REQUIREMENTS ANALYSIS DOCUMENT

Massive Dynamic

Liron Mizrahi - 708810
Marko Vidalis - 831987
Jason Chalom - 711985

Contents

Executive Summary

Massive Dynamic (MD) proposes a system which will help students trade, borrow and sell second hand books – mainly textbooks. The system's main objective is to allow users to easily search for, locate and purchase the books through it. It will have gamification aspects like Uber where users can rate other users. Additionally, it will include a reporting function in order to allow reporting of abusive users. There will be an administrative profile/s in the system, to aid for the maintenance of the system. The technology we have chosen to implement this on is a web based technology, namely the MEAN stack. The system will not cover using online payment methods and will only facilitate the locating and transferral of physical goods.

1. Introduction

1.1 Purpose of the system

MD is focused on building a system that will help facilitate the borrowing and/or selling of textbooks and other books between students. The system will not allow the transfer of copyrighted materials but will rather facilitate the lending and selling of second-hand textbooks. i.e. trading.

If a user has found a textbook which they would like to purchase, the seller is then contacted through contact details provided through the system and the purchaser can then buy the desired book.

The system will have a gamification function which will allow users to rate each other, similar to how the Uber app works. This rating will represent how users behave in their communications, transactions and reliability in the process of trading through the system. Any transactions made will be logged, which will allow the system to provide a history to the user.

Users can rate each other after a transaction is completed and also report users who abuse the system. These 'bad' users can then be suspended from the system, which is an administrative feature.

1.2 Scope of the system

The system will not include a mobile app. It will also not include any complex login profiles or file transferring. However there will be secure login with password salting and hashing. Besides regular user logins, there will also be admin logins with extra privileges.

It will consist of a web server and an html browser front-end. There will also be a post-transaction rating system in which other users are rated according to how well the transaction went. There will also be a reporting feature where a user may report another user. The system will also have a bug reporting system where users can report any bugs found on the site.

1.3 Objectives and success criteria of the project

- Secure login credentials using password salting
- Ability for admin to manage users
- Ability for users to communicate and trade books
- Ability to rate users
- Ability to report users

1.4 Measures of Success:

Efficiency:

The system must have efficient work flows for users and admins.

Responsiveness:

The system must be responsive and perform well whilst delivering content to multiple users.

Quality:

Users and stakeholders must be happy with what is delivered every sprint and the final deliverable.

2. Current system

The current system of obtaining any books or textbooks for students is by going to either a library or to a book store, for both first and second-hand books. This means that there are advantages to the sellers of books as there are many more buyers in comparison, due to the fact that there are limited centralised locations at which a specific book may be found. However it is not always easy to find the exact book a customer is looking for. While this current system has worked in the past, there are very few receipts and logs of transactions made, which makes tracing any purchases very difficult. Finding books that a specific student is looking for is also very difficult and time consuming and this is what MD serves to solve through this system.

3. Proposed system

3.1 Overview

The system will consist of a server which will serve webpages to users and also process user requests. EJS will be used as the templating engine which will dynamically render webpages to the user. MongoDb will be used for the database backend. User information, including salted passwords, transaction logs and any other functional system information will be stored here.

3.2 Functional Requirements

3.2.1 As a user, I can create an account on the system.

3.2.2 As a user, I can login to the system.

3.2.3 As a user, I can change my password.

3.2.4 As a user, I can explore books other users have posted to the system.

3.2.5 As a user, I can post books Im willing to trade or loan on the system.

3.2.6 As a user, I can post books Im willing to sell on the system.

3.2.7 As a user, I can rate my experience during a specific transaction.

3.2.8 As a user, I can report a user who has abused the system.

3.2.9 As an Admin, I can reset user passwords.

3.2.10 As an Admin, I can see and review which users have been reported for bad behaviour.

3.2.11 As an Admin, I can reverse transactions when something unexpected happens in the system.

3.2.12 As a user, I can contact admin on the system.

3.2.13 As an Admin, I can see messages sent by users and respond to them.

3.3 Nonfunctional requirements

3.3.1 Usability

The system must have a pleasing interface which is simplistic in design and intuitive to use.

3.3.2 Reliability

The system must support multiple users simultaneously and not be prone to hanging or crashing during operation.

3.3.3 Supportability

The web based approach allows the system to be opened by any HTML5 supporting web browser. This enables the system to run on many different devices and device eco-systems.

3.3.4 Implementation

The server backend of the system will be built using the MEAN stack, which includes MongoDb for the database, nodejs for the server backend, and ExpressJS for the frontend. We want to also use PassportJS to help us with user logins. Our tests will be done with Mocha and Chai libraries. WinstonJS will be used to generate server logs to help us maintain server operation.

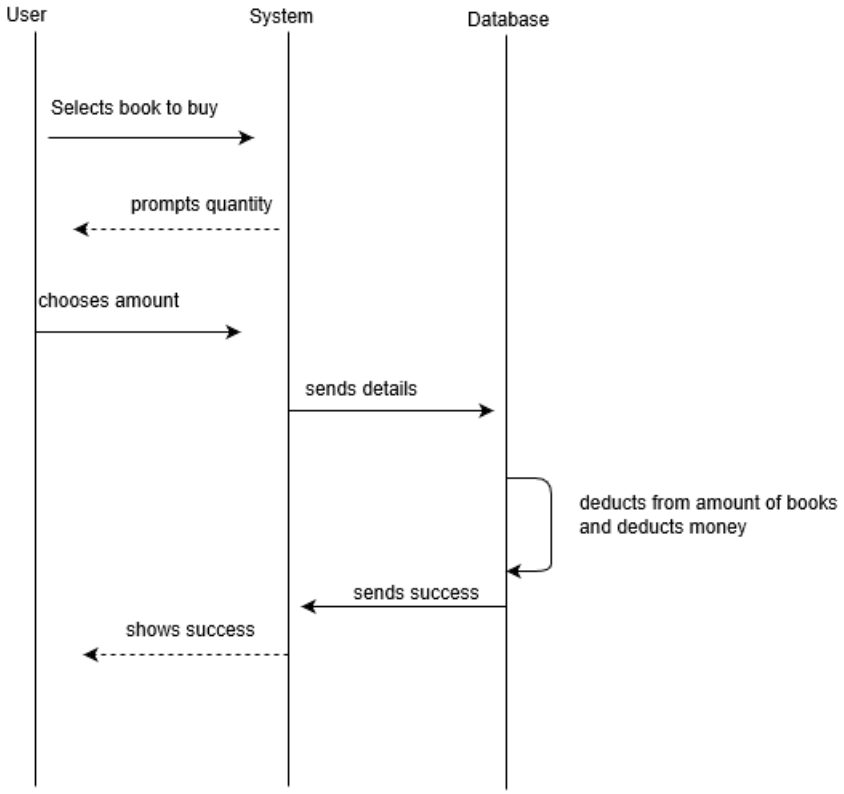
4. System Diagrams

4.1 Sequence Diagrams

In this section, the Sequence diagrams of the various system functions are documented.

Purchase Sequence Diagram:

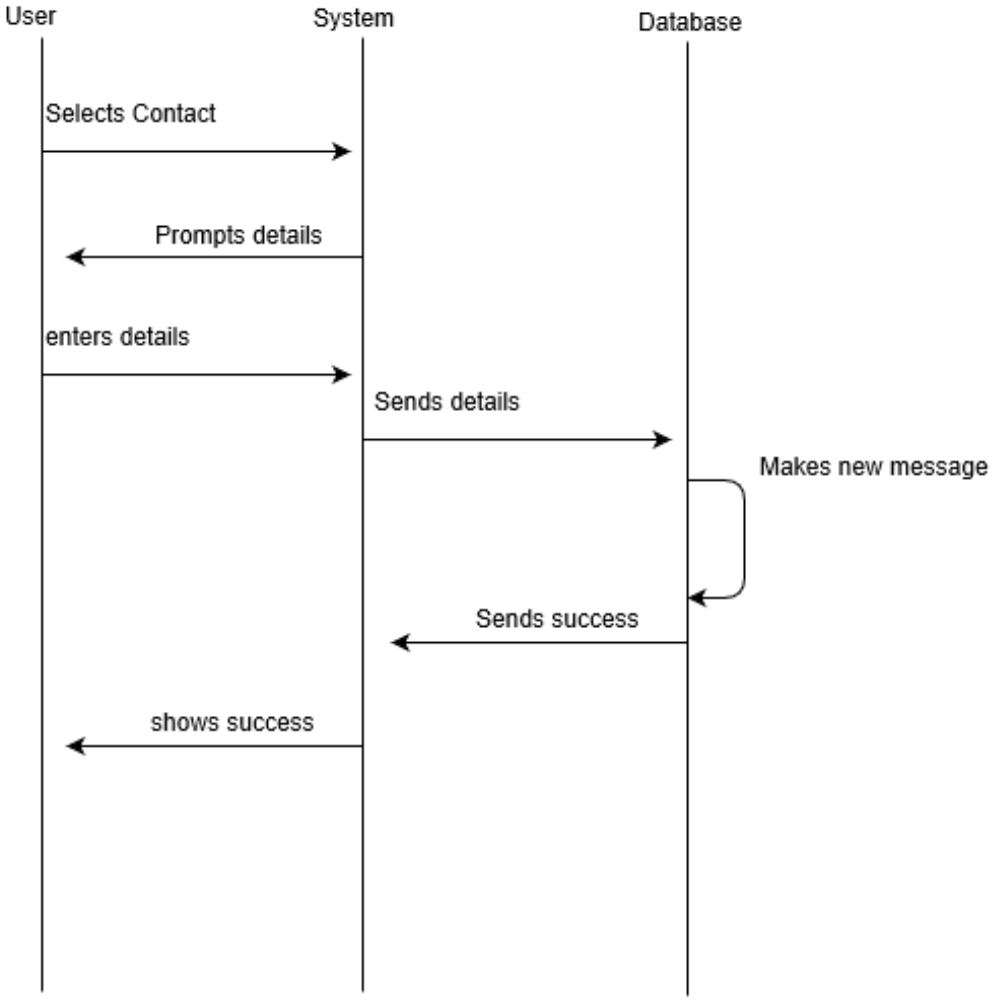
The sequence diagram below describes the process which the system follows when a user is requesting to make a purchase of a specific book.



The purchaser selects a book which they would like to buy. The user is then prompted to enter the quantity of the specific book which they would like to purchase. Once the quantity is entered, the database inventory and balance is updated. The user is then informed of the successful transaction made.

Contact Admin sequence diagram:

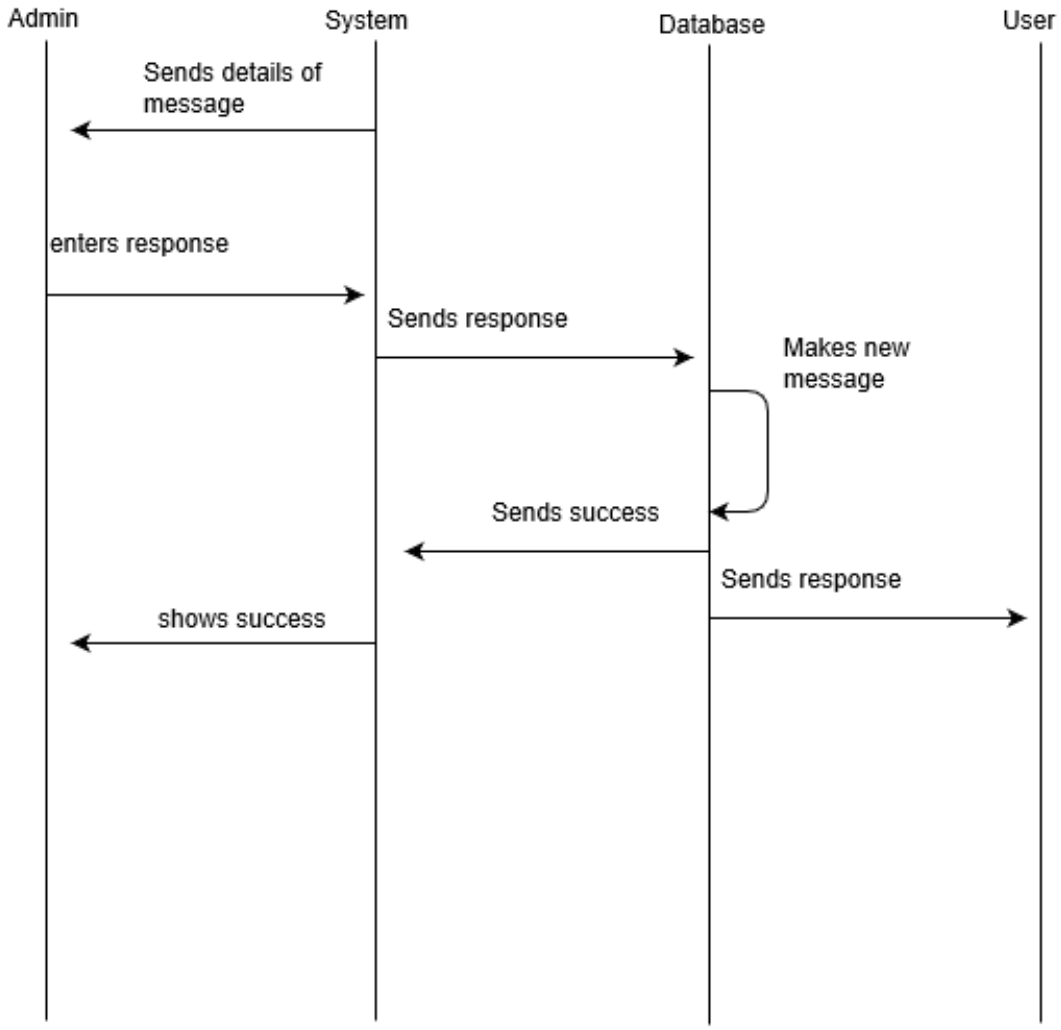
The following sequence diagram describes the process followed by the system when a user requests to come into contact with an administrator.



The user selects the option to contact an administrator. The system then asks the user for their personal details, and these details are sent to the database which creates a message. The user is then informed that their request was submitted successfully.

Contact User sequence diagram:

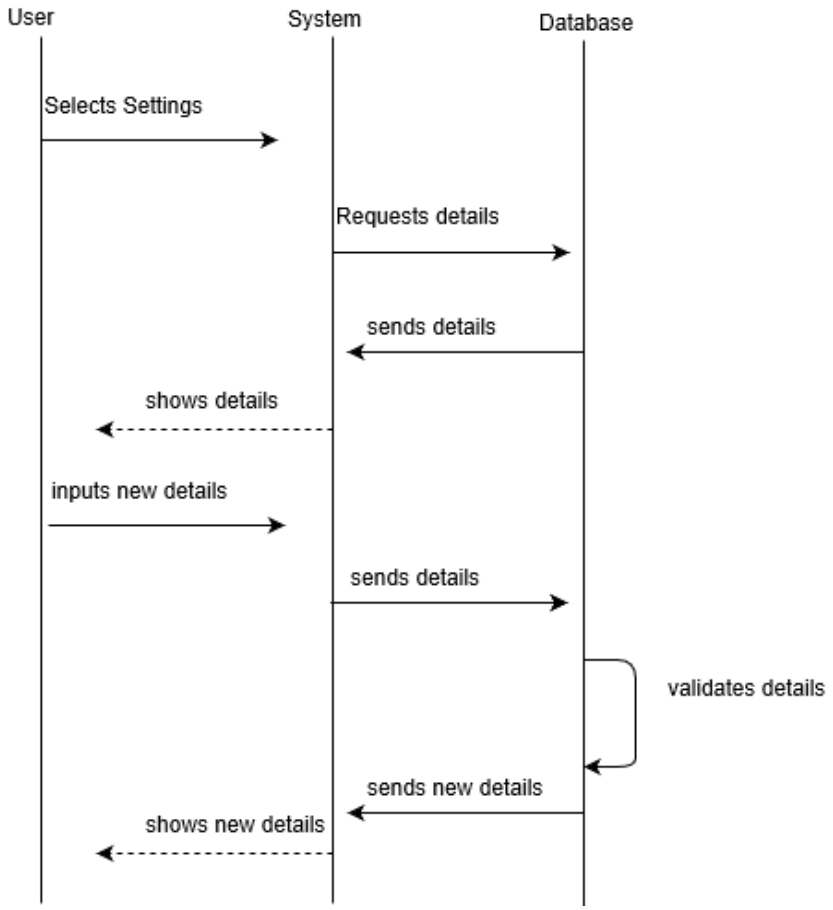
The following sequence diagram shows the process of the system contacting a user with the administrator's response to the system request.



The system needs to send a notification to a user. Therefore, it sends a message to the administrator with the nature of the message to be sent. The administrator responds to this request. The system then sends the response to the database. A new message is then generated and sent to the user and a success notification is sent to the system and forwarded to the administrator.

Update user details sequence diagram:

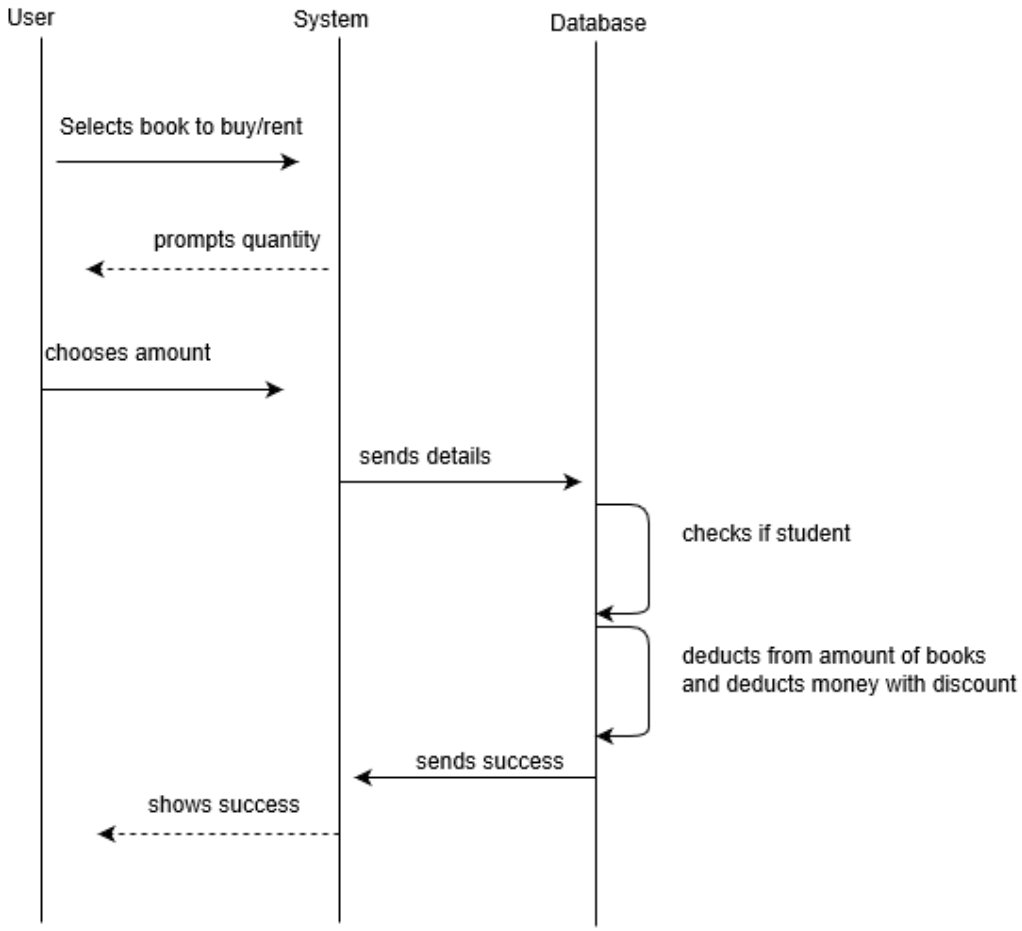
The following sequence diagram depicts the process followed by the system and user upon the user's request to update their details in the systems database.



The user needs to update their details on the system. They select the 'settings' option on the page which then query's the user's existing details from the database. These details are displayed to the user on the page. The user can then enter their new details and send them through. The system processes the details and forwards them to the database where these new details are stored. Data validation is performed and the new details are stored, if valid, in the database. The user is then notified by the system of the successful update.

Discount Evaluation Sequence Diagram:

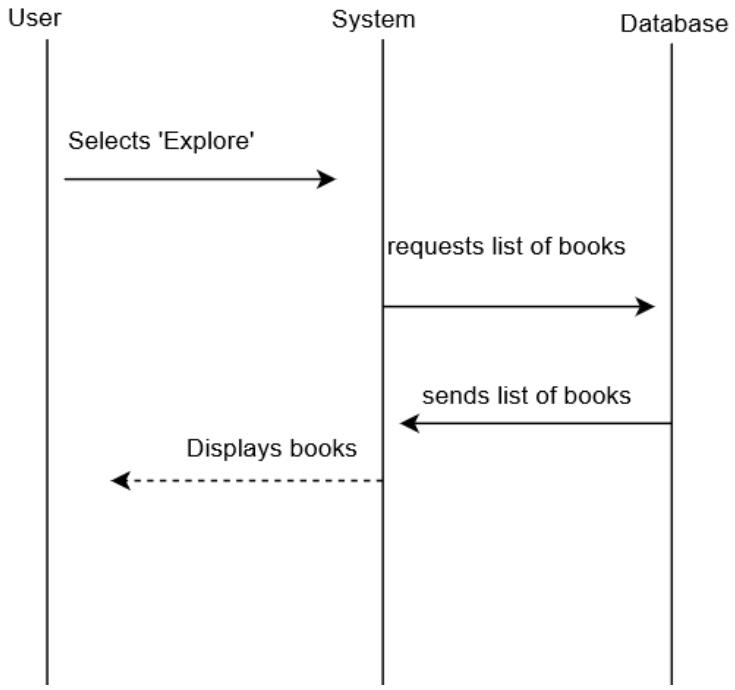
The following sequence diagram shows the process which the system follows in order to determine a discount for a purchaser.



The user selects a book to buy or rent on the page. The system then prompts the user to enter the quantity of the books required by the user. These details are then sent to the database. The database checks to see if the user is a student. If so, the number of remaining available books in the database is updated and the value of the transaction made by the user is deducted from the user's account, less the discount calculated by the database. The user is then notified of the successful transaction by the system.

Explore Sequence Diagram:

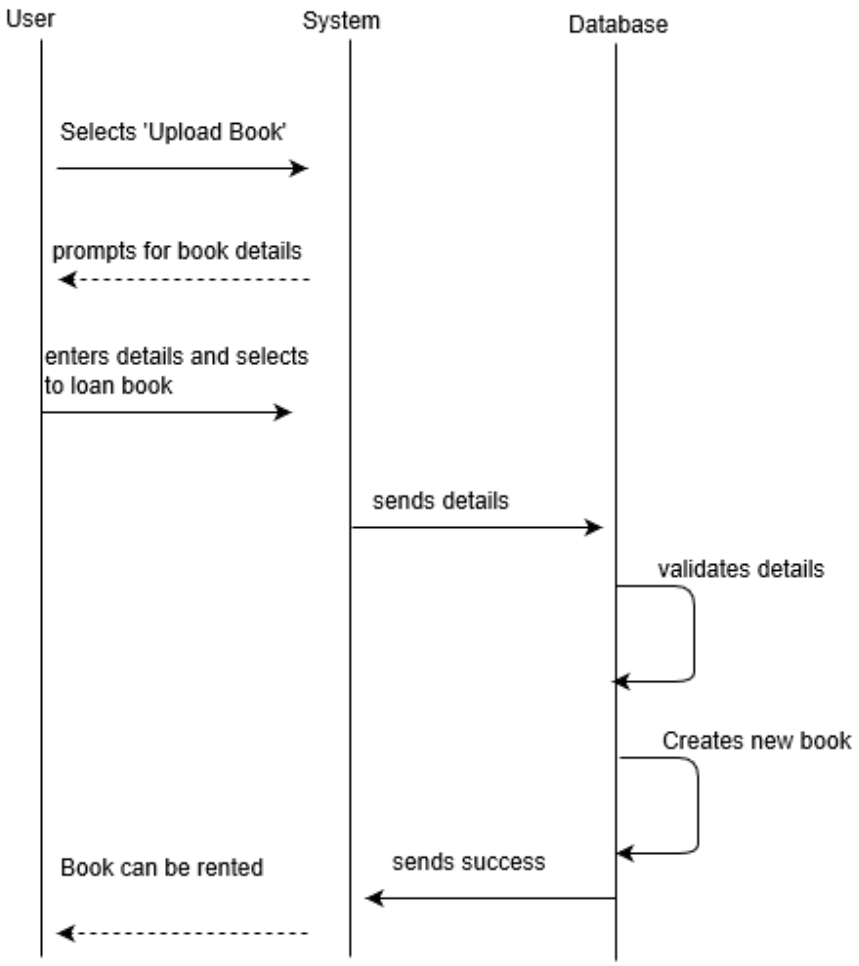
The following sequence diagram shows the process of the Explore page being populated on the site.



The user selects the ‘explore’ button on the site. The system then requests the list of books and their details from the database. The database returns all these details and the system displays the books once loading the explore page on the user’s side of the system.

Loan Sequence Diagram:

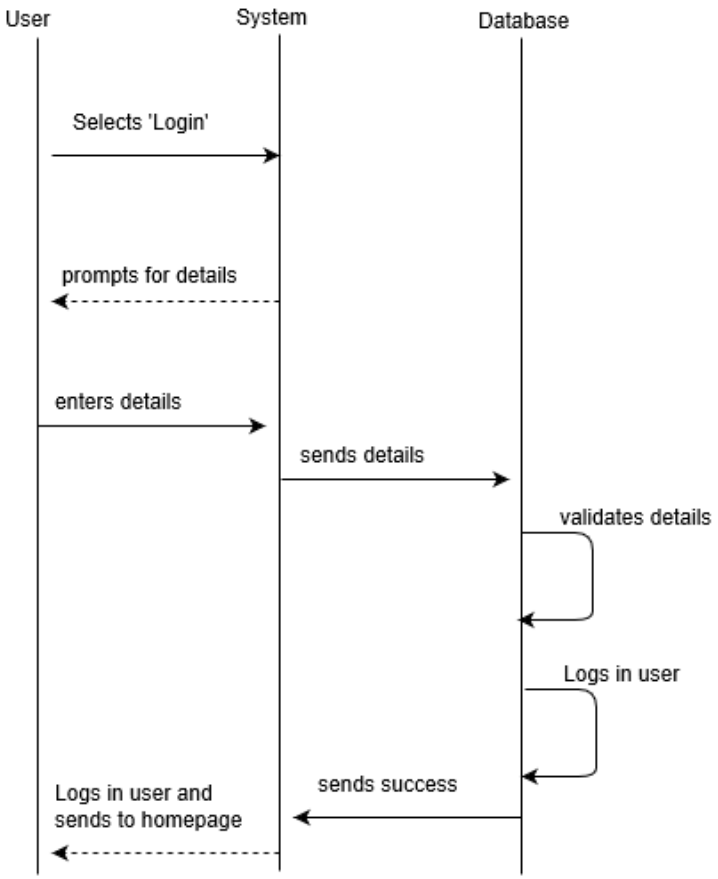
The following Sequence diagram shows the process followed by the stakeholders when a user requests to upload a book for loaning on the system.



The user requests to upload a new book. The system requests all the details of the book from the user. It then sends the details, once entered, to the database. After data validation is performed on the details, the database creates a new book in the datastore and responds with a success message to the system. The book can now be rented out to other users.

Login sequence diagram:

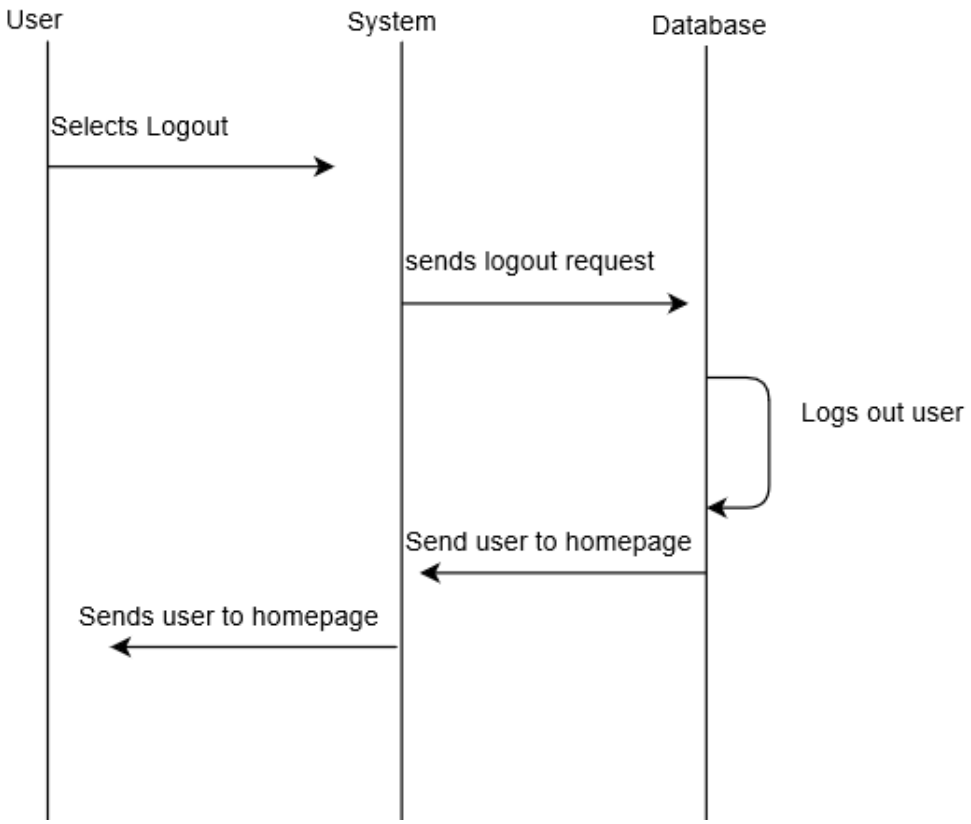
The following Sequence Diagram shows the process followed by the system when a user requests to login.



The user requests to login to the system. The system processes the submitted login details and sends them to the database. If the validation of the credentials is successful, the database logs the user in. The system is notified by the database of the successful login attempt, and the user is redirected to the home page.

Logout sequence diagram:

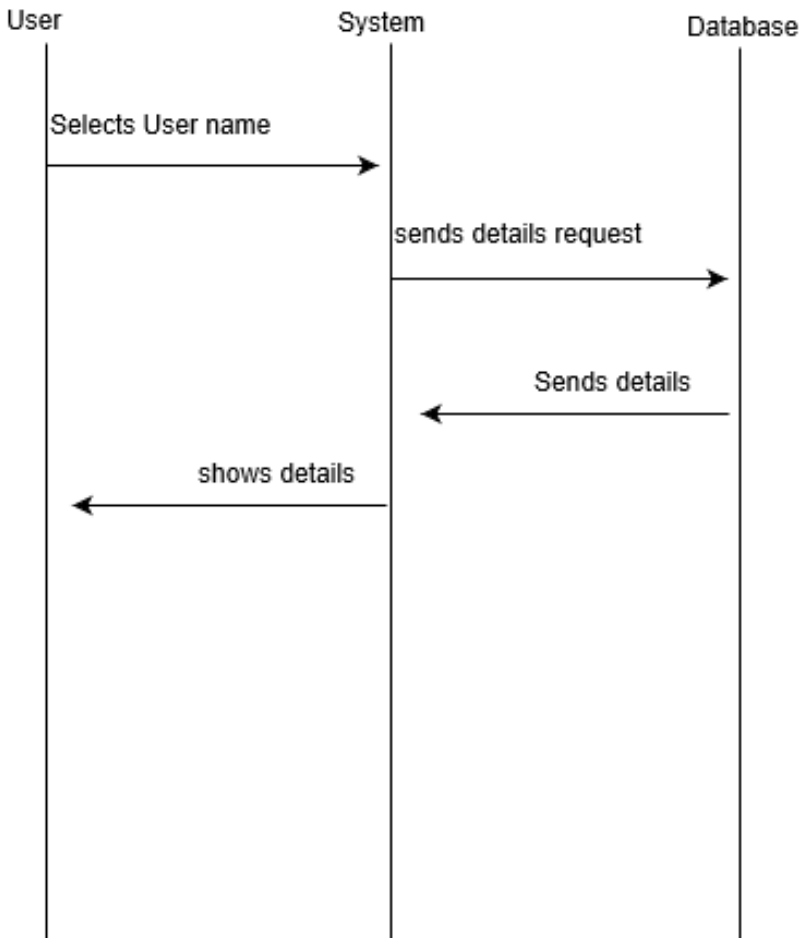
The following sequence diagram depicts the process of a user logging out of the system.



The user selects the logout button. The system then sends a logout request to the database, which logs the user out and sends a notification to the system that the user must be sent to the homepage. The system then redirects the user to the homepage.

Profile Sequence Diagram:

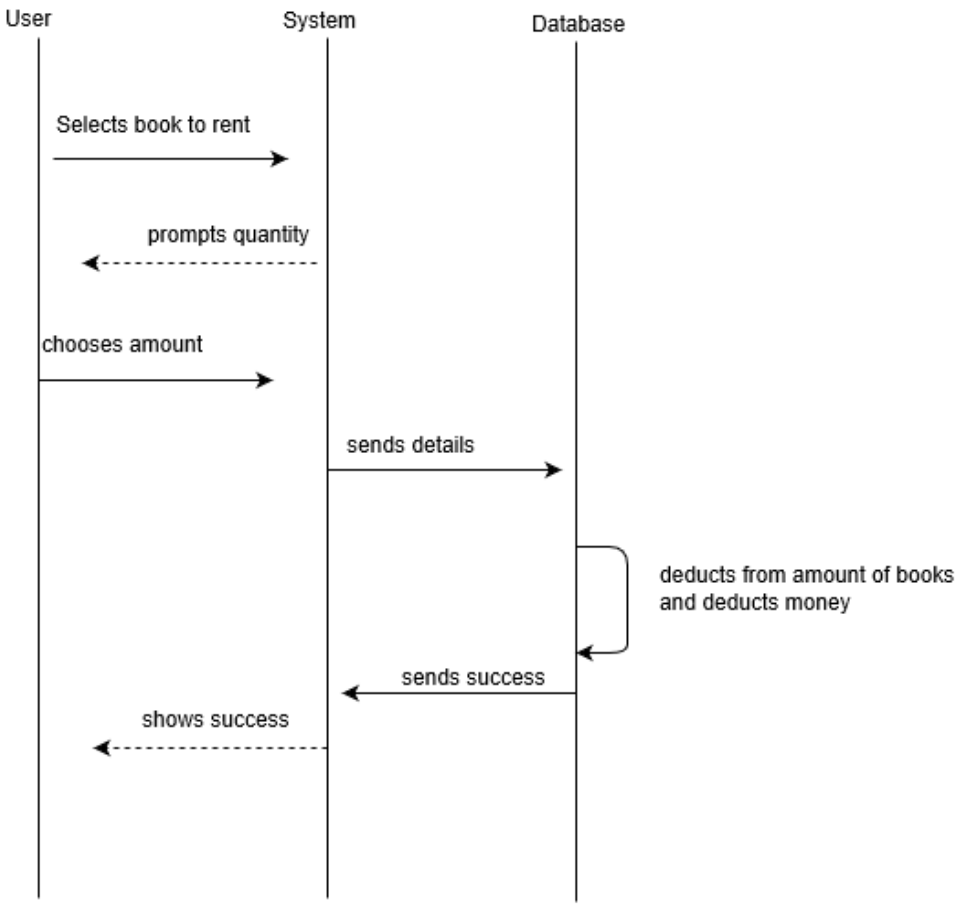
The following sequence diagram shows the process of the system displaying a users details upon request.



The user selects their name on the system page. The system then requests their details from the database, which responds with all the user's details. The system then displays these details to the user.

Rent sequence diagram:

The following sequence diagram shows the process of a user renting a book out from the system.

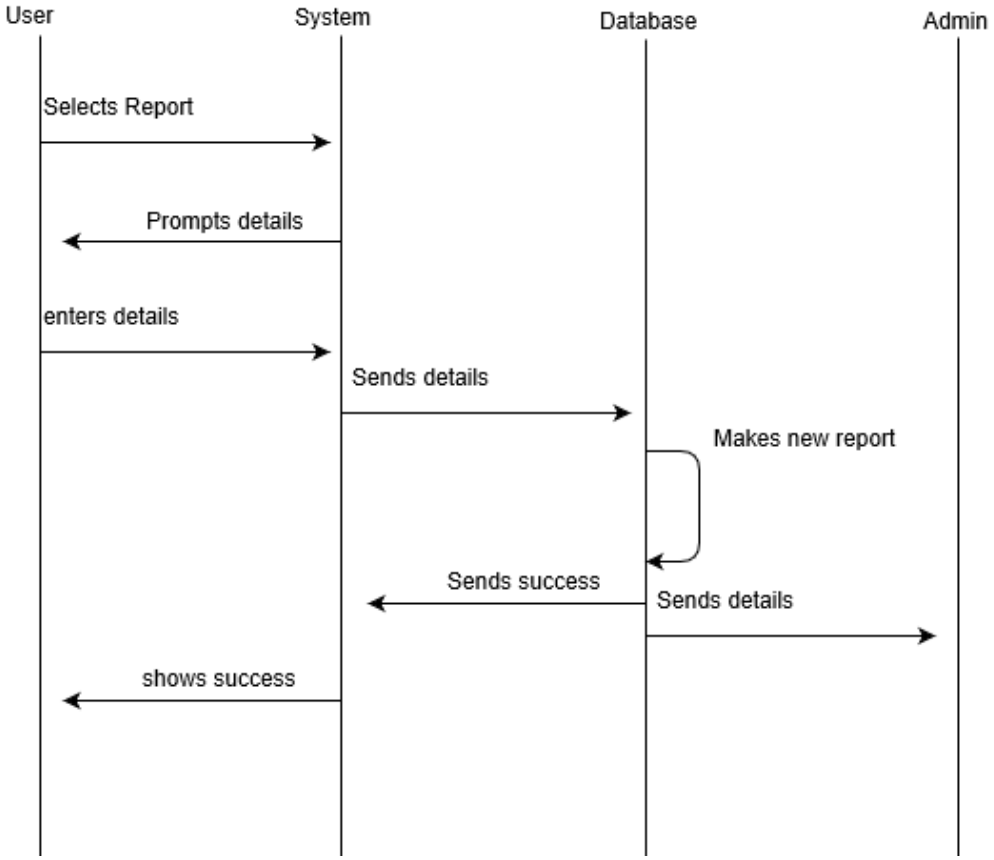


The user selects the book which they would like to rent out. They are then prompted for the quantity by the system. Once the quantity is entered, the system sends the details to the database. The database then reduces the availability of the book and monetary value from the datastores accordingly.

The system is then sent a success notification, which is forwarded to the user.

Report User Sequence Diagram

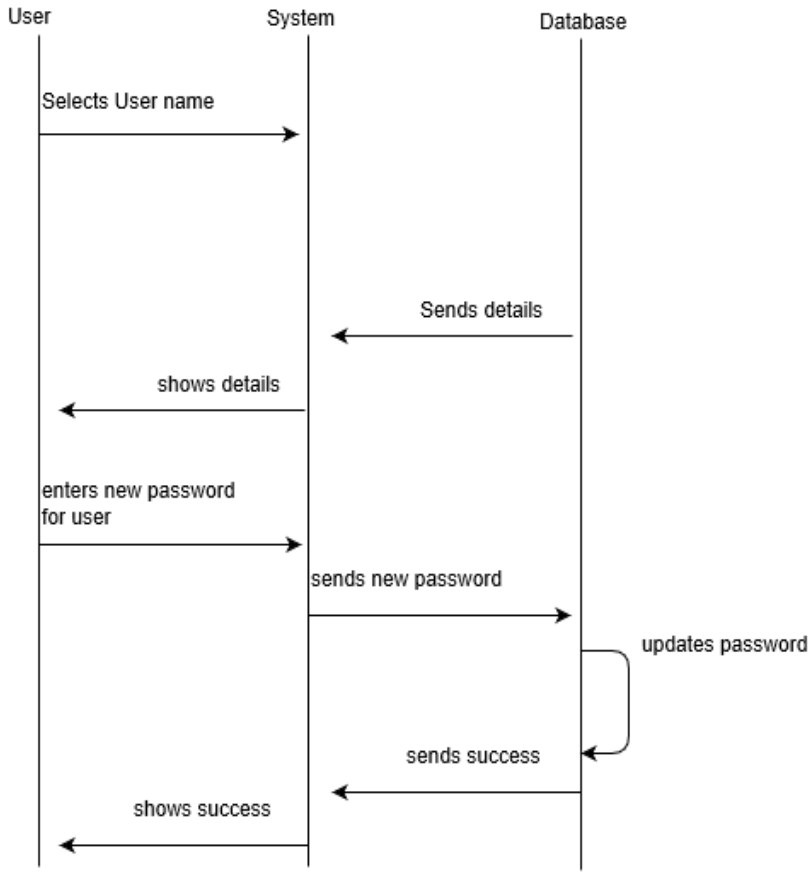
The following sequence diagram shows the process followed by a user to report another user through the system.



The user requests to report another user on the system. The system requests the details of the other user. These details are then forwarded to the database, which generates the report and sends it to the administrator. The user is notified of the report creation.

Reset Password sequence diagram:

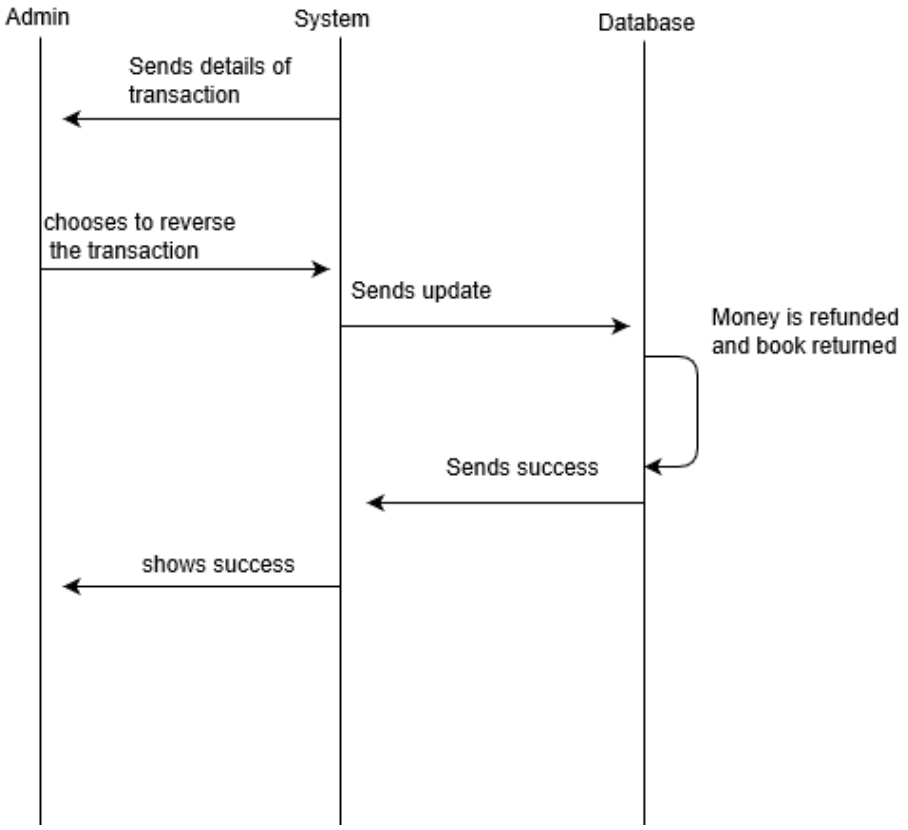
The following sequence diagram describes the process followed by a user that wishes to reset their password.



The user selects their user name on a system page. The database then returns the users details, and the system displays the details on the user's screen. The user then enters their new password, which the system sends to the database. The database updates the existing password and the user is notified of the successful update.

Reverse Transaction Sequence Diagram

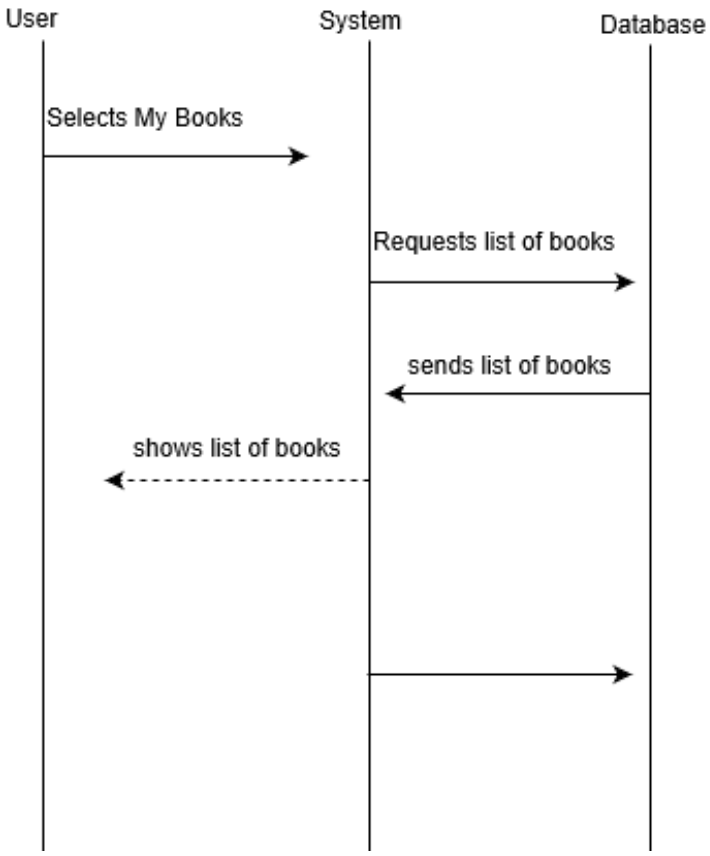
The following sequence diagram shows the process followed for a transaction on the system to be reversed.



The system sends the details of the transaction to the administrator. The administrator selects to reverse the transaction. The update is sent to the database and the money is refunded to the user, and the book is added back into the database. The administrator is then notified of the successful reversal.

View uploaded books sequence diagram:

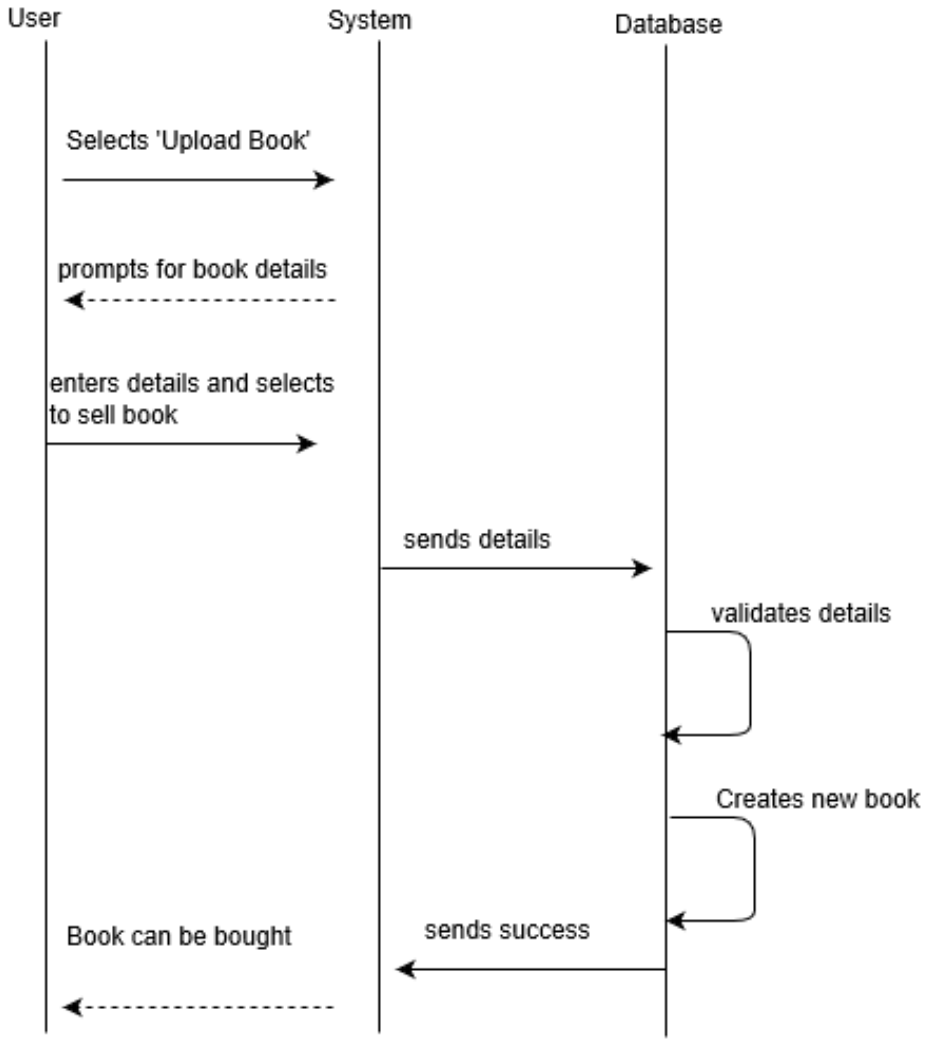
The following sequence diagram shows the process followed to display the books uploaded by a user.



The user requests to view all the books they have uploaded. The database returns all the details of these books and displays them to the user through the system.

Sell sequence diagram

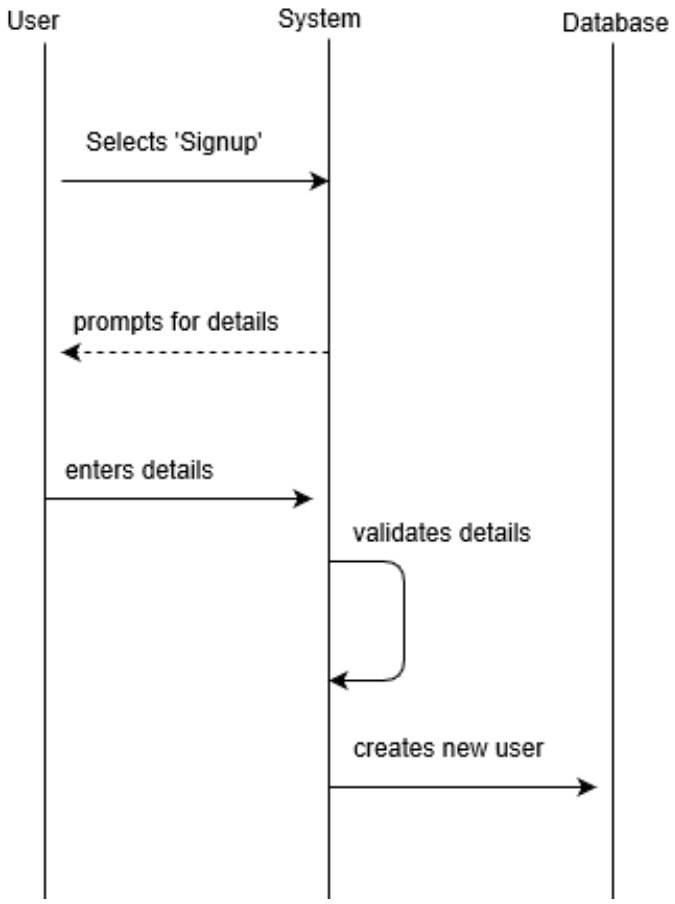
The following sequence diagram shows the process followed in the system when a book is being uploaded to be sold.



The user selects the ‘upload book’ option. The user then enters all the details of the book which are then sent to the database. The database validates the data and creates a new book in the book datastore. The system is notified of the successful creation of the book which is then forwarded to the user.

Sign up sequence diagram:

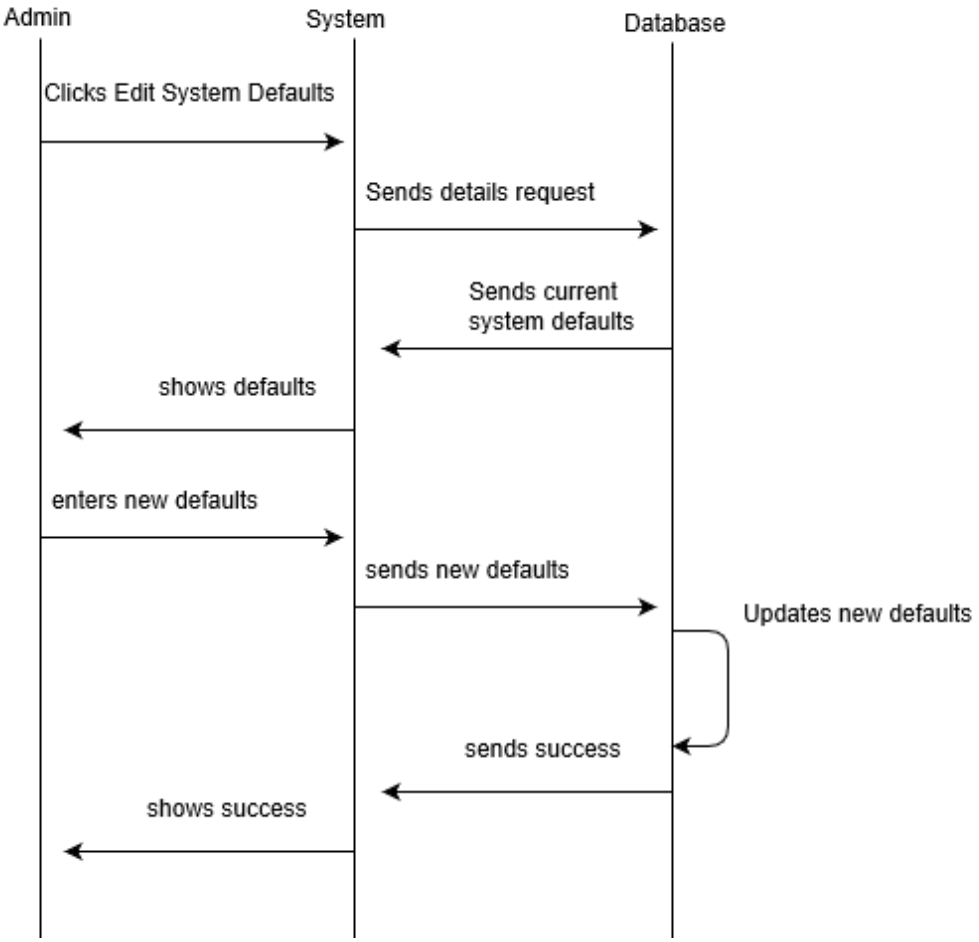
The following sequence diagram shows the process followed to sign a new user up on the system



The user selects the option to sign up. They enter their details onto the system, where the system validates them and sends them to the database. The new user is then created and stored in the datastore.

System defaults sequence diagram

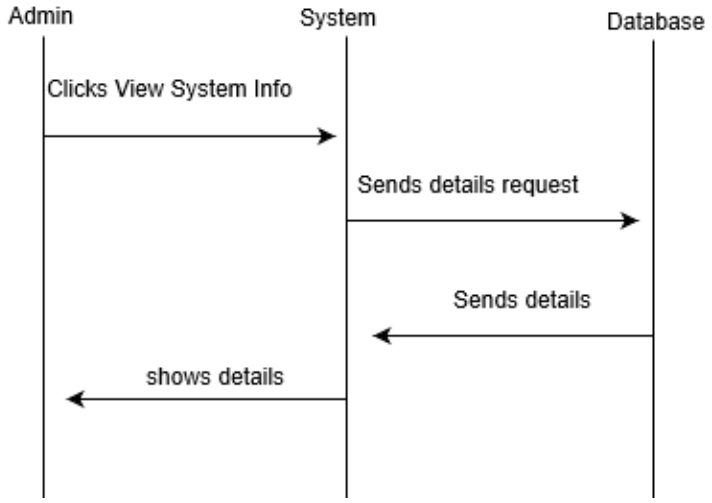
The following diagram shows the process followed to update the system defaults on a user's browser.



The user selects the system defaults option. The details are returned from the database and displayed to the user by the system. The user enters the new defaults preferences and these are sent to the database by the system. The system and user are then notified of the successful update.

System information sequence diagram

The following diagrams shows the process of requesting the system information.

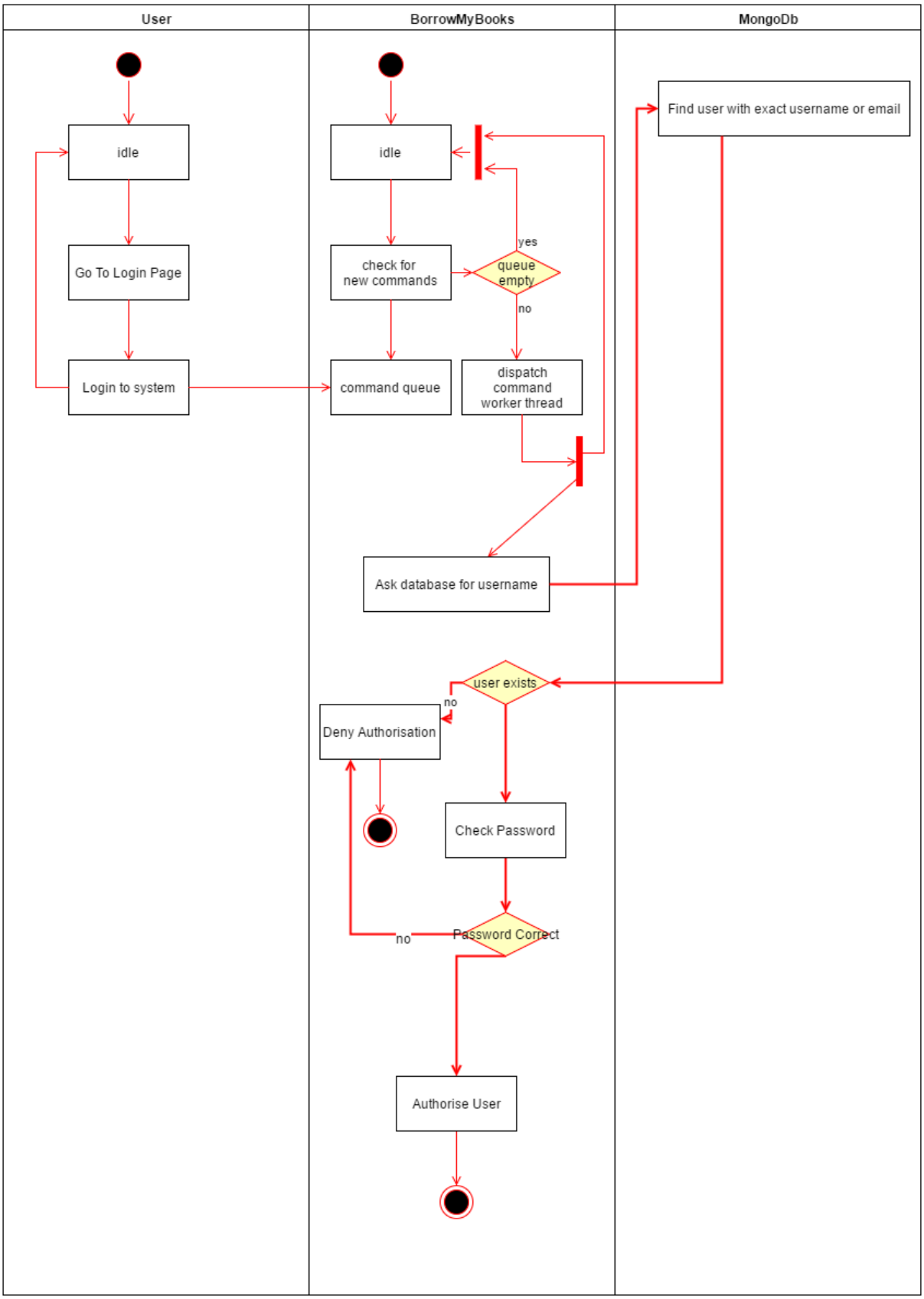


The administrator chooses to view the system information. The system requests the system information from the database and these details are sent to the system. They are then displayed to the administrator.

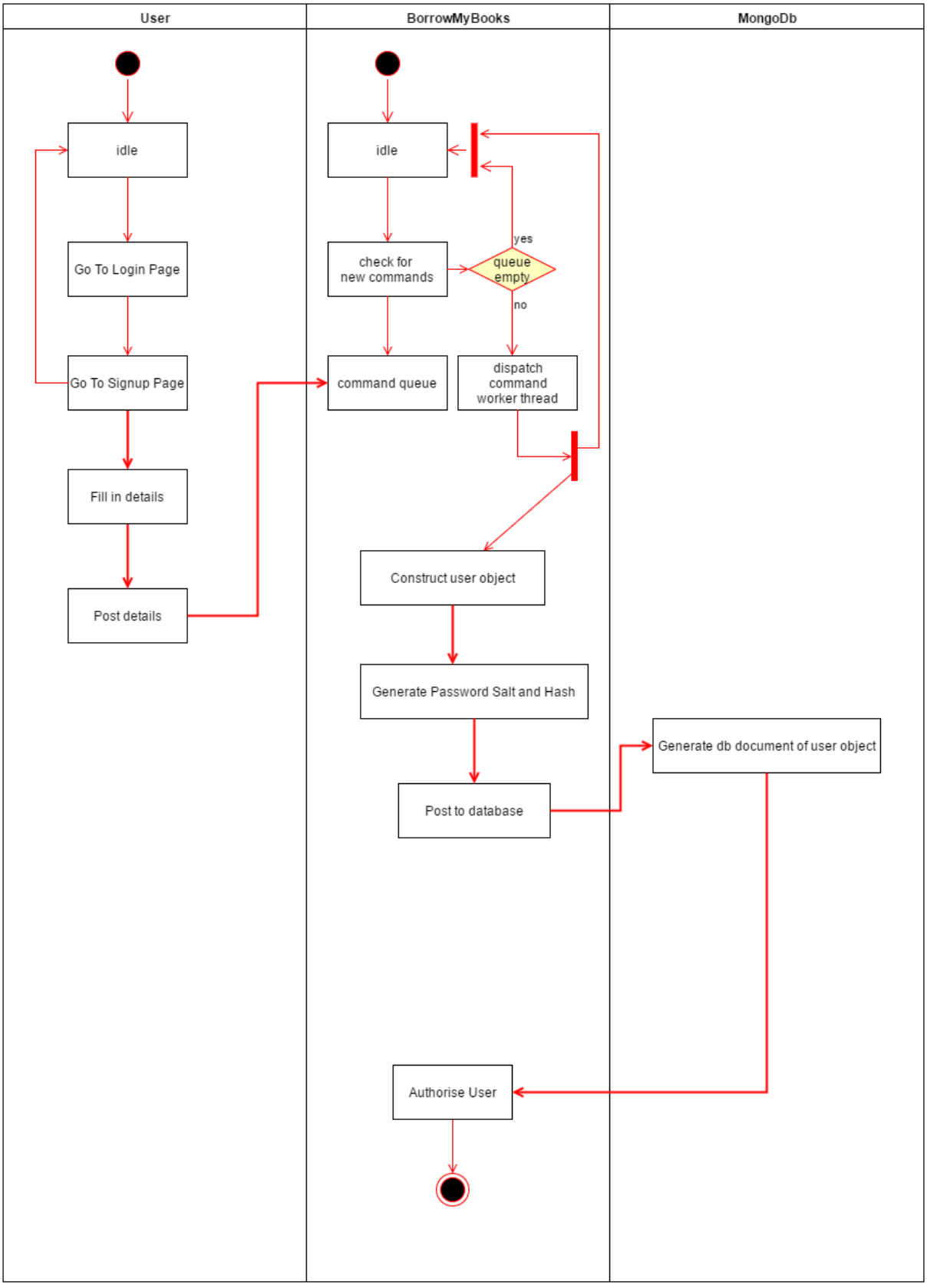
5. Activity Diagrams

In this section, the Activity Diagrams relating to the system's functionality are documented.

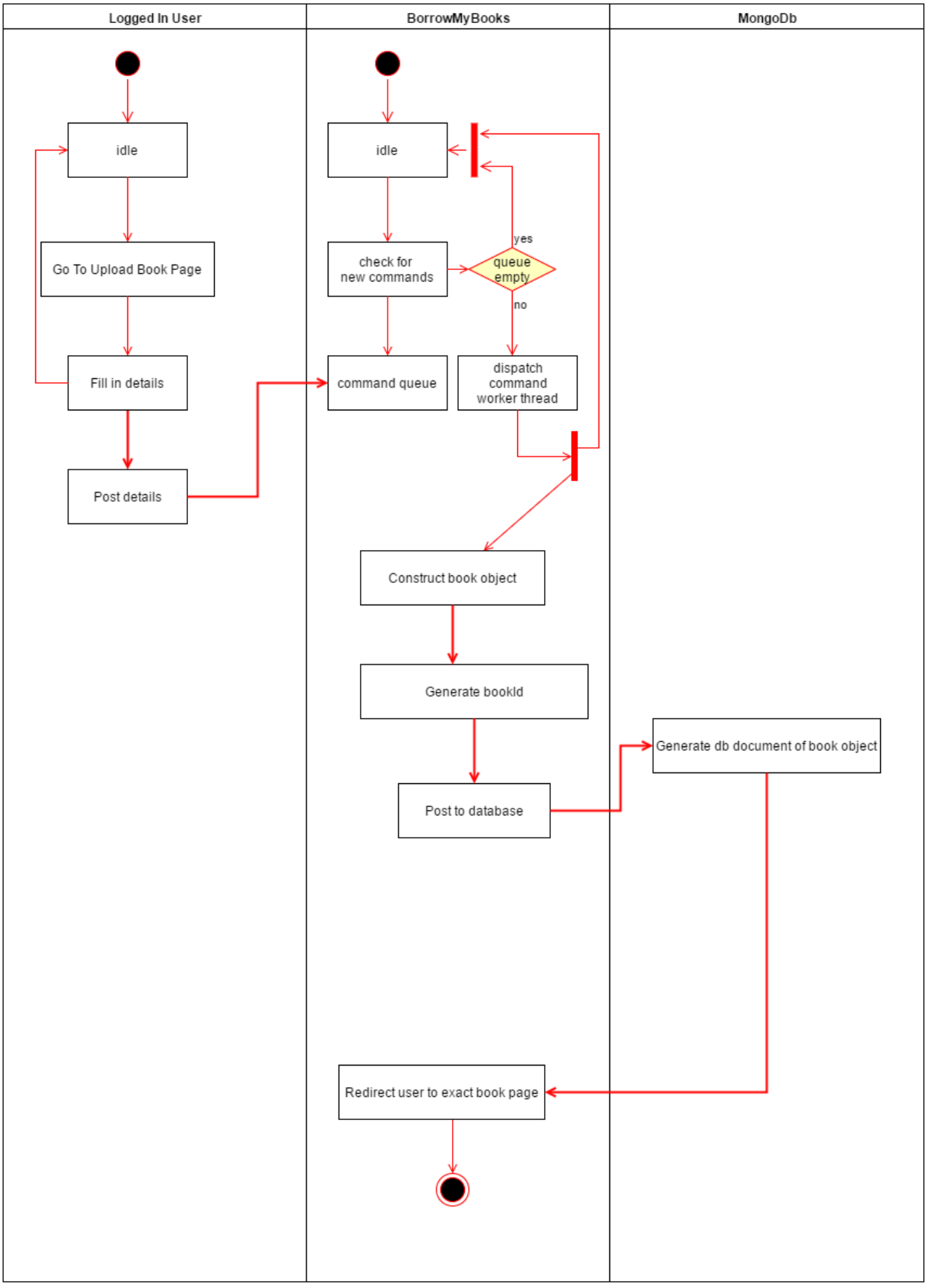
5.1 Login Activity Diagram



5.2 Sign Up Activity Diagram

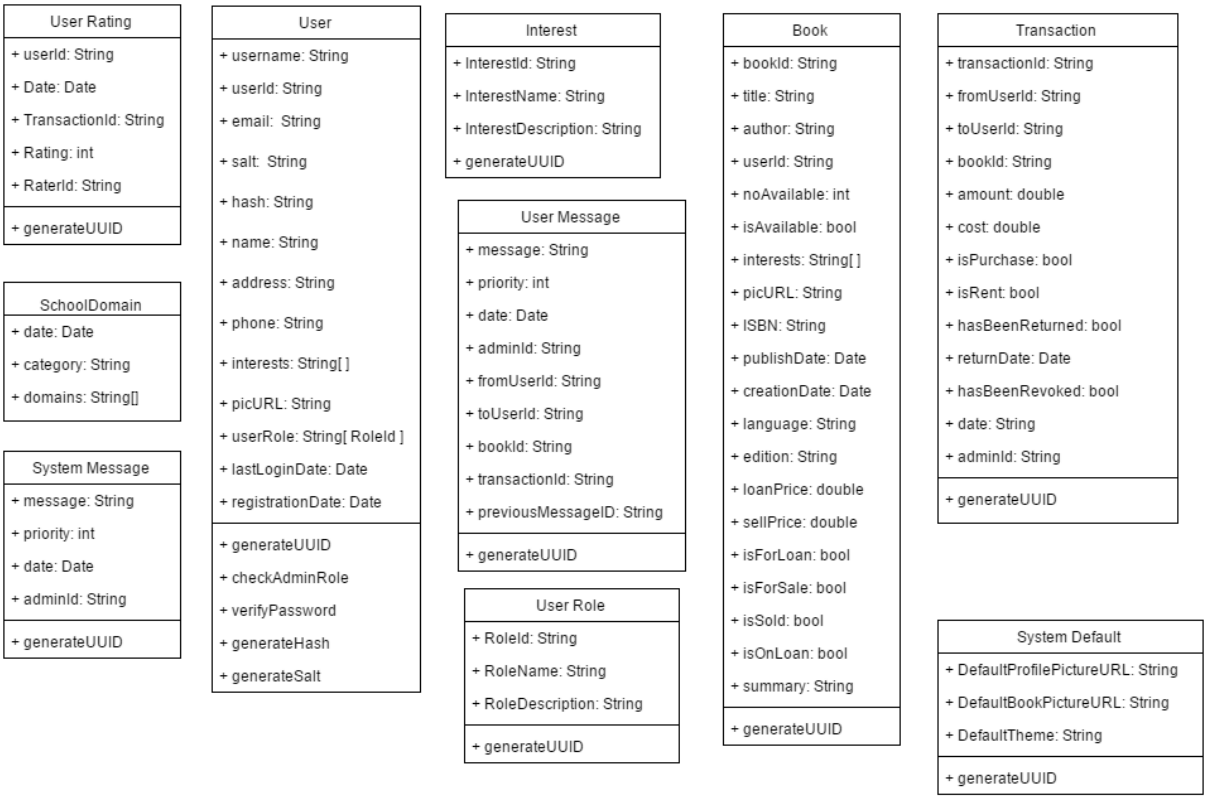


5.3 Upload Book Activity Diagram



6. Database Class Diagram

The following diagram is a Class Diagram depicting the layout of the Database used for the Borrow My Books system.



7. Use Cases & requirements

The following is a description of the use cases which describe the system's core functionality. They will be used in the test cases in section 7.

7.1 Use cases

Actor	Actor Goals	Use case Name
User	To create an account on the system.	Signup(UC 1)
User	To login to the system.	Login(UC 2)
User	To change my password.	ChangePassword(UC 3)
User	To explore books other users have posted to the system.	Explore(UC 4)
User	To post books Im willing to trade or loan on the system.	Loan(UC 5)
User	To post books Im willing to sell on the system.	Sell(UC 6)
User	To rate my experience during a specific transaction.	Rate(UC 7)
User	To report a user who has abused the system.	ReportUser(UC 8)
User	To contact admin on the system.	ContactAdmin(UC 9)
Admin	To reset user passwords.	ResetPassword(UC 10)
Admin	To see and review which users have been reported for bad behaviour.	ReviewUsers(UC 11)
Admin	To reverse transactions when something unexpected happens in the system.	ReverseTransactions(UC 12)
Admin	To see messages sent by users and respond to them.	ContactUser(UC 13)
User	To buy uploaded books	Buy(UC 14)
User	To rent uploaded books.	Rent(UC 15)
User	To change my details of my profile.	Details(UC 16)
User	To view profile of other users.	Profile(UC 17)
User	To view books uploaded by me.	SelfUploadedBooks(UC 18)
User	To get a discount as a student.	Discount(UC 19)
User	To logout of my session	Logout(UC 20)
Admin	To view information about the system	SystemInfo(UC 21)
Admin	To edit the system defaults	SystemDefaults(UC 22)

7.2 Requirements

Identifier	Priority	Requirement
REQ 1	10	The system shall create a new user with the given details.
REQ 2	10	The system shall login a user if the username and password submitted are valid
REQ 3	9	The system shall allow users to browse uploaded books.
REQ 4	9	The system shall allow users to see details of a chosen book.
REQ 5	10	The system shall allow a user to buy/rent a book given they are logged in, and it is not a book the user has uploaded.
REQ 6	6	The system shall allow users to change their details at any time.
REQ 7	5	The system shall allow users to browse other users profiles.
REQ 8	10	The system shall allow users to upload books to sell/rent.
REQ 9	6	The system shall allow users to browse their uploaded books.
REQ 10	8	The system shall allow users to rate transactions.
REQ 11	7	The system shall allow users to send messages to admins.
REQ 12	7	The system shall give students discounts.
REQ 13	10	The system shall allow users to logout from a seesion.
REQ 14	8	The system shall allow admins to view system information.
REQ 15	9	The system shall allow admins to edit the system defaults
REQ 16	10	The system shall allow admins to reverse transactions, within reason.
REQ 17	7	The system shall allow admins to respond to user messages.
REQ 18	4	The system shall allow users to report other users for abusive behaviour.
REQ 19	4	The system shall allow admins to review reported users and give neccassary punishment.

7.3 Traceability Matrix for Use cases and Requirements

The below traceability matrix shows the relationships between the various use cases and requirements.

REQ'S	**PW**	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11
REQ 1	10	X	X	X								
REQ 2	10	X	X	X								
REQ 3	9				X	X	X					
REQ 4	9				X	X	X					
REQ 5	10				X	X	X					
REQ 6	6										X	
REQ 7	5											
REQ 8	10				X	X	X					
REQ 9	6											
REQ 10	8							X				
REQ 11	7									X		
REQ 12	7											
REQ 13	10											
REQ 14	8											
REQ 15	9											
REQ 16	10											
REQ 17	7									X		
REQ 18	4								X			X
REQ 19	4								X			X
Max PW		10	10	10	10	10	10	8	4	7	6	4
Total PW												

Traceability Matrix Continued:

REQ'S	UC12	UC13	UC14	UC15	UC16	UC17	UC18	UC19	UC20	UC21	UC22
REQ 1											
REQ 2											
REQ 3											
REQ 4											
REQ 5			X	X							
REQ 6					X						
REQ 7						X					
REQ 8			X	X							
REQ 9							X				
REQ 10											
REQ 11		X									
REQ 12								X			
REQ 13									X		

REQ 14										X	
REQ 15											X
REQ 16	X										
REQ 17		X									
REQ 18											
REQ 19											
Max PW	10	7	10	10	6	5	6	7	10	8	9
Total PW											

8. Test Cases

The following test cases elaborate on the testing processes followed by the Borrow My Books system by use case, and their expected results.

Login test case:

Test-case Identifier:	**TC-1**
Use Case Tested:	**Login(UC-2)**
Pass/fail Criteria:	**The test passes if the user enters a valid username and password into the system, within less than the specified amount of allowed failed attempts.**
Input Data:	**Alphanumeric username, alphanumeric password.**

Test Procedure:	Expected Result:
Step 1. Type in a correct username and incorrect password.	**System shows unsuccessful login attempt message. records unsuccessful attempt in the database; prompts the user to try again**
Step 2. Type in an incorrect username and any password	**System shows unsuccessful login attempt message. records unsuccessful attempt in the database; prompts the user to try again**
Step 3. Type in a correct username and password	**System shows a successful login notification. Redirects the user to the home page of the site.**

User sign up test case:

Test-case Identifier:	**TC-2**
Use Case Tested:	**Signup(UC-1)**
Pass/fail Criteria:	**The test passes if the user successfully creates an account with a valid email address and password**
Input Data:	**User Data (username, userId, email, name, address, phone, interests, picUrl)**

Test Procedure:	Expected Result:
Step 1. Try to sign up with some required fields missing	**System shows unsuccessful account creation message. Records unsuccessful attempt in the database; prompts the user to try again with valid credentials.**
Step 2. Try to sign up with all fields completed but erroneous data in some fields	**System shows unsuccessful account creation message. Records unsuccessful attempt in the database; prompts the user to try again with valid credentials.**
Step 2. Try to sign up with valid credentials and all mandatory fields completed.	**System shows successful login attempt message. Informs the user that the account was created successfully.**

Change password test case

Test-case Identifier:	**TC-3**
Use Case Tested:	**ChangePassword(UC-3)**
Pass/fail Criteria:	**The test passes if the user updates their password to a new password which meets the security standards of the system.**
Input Data:	**Alphanumeric username, email address**

Test Procedure:	Expected Result:
Step 1. Type in an invalid new password (which does not meet minimum requirements)	**System shows unsuccessful password change message. records unsuccessful attempt in the database; prompts the user to try again with a valid password and shows the specifications of a new password.**
Step 2. Type in a correct password.	**System shows a successful password change notification. Password salting & hashing takes place and the user is logged into the system with their new password.**

Upload book for sale test case:

Test-case Identifier:	**TC-4**
Use Case Tested:	**UC 5**
Pass/fail Criteria:	**The test passes if the user can successfully upload a book which they wish to loan or trade on the system.**

Input Data:	**Book data (title, author, interests, picUrl, ISBN, publishDate, creationDate, language, edition, loanPrice, sellPrice, summary)**
Test Procedure:	Expected Result:
Step 1. Enter book data with missing and/or incorrect fields. **Step 2. Enter all valid book data**	**The system notices invalid fields for the book upload process. Prompts user to re-enter book details and try again.** **The system detects all valid information entered.** **Records the new book as an available book for trade/loan on the system** **Adds the book to the user's list of uploaded books.**

Upload book for sale test case:

Test-case Identifier:	**TC-5**
Use Case Tested:	**UC 6**
Pass/fail Criteria:	**The test passes if the user successfully uploads a book on the system for selling.**
Input Data:	**Book data (title, author, interests, picUrl, ISBN, publishDate, creationDate, language, edition, loanPrice, sellPrice, summary)**
Test Procedure:	Expected Result:

Step 1. Enter book data with missing and/or incorrect fields.	**The system notices invalid fields for the book upload process. Prompts user to re-enter book details and try again.**
Step 2. Enter all valid book data	**The system detects all valid information entered.** **Records the new book as an available book for sale on the system** **Adds the book to the user's list of uploaded books.**

User reporting test case:

Test-case Identifier:	**TC-6**
Use Case Tested:	**UC 8**
Pass/fail Criteria:	**The test passes if a user successfully follows the process of reporting another user on the system.**
Input Data:	**Reporting user data (username), report data (message, reason, date, reportingUserID, userId, bookId, transactionId)**
Test Procedure:	Expected Result:
Step 1. User tries to submit a report without completing all necessary fields in the correct manner.	**The system notices invalid fields for the reporting process.** **Prompts the user to try again with all valid details.**
Step 2. Enter all valid report data	**The system records all the data entered by the reporting user.** **Notifies the administrator of the user which has been reported with all report details..**

Test-case Identifier:	**TC-7**
Use Case Tested:	**UC 12**
Pass/fail Criteria:	**The test passes if the administrator can successfully reverse a recorded transaction on the system.**
Input Data:	**Transaction data (transactionId)**

Test Procedure:	Expected Result:
Step 1. Admin searches for transaction by transactionId but enters an invalid id, and requests to reverse it	**The system searches but does not find a valid transaction with that transactionId. Prompts admin of failed search and to re-enter the id.**
Step 2. The user enters a valid transactionId	The system detects the transaction. Records it as a reversed transaction in the database, notifies related users and admin of successful reversal002E

8. Glossary

- MEAN stack: A software bundle used in making dynamic web systems and information systems. It includes NodeJS, ExpressJS, AngularJS and MongoDb.
- PassportJS: A library for NodeJS which is used for OAuth and user accounts on a node system.
- ExpressJS: A library for NodeJS which provides dynamically generated frontends and RESTful APIs.
- EJS: A templating library and language used with ExpressJS for web page generation.
- Templating Engine: A tool used to separate program-logic and presentation into two independent parts.
- WinstonJS: A logging library used to make system logs for NodeJS.
- MongoDb: A NoSQL or object oriented database which uses JSON objects and RESTful Api calls.
- NodeJS: A runtime environment which is used to build server side applications in Javascript.
- Mocha and Chai JS: These are unit and business process testing frameworks (libraries) for NodeJS.
- Node system: A system built on NodeJS.

9. Team

- Our Team name is: **Massive Dynamic**
- The Scrum Master: Liron
- The Project Owner: Jason
- Software developer/designer: Marko

10. Statement of Effort

Project:

	Jason Chalom (711985)	Liron Mizrahi (708810)	Marko Vidalis (831987)
Proof of Scrum Meetings	0%	100%	0%
Software Code	90%	10%	0%
Continuous Integration	100%	0%	0%
Test Driven Development	100%	0%	0%

Requirements Analysis Document:

	Jason Chalom (711985)	Liron Mizrahi (708810)	Marko Vidalis (831987)
Finishing	0%	0%	100%
Traceable	33%	33%	33%
Measurable	33%	33%	33%
Testable	33%	33%	33%
Introduction	33%	33%	33%
Functional Requirements	33%	33%	33%
Non-functional Requirements	33%	33%	33%
System Models	0%	70%	30%

Architecture Description:

	Jason Chalom (711985)	Liron Mizrahi (708810)	Marko Vidalis (831987)
Finishing	0%	0%	100%
Traceable	33%	33%	33%
View Points	0%	0%	100%
Logical View	40%	0%	60%
Process View	10%	0%	90%
Development View	0%	0%	100%
Physical View	0%	0%	100%