Architecture description template for Borrow My Books

Architecture Description of *Express JS, Client-server and asynchronous calls* For *Borrow My Books*

"Bare bones" edition version: 2.3

Template prepared by:
Rich Hilliard
r.hilliard@computer.org

License

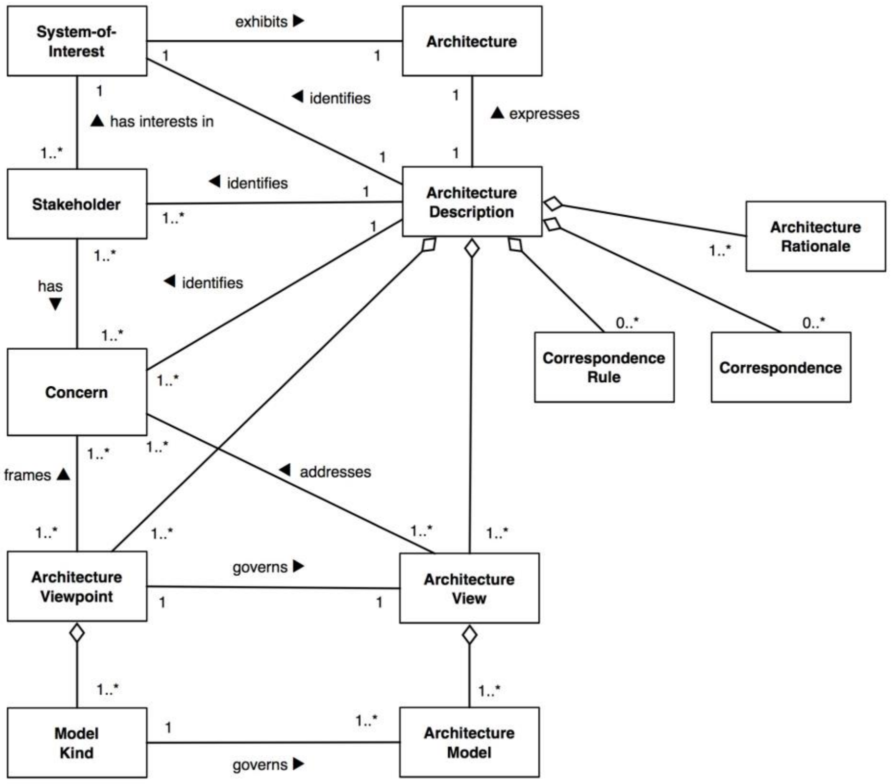


Figure 1: Content model of an architecture description

Contents

Chapter 1

Introduction

This chapter describes introductory information items of the AD, including identifying and supplementary information.

1.1 Identifying information

The Software Architecture chosen for the development of the BorrowMyBooks system is a hybrid one, consisting of a Model View Controller (MVC) type of system called ExpressJS for the implementation of relationships between user interfaces and underlying data models, and a Client Server in order to manage the network aspects of the system.

1.2 Supplementary information

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 transferal of physical goods.

1.3 Other information

This system makes use of web 2.0 technologies. We are using JavaScript processed by the V8 engine through NodeJS. We make heavy use of the callback code pattern. The system is based on a single process model which is event driven. These event occur mainly with user input and/or asynchronous calls. Other code patterns we have used when they are necessary are both the yield/await directives and also the async.waterfall model. (Found here: http://caolan.github.io/async/) These other patterns have been used to reduce the amount of nesting and callbacks used in the code. They are not anti-patterns but they are not the standard that comes with our use of the database connection driver (mongoose) and ExpressJS.

The live version of the site is hosted on Heroku (https://www.**heroku**.com/). The URL is https://borrowmybooks.herokuapp.com/. This site provides hosting for the server side code and public assets. The MongoDb database is stored on another site who is affiliated

Chapter 2

Stakeholders and concerns

2.1 Stakeholders

There are 4 main stakeholders which take part in the functioning of the system. These are:

Stakeholder 1: General user (Seller)

Stakeholder 2: General user (Purchaser)

Stakeholder 3: Administrator

Stakeholder 4: Owner

2.2 Concerns

The purpose of the BorrowMyBooks system is to create an easy to use platform where people can trade their textbooks. The architecture we have chosen to apply to this system is suitable due to the fact that the system will be developed as a Web application, where the separation of concerns (as supported by the MVC architecture) and the isolation of application logic from the user interface is necessary.
Furthermore, the Client-server network architecture will be suitable for the functioning of the BorrowMyBooks system as the system is mainly based on a large database of data relating to users, administrators and books listed for trading. This architecture will aid in the handling of the database and the communications of each user with it.
The feasibility of the system is dependent on the demand which students have for second hand textbooks. At tertiary education level, the demand is high due to the fact that brand new textbooks are expensive, and in some cases, textbooks are only used for 6 months, after which they become useless unless they are sold. This system will therefore make it easier for students to trade their textbook with others which is currently a cumbersome task. The deployment of the system is feasible.
There are few risk factors involved for the stakeholders of the system. The system will include password protection for the users, namely password salting, which will minimize the risk of user accounts being hacked into.
Furthermore, users will not be requested to submit financial data, such as credit card numbers and bank details. No transaction handling will be done through the system, it is solely developed for the location and transferal of textbooks between pupils.

The maintenance of the system will be carried out by the main developers, who will make sure that the system runs fluidly, any bugs are eliminated and that database communications are fast and reliable.
Lastly, the evolution of the system will depend on the future requirements of its users.

The main concerns are laid out as follows:

- CN1: All users must be able to log in to the system in a secure manner.
- CN2: An administrator must be able to log in to the system with their credentials which will grant them access to admin functionality.
- CN3: An unregistered user must be able to sign up through the system's sign up page.
- CN4: Any user on the system must be able to browse available books, their prices and descriptions.
- CN5: A seller on the system must be able to upload a book which they wish to trade and provide all relevant details for the book.
- CN6: A purchaser on the system must be able to browse all available books and view their relevant details through the explore page.
- CN7: A purchaser on the system must be able to search for a specific book on the system through the built in search engine.
- CN8: A purchaser must be able to perform a successful transaction on the system for a book which they are interested in purchasing.
- CN9: A seller must be able to view transaction requests for books which they are selling and accept or reject them.
- CN10: A user of the system must be able to report another user for bad conduct through the site's functionality.
- CN11: An administrator must be able to view and act upon any reports made relating to bad user conduct.
- CN12: An administrator must be able to view all transactions made for all books through the system and sort out any issues which may arise which are beyond the control of the purchaser and seller, and are system-related.

2.3 Concern–Stakeholder Traceability

The table below depicts the relationship between the various stakeholders in the system and the concerns which they directly relate to them.

	Purchaser	Seller	Administrator	.Unregistered User	Owner
CN1	x	x	x	-	-
CN2	-	-	x	-	-
CN3	-	-	-	x	-
CN4	x	X	x	x	-
CN5	-	X	x	x	-
CN6	X	-	-	-	-
CN7	X	-	-	-	-
CN8	X	-	-	-	-
CN9	-	x	-	-	-
CN10	x	x	x	-	-
CN11	-	-	x	-	-
CN12	-	-	x	-	-

Chapter 3

Viewpoints+

3.1 *Logical Viewpoint*

3.1.1 Overview

All systems exist in some larger environment, be it a department, an organisation's IT environment, a mobile communications system or even a virtual world. The Logical view aims to elaborate on the existence and technical relationships that this system has with elements of the wider environment. It describes the functionality that the system provides to end users.

3.1.2 Concerns and stakeholders

3.1.3 Concerns

Identity and Responsibilities of External Entities:

The BorrowMyBooks system has a set of entities, internally and externally, with which it interacts through its processes. This set consists of human entities, mainly general users of the application. The responsibilities of the general user which interacts with the system is to:

- Sign up on the system as an active user
- Login to the system with their personal account details
- Browse the available books and their details
- Transact through the system, through either purchasing or buying a book
- Upload books to be sold or rented out
- Rate users for their reliability as buyers/sellers on the system
- Report users for abusive or unethical behavior on the system

Another external user is the administrator of the system. The administrator of the system has more complex responsibilities regarding the functioning of the system which are stated below.

- Signing in to the system with their relevant admin details.
- View reported users and block or suspend their interaction with the system depending on the severity of their abusive behavior on the system.
- Solve problems that arise with transactions which user attempt to make through the system
- Monitor the performance of the system and make sure that any bugs which may arise are eradicated.

Most important for the proper functioning of the system, is the database system which BorrowMyBooks relies on to store and manage all the data. The responsibilities of the database are as follows:

- Storing all information of users who are signed up on the system in a secure way.
- Storing all information of books which are registered on the system for sale or rent, and the availability of each specific book.
- Provide a reliable and secure connection to the system to avoid external entities tapping into private system data

Nature of External Connections:

Each external actor on the system has its own way of interacting with the system. The different communication methods for each user is stated below.

- General user interaction: There is a dedicated User Interface controller developed using the Model View controller architecture through which users of the system will interact with the system for input and output data. This interface has limited functionality to cater only for the needs and responsibilities of a general user of the system as described above.
- Admin user control: There is a dedicated user interface through which the administrators of the system can carry out their responsibilities for the functioning and maintenance of the system, and for any other issues concerning general users, system properties, system functionality and database management.
- Database system communications: The system communicates with the database through an external API which allows for secure and reliable database connections.

3.1.4 Typical stakeholders

The potential stakeholders in this view are listed below:

- General users of a system willing to purchase or rent a book
- General users of the system who are willing to sell or rent a book out through the system.
- Administrators.
- Database administrators

3.2 *Process Viewpoint*

3.2.1 Overview

It is essential that the system's functional elements, their responsibilities, interfaces and primary interactions are described in the process viewpoint. Its purpose is to drive the shape of other system structures such as the information structure, concurrency structure and deployment structure.

3.2.2 Concerns and stakeholders

3.2.3 Concerns

Identity of the system's functional elements:

The Borrow My Books system has various elements which lead to its overall functioning.

The elements are described below:

- A web based interface/browser (on the end user's side) with the following responsibilities:
  - To allow a general user to interact with the system and its functionality.
  - To allow an administrator to perform administrator-related actions.
- A server side system which allows for:
  - The storage of system information and data which can be accessed through the web based interface mentioned above.
  - The acquisition of the web interface for users to access the system's full functionality through their web browsers.

Responsibilities of the functional elements:

Server side:

The server side of the system is managed and built through the Heroku application management system. This system is responsible for management and development of the Borrow My Books application in the cloud. Furthermore, the MongoDB DBMS data model is used, and is responsible for storing all information related to the functioning of the application, and is accessed through the cloud.

Web Browser:

Any web browser can be used to access the site. The Borrow My Books application is created using Javascript which can be deployed to any web-based interface.

Primary interactions of the functional elements:

In order for the Borrow My Books system to be fully functional, it is essential that the cloud database used is fully accessible by the server side system which deploys the application to a user's web interface. The interactions of the elements all take place through online communications and server connections.

3.2.4 Typical stakeholders

The potential stakeholders in this view are listed below:

- Requirements Analysts
- System Developers
- Administrators
- General users

3.3 *Development viewpoint*

3.3.1 Overview

The development view aims to depict the system from a Developer's perspective, and is mostly concerned with the software management aspect of the Borrow My Books system.

3.3.2 Concerns and stakeholders

3.3.3 Concerns

The following concerns are addressed in the Development View:

- Software management
- Internal entities of the system

3.3.4 Typical stakeholders

The potential stakeholders in this view are listed below:

- Developers
- Interface Designers
- Testers
- Product Engineers

3.4 *Physical viewpoint*

3.4.1 Overview

The physical view elaborates on the topology of the software components on the physical layer, as well as the communications between these components.

3.4.2 Concerns

The following concerns are addressed in the Development View:

- Physical topology of software components in the system.
- Communication types between software components in the Borrow My Books system.

3.4.3 Stakeholders

- System's engineers.
- Developers.

Chapter 4

Views+

4.1 View: *Logical View*

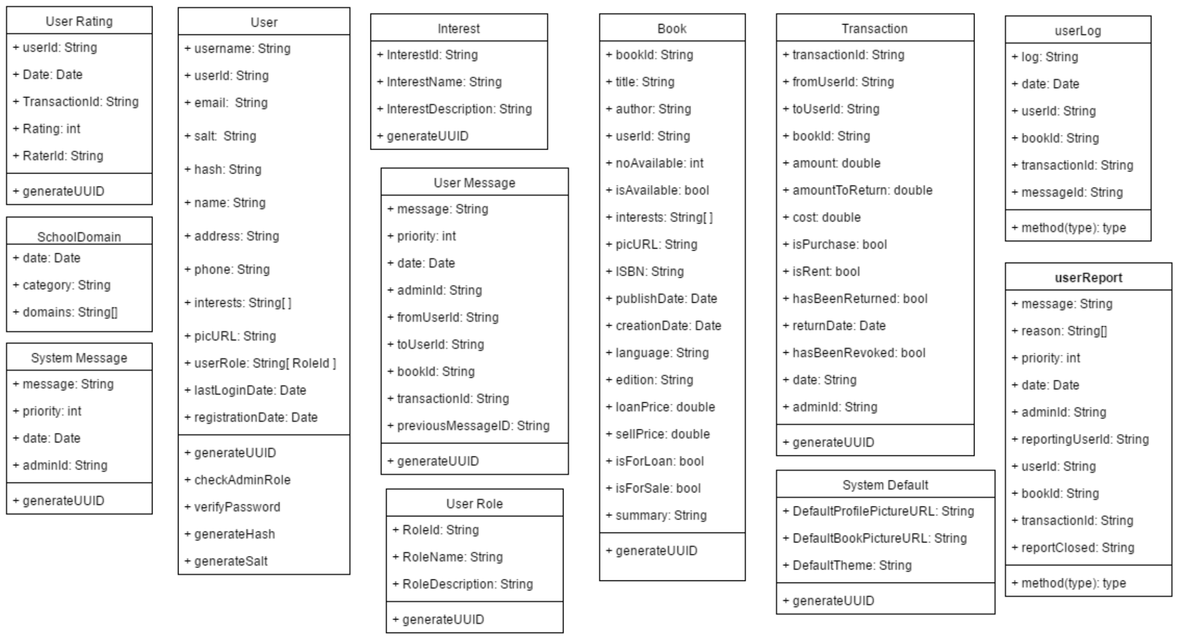
As mentioned above, the purpose of the logical view is to elaborate on the communications between the system and the external stakeholders.

Here, the database schema will be depicted using a UML notation class diagram. Also, certain processes shall be depicted in an activity diagrams section of the document using UML activity diagrams.

4.1.1 Diagrams:

4.1.2 Class diagram

The following class diagram shows the structure of the database which the system uses to store all related system information.



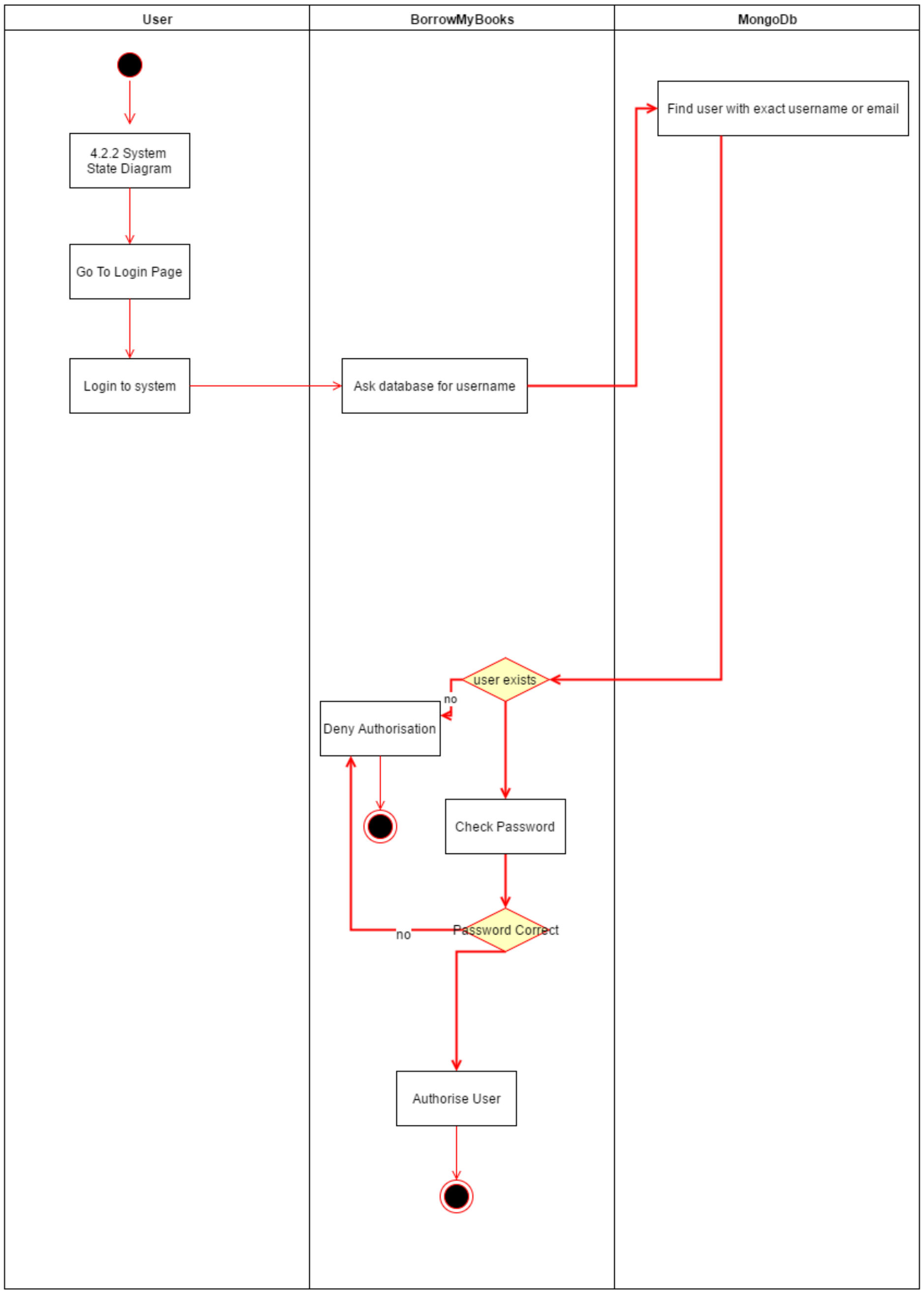
The database structure has been designed in such a way that ensures all necessary and relevant information to the Borrow My Books system is stored and accessible. This is a key feature to guarantee the successful operation of the Borrow My Books system.

4.1.3 Activity Diagrams

4.1.3.1 Login Activity Diagram

The following activity diagram explains the relationships between the system, the interface and the database when a user of the system tries to login to the system.

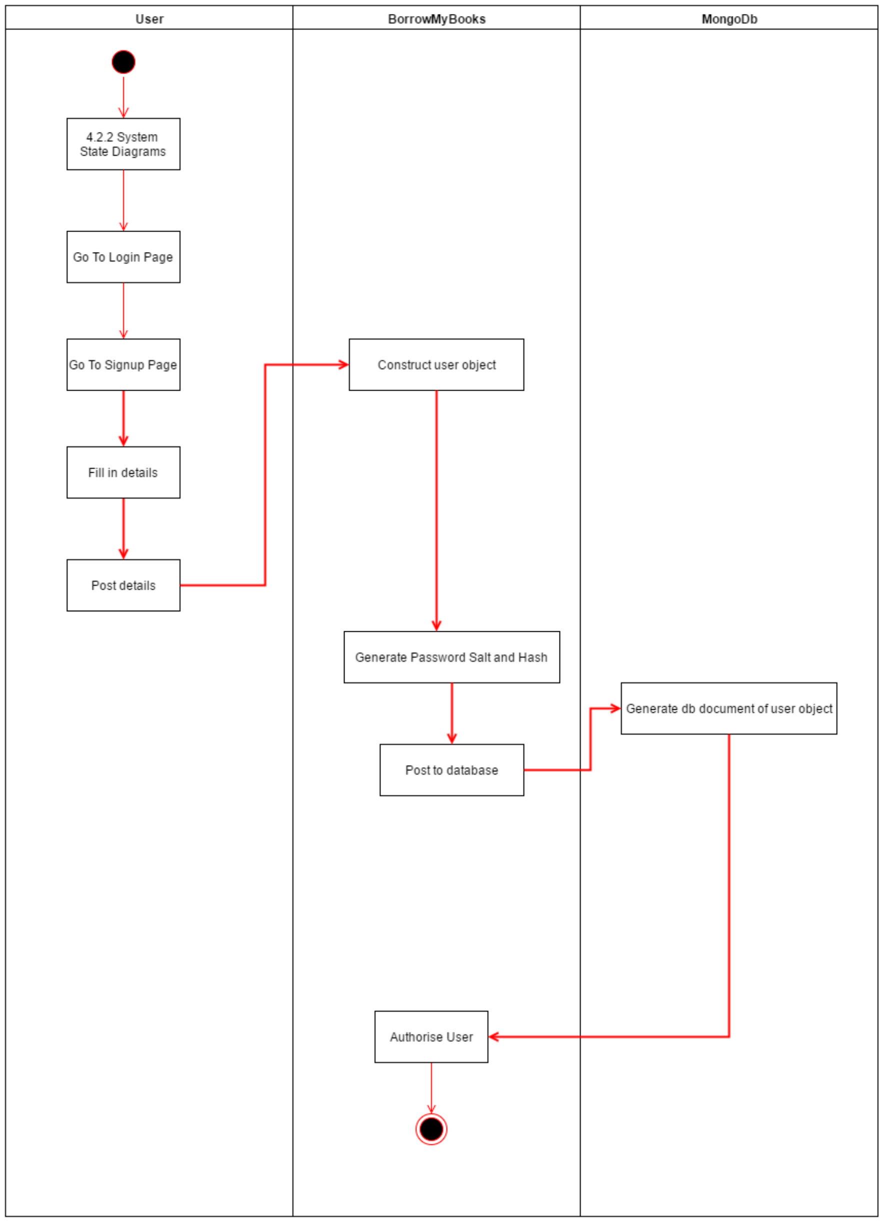
The system and user are initially in an idle state. When the user wishes to log in, they are moved to the log in page. The system's command queue is then populated with the user's login request. The command worker thread then processes the username entered by the user, and requests to check the existence of that user in the database.



4.1.3.2 Sign Up Activity Diagram:

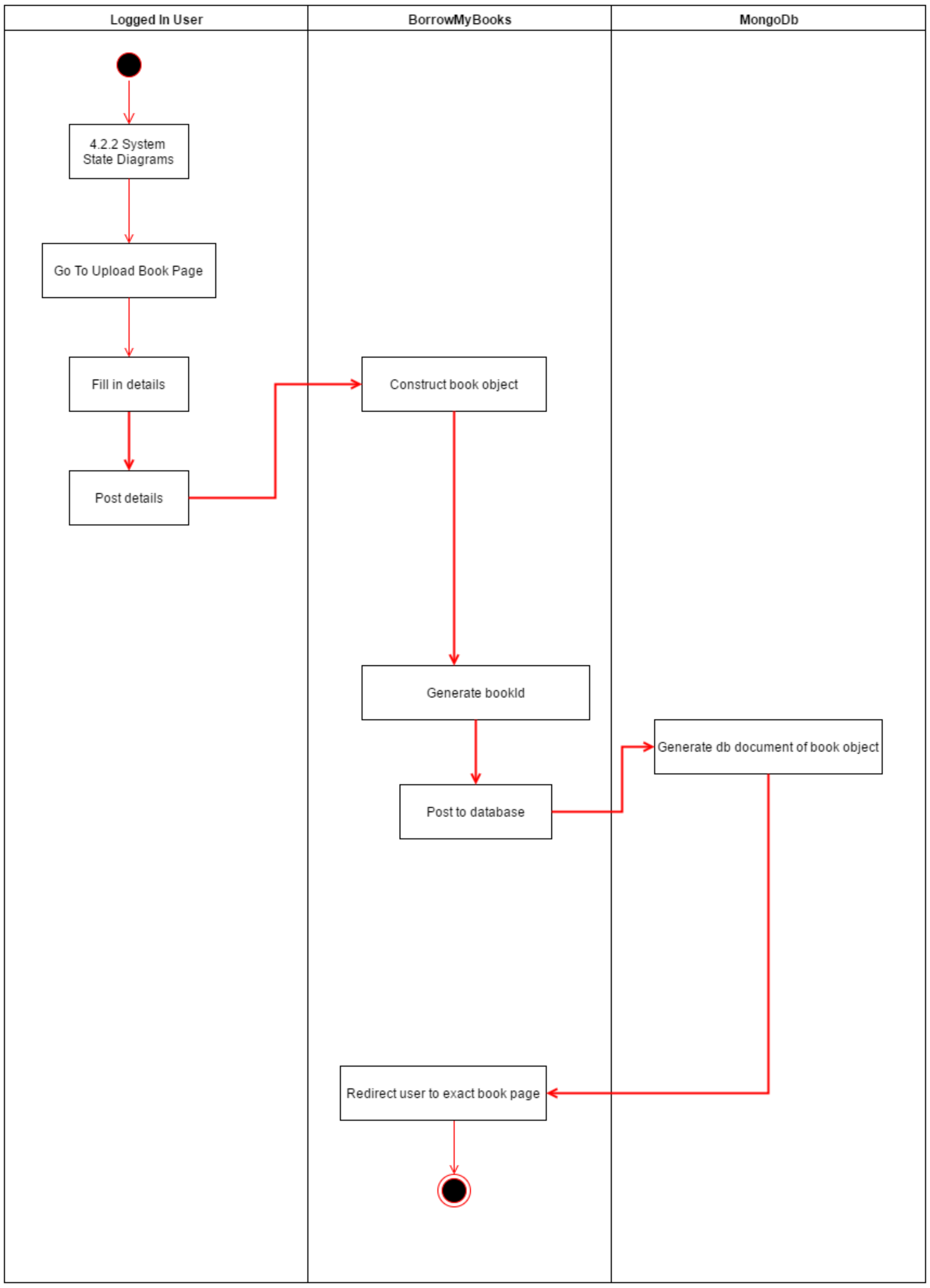
The following activity diagram explains the relationship between the user, the internal system and the database.

It depicts the process followed when a user requests to sign up on the system. The user and system are currently in an idle state. Upon request of the user, the system redirects the user to the sign up page. Once the user enters their details and requests to be registered on the system, the system checks that there is a command in the command queue. The command worker thread then creates a new user object. It generates a user password salt and hash. The database is then updated with the new user, all the relevant details, and the user is authorized to access the system with a personal account.



4.1.3.3 Upload Book Activity Diagram

The user, system and database server are initially in an idle state. The user then requests to upload a book and they are redirected to the upload book page. Once they enter the information, the details are processed by the system as a new command in the command queue. The command is dispatched to the worker thread and a new book object is constructed. A bookId is generated and the book's details are posted to the database. The user is then redirected to the new book's page.



4.2 View: Process View

As mentioned above, the Process view takes non-functional requirements of the system into account and elaborates on their importance to the success of the system.

4.2.1 Non-functional requirements and elaborations:

This section will mention each of the non-functional requirements and their relativity to the Borrow My Books system.

1) **Performance:**

The availability of the system, which also relates to its reliability is a crucial aspect for success. When the system is active (Users are logged in and connected to the system), the downtime tends to 0%. The uptime tends to 100%. Based on these statistics, we can quote the availability as a ratio of the expected value of uptime of the system, to the sum of the expected values of uptime and downtime of the system. In this case, based on our assumptions, and in order to cater for unexpected downtime of the system, this ratio can be expressed as a 99.999% availability of the system.

2) **Response time:**

The average response time of the system is 25ms. This response time is low enough to ensure that there are no unnecessary delays during the execution of the system.

3) **Security:**

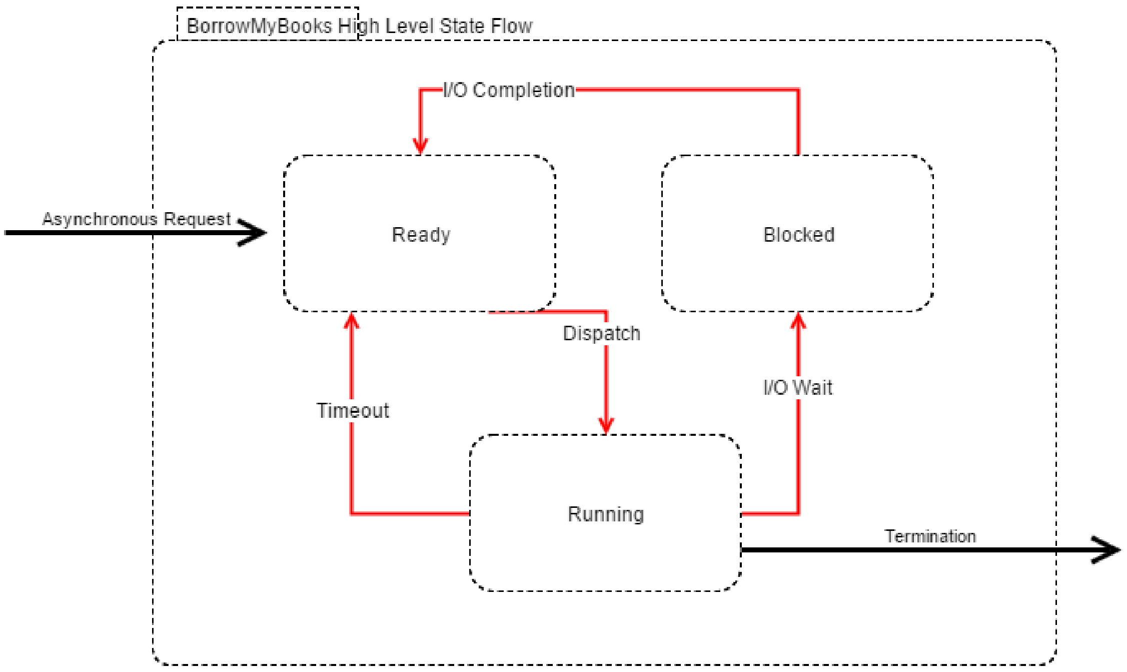
The system uses password salting and hashing in order to ensure the safety and security of each user's login details. The use of password salting and hashing minimizes the risk of dictionary attacks on a user's account. A user-specific salt is stored in the database that matches only to that user. This further secures each user's login details as the salt is randomly generated for each user, which ensures that the hash for the specific user is also unique and even more secure.

4) **Testability:**

The system uses testing (which is covered in detail in Section 4.3: Development View) in order to avoid system errors and problems which could arise.

4.2.2 System State Diagram

The following System State Diagram depicts the high-level state flow of the internal entities of the Borrow My Books system. This flow has an effect on the performance and response time of the system while in operation.



4.3 View: Development View

The aim of the development view is to illustrate the software management which exists in the system. The system is governed by various software powerhouses which all communicate effectively in order to realise the goal of the BorrowMyBooks system.

4.3.1 Software Management

The system is developed on a server which serves webpages to users and executes user commands. The templating engine for this system is EJS which dynamically renders webpages to its users.

The database backend is managed by MongoDB.

The system is covered by testing which ensures successful deployment when accessed by a user. The following areas are covered by testing:

- Functionality Testing: the following functionality aspects of the system are tested:
    - Forms: All form functionality in the system is tested for errors before deployment. User input to the system is tested through error checking and validation in order to avoid inconsistencies in the database.
    - File manipulation
    - Calculations
- User interface and usability testing:
    - Navigation between pages
    - Accessibility
    - Layouts for different device sizes and browsers

4.3.2 Internal system entities:

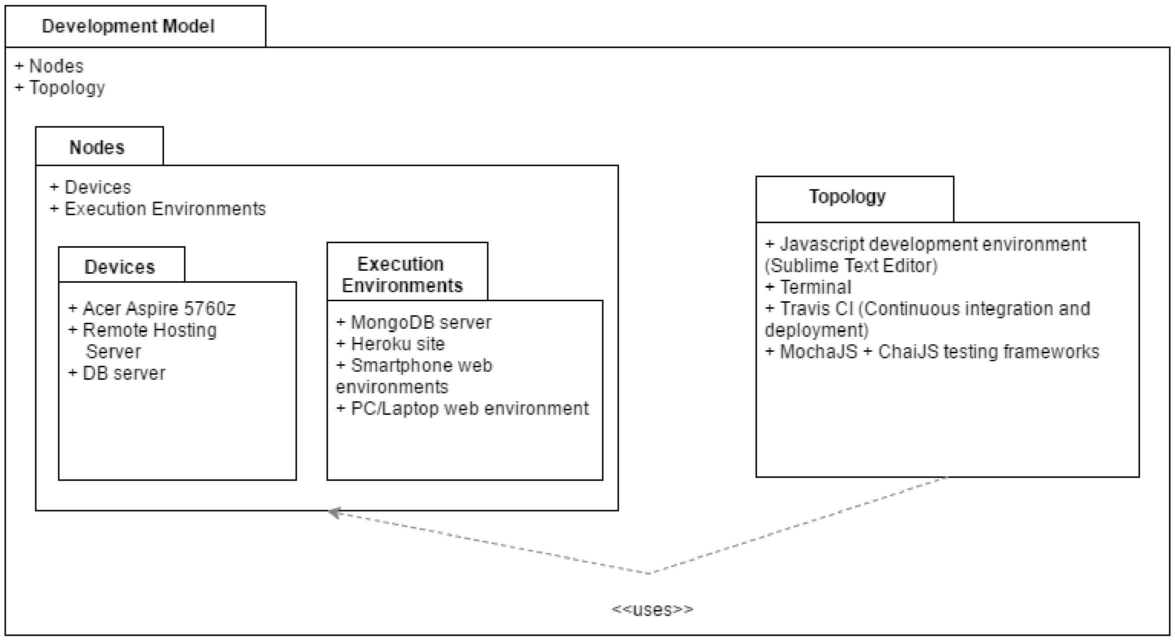
The system functioning is governed by the MVC architectural model.

The model aspect of the architecture is developed using the ExpressJS system. It is coded in Javascript and hosted on the Heroku hosting site. This allows for the best deployment of the system as a web application. Through the Client server controller, ExpressJS allows for the best communications between interface and backend system in order to aid for all the non-functional requirements of the system.

The view aspect of the system is catered for by any web browsing application, as the system is a web application.

The following package diagram shows all the specific entities used in the development of the Borrow My Books system.

4.3.3 Package Diagram



As pictured above, the development model consists of nodes and topologies which run on a specific node.

The Nodes consist of devices and execution environments. The devices are namely those used for the development of the system. The execution environments name the environments on which the system itself is simultaneously executed.

The topology consists of the various tools and development environments used in order to develop the system.

4.4 Physical View

As mentioned in the Physical Viewpoint, the Physical view will elaborate on the topology of the software components in the Borrow My Books system, as well as the communications which take place between them.

4.4.1 Description of software components and communications

The software components of the Borrow My Books system are listed below:

- ExpressJS web application framework
- MochaJS and ChaiJS testing frameworks
- TravisCI – continuous integration and deployment
- MongoDB database server

The communications between the above software components happen on a local level between the web and testing frameworks.

The communications, however, which occur between the software components and the cloud components (such as the DB server) occur through TCP/IP protocols via the client/server architecture.

Appendix A

Architecture decisions and rationale

A.1 Decisions

The Borrow My Books team decided that the MVC and client-server architectures were the best fit for the purpose of the system in development.

The MVC architecture provides an effective model for development of a system that runs on all devices, irrespective of operating system and specifications. The web-based application is executable on any web browsing app or program on any device, whether it be any smartphone, laptop, desktop etc. The only requirement in order to have access to the site is having a web browser.

With regards to project planning and management, the architecture leaves room for future evolution of the system. The interface, system functioning and database server are all independent of each other, and can therefore be developed and improved independently, without having to make major changes across all aspects of the system. Furthermore, the Client server architecture allows for secure and effective communications between the users and system backend to take place.

With regards to expenditure of time and capital, the MVC architecture allows for time-efficient development and improvement in the future evolution of the system. The capital expenditure of the system is minimal as the only hardware components needed for its development are a server backend (which is in fact hosted remotely) and a machine on which development can occur.

Bibliography

Clements, Paul C. et al. *Documenting Software Architectures: views and beyond*. 2nd. Addison Wesley, 2010.

Finkelstein, A. et al. “Viewpoints: a framework for integrating multiple perspectives in system development”. In: *International Journal of Software Engineering and Knowledge Engineering* 2.1 (Mar. 1992), pp. 31–57.

Heesch, Uwe van, Paris Avgeriou, and Rich Hilliard. “A Documentation Framework for Architecture Decisions”. In: *The Journal of Systems & Software* 85.4 (Apr. 2012), pp. 795–820. DOI: 10.1016/j.jss.2011.10.017.

IEEE Std 1471, IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. Oct. 2000.

ISO/IEC/IEEE 42010, Systems and software engineering — Architecture description. Dec. 2011, pp. 1–46.

Ran, Alexander. “ARES Conceptual Framework for Software Architecture”. In: *Software Architecture for Product Families Principles and Practice*. Ed. by M. Jazayeri, A. Ran, and F. van der Linden. Addison-Wesley, 2000, pp. 1–29.

Rozanski, Nick and Eoin Woods.´ *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*. 2nd. Addison Wesley, 2011.