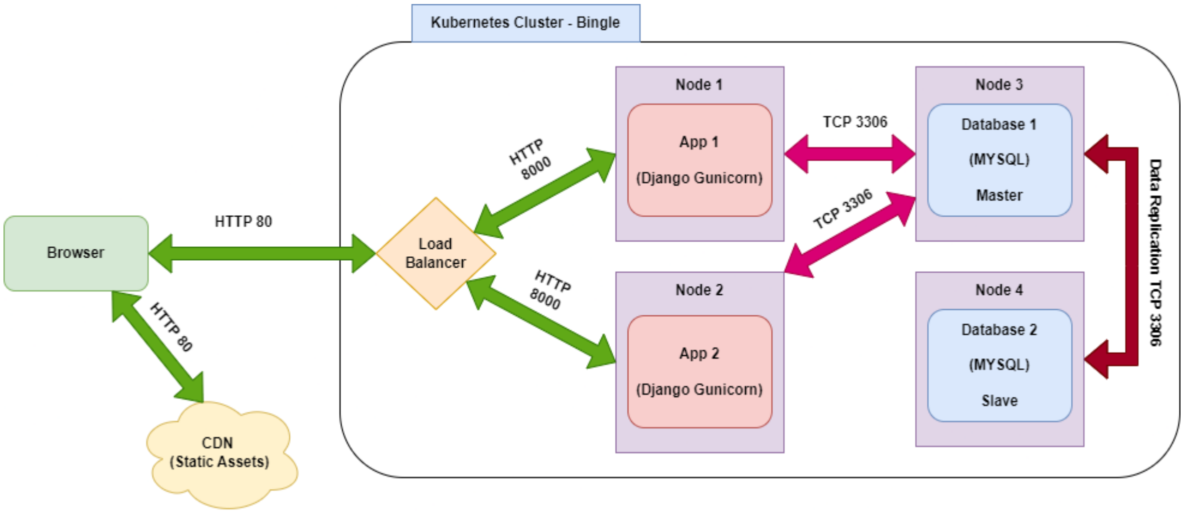


Figure 1: Application Architecture Overview

Based on the Scalable and Reliable problem, we are considering adopting a Model-View-Template architecture during the development and testing. The whole system will be deployed on a Kubernetes cluster of four nodes: 2 for database and 2 for application. The database will be configured in Master-Slave configuration where at most one database will be active at all times performing read-write operations and the data will be replicated constantly to the slave DB to be up to date. If master fails than the slave is automatically promoted to master DB. The traffic would be equally distributed using a load balancer and static assets would be used from CDN servers to minimise slow response time.

2.2 Application Responsibility:

Model: The application's models define the structure of the data, encapsulating entities such as users, books, reviews, and messages. These models specify the fields and relationships necessary for functionalities like user registration, library management, and communication between users. For instance, the User model stores user-related information such as usernames, emails, and review scores, facilitating user interactions and trustworthiness assessment. Similarly, the Book model includes details like titles, authors, and statuses (available/on loan) to manage the lending and borrowing of books among users. Furthermore, the Review and Message models enable users to leave feedback and communicate with each other, enhancing the collaborative nature of the platform.

View: Views in the application handle the logic behind user interactions and business processes, orchestrating actions based on user requests and system requirements. These views implement functionalities like user authentication, library updates, search operations, and book borrowing processes. For example, the library view orchestrates the display of a user's book collection and supports actions like adding or removing books. Additionally, views for messaging facilitate communication between users, allowing them to coordinate book lending arrangements seamlessly. Each view is designed to respond appropriately to user inputs, ensuring smooth navigation and efficient utilization of application features.

Template: Templates in the application provide the visual representation of user interface elements, presenting information and interactive components to users. These templates encompass registration forms, library displays, search result pages, messaging interfaces, and more, offering a cohesive and intuitive user experience. For instance, the registration template presents fields for users to input their credentials during account creation, guiding them through the registration process efficiently. Similarly, library templates showcase a user's book collection in an organized manner, allowing for easy navigation and management of books. Each template is designed with usability and aesthetics in mind, aiming to enhance user engagement and satisfaction throughout their interaction with the application.

2.3 Application Flowchart

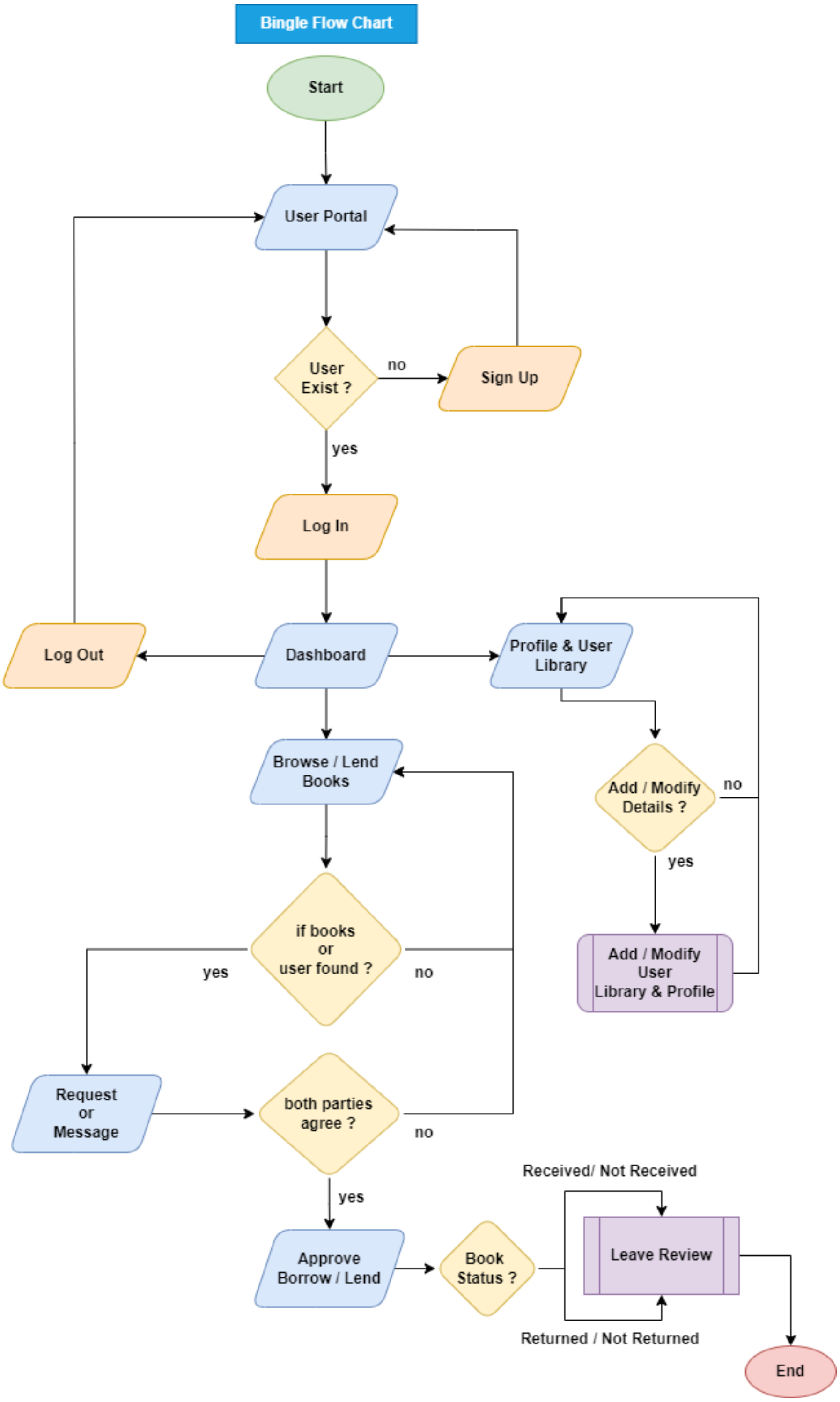


Figure 2: Flowchart for the System

2.4 Use Case Diagram

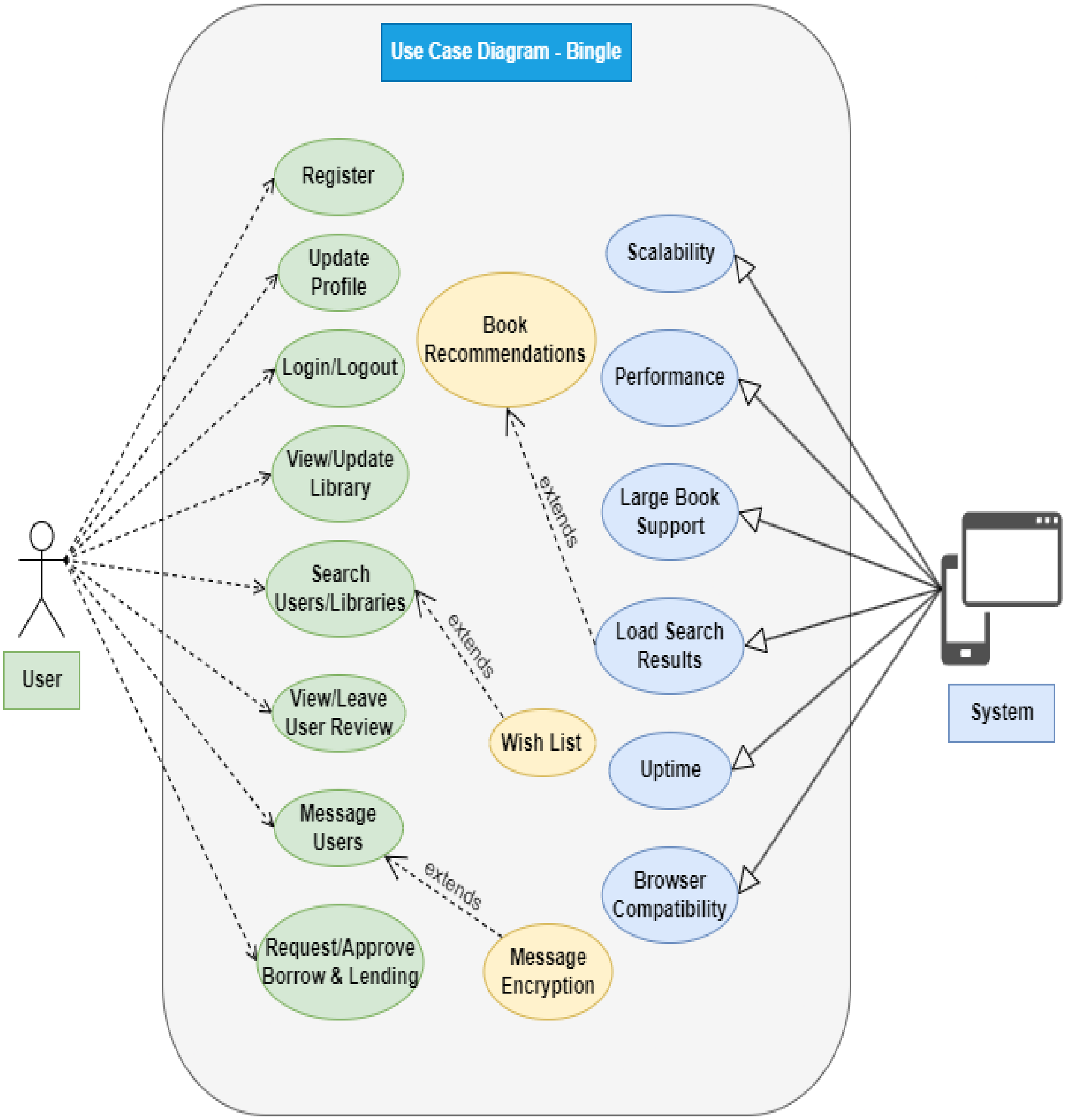
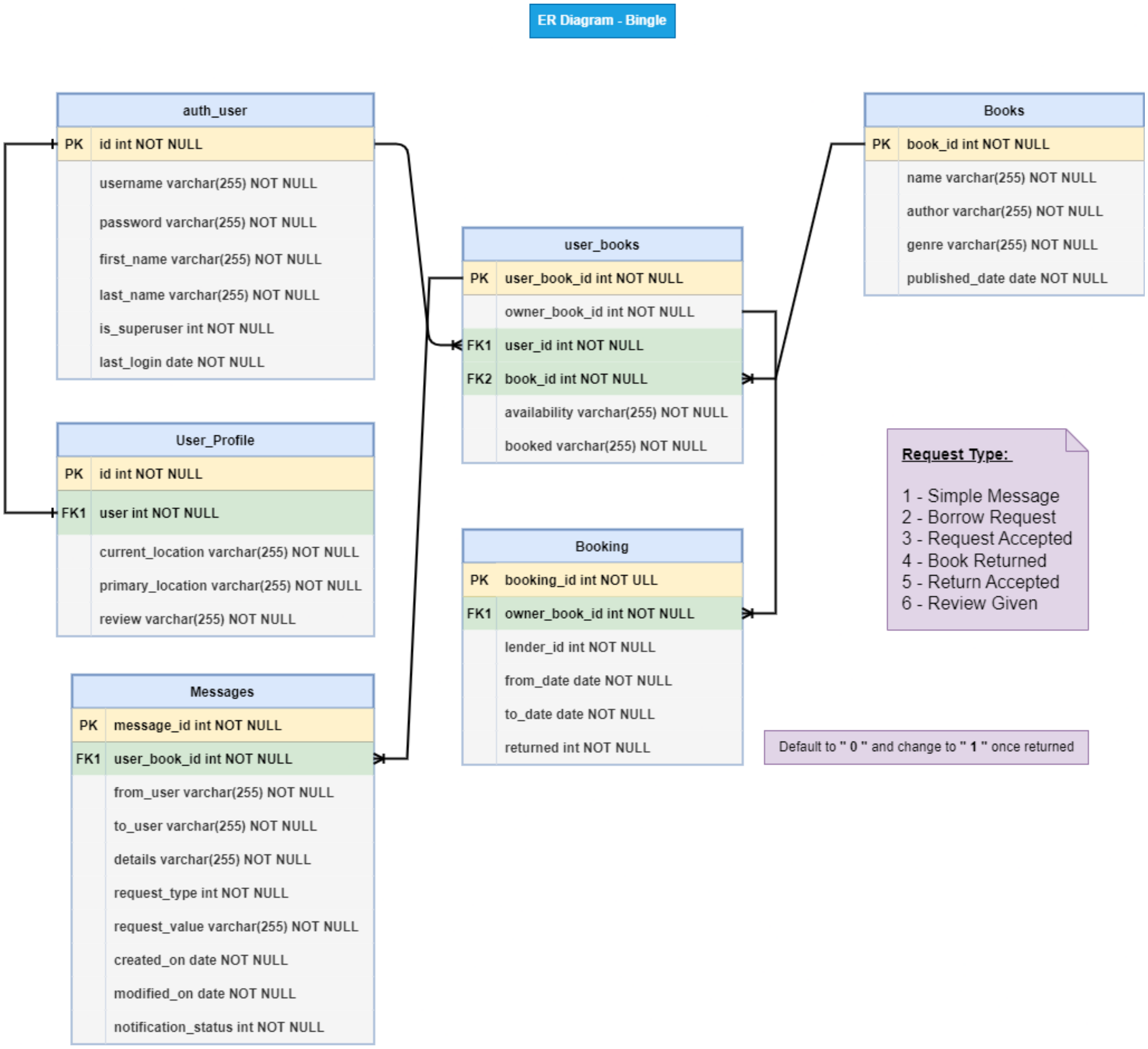


Figure 3: Use Case Diagram for the System

2.5 ER Diagram



ER Diagram - Bingle

auth_user

PK	id int NOT NULL
	username varchar(255) NOT NULL
	password varchar(255) NOT NULL
	first_name varchar(255) NOT NULL
	last_name varchar(255) NOT NULL
	is_superuser int NOT NULL
	last_login date NOT NULL

User_Profile

PK	id int NOT NULL
FK1	user int NOT NULL
	current_location varchar(255) NOT NULL
	primary_location varchar(255) NOT NULL
	review varchar(255) NOT NULL

Messages

PK	message_id int NOT NULL
FK1	user_book_id int NOT NULL
	from_user varchar(255) NOT NULL
	to_user varchar(255) NOT NULL
	details varchar(255) NOT NULL
	request_type int NOT NULL
	request_value varchar(255) NOT NULL
	created_on date NOT NULL
	modified_on date NOT NULL
	notification_status int NOT NULL

user_books

PK	user_book_id int NOT NULL
	owner_book_id int NOT NULL
FK1	user_id int NOT NULL
FK2	book_id int NOT NULL
	availability varchar(255) NOT NULL
	booked varchar(255) NOT NULL

Booking

PK	booking_id int NOT ULL
FK1	owner_book_id int NOT NULL
	lender_id int NOT NULL
	from_date date NOT NULL
	to_date date NOT NULL
	returned int NOT NULL

Books

PK	book_id int NOT NULL
	name varchar(255) NOT NULL
	author varchar(255) NOT NULL
	genre varchar(255) NOT NULL
	published_date date NOT NULL

Request Type:

1 - Simple Message
2 - Borrow Request
3 - Request Accepted
4 - Book Returned
5 - Return Accepted
6 - Review Given

Default to " **0** " and change to " **1** " once returned

Figure 4: ER Diagram for the Web App

2.6 Risk Management Mind Map

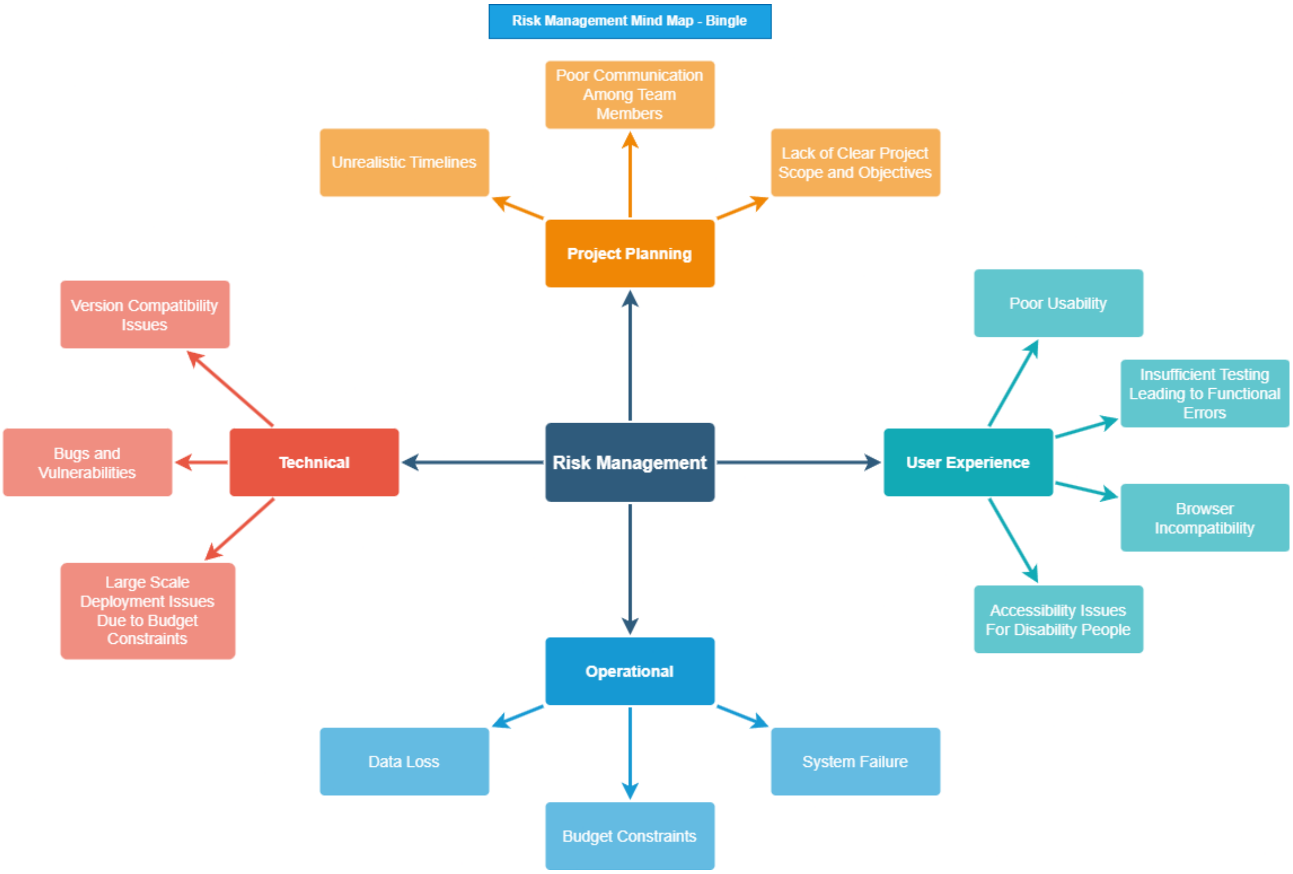

Risk Management Mind Map - Bingle
Poor Communication Among Team Members
Unrealistic Timelines
Lack of Clear Project Scope and Objectives
Project Planning
Version Compatibility Issues
Poor Usability
Insufficient Testing Leading to Functional Errors
Bugs and Vulnerabilities
Technical
Risk Management
User Experience
Browser Incompatibility
Large Scale Deployment Issues Due to Budget Constraints
Accessibility Issues For Disability People
Operational
Data Loss
System Failure
Budget Constraints

Figure 5: Risk Management Mind Map for the Project

3 Requirements

3.1 Functional

*** C/E – Core / Extended**

SN	C/E	Requirement	Description	Notes
1	C	GitHub Repo	Each group gets a GitHub account and creates a repo where the project is hosted. Each group emails a link to the repo to the module convenor (Hsi-Ming.Ho@sussex...).	N/A
2	C	User can register	Users can create an account by providing necessary information such as username, email address, and password. This signup process allows users to establish their unique identity within the application, enabling them to access various features and functionalities.	should not have to be created for them
3	C	Login	Once registered, users can log in to their accounts securely using their credentials. This login functionality ensures that users can access their personal library, interact with other users, and utilize the app's features. Additionally, users have the option to log out when they finish their session, enhancing account security.	N/A
4	C	Library	Within the application, users have access to their personal library, where they can browse, manage, and organize their collection of books. This includes viewing books they have lent or borrowed, adding or removing books from their library, and exploring the libraries of other users to discover new reading materials.	N/A
5	C	Reviews	Users have the ability to review and rate other users based on their interactions within the application. These reviews contribute to generating a score for each user, reflecting their reliability and trustworthiness within the book lending community.	N/A

6	C	Messages	To facilitate book borrowing and lending, users can communicate with each other via messaging. This feature enables users to coordinate logistics, such as arranging meeting points or discussing book availability, enhancing the efficiency and convenience of book exchanges.	N/A
7	E	Book Recommendations	The application offers personalized book recommendations based on the user's reading preferences and history. By analyzing user data and behavior, the app suggests relevant titles, helping users discover new books aligned with their interests.	N/A
8	E	User Messages Encryption	To ensure the security and privacy of user communications, all messages exchanged between users are encrypted. This encryption technology protects sensitive information from unauthorized access, safeguarding user data and enhancing trust in the platform.	N/A
9	E	Wishlist	Users have the option to create a wishlist of books they are interested in borrowing or acquiring. This wishlist feature allows users to keep track of desired titles and notifies them when those books become available for lending within the community.	N/A

3.2 Non-functional

*** C/E – Core / Extended**

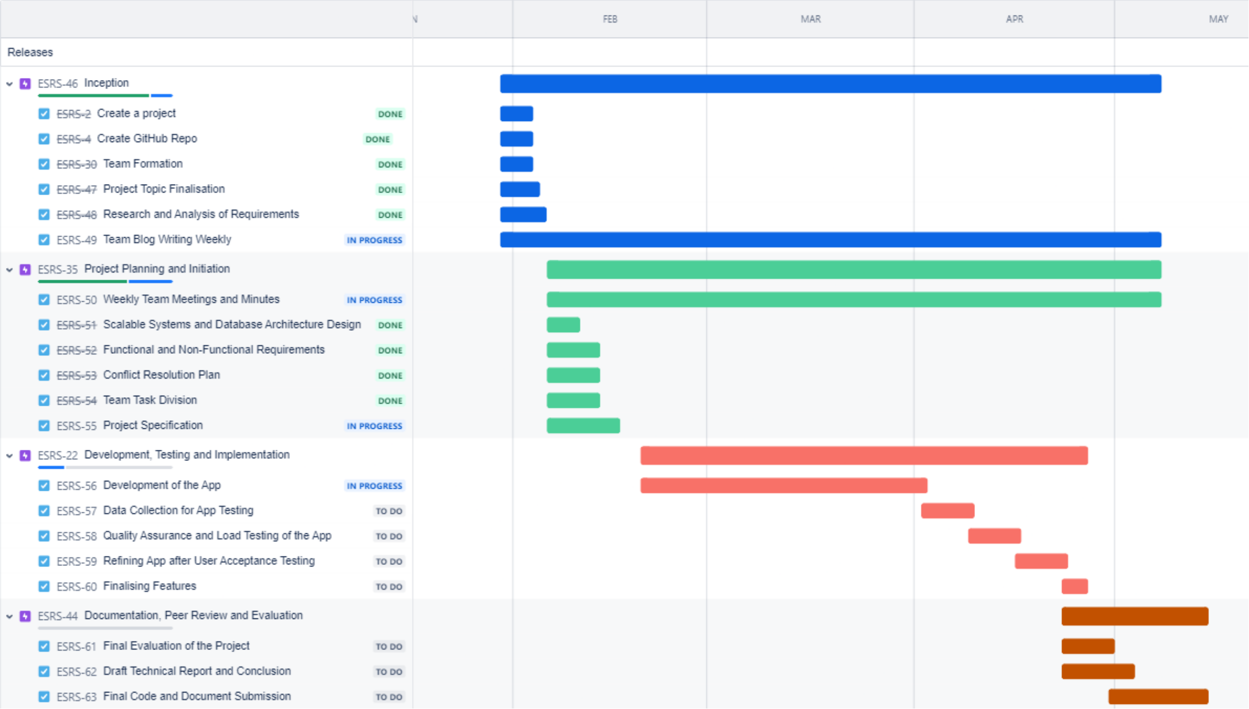
SN	C/E	Requirement	Description	Notes
1	C	The system must load search results in a timely fashion	This should load search results in under 2 seconds under normal operational circumstances.	N/A
2	C	User interface must be intuitive	Most users should be able to use the application without training.	N/A
3	C	The system should support a large number of books	It should support 100000 books minimum.	N/A
4	C	The system should be able to handle a large number of users without significant impact to performance	The system should have at least 99.9% uptime	N/A
5	C	The system should be capable of remaining up and operational without interruption	Should be able to run on chrome, safari, Firefox, edge	N/A

3.3 Software Requirements

*** S/L – Software / Libraries**

SN	S/L	Name	Details
1	S	Visual Studio / Pycharm CE	Integerated Development Environment for all sorts of programming
2	S	Python	Language and Interpreter used to program our web app
3	L	Django	Web Framework using Python, HTML
4	S	Nginx	Web hosting and Load Balancing
5	S	MySQL	Database Management System

4 Project Timeline - Gantt Chart



Releases

ESRS-46 Inception		
☑	~~ESRS-2~~ Create a project	DONE
☑	~~ESRS-4~~ Create GitHub Repo	DONE
☑	~~ESRS-30~~ Team Formation	DONE
☑	~~ESRS-47~~ Project Topic Finalisation	DONE
☑	~~ESRS-48~~ Research and Analysis of Requirements	DONE
☑	ESRS-49 Team Blog Writing Weekly	IN PROGRESS
ESRS-35 Project Planning and Initiation		
☑	ESRS-50 Weekly Team Meetings and Minutes	IN PROGRESS
☑	~~ESRS-51~~ Scalable Systems and Database Architecture Design	DONE
☑	~~ESRS-52~~ Functional and Non-Functional Requirements	DONE
☑	~~ESRS-53~~ Conflict Resolution Plan	DONE
☑	~~ESRS-54~~ Team Task Division	DONE
☑	ESRS-55 Project Specification	IN PROGRESS
ESRS-22 Development, Testing and Implementation		
☑	ESRS-56 Development of the App	IN PROGRESS
☑	ESRS-57 Data Collection for App Testing	TO DO
☑	ESRS-58 Quality Assurance and Load Testing of the App	TO DO
☑	ESRS-59 Refining App after User Acceptance Testing	TO DO
☑	ESRS-60 Finalising Features	TO DO
ESRS-44 Documentation, Peer Review and Evaluation		
☑	ESRS-61 Final Evaluation of the Project	TO DO
☑	ESRS-62 Draft Technical Report and Conclusion	TO DO
☑	ESRS-63 Final Code and Document Submission	TO DO

Figure 6: Estimated Project Timeline

5 Conflict Resolution Plan

5.1 Team Member Drops Out

In the event of a team member's dropout, we will redistribute their tasks among remaining team members to maintain project stability. If needed, we'll adjust the project timeline and keep the client informed. After resolving the issue, we'll update project documentation to record the dropout's contributions and the steps taken to ensure project continuity.

5.2 Deadline Slippage

In the event that a team member consistently misses deadlines, our approach involves first understanding the root causes, whether they be technical challenges, personal issues, or workload management problems, through open and empathetic communication. Once identified, we provide necessary support and assistance, which may include additional guidance or adjusting their responsibilities. If necessary, we will reallocate tasks among the team to accommodate their needs and revise the project timeline accordingly. Following issue resolution, we will document the steps taken to address the problem comprehensively.

5.3 Handling Disagreements

When disagreements arise about how to approach a task or project, our process involves having a thorough discussion to understand differing viewpoints. We aim to reach an agreement by considering everyone's perspective, evaluating the proposed methods based on project goals, and looking for compromises. If we can't resolve the issue, we might involve higher ups or seek mediation while keeping open and clear communication throughout.

5.4 Technical Issues

In the event of a GitHub repository issue, our plan includes identifying the problem's nature, attempting rollback changes if feasible, and, if necessary, restoring data from locally backed-up repository copies while maintaining thorough documentation. In the case of a data breach or security incident, our response encompasses immediate containment, investigation, notification, remediation, and comprehensive documentation to address the situation effectively.