BOOK RANKS: ONLINE BOOK REVIEWS WITH STAR RATINGS FOR DIGITAL READERS

MUHAMMAD IMRAN BIN ISMAIL

BOOK RANKS: ONLINE BOOK REVIEWS WITH STAR RATINGS FOR DIGITAL READERS

MUHAMMAD IMRAN BIN ISMAIL

This report is submitted in partial fulfillment of the requirements for the Bachelor of Computer Science (Software Development) with Honours.

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2021

DECLARATION

I hereby declare that this project report entitled

BOOK RANKS: ONLINE BOOK REVIEWS WITH STAR RATINGS FOR DIGITAL READERS

is written by me and is my own effort and that no part has been plagiarized without citations.

STUDENT : _____ Date : 07/09/2021

(MUHAMMAD IMRAN BIN ISMAIL)

I hereby declare that I have read this project report and found this project report is sufficient in term of the scope and quality for the award of Bachelor of Computer Science (Software Development) with Honours.

SUPERVISOR : _____ Date : 12/09/2021

(TS. MUHAMMAD SUHAIZAN SULONG)

DEDICATION

Dedicated to my parents who have been my source of inspiration and gave me strength and courage to keep going when I though to give up.

To my brothers, sisters, and friends who shared their words of advice and encouragement to finish this study.

ACKNOWLEDGEMENTS

I would like to express my special thanks of gratitude to my supervisor, Ts. Muhammad Suhaizan bin Sulong for his sincere, expert, and valuable guidance. I am grateful for his professional guidance.

I also would like to thank my family and friends for their valuable support.

ABSTRACT

Books can be written at any time on any topics. To see whether a book is worth reading, book review can be very helpful for potential readers to know about the book. Thus, under this project, the Book Ranks is developed – a web-based application for readers to share their opinions about a book they read by providing insights and ratings. This application also provides a faster way of retrieving book reviews through barcode scanner and searching for book reviews while on the go. Through an admin, the book reviews will be up to date and well managed. The app is developed in open-source PHP and MySQL with Laravel framework through a waterfall model until the application is delivered. It is expected that this Book Ranks may be able to help book readers to get better reviews for books and even make it easier to search for reviews.

ABSTRAK

Buku boleh ditulis pada bila-bila masa tentang apa-apa topik. Untuk mengetahui bahawa jika sesebuah buku berbaloi untuk dibaca, ulasan tentang buku adalah sangat berguna. Oleh itu, Book Ranks dibangunkan. Book Ranks adalah aplikasi berasaskan web untuk pembaca berkongsi pendapat mereka tentang sesebuah buku dengan memberikan pandangan dan penilaian. Aplikasi ini juga menyediakan cara yang lebih pantas untuk mendapatkan ulasan buku melalui pengimbas kod bar, dan mencari ulasan buku pada bila-bila masa. Melalui pentadbir, ulasan buku akan dikendalikan dengan baik. Aplikasi ini dibangunkan menggunakan PHP dan MySQL dengan rangka kerja Laravel melalui model pembanguan air terjun untuk membangunkan aplikasi. Diharapkan Book Ranks dapat membantu pembaca buku untuk mendapatkan ulasan yang lebih baik untuk buku dan mempermudah mereka mencari ulasan.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1 Introduction

This project's purpose is to develop a website that will allow book readers to review and easily retrieve reviews of a certain book. Most book review sites these days are left unmoderated that will leave irrelevant comments and reviews that do not provide any relevant information on the book itself. Other than that, it is also hard for readers to search quickly to look up review for a book while shopping for books in a sale for example. So, the objectives of this project are to allow our admin to moderate the reviews left by readers, so only relevant reviews are allowed. Then, we would also like to allow users to quickly can book's ISBN barcode to look up a book. In this website, an admin will manage books and functions as a moderator. Admin can remove a review if deemed necessary. A reader can read reviews and add a review for a book. A reader would also be suggested what books should they review next based on their interest. Another user of this system is author. An author can request for their books to be added and view reports on how their books are perceived. An author can also response to the reviews. The expected output of this project is a website that stores book reviews by readers and give easy access to readers to read the reviews.

1.2 Problem Statements

- Difficult to avoid irrelevant reviews
    - Users of existing system today may find it hard to avoid irrelevant reviews that contribute nothing to the discussion of the book. This is

because the existing systems are not heavily moderated which allows reviewers to say anything they want.

- Time consuming to search for book reviews while book shopping.
    - Readers may not have a pre-selected books that they want to buy when shopping for books. So, they may want to read online reviews while shopping for books. Existing system doesn't allow a quick way to go to a certain book's reviews, they usually require user to manually type the books title.

- Difficult for authors to interact with readers.
    - On other systems, they do not allow for authors to directly reply to a user's review for their books. An author might want to give a feedback on user's review for their criticism or appreciation.

1.3 Objectives

- To review and design a suitable web-based system for allowing readers to review and rate published books.

- To develop and implement the web-based system according to requirements.

- To test and evaluate the system for potential bugs and functionality at all levels.

1.4 Scope

Users: 1. Admin

2. Readers

3. Authors

System modules:

1. Authentication Module (All users)

- User registration

- Login/logout

- Password recovery

- Profile update with interest category

2. Book Records Module

- Manage (CRUD) book records with all details (User: admin)

- Add books (User: authors) and verify books (User: admin)

3. Search and Review Module

- Search books by title, author, publisher, isbn, and scanning barcode (User: Readers)

- Review books (CRUD) comments and ratings (User: Readers)

- Moderate reviews and verify reviews (User: Admin)

4. Special Module

- Recommend books to review based on interest and other readers (User: readers)

- Readers' choice/best reviews award

- poll (All users)

- Feedback on reviews (User: Authors)

5. Dashboard Reporting Module

- Graphs about visitors (User: Admin)

- No. of reviews monthly/yearly as a whole (User: Admin)

1.5 Project Significance

This project will help users to get better and more reliable book reviews by allowing the admins of the website to moderate the reviews. It will also allow book readers to quickly retrieve the reviews for books that they desire by searching for their books using book title or ISBN. Other than authors can also get better feedbacks by having two-way communications with the readers.

1.6 Expected Output

The expected output of this project is a website that stores book reviews by readers and give easy access to readers to read the reviews. The output of this project is to create a website that will be able to allow users to easily search and read reviews for a book that they intend to. This site will also be a safe space for everybody since our admins will moderate the reviews.

1.7 Conclusion

In conclusion, Book Ranks is a web-based book reviews hub that will have three users, admin, users, and authors. Book Ranks will allow readers to review the

books that they have read and also find reviews for their upcoming books. Authors can request for admins to add their desired books and reply to readers' reviews. Meanwhile admin can moderate the reviews. This website will also hopefully will be able to make relevant recommendations to readers based on their interests.

CHAPTER 2: LITERATURE REVIEW AND PROJECT METHODOLOGY

2.1 Introduction

This chapter will discuss about the domains involved in this project. This chapter will also define the software development methodologies used to develop the project. Other than that, the milestones and schedule of the project timeline is going to be set.

2.2 Facts and findings

2.2.1 Domain

The domain of this project is the book reviews domain. A book review is a guide for potential readers. In a concise manner, a review summarizes the author's qualifications and main points, often providing examples from the text. A review also provides an opinion on whether the author succeeds or not in convincing readers of their points. The reason this domain is chosen is to help casual readers find reviews on books that they want to read. Certain existing system nowadays are too bloated and works more like social media such as GoodReads where users are given too much information unrelated to the books. Then, other website may only allow reviews from verified journalist and users may feel detached from their reviews and want reviews from everyday people.

2.2.2 Existing System

Book Ranks target to be a website where users can search for book reviews and leave their own revies.

2.2.2.1 Goodreads

Most book review sites these days such as Goodreads are left unmoderated that will leave irrelevant comments and reviews that do not provide any relevant information on the book itself. Other than that, it is also hard for readers to search quickly to look up review for a book while shopping for books in a sale for example. Another common problem for existing system such as Goodreads is that because of its many sections and clunky web design, it is confusing, at times to the point of being almost unusable. Both authors and readers on Goodreads have noted an increase in political banter, trolling, cancel culture and cyberbullying afflicting the website. This is caused by the social media aspect of the website. Goodreads has grown too much from its original state as a book catalog where users can add books to their reading list.

2.2.2.2 BookMark.Reviews

Book Marks is book reviews aggregating website. Unlike other existing system on the market, the system does not allow any type of users to leave reviews for books. The reviews that are submitted on the website come only from verified reviewers or journalist who write for magazines or other websites. So it may hard for casual readers to relate to the reviews on the site since professional critics are more critical of the books.

2.2.2.3 Google Play Books

Google Play Books is a digital reading service provided by Google. The service does allow users to leave reviews, but only for people who buy the books on the service. Someone who gets their book from other digital books provider are not able to leave their reviews. People who prefer physical books also are not able to since there is no system in place to proof that they own the books.

2.2.2.4 Comparison

In comparison of these three existing systems, GoodReads allow everyone to leave reviews on their site, BookMarks only allow verified writers or journalist and Google only allow their digital customer. GoodReads also work more like a social

media compared to BookMarks and Google Play Books where it is more like an advertisement hub for the books and may be biased.

2.3 Project Methodology

The system development approach taken to build this project is the waterfall model of system development life cycle. This method is chosen because it is suitable when a project is short and for a project that is not too big.

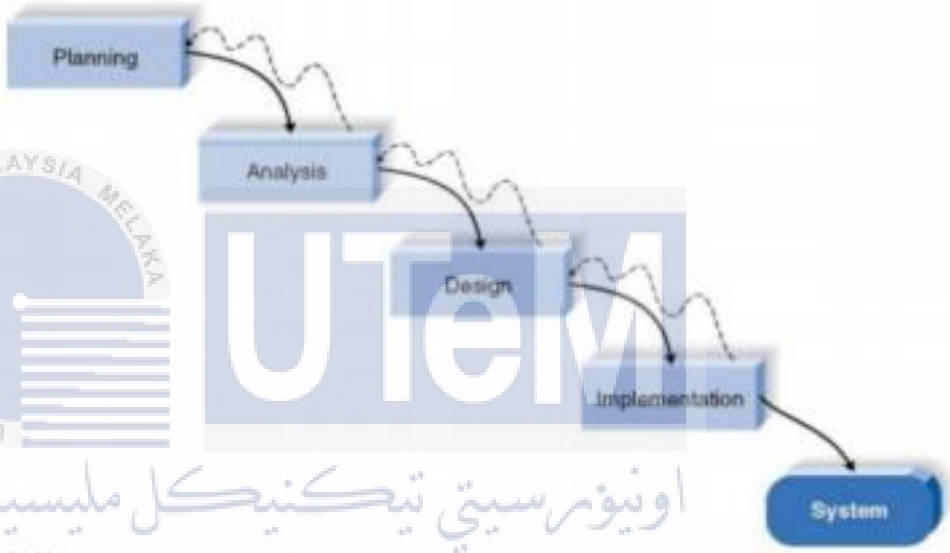


Figure 2.1: Waterfall Model

• Planning

Plan on suitable and feasible topic.

• Analysis

Begin collecting required data to build the database. After that plan what processes are required to achieve the objectives of the project.

• Design

Sketch a design for the user interface based on the collected required data and processes.

• Implementation

Begin translating the planned processes and interface in programming languages. Beginning with creating the database, with tables, triggers, and procedures. After finished with setting up the database, the user interface is created. Then, the system is tested.

2.4 Project Requirements

2.4.1 Software Requirement

- XAMPP
- PHPMyAdmin
- Visual Studio Code
- Laravel
- Composer

2.4.2 Hardware Requirement

- Personal Computer

2.5 Project Schedule and Milestones

Table 2.1: Gantt Chart

Milestone/Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Proposal															
Complete ERD & Design															
Authentication module															
Book records module															
Search and review module															
Special module															
Report module															
Presentation															

2.6 Conclusion

In conclusion, this project is closely related to web and software development. Waterfall model methodology is used because this method is suitable when a project is short and for a project that is not too big. It also helps that this project requirements are fixed from the beginning and would not go through any changes during the development process. So, this approach helped to provide more predictable result. This

methodology is also easy to conduct and understand due to its sequential nature as waterfall methodology focuses most on a clear, defined set of steps.

CHAPTER 3: ANALYSIS

3.1 Introduction

This chapter will define the analysis result of the existing system and the proposed system. It will show the data flow diagrams for both systems. Then it will define the requirements of the proposed system in forms of data dictionary and data flow diagrams. Other than that, non-functional requirements also will be stated.

3.2 Problem Analysis

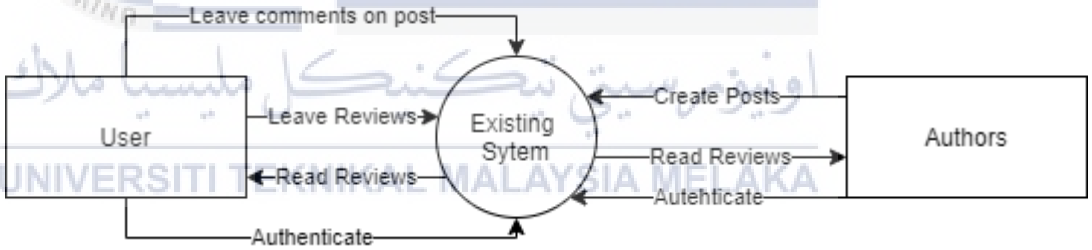


Figure 3.1: Existing System Context Diagram

Most book review sites these days are left unmoderated that will leave irrelevant comments and reviews that do not provide any relevant information on the book itself. As shown in the data flow diagram for the existing system, there is no use case for content moderation. There are no moderators that will verify reviews, this will cause unhelpful or unconstructive reviews to be left by irresponsible users. Other than that, it is also hard for readers to search quickly to look up review for a book while

shopping for books in a sale for example. Lastly, there is also no direct way for authors to interact with user's reviews unless they create a separate posting.

3.3 Requirement Analysis

3.3.1 Data Requirement

The data requirement is based on the input and output necessary for the system to work. For this system, the necessary input must be an email and password for the user to logged in. Then, when managing book records, the details needed are title, author, ISBN, picture, and description. For reviewing books, user must provide reviews and ratings from one to five star. The system also has a polling system where user can vote a book based on the poll category. The input is the book choice and poll ID.

Next for the output, based on the input such as book reviews and ratings, the system may calculate the average ratings for each book. This is important to give an insight to users to as how the books are perceived by other readers. The system is also capable to calculate and display the number of new users, and new reviews per month. Next, the system can calculate the poll winner based on the vote by the users.

3.3.1.1 Data Dictionary

A data dictionary includes metadata or database data. The data dictionary is very important because it contains information such as what is in the database, who can access it, where the physically stored database is. The user of database usually does not interact with the data dictionary; it is only handled by the database administrators.

Table 3.1: Role Data Dictionary

Column Name	**Data Type**	**PK/FK**	**Description**	**Required**
Role_ID	INT	PK	Unique ID for role	Yes
Role_Name	VARCHAR(20)		Role name for different users	Yes

Table 3.2: User Data Dictionary

Column Name	**Data Type**	**PK/FK**	**Description**	**Required**
User_ID	BIGINT(20)	PK	Unique ID for role	Yes
User_Password	VARCHAR(191)		User's password for logging in.	Yes
User_Name	VARCHAR(191)		User's name	Yes
User_Email	VARCHAR(191)		User's email.	Yes
Role_ID	BIGINT(20)	FK	Unique ID for role	Yes

Table 3.3: Category Dictionary

Column Name	**Data Type**	**PK/FK**	**Description**	**Required**
Category_ID	BIGINT(20)	PK	Unique ID for category	Yes
Category_Name	VARCHAR(191)		Category name	Yes

Table 3.4: User Interest Data Dictionary

Column Name	Data Type	PK/FK	Description	Required
User_ID	BIGINT(20)	PK, FK	Unique ID for user	Yes
Category_ID	BIGINT(20)	PK, FK	Unique ID for category	Yes

Table 3.5: Book Data Dictionary

Column Name	Data Type	PK/FK	Description	Required
Book_ID	BIGINT(20)	PK	Unique ID for book	Yes
Book_ISBN	VARCHAR(13)		Role name for different users	Yes
Book_Title	VARCHAR(191)		Book's title	Yes
Book_Description	TEXT		Book's description	Yes
Book_Cover	VARCHAR(191)		Book's cover location	Yes
Book_Status	VARCHAR(191)		Book's status	Yes

User_ID	BIGINT(20)	FK	Unique ID for user	Yes
Publisher_ID	BIGINT(20)	FK	Unique ID for publisher	Yes

Table 3.6: Author Data Dictionary

Column Name	**Data Type**	**PK/FK**	**Description**	**Required**
Author_ID	BIGINT(20)	PK	Unique ID for author	Yes
Author_F_Name	VARCHAR(191)		Author's first name	Yes
Author_L_Name	VARCHAR(191)		Author's last name	
User_ID	BIGINT(20)		Unique ID for user	No

Table 3.7: Author_Book Data Dictionary

Column Name	**Data Type**	**PK/FK**	**Description**	**Required**
Book_ID	BIGINT(20)	PK	Unique ID for book	Yes
Author_ID	VARCHAR(191)		Unique ID for author	Yes

Table 3.8: Poll Data Dictionary

Column Name	Data Type	PK/FK	Description	Required
Poll_ID	BIGINT(20)	PK	Unique ID for poll	Yes
Poll_Description	TEXT		Poll's description	Yes
User_ID	BIGINT(20)	FK	Unique ID for user	

Table 3.9: Book_Poll Data Dictionary

Column Name	Data Type	PK/FK	Description	Required
Poll_ID	BIGINT(20)	PK, FK	Unique ID for poll	Yes
Book_ID	BIGINT(20)	PK, FK	Unique ID for Book	Yes

Table 3.10: Feedback Data Dictionary

Column Name	Data Type	PK/FK	Description	Required
Feedback_ID	BIGINT(20)	PK	Unique ID for a feedback	Yes
Feedback	TEXT			Yes
User_ID	BIGINT(20)	FK	Unique ID for a user	Yes
Review_ID	BIGINT(20)	FK	Unique ID for a review	Yes

Table 3.11: Report Data Dictionary

Column Name	Data Type	PK/FK	Description	Required
Report_ID	BIGINT(20)	PK	Unique ID for a report	Yes
Report_Reason	Varchar(30)			Yes
User_ID	BIGINT(20)	FK	Unique ID for a user	Yes
Review_ID	BIGINT(20)	FK	Unique ID for a review	Yes

Table 3.12: Vote Data Dictionary

Column Name	Data Type	PK/FK	Description	Required
Vote_ID	BIGINT(20)	PK	Unique ID for a vote	Yes
Poll_ID	BIGINT(20)	FK	Unique ID for a poll	Yes
User_ID	BIGINT(20)	FK	Unique ID for a user	Yes
Book_ID	BIGINT(20)	FK	Unique ID for a book	Yes

3.3.2 Functional Requirement

BookRanks target to allow users to add review and retrieve reviews to any book they want. First, admins and users must be able to register into the system. Admin will be able to manage books by adding and editing the details. Then, users can add reviews with ratings to any books. Users are also able to search for books by searching the books by title, description, author, and ISBN. Other than that, users may also scan a

book’s barcode. Finally, the system can generate report such as reviews received and new registered for administration uses.

3.3.2.1 Data Flow Diagram

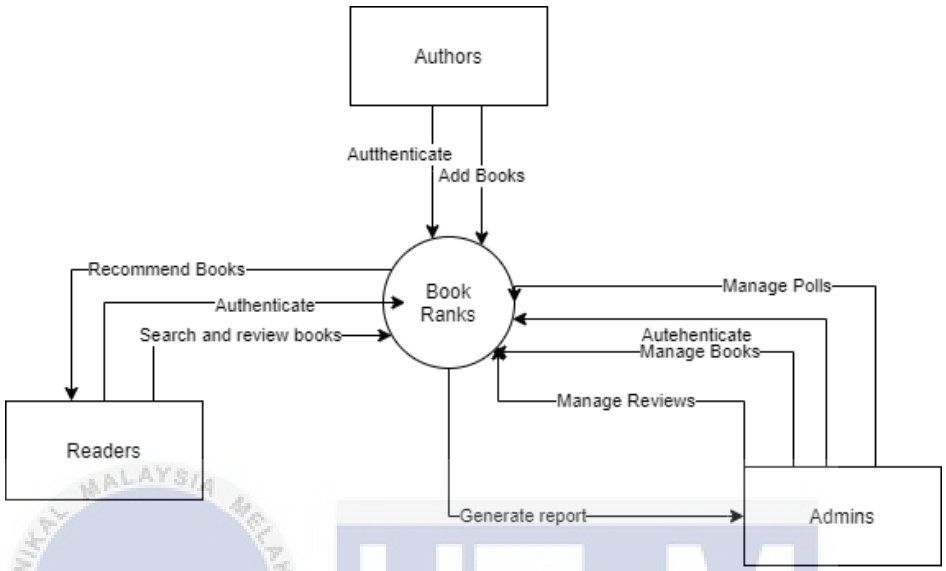


Figure 3.2: Context Diagram

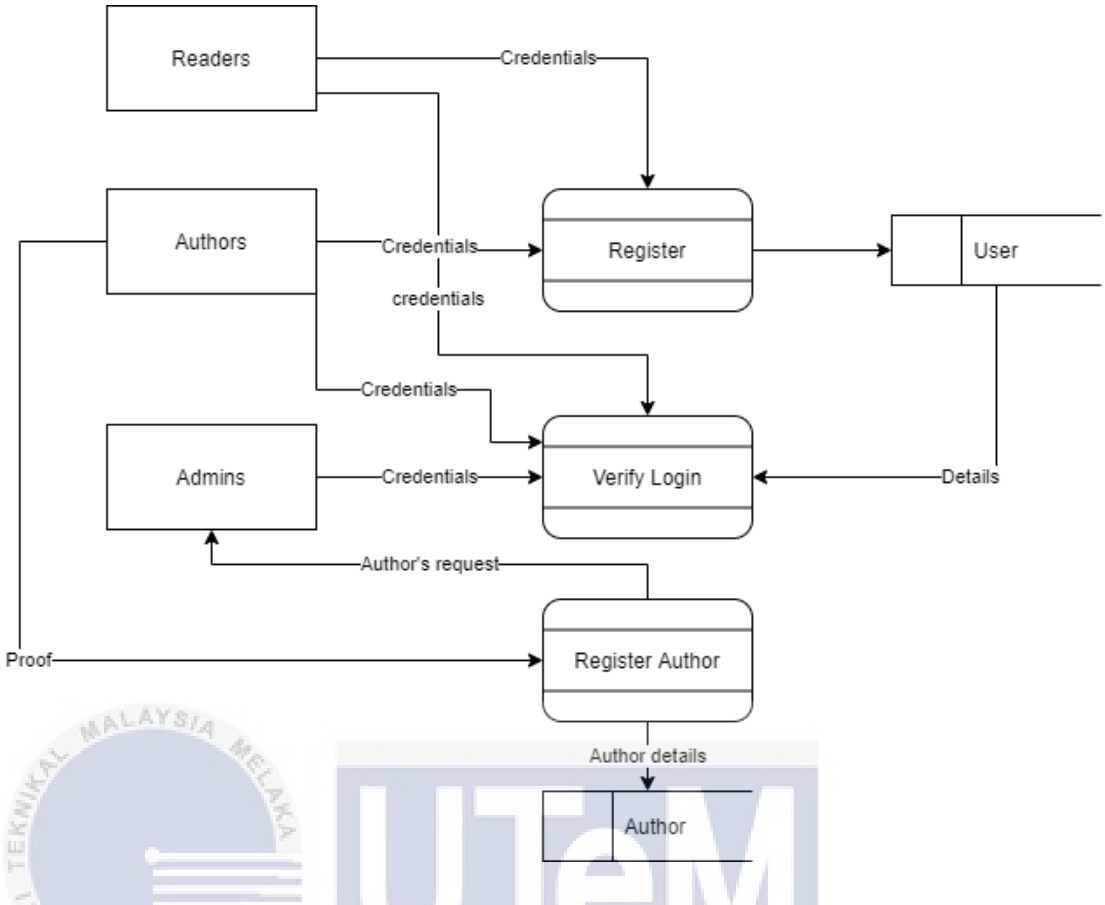


Figure 3.3: Level 1 DFD for Authentication Module

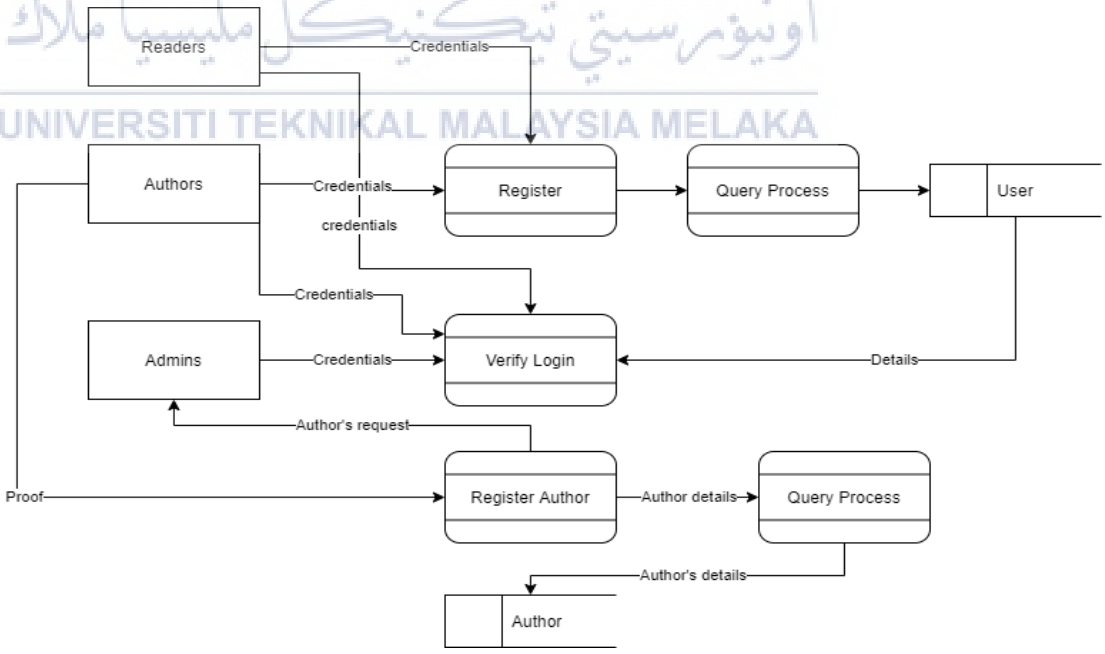


Figure 3.4: Level 2 DFD for Authentication Module

The DFD for authentication module shows that users can register into the system and log in. When registering into the system, the system will validate the user's information and insert the user's information into the 'User table'. When logging in, the system will fetch the user's information from the database. Users can also insert their own proof of authorship into the database.

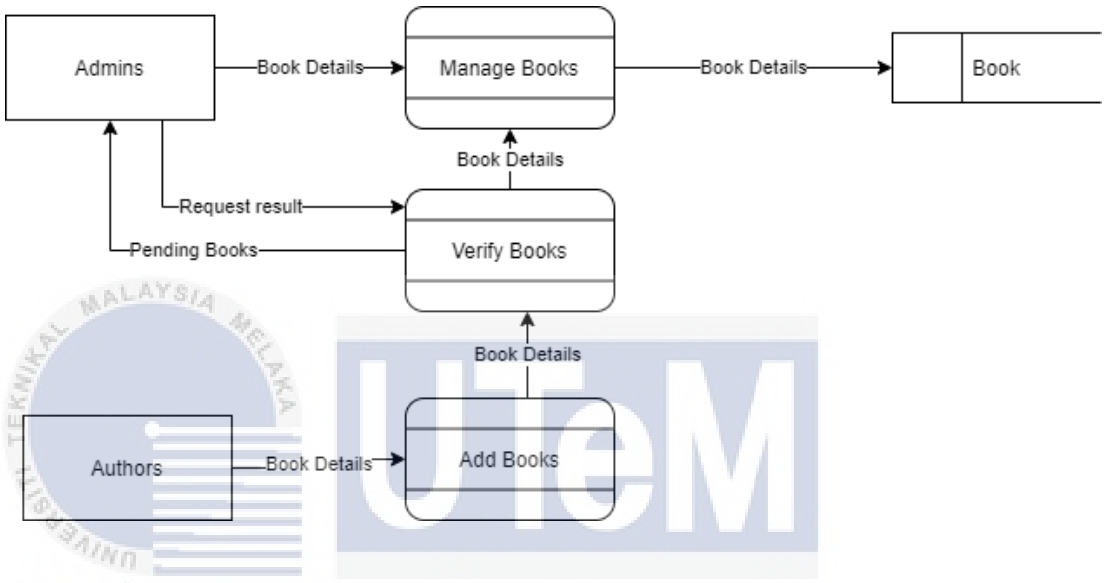


Figure 3.5: Level 1 DFD for Books Record Module

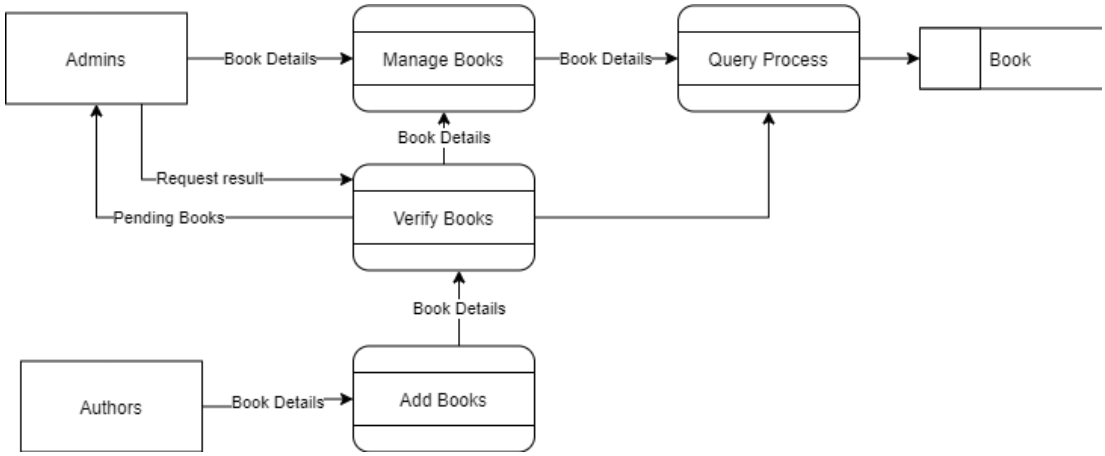


Figure 3.6: Level 2 DFD for Books Record Module

The DFD for books record module shows that an admin and author can insert new books into the system with details such as title, author, ISBN, and description. The system will receive the data and insert the books through a database query process into the Book table.

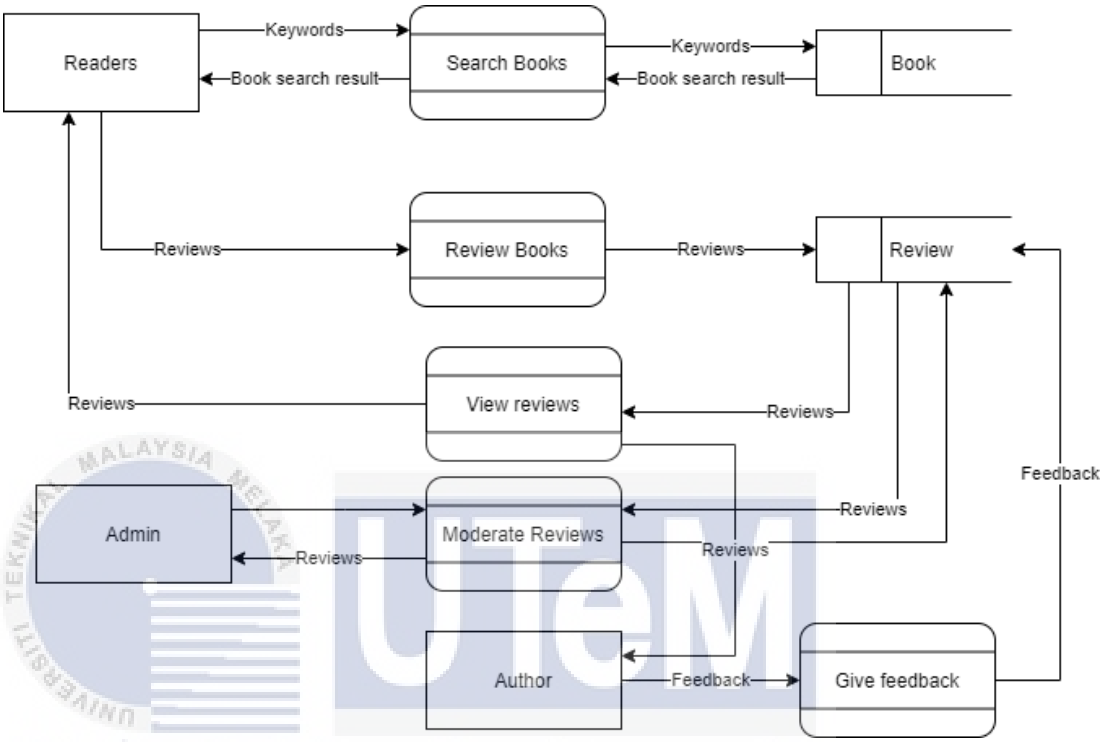


Figure 3.7: Level 1 DFD for Reviews Module

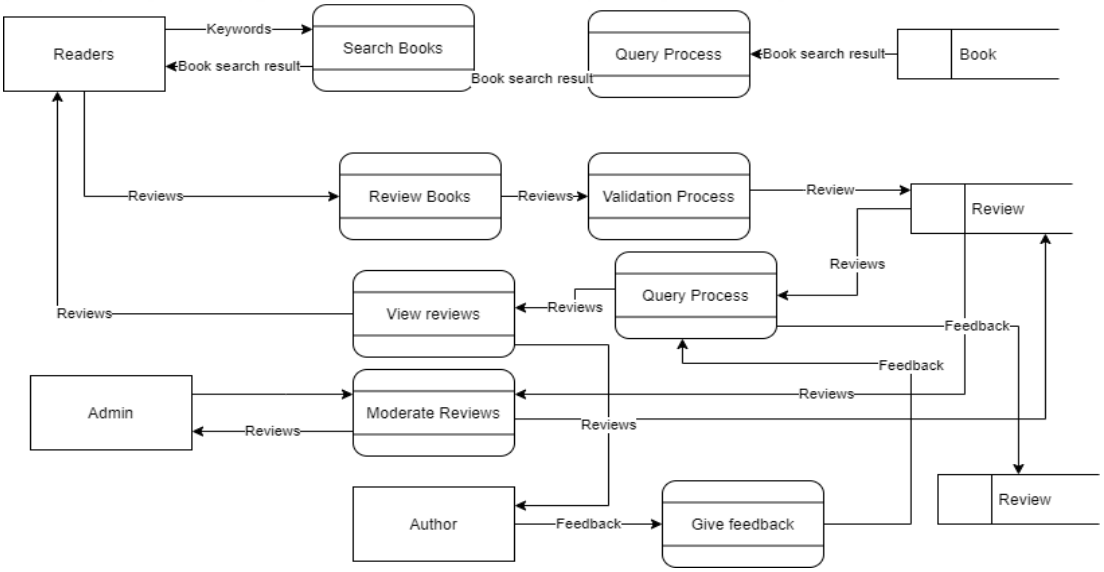


Figure 3.8: Level 2 DFD for Reviews Module

Reviews module DFD shows how readers add a review for a book and search for a book. The system receives the keyword from the user and will fetch the books from the database containing the keyword. When adding review, the user will input the review before being validated and added into the database. Through a database query, a user may see all the reviews for a book. And admin can also moderate reviews through the moderate reviews process which take reviews from a query process.

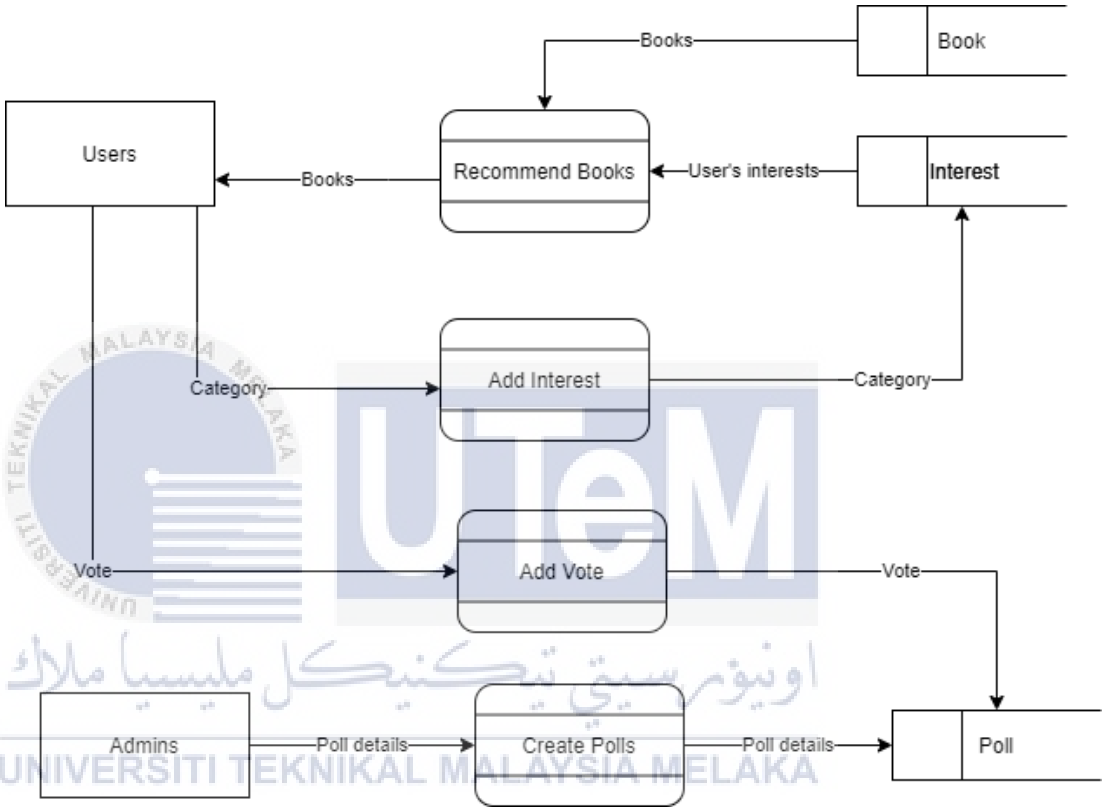


Figure 3.9: Level 1 DFD for Special Module

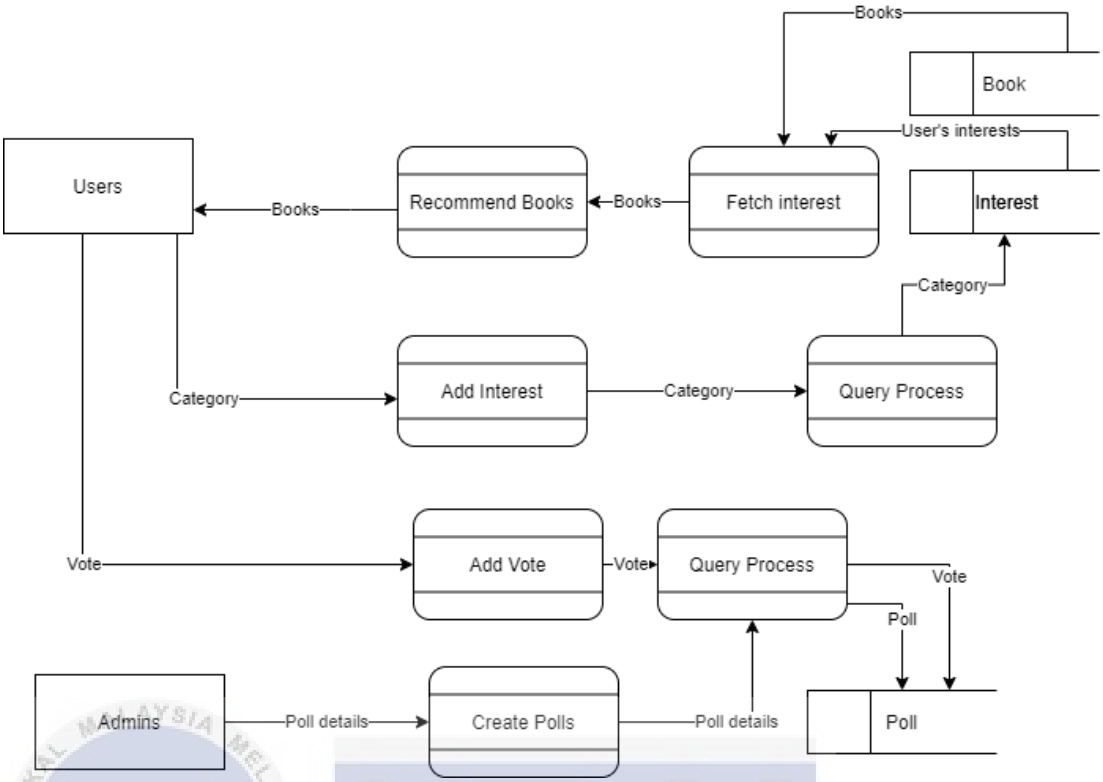


Figure 3.10: Level 2 DFD for Special Module

In the special module, a user can add a category into their interest. A process then will fetch books from the database bases on the logged in user's interest and recommend them to the user. The special module also provides a voting system. An admin can create a poll and add options of books into the poll, and a user may make a vote.

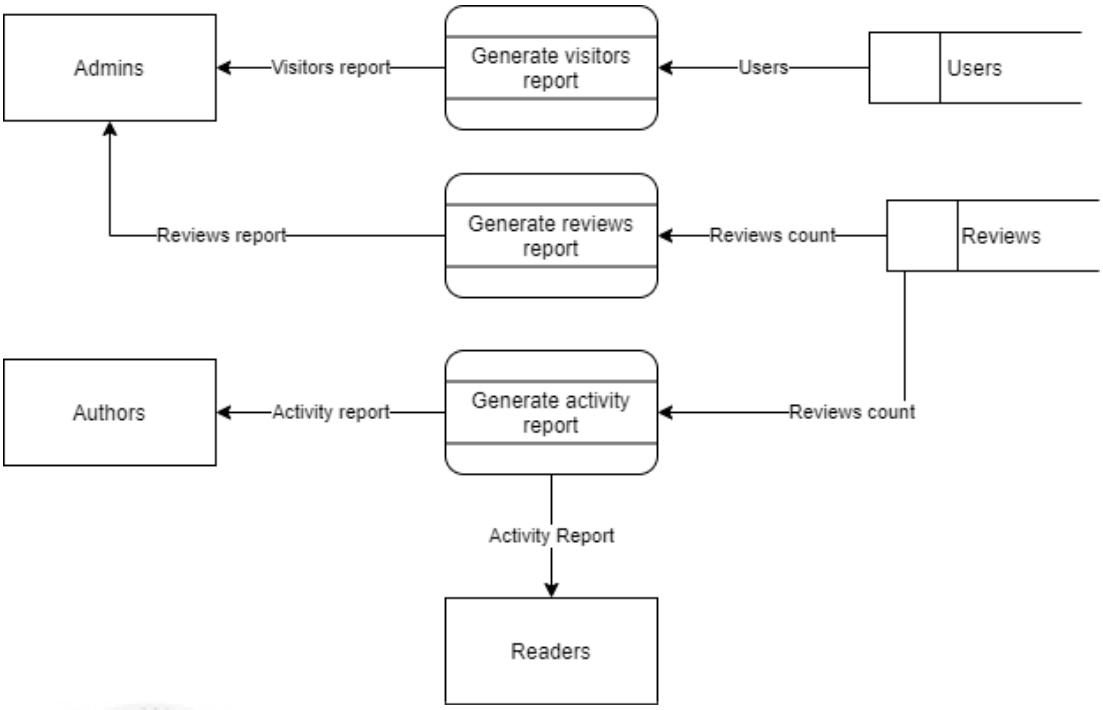


Figure 3.11: Level 1 DFD for Report Module

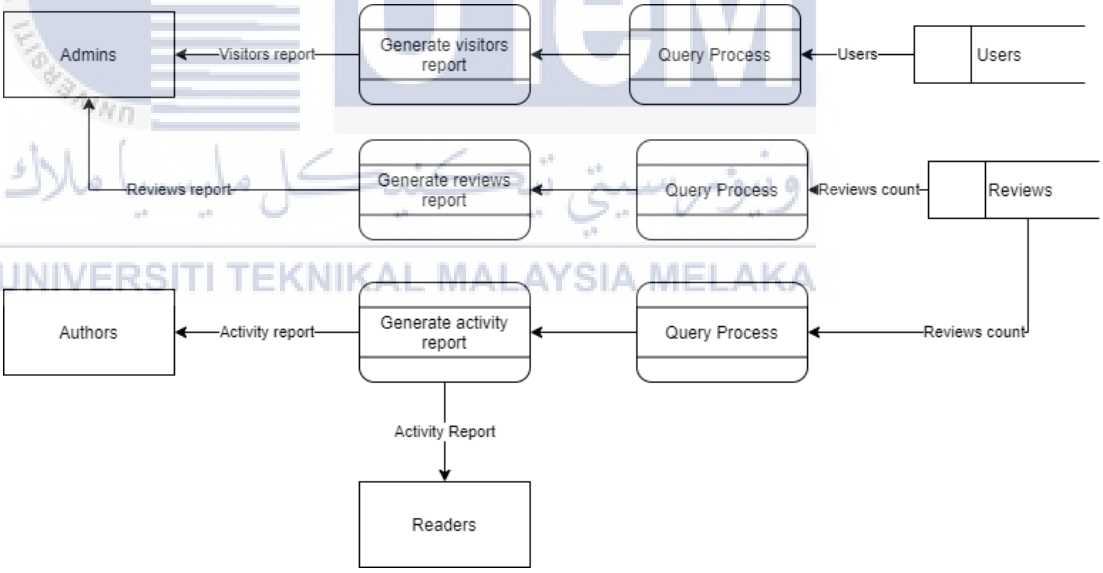


Figure 3.12: Level 2 DFD for Report Module

In the report module, multiple query process will fetch data from the users and reviews table to calculate statistics such as new users growth, and new reviews added into the database.

3.3.3 Non-functional Requirement

Table 3.13: Non-Functional Requirements

Non-Functional Requirement	Description
Security	Only registered users are allowed to leave reviews. Only logged in admins and admins only can access the admin menu.
Integrity	Data stored in the database must be valid and accurate. The data also need to be always consistent.
Performance	Review should reflect on the review score immediately after a review is submitted.
Usability	The UI must be easy to be understood by users and require no training to use.
Availability	The website is always available to users.

3.4 Conclusion

From chapter 3, the analysis between the existing system and to-be system is discussed to overcome the weaknesses of the existing system. The data flow of the

proposed system is illustrated by using data flow diagram. The analysis of each system is important as it will become the guideline for database design.

In the upcoming chapter, the project will proceed with design phase. Further discussion of database design that includes three types of design are conceptual design, logical design and also physical design. The entity relationship diagram, business rules, data dictionary and the graphical user interface will be displayed and explain more in chapter 4.

CHAPTER 4: DESIGN

4.1 Introduction

It is important to concern about the architecture of the system. because systems design is the process of defining elements of a system like modules, architecture, components, and their interfaces about the interface, an interactive interface will attract users to using this system and data for a system based on the specified requirements. It is the process of defining, developing, and designing systems that satisfy the specific needs and requirements of a business or organization.

4.2 High Level Design

4.2.1 Architecture Design

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. For Book Ranks, a three-tier architecture is being implemented. A three-tier architecture is divided into three layers which are the presentation layer, application layer and data layer.

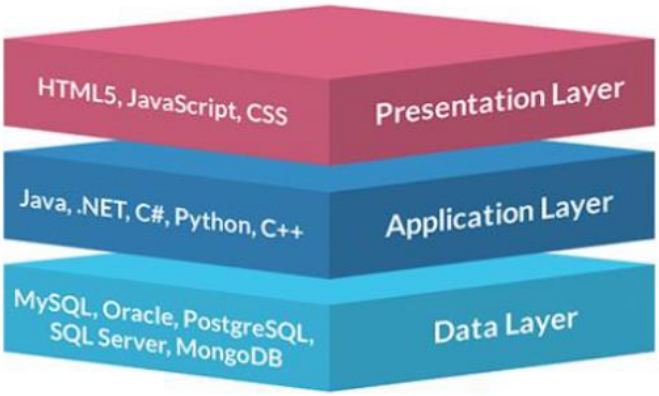


Figure 4.1: Three-Tier Architecture Layers

Figure 4.1 displays the layers in the three-tier architecture. The presentation layer is the front-end layer where it consists of the user interfaces. The user interface is accessible through web browser. It displays the content and information for the end user. In this layer Bootstrap is used as the web development framework. Next is the application layer. The application layer includes the functional business logic which drives the core capabilities of a developed system. The system is written in PHP with the help of Laravel framework. Lastly, the data layer. Data layer composes of the database and data access layer. MySQL is being used as the DBMS.

4.2.2 User Interface Design

A graphical user interface (GUI) is a type of user interface through which users interact with electronic devices via visual indicator representations. The basic elements of GUI are check boxes, buttons, label buttons, radio buttons, dropdown lists, text boxes and many more. This system implements Bootstrap as the front-end framework to ensure all GUI are created consistently and have responsive structures and styles. Below are some of the interfaces of the Book Ranks following with short explanation of each page.

Refer **Appendix A** for more UI screenshots.

4.2.2.1 Navigation Design

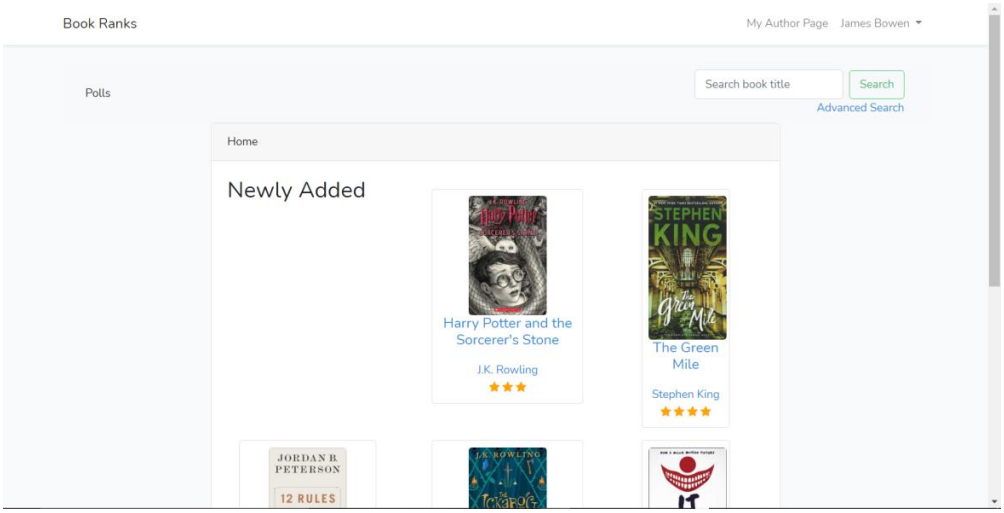


Figure 4.2: Home Page

Figure 4.2 shows the homepage. From this page, a user can navigate to a book page, log in page, poll, page, and register page.

4.2.2.2 Input Design

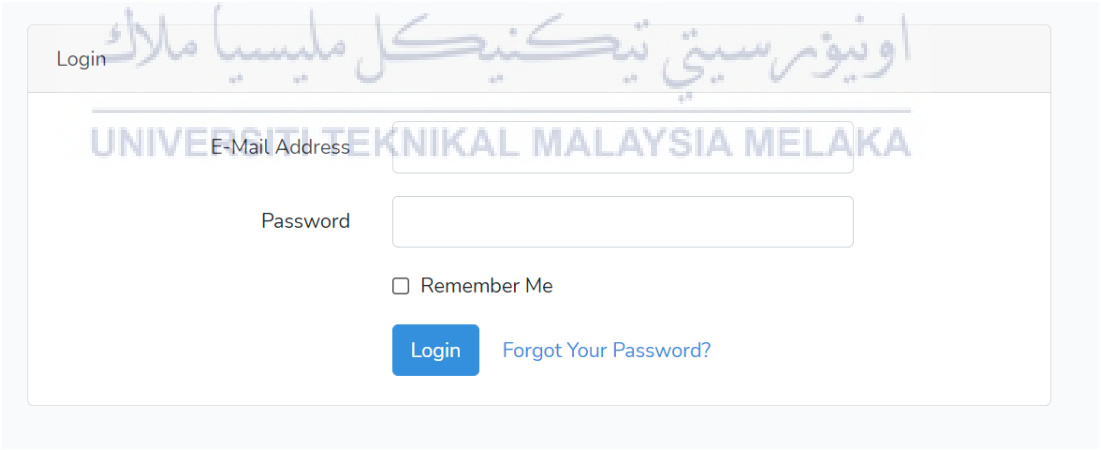


Figure 4.3: Login Page

Figure 4.2 shows the login page that can be used by all three users which are admins, staff, and readers. In the login page, form control styles are used to allow user to insert email address and password for login purpose. A login button is available for user to click to submit the login form.

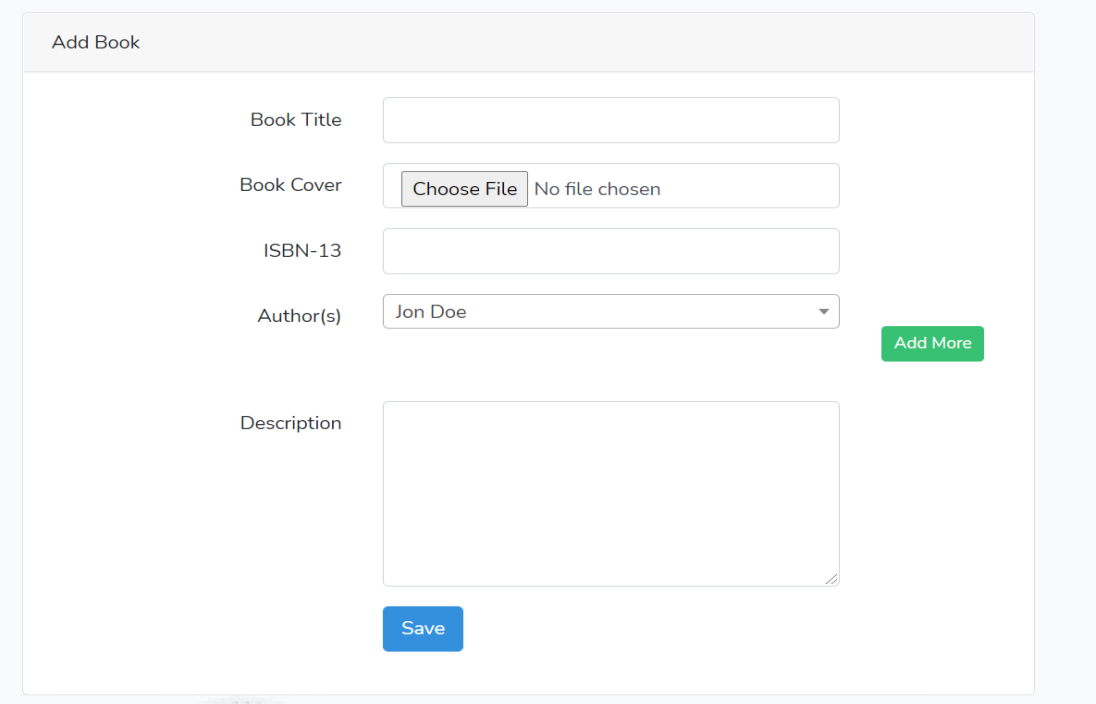


Figure 4.4: Add Book Page

Figure 4.4 shows the form used to insert a new book. It consists of multiple inputs such as title, isbn, book cover, author, and description.

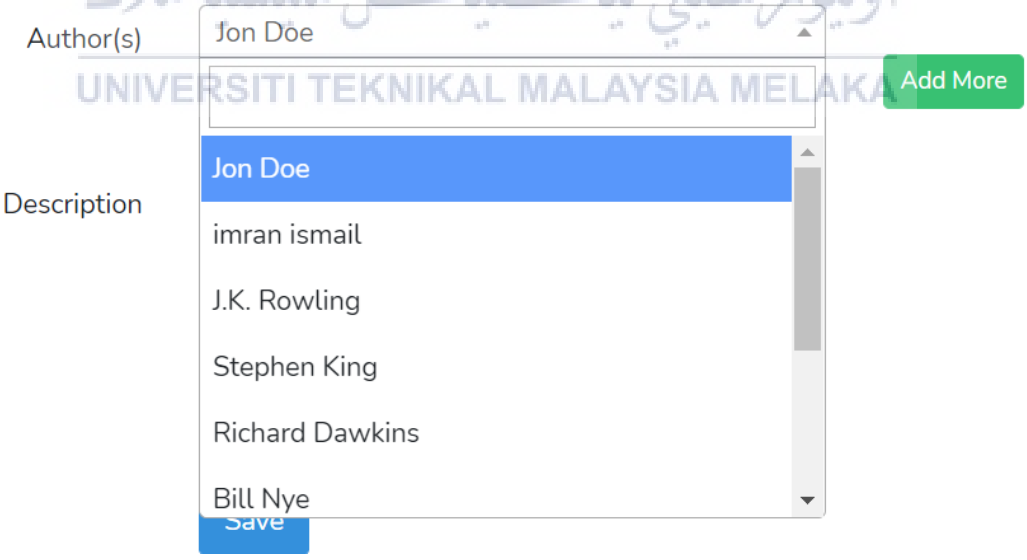


Figure 4.5: Dropdown List To Add Author

Figure 4.5 shows a dropdown to select author while inserting a new book. A dropdown list is a graphical control element that allow user to choose only one value from the list. However, use can also searched for the author in list.

4.2.2.3 Output Design

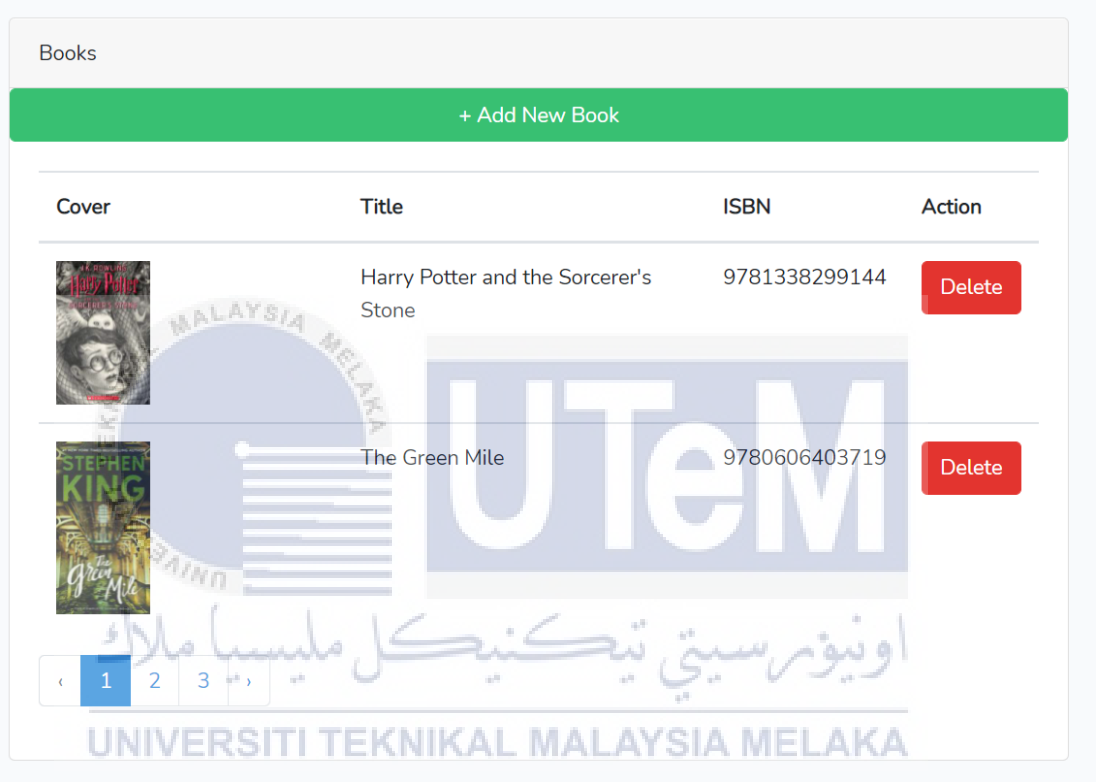


Figure 4.3: Manage Books

Figure 4.3 shows the Manage Book Page. It can only be seen by user admin. Each book is being list out in the form of table which are being sort according to the created date in descending order. Thus, the latest inserted part will be on the top. Each book can be deleted. User can also go to next page.

4.2.3 Database Design

4.2.3.1 Conceptual and Logical Database Design

Figure 4.6 shows the conceptual ERD. The conceptual ERD models information gathered from business requirements. Entities and relationships modeled in such ERD are defined around the business's need. The need of satisfying the database design is not considered yet. Conceptual ERD is the simplest model among all.

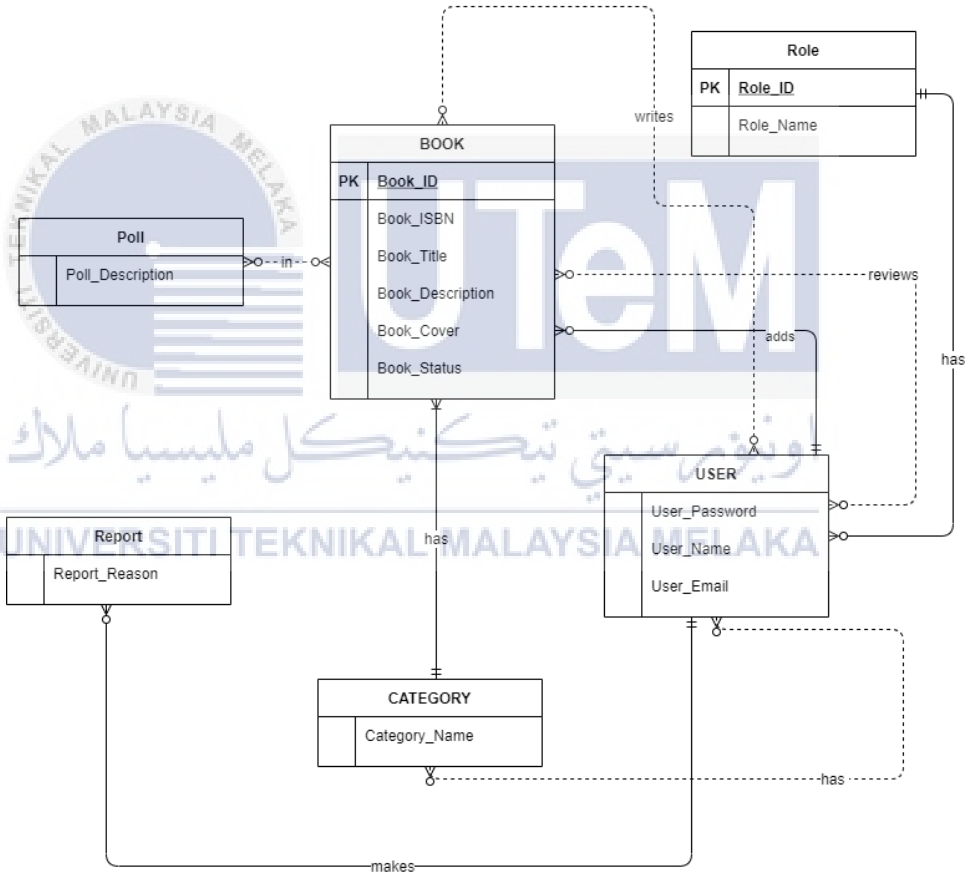


Figure 4.6: Conceptual ERD

Figure 4.7 shows the logical ERD Logical ERD also models information gathered from business requirements. It is more complex than conceptual. It has nothing to do with database creation yet. In, logical ERD, we identify the needed columns such as Primary Keys columns.

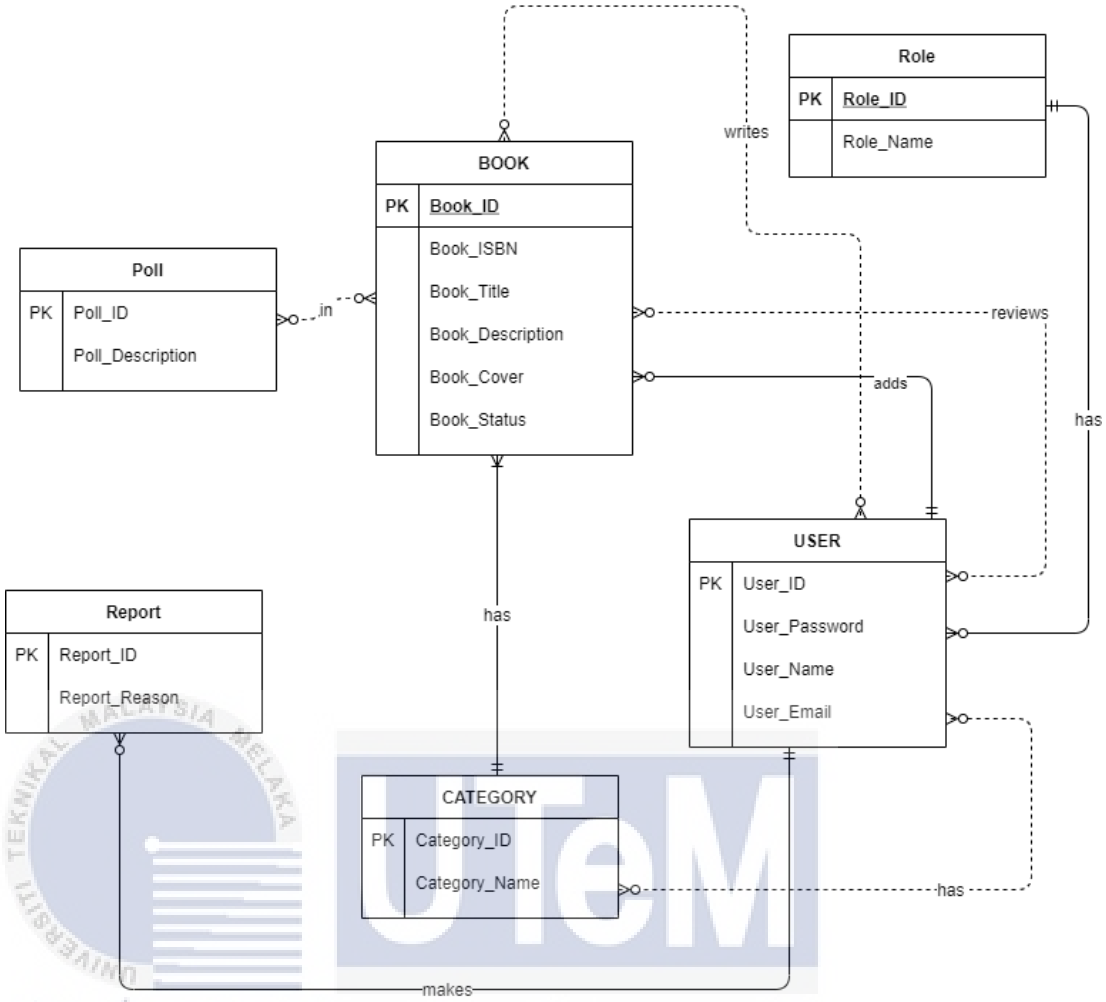


Figure 4.7: Logical ERD

4.2.3.2 Business Rules

Business rule is a list of statement that defines the requirement from the business operation and its constraints. It must be applied on the information system to ensure that the system is developed according to the business operations and requirements. The business rules for Book Ranks are as below:

a) A user may have one and only one role.

b) A user may have zero or many reviews.

c) A user who is an admin may insert zero or many books.

d) An author may or may not be a user.

e) A book has one or more authors.

f) A user can create one or more polls.

g) A poll can have many books and a book may be in many polls.

h) A book belongs to one and only one category, a category can have zero or many books.

4.3 Detailed Design

4.3.1 Software Design

In software design, we explain methods/operations such as its responsibility, input/output parameter, pre/post condition and algorithm. Refer to **Appendix B** for more sample source codes.

This Vote model represents the vote table in the database including its relationships.

The model sets the Vote relationship with the Book and Poll models. That one vote contains one book and belongs to one poll.

```
<?php


namespace App\Models;


use Illuminate\Database\Eloquent\Factories\HasFactory;
```

```
use Illuminate\Database\Eloquent\Model;



class Vote extends Model

{

    use HasFactory;

    protected $fillable = ['user_id','poll_id', 'book_id'];



    public function book(){

        return $this->belongsTo(Book::class, 'book_id');

    }

    public function poll(){

        return $this->belongsTo(Poll::class, 'poll_id');

    }

}
```

This function belongs in the Vote Controller. The store function would store a vote made by a user. If a user has already voted for a poll, it will update the user's vote. It receives the input from the request and run the function update or create. If

record already exist, it will update the previous vote based on the poll and user ID. Then it will run the save function to store the record.

```
public function store(Request $request)
{
    //dd($request->b_id);


    $vote = Vote::updateOrCreate(
        ['poll_id' =>$request->poll_id, 'user_id' =>Auth()->id()],
        ['book_id' =>$request->b_id]
    );

    if($vote->save()){
        return redirect()->back()->with('success', 'Voted');
    }
}
```

4.3.2 Physical Database Design

Physical database design is the process of producing a description of the implementation of the database on secondary storage; it describes the base relations, file organizations, and indexes design used to achieve efficient access to the data, and any associated integrity constraints and security measures. Sources of information for physical design process includes logical data model and documentation that describes model. In this phase, the database design is implemented, and a DBMS must be chosen to use. The result of this phase is Data Definition Language (DDL) and the physical ERD.

Physical ERD represents the actual design blueprint of a relational database. It is the result of identifying the necessary columns such as foreign key from the logical ERD. Physical ERD also adds a bridge table between many to many relationships.

Refer **Appendix C** for DDL.

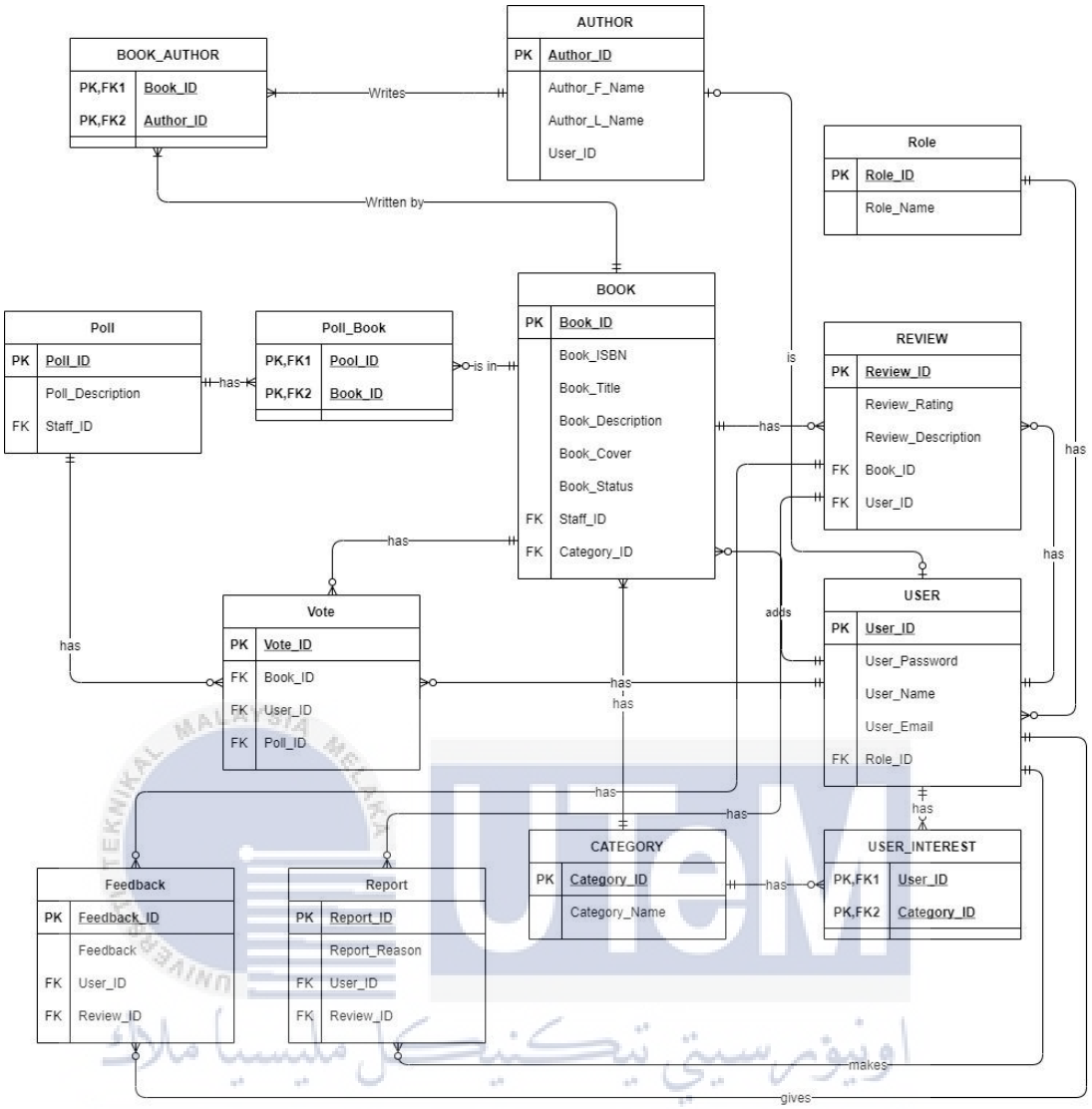


Figure 4.8: Entity Relationship Diagram

4.4 Conclusion

In chapter 4, there are several aspects that need to be considered when designing the database, since each of them would either influence how some of the information that needs to be translated into properly formed data would have an impact on the types of linking, counting and analyzing it would be possible to perform on the data. In designing a database, apart from involving the modelling of information from the sources, it also involves the conceptual approach on the database, the functions and behavior of the database and how the table and fields of the database operate. After

designing the database, the GUI design is then produced as the product of the information system that end-users will interact with.

For chapter 5, the system implementation starting from the development environment to the database implementation of the selected DBMS will be discussed and explained in detailed.

CHAPTER 5: IMPLEMENTATION

5.1 Introduction

The project will now proceed to the implementation of software development environment and database implementation. Environment refers to the set of hardware and software tool used by a system developer to develop a software system.

5.2 Software Development Environment Setup

A system development environment consists of broad sets of processes and programming tools in developing a software. To develop this system, XAMPP is chosen as the development environment as it is completely free and easy to install and configure. Next, tools required to create, customize, and maintain web-based system, a web development framework also need to be considered thus a Laravel framework is used resulting from the system development process. Laravel is a software architecture standard that divides the representation of information from users' interaction with it. The MVC (Model View Controller) architecture model is implemented. MVC method is more structured, arrange orderly and easier to be developed where it separates the main components for instance the data manipulation which represents the Model, the display or interfaces represents the View and the process that portrays the Controller.

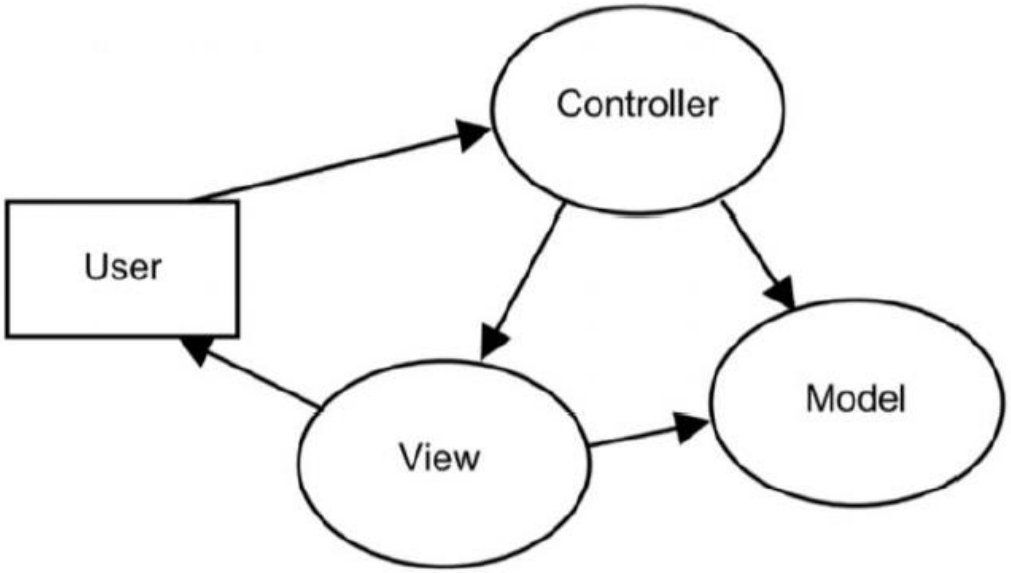


Figure 5.1: Model View Architecture (MVC) Model

To develop this project, firstly a directory is set up inside the 'C' directory named 'PSM'. This is where all the source code, resources used by the system is located.

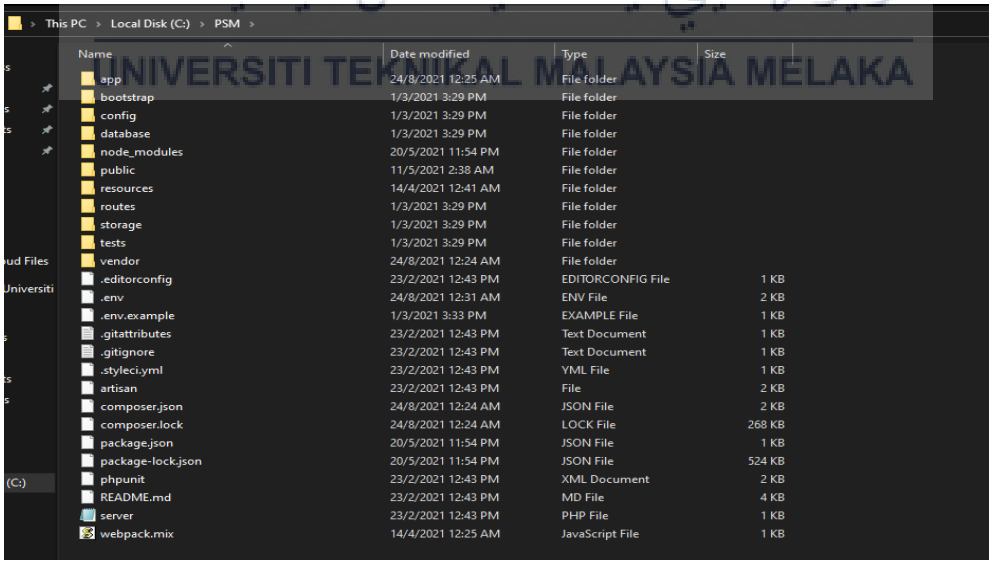


Figure 5.2: PSM Directory.

Next, the system developer must make sure all external hardware to be used in the system is available. In this case, a camera is needed to scan books' barcode.

The system developer also needs to make sure that all ports such as 8000 and 3306 are clear to be used by PHP artisan, the database, and Apache.

Related documentation such as Laravel documentation must be prepared as a guide to the developer.

5.3 Software Configuration Management

5.3.1 Configuration Management Setup

5.3.1.1 XAMPP

XAMPP provide a reliable, easy and fast way to set up environment for PHP programming. All the components needed for developing, running, debugging and unit testing for PHP applications are provided by XAMPP. It is a cross-platform package consisting of an Apache HTTP server, MySQL database and PHP interpreter.

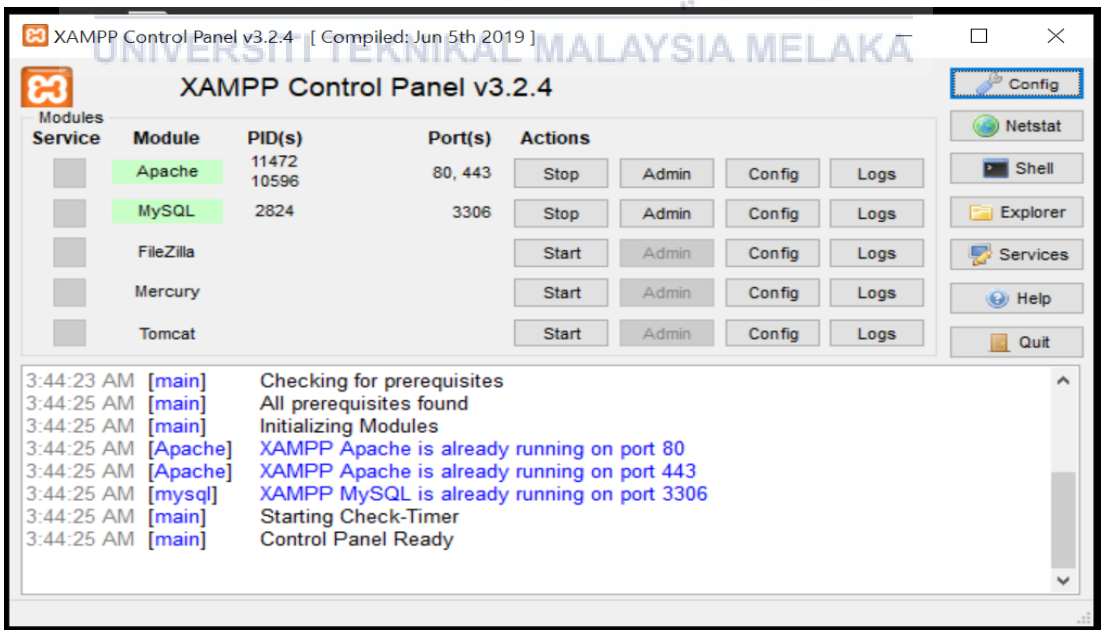


Figure 5.3: XAMPP Control Panel

5.3.1.2 Composer

Composer is a tool for dependency management in PHP. It enables users to declare the libraries that the project depends on and it will handle the process of install or update for users. Figure is the final view once the Composer has been installed and configured.

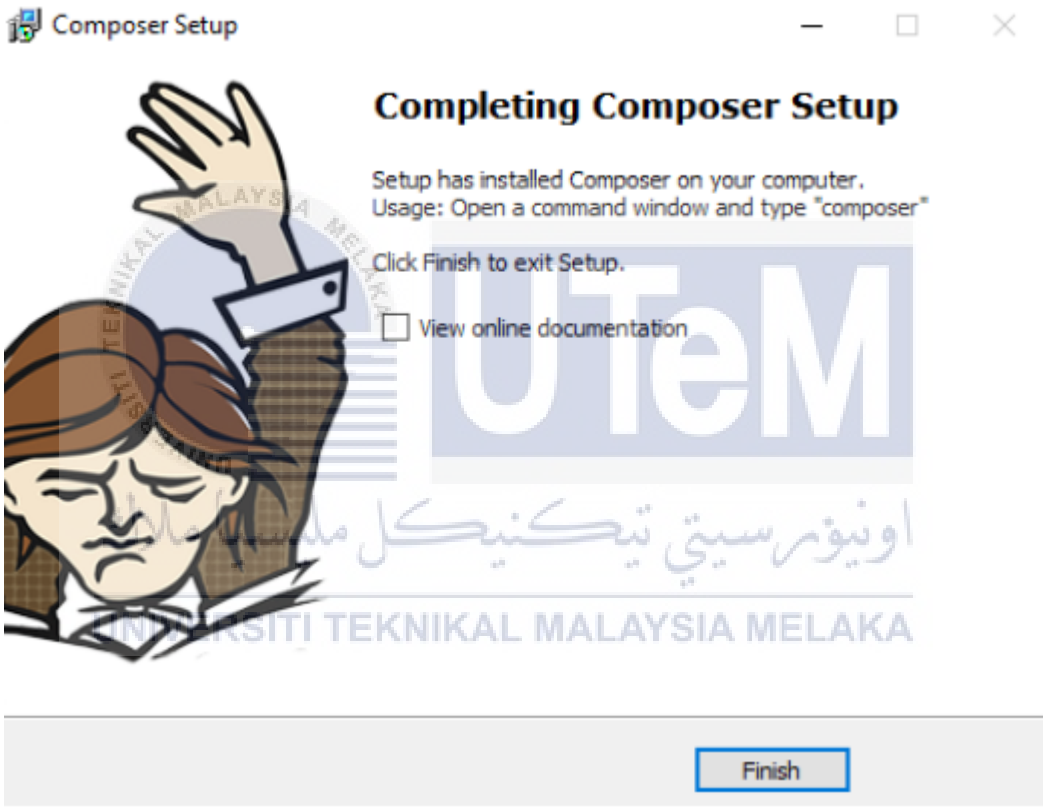


Figure 5.4: Composer Installed

5.3.1.3 Laravel Framework

Laravel utilizes composer to manage its dependencies thus explain the reason why composer need to be installed first. To proceed with installing the Laravel, issue the following command in the terminal.

```
C:\Users\Imran>composer create project PSM
```

Figure 5.5: Creating Laravel Project

After installing, all the directories of Laravel are being created and here is where the MVC architecture can be seen. For instance, the MVC is represented by the following folders and files where a user is able to login: Table 5.1 is the example to show the implementation of MVC architecture in Laravel directories.

Table 5.1: MVC Architecture

MVC Architecture	Application Path	File
Model	app/	User.php
View	resources/views	Welcome.blade.php
Controller	app/Http/Controllers	Auth/AuthController.php

5.3.1.3 Visual Studio Code

Visual Studio Code is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

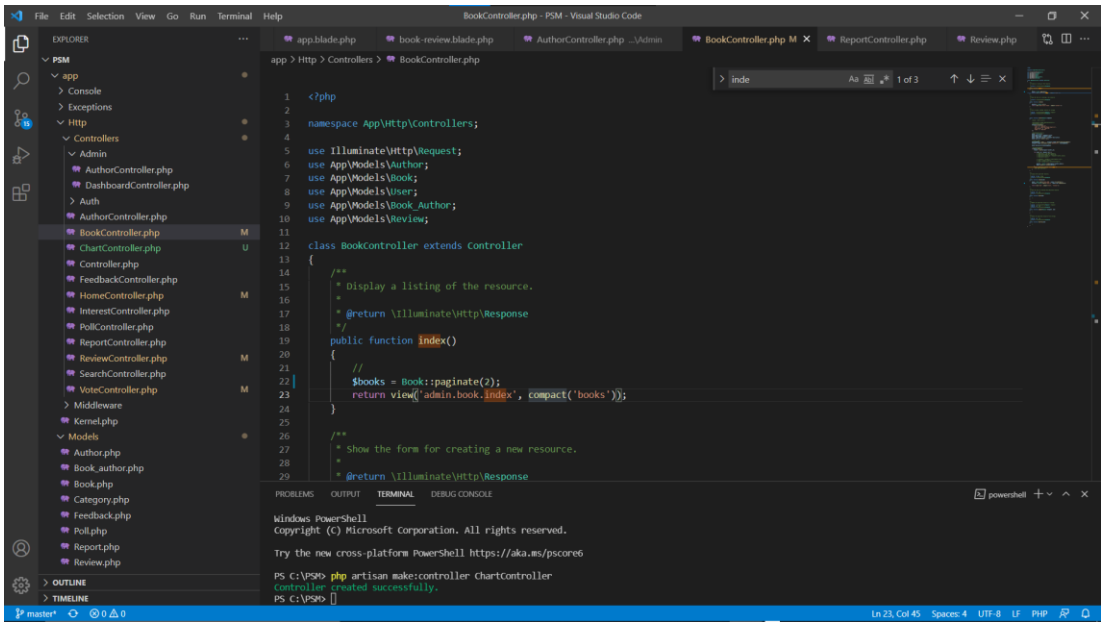


Figure 5.6: VS Code

5.3.2 Version Control Procedure

Version control is done through local repository and a private repository on GitHub.

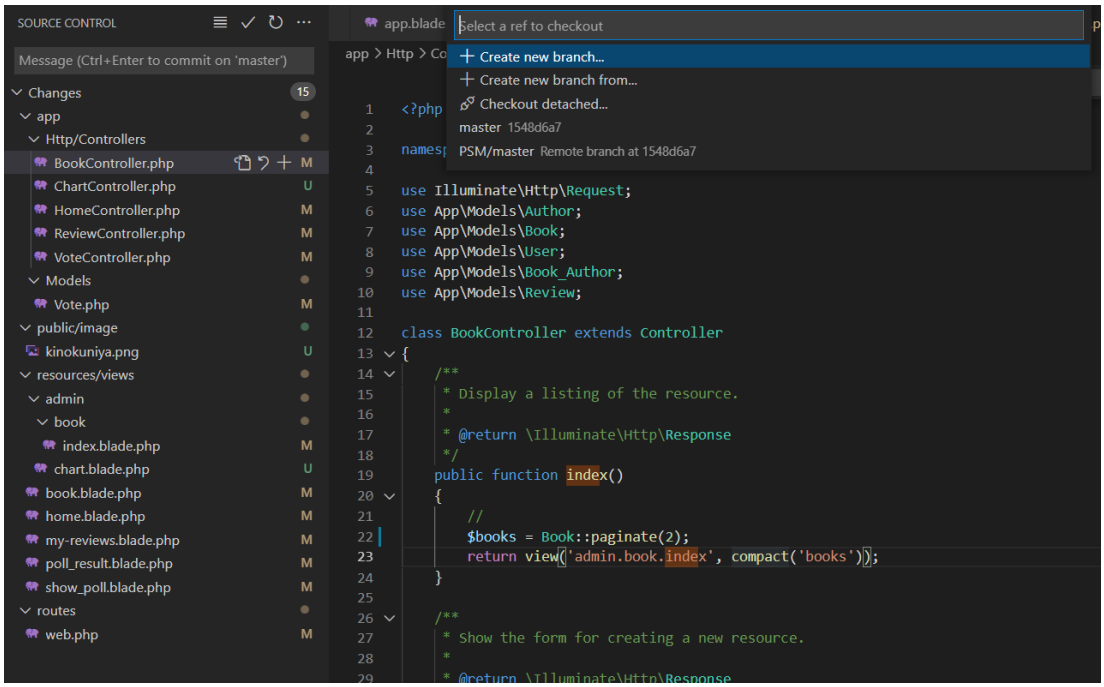


Figure 5.7: VS Code Source Control

Figure 5.8: Project's GitHub Page.

5.4 Implementation Status

This section describes the progress of the development status for each of the module in the system. The implementation status of each of the main module and sub module are presented in Table 5.2.

Table 5.2: Implementation Status

Module Name	Description	Duration	Date completed
Authentication Module	Users, can register, log in, update profile, and reset password	1 week	March – June 2021
Books Module	Admin can manage books by adding and editing details.	2 week	
Review Module	Users can leave reviews with ratings for any books in the database.	1 week	
Search Module	Users may search books by title, author, isbn, and even scanning a book's barcode	1 week	
Special Module	The system can give suggestion to users based on their interests. The site has a polling system where users can vote.	2 weeks	

5.5 Conclusion

In this chapter, the implementation is described by starting with the system development environment setup. XAMPP is chosen as the local web development along with Laravel as the PHP framework to ease the development process.

In the next chapter, testing phase of the software product is going to be discussed. Software testing is necessary to detect faults, defects and errors made throughout the development phases. It is also essential to find out does the system developed follows user's requirements and meet their expectations.

CHAPTER 6: TESTING

6.1 Introduction

To overcome the defect by fixing the errors, testing in software development is being introduced. Testing helps to evaluate the quality of software products in form of the number of defects found, the number of tests run, and the amount of system covered by the tests. Software testing can be done with the involvement of functional requirements and non-functional requirements of the software.

6.2 Test Plan

Test plan is a document that defines the scope, approach, resources and schedule of planned test activities. It will provide information on testing to be carried out such as test organization, test environment and test schedule.

6.2.1 Test Organization

In software testing, test organization is an organization, person, or company that are given with responsibilities to tests software products by following to agreed requirements. The organization will define the activities throughout the test process. For Book Ranks, it involves the software developer to create the test cases and a tester. This will make it easier for the developer to test and finding the errors and faults from the system. Testing need to be done to ensure the developed software

follows the given requirements and meet user's expectation. Table 6.1 shows the roles and responsibilities of the test organization.

Table 6.1: Roles and Responsibilities

Roles	Responsibilities
System Developer	• Defining test plan • Designing test case • Executing test activities • Test the system based on given test script • Analyzing test result • Take action to correct the faults and errors • Ensuring the system follows all users' requirements
System tester	Executing test activities

6.2.2 Test Environment

Test environment is designed to set-up and identify any required infrastructure and tools to run the test cases. The includes the software and hardware needed for software testers to execute the test cases. Among the factors that need to be identified to design the test environment are identifying the required operating system, clarify the databases used, and the local server to execute the system for testing purpose.

6.2.3 Test Schedule

The testing starts on the simpler module or unit first.

Table 6.2: Test Schedule

	Task	Start Date	Duration (Day)
1.	Identify test activity	19/07/2021	2
2.	Describing test cases	21/07/2021	3
3.	Test authentication module	26/07/2021	1
4.	Test book records module.	27/07/2021	1
5.	Test reviews module.	28/07/2021	2
6.	Test feedback system.	30/07/2021	1
7.	Test polling system.	02/08/2021	2

Table 6.3: Software Environment

Software	**Description**
XAMPP Server	A local web server used by Book Ranks
MySQL	An open-source relational database management system (RDBMS).
Google Chrome	A web browser to run the user interface

6.3 Test Strategy

Test strategy in software testing is documented to specify the test design involves and regulates how the testing process will be done. It is the most important document in software testing as it explains how the software product will be tested. For Book Ranks, test approaches or test strategies that is carried out are both white-box testing and black-box testing. Both is required to ensure software delivery.

Black-box testing is also known as behavioral testing. It is a software testing method where the internal structure, design, implementation of the test items is tested without having knowledge about it. This type of testing emphasized on input and output of software applications by following to software requirements and specifications. To use this method of testing, a tester does not need to have the knowledge of implementation and programming. The method is used to look bugs in the way system behaves by trial and error of ways. The method used is the use case testing method.

White-box testing is categorized as dynamic testing in which the testing involves the execution of the software of a component or a system. A method of white box testing we will be using is the statement coverage method. It is where we aim to run all the statements inside a process at least once.

The approach taken in testing is the bottom up approach where the smaller and simpler modules are tested first before moving on to more complex and integrated modules.

6.3.1 Test Classes

The classes of test that is selected to test BookRanks is described as the following. Firstly, the functional testing. Functional testing is a type of testing considers the behaviour of the system and tested against the functional requirements and specifications. Functional testing is done to ensure the end-user is satisfied as all the requirements are fulfilled. Next is security testing, Security testing is a process intended to reveal flaws in the security mechanisms of an information system that protect data and maintain functionality as intended. We will be testing the authentication module of the system when logging in and the admin log in.

6.4 Test Design

6.4.1 Test Description

In this section, test case identification, test cases and expected result for each module are designed and documented. The following are some of the test descriptions of the system testing which are the testing for authentication module, books module, review module, and special module. Here we design the steps necessary to carry out the testing and what is expected result that we want to achieve.

6.4.1.1 Use Case Testing (Black Box Testing)

Use Case Testing is a software testing technique that aids in the identification of test cases that span the full system from beginning to end on a transaction-by-transaction basis. Interactions between users and software applications are called test cases. Use case testing identifies holes in software applications that would otherwise go undetected if individual software components were tested.

Table 6.4: Test Description A001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
A001_01	Register User	Register button is clicked with all field left empty		1.Leave all input blank 2.Click register	Alert message of field is required appears.
A001_02	Register user	Register with taken email	1.Register with an email	1.Fill all input fields 2.Use a taken email	Error “email is already taken appears”
A001_03	Register user	Register with a short password		1.Insert password shorter than 8 characters 2.Click register	Error “The password must be at least 8 characters” appears

A001_04	Register user	Register with non-matching passwords		1.Insert non-matching passwords	Error "The password confirmation does not match." appears

Table 6.5: Test Description A002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
A002_01	Log In	Log in with valid reader email and password	Register user.	1.Insert email and password. 2.Click Log in	Log In and Redirect to home
A002_02	Log In	Log in with valid admin email and password	Register admin account.	1.Insert email and password. 2.Click Log in	Log in and redirect to admin menu
A002_03	Log In	Log In with non-existing email		1.Insert non-existing email 2.Click Login	Error "These credentials do not match our records" appears.
A002_04	Log in	Log in with wrong password	1.Register an accunt	1.Insert valid email 2.Insert wrong password 3.Click Login	Error "These credentials do not match our records" appears

Table 6.6: Test Description B001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
B001_01	Record new book	Leave all input blanks		1.Leave all input blanks 2.Click "Save"	Alert message of field is required appears
B001_02	Record new book	Insert invalid ISBN		1.Insert ISBN shorter than 13 digits 2.Insert all inputs 3.Click "Save"	Error "The isbn must be 13 digits" appears.
B001_03	Record new book	Insert invalid ISBN		1.Insert ISBN with letters 2.Insert all inputs 3.Click "Save"	Error "The isbn must be an integer" appears.
B001_04	Record new book	Insert valid inputs		1.Insert valid inputs 2.Click "save"	Book is added

Table 6.7: Test Description C001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
C001_01	Add a book review	Leave review blank		1.Leave all input blanks 2.Click "Add Review"	Alert message of field is required appears.
C001_02	Add a book review	Insert short review		1.Insert review shorter than 20 characters. 2.Click "Add Review"	Error "The review must be at least 20 characters" appears.
C001_03	Add a book review	Insert review for a book		1.Insert review	Review added

				2.Click "add review"	
C001_04	Add a book review	Insert review for same book	1.Add a review for the same book	1.Insert review 2.Click "Add Review"	Review is updated

Table 6.8: Test Description C002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
C002_01	Add a book review	Report a review		1.Click "report" 2.Select reason 3.Submit report"	Report is submitted
C002_02	Add a book review	Ignore report (Admin)	1. Make a report	1.Click 'ignore'	Report is deleted.
C002_03	Add a book review	Remove review from report (Admin)	1.Make a report	1.Click Remove 2.Confirms	Report and review are deleted.

Table 6.9: Test Description D001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
D001_01	Add a book review	Search book		1.Enter keyword. 2.Click search	Results are displayed.
D001_02	Add a book review	Advanced Search	1. Make a report	1.Select search by option.	Results are displayed.

				2.Enter keyword. 3.Click "search"	
D001_03	Add a book review	Advanced Search	1.Make a report	1.Click Remove 2.Confirms	Redirect to book page.

Table 6.10: Test Description F001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
F001_01	Insert poll	Insert poll (Admin)		1.Insert all inputs	Poll created
F001_02	Close Poll	Close Poll (Admin)	1. Create a poll	1.Click "Close"	Poll status updated to closed
F001_03	User voting	Insert Vote	1.Create a poll	1.Click an option. 2.Click "Vote"	Vote inserted
F001_04	User voting	Insert vote for a same poll	1.Voted in a same poll	1.Click an option in previous poll 2.Click "vote"	Vote updated

Table 6.11: Test Description F002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
F002_01	Reply to a review	Insert feedback.		1.Insert feedback. 2.Click "reply"	Feedback inserted.
F001_02	Reply to a review	Insert feedback for same review.	1. Create a poll	1.Insert feedback for same review 2.Click "reply"	Reply updated

6.4.1.2 Statement Coverage (White Box Testing)

White Box Testing is software testing technique in which internal structure, design and coding of software are tested to verify flow of input-output and to improve design.

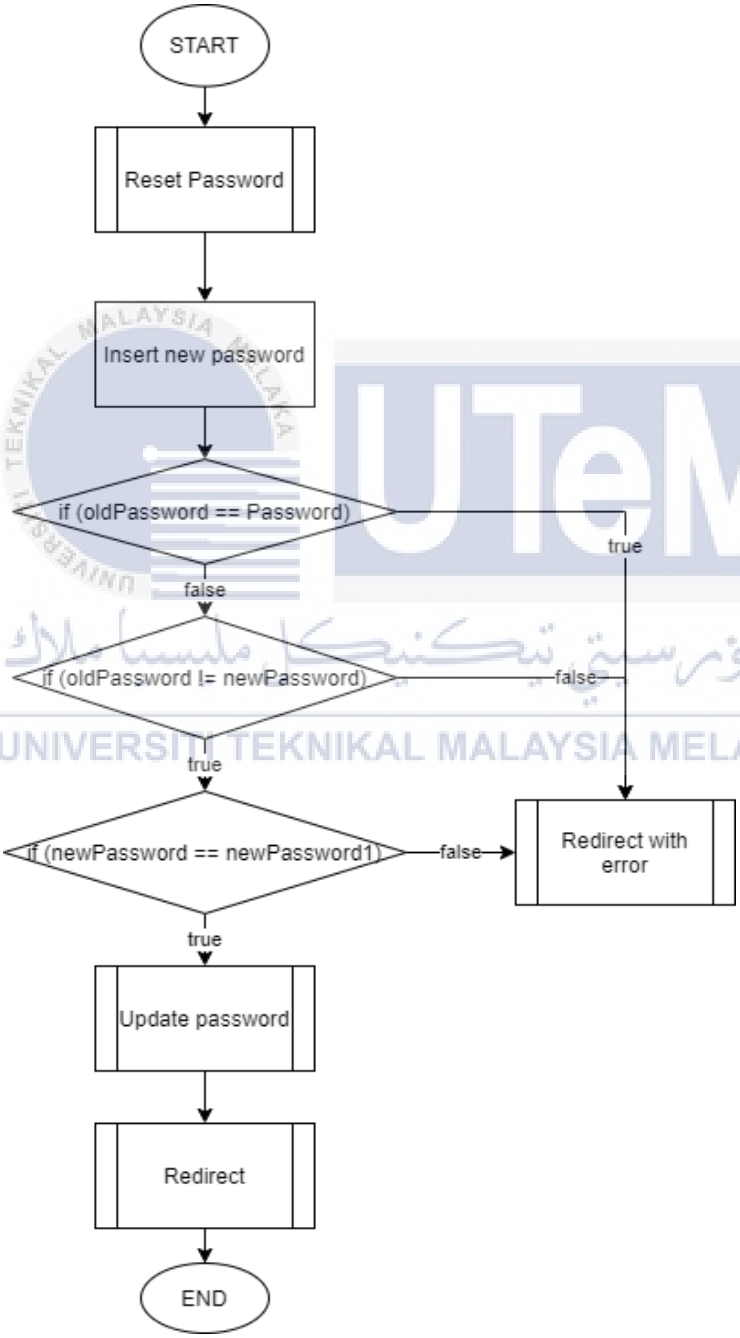


Figure 6.1:Reset Password Statements.

Table 6.12: Test Description G001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
G001_01	Reset password	Insert feedback.	1.Logged in	1.Insert old password and new password. 2.Click "reset"	Redirect with error
G001_02	Reset password	Insert feedback for same review.	1. Logged in	1.Insert old password and new password. 2.Click "reset"	Password updated.

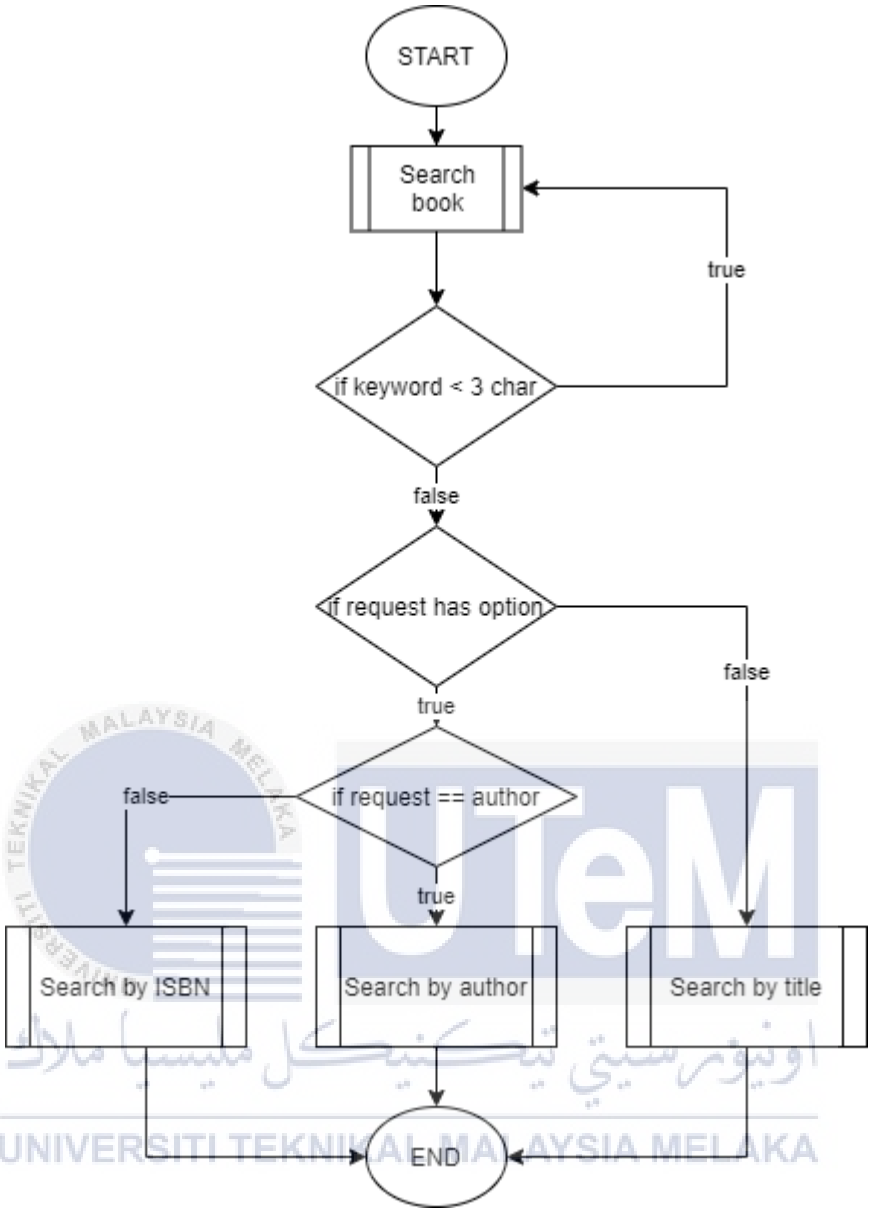


Figure 6.2: Advanced search statements.

Table 6.13: Test description G0002.

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result
G002_01	Search book	Search by title.		1.Insert title 2.Click "search"	Books containing keyword is displayed
G002_02	Search book	Search by author.		1.Insert author's name	Books containing

				2.Click "search"	keyword is displayed
G002_03	Search book	Search by ISBN.		1.Insert ISBN 2.Click "search"	Books containing keyword is displayed

6.4.2 Test Data

Test data is list of data that is used as an input to execute the test cases of a software testing. to uncover the faults or defects during testing, test data must be precise and comprehensive. Some of the data may be in form of positive testing in order to verify that the actual result meets the expected result.

Table 6.14: Test Data A001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
A001_01	Register User	Register button is clicked with all field left empty		1.Leave all input blank 2.Click register	-	Alert message of field is required appears.
A001_02	Register user	Register with taken email		1.Fill all input fields 2.Use a taken email	Name: Imran Email: Muhdimran1810@gmail.com Password: Abcd1234 Confirm Password: Abcd1234	Error "email is already taken appears"
A001_03	Register user	Register with a short password		1.Insert password shorter than 8 characters 2.Click register	Name: Jon Email: jondoe@gmail.com Password: Abc	Error "The password must be at least 8 characters" appears

					Confirm Password: Abc	
A001_04	Register user	Register with non-matching passwords		1.Insert non-matching passwords	Name: Jon Email: jondoe@gmail.com Password: Abcd1234 Confirm Password: Abce1234	Error "The password confirmation does not match." appears

Table 6.15: Test Data A002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
A002_01	Log In	Log in with valid reader email and password	Register user.	1.Insert email and password. 2.Click Log in	Email: imranhuha@gmail.com Password: abcd1234	Log In and Redirect to home
A002_02	Log In	Log in with valid admin email and password	Register admin account.	1.Insert email and password. 2.Click Log in	Email: muhdimran@gmail.com Password: asdf4321	Log in and redirect to admin menu
A002_03	Log In	Log In with non-existing email		1.Insert non-existing email 2.Click Login	Email: Kepala@gmail.com Password: abcd1234	Error "These credentials do not match our records" appears.

A002_04	Log in	Log in with wrong password		1.Insert valid email 2.Insert wrong password 3.Click Login	Email: Kepala@gmail.com Password: asdf4312	Error "These credentials do not match our records" appears

Table 6.16: Test Data B001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
B001_01	Record new book	Leave all input blanks		1.Leave all input blanks 2.Click "Save"	-	Alert message of field is required appears
B001_02	Record new book	Insert invalid ISBN		1.Insert ISBN shorter than 13 digits 2.Insert all inputs 3.Click "Save"	Title: The God Delusion Cover: 97801234567890 .jpeg ISBN: 97801234 Author: Richard Dawkins Description: The God Delusion is a 2006 book by British evolutionary biologist, ethologist Richard Dawkins	Error "The isbn must be 13 digits" appears.
B001_03	Record new book	Insert invalid ISBN		1.Insert ISBN with letters 2.Insert all inputs 3.Click "Save"	Title: The God Delusion Cover: 97801234567890 .jpeg ISBN: Okay Author: Richard Dawkins Description: The God Delusion is a 2006 book by British	Error "The isbn must be an integer" appears.

					evolutionary biologist, ethologist Richard Dawkins	
B001_04	Record new book	Insert valid inputs		1.Insert valid inputs 2.Click “save”	Title: The God Delusion Cover: 97801234567890 .jpeg ISBN: 97801234567890 Author: Richard Dawkins Description: The God Delusion is a 2006 book by British evolutionary biologist, ethologist Richard Dawkins	Book is added

Table 6.17: Test Data C001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
C001_01	Add a book review	Leave review blank		1.Leave all input blanks 2.Click "Add Review"	-	Alert message of field is required appears.
C001_02	Add a book review	Insert short review		1.Insert review shorter than 20 characters. 2.Click "Add Review"	Review: OK	Error "The review must be at least 20 characters" appears.
C001_03	Add a book review	Insert review for a book		1.Insert review 2.Click "add review"	Review: I liked it. It is a good book.	Review added
C001_04	Add a book review	Insert review for same book	1.Add a review for the same book	1.Insert review 2.Click "Add Review"	Review: I did not like it. It's a bad book.	Review is updated

Table 6.18: Test Data C002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
C002_01	Submit a report.	Report a review		1.Click "report" 2.Select reason 3.Submit report"	Reason: Spoilers	Report is submitted
C002_02	Act on a report	Ignore report (Admin)	1. Make a report	1.Click 'ignore'	Input: Ignore	Report is deleted.
C002_03	Act on a report	Remove review from report (Admin)	1.Make a report	1.Click Remove 2.Confirms	Input: remove	Report and review are deleted.

Table 6.19: Test Data D001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
D001_01	Search book	Search book	Insert book	1.Enter keyword. 2.Click search	Title: Harry	Results are displayed.
D001_02	Search book	Advanced Search	Insert book	1.Select search by option. 2.Enter keyword. 3.Click "search"	Author: Stephen Option: Author	Results are displayed.
D001_03	Scan a barcode	Advanced Search	Insert book	1.Click Remove 2.Confirms	Input: Barcode	Redirect to book page.

Table 6.20: Test Data F001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
F001_01	Create poll	Insert poll (Admin)		1.Insert all inputs	Title: Best Non-Fiction Book Description: What is your favorite Non-Fiction Book Books: Multiple books	Poll created
F001_02	Close poll	Close Poll (Admin)	1. Create a poll	1.Click "Close"	-	Poll status updated to closed
F001_03	Vote	Insert Vote	1.Create a poll	1.Click an option. 2.Click "Vote"	-	Vote inserted
F001_04	Vote	Insert vote for a same poll	1.Voted in a same poll	1.Click an option in previous poll 2.Click "vote"	-	Vote updated

Table 6.21: Test Data F002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
F002_01	Reply to review	Insert feedback.		1.Insert feedback.	Review ID: 1	Feedback inserted.

				2.Click "reply"	Feedback: Thank you	
F001_02	Reply to review	Insert feedback for same review.	1. Create a poll	1.Insert feedback for same review 2.Click "reply"	Review ID: 1 Feedback: Thank so much	Reply updated

Table 6.22: Test Data G001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
G001_01	Reset password	Non-matching new password	1. Logged in	1. Insert old and new password. 2. Click change password.	Old password: abcd1234 New password: asdf1234 New password confirm: asdf1235	Redirect with error
G001_02	Reset password	Reset password successful	1. Logged in	1. Insert old and new password. 2. Click change password.	Old password: abcd1234 New password: asdf1234 New password confirm: asdf1234	Password updated.

Table 6.22: Test Data G002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Test Data	Expected Result
G002_01	Search book	Search by title		1.Insert title. 2.Click search	Title: Harry	Display search results

G002_02	Search book	Search by author		1.Insert author. 2.Click search	Author: Bowen	Display search results
G002_03	Search book	Search by ISBN		1.Insert ISBN. 2.Click search	ISBN: 9781624758261	Display search results

6.5 Test Result and Analysis

All the result from test cases may vary with the involvement of expected result and actual result. Test result for BookRanks testing is identified by classifying either pass or fail based on the actual result. If the actual result differs from the expected result, then the test status is classified as fail and the problem needed to be fixed. The following are the test result for test cases on authentication module, books module, review module, and special module.

Table 6.23: Test Result A001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
A001_01	Register User	Register button is clicked with all field left empty		1.Leave all input blank 2.Click register	Alert message of field is required appears.	Alert message of field is required appears.	PASS
A001_02	Register user	Register with taken email		1.Fill all input fields 2.Use a taken email	Error "email is already taken appears"	Error "email is already taken appears"	PASS
A001_03	Register user	Register with a short password		1.Insert password shorter than 8 characters 2.Click register	Error "The password must be at least 8 characters" appears	Error "The password must be at least 8 characters" appears	PASS

A001_04	Register user	Register with non-matching passwords		1.Insert non-matching passwords	Error "The password confirmation does not match." appears	Error "The password confirmation does not match." appears	PASS

Table 6.24: Test Result A002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
A002_01	Log In	Log in with valid reader email and password	Register user.	1.Insert email and password. 2.Click Log in	Log In and Redirect to home	Log In and Redirect to home	PASS
A002_02	Log In	Log in with valid admin email and password	Register admin account.	1.Insert email and password. 2.Click Log in	Log in and redirect to admin menu	Log in and redirect to admin menu	PASS
A002_03	Log In	Log In with non-existing email		1.Insert non-existing email 2.Click Login	Error "These credentials do not match our records" appears.	Error "These credentials do not match our records" appears.	PASS
A002_04	Log in	Log in with wrong password		1.Insert valid email 2.Insert wrong password 3.Click Login	Error "These credentials do not match our records" appears	Error "These credentials do not match our records" appears	PASS

Table 6.25: Test Result B001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
B001_01	Record new book	Leave all input blanks		1.Leave all input blanks 2.Click "Save"	Alert message of field is required appears	Alert message of field is required appears	PASS
B001_02	Record new book	Insert invalid ISBN		1.Insert ISBN shorter than 13 digits 2.Insert all inputs 3.Click "Save"	Error "The isbn must be 13 digits" appears.	Error "The isbn must be 13 digits" appears.	PASS
B001_03	Record new book	Insert invalid ISBN		1.Insert ISBN with letters 2.Insert all inputs 3.Click "Save"	Error "The isbn must be an integer" appears.	Error "The isbn must be an integer" appears.	PASS
B001_04	Record new book	Insert valid inputs		1.Insert valid inputs 2.Click "save"	Book is added	Book is added	PASS

Table 6.26: Test Result C001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
C001_01	Add a book review	Leave review blank		1.Leave all input blanks 2.Click "Add Review"	Alert message of field is required appears.	Alert message of field is required appears.	PASS

C001_02	Add a book review	Insert short review		1.Insert review shorter than 20 characters. 2.Click "Add Review"	Error "The review must be at least 20 characters" appears.	Error "The review must be at least 20 characters" appears.	PASS
C001_03	Add a book review	Insert review for a book		1.Insert review 2.Click "add review"	Review added	Review added	PASS
C001_04	Add a book review	Insert review for same book	1.Add a review for the same book	1.Insert review 2.Click "Add Review"	Review is updated	Review is updated	PASS

Table 6.27: Test Result C002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
C002_01	Add a book review	Report a review		1.Click "report" 2.Select reason 3.Submit report"	Report is submitted	Report is submitted	PASS
C002_02	Add a book review	Ignore report (Admin)	1. Make a report	1.Click 'ignore'	Report is deleted.	Report is deleted.	PASS
C002_03	Add a book review	Remove review from report (Admin)	1.Make a report	1.Click Remove 2.Confirms	Report and review are deleted.	Report and review are deleted.	PASS

Table 6.28: Test Result D001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
D001_01	Add a book review	Search book		1.Enter keyword. 2.Click search	Results are displayed.	Results are displayed.	PASS
D001_02	Add a book review	Advanced Search	1. Make a report	1.Select search by option. 2.Enter keyword. 3.Click "search"	Results are displayed.	Results are displayed.	PASS
D001_03	Add a book review	Advanced Search	1.Make a report	1.Click Remove 2.Confirms	Redirect to book page.	Redirect to book page.	PASS

Table 6.29: Test Result F001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
F001_01	Create poll	Insert poll (Admin)		1.Insert all inputs	Poll created	Poll created	PASS
F001_02	Close poll	Close Poll (Admin)	1. Create a poll	1.Click "Close"	Poll status updated to closed	Poll status updated to closed	PASS
F001_03	User votes	Insert Vote	1.Create a poll	1.Click an option. 2.Click "Vote"	Vote inserted	Vote inserted	PASS
F001_04	User votes	Insert vote for a same poll	1.Voted in a same poll	1.Click an option in previous poll 2.Click "vote"	Vote updated	Vote updated	PASS

Table 6.30: Test Result F002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
F002_01	Reply to a review	Insert feedback.		1.Insert feedback. 2.Click "reply"	Feedback inserted.	Feedback inserted.	PASS
F001_02	Reply to a review	Insert feedback for same review.	1. Create a poll	1.Insert feedback for same review 2.Click "reply"	Reply updated	Reply updated	PASS

Table 6.31: Test Result G001

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
G001_01	Reset password	Non-matching new password	1. Logged in	1. insert non matching passwords	Redirect with error	Redirect with error	PASS
G001_02	Reset password	Reset password successful	1. Logged in	1.Insert matching passwords	Password updated.	Password updated.	PASS

Table 6.32: Test Result G002

Test Case ID	Test Scenario	Test Case	Pre-requisite	Test Step	Expected Result	Actual Results	Status (PASS/ FAIL)
G002_01	Search book	Search by title		1.Insert title 2.click search	Display search results	Display search results	PASS
G002_02	Search book	Search by author		1.Insert author 2.click search	Display search results	Display search results	PASS
G002_03	Search book	Search by ISBN		1.Insert ISBN 2.click search	Display search results	Display search results	PASS

After the testing has been completed, based on the status of all the tests, the system does not encounter any error that may disrupt the functional requirements. All the functional requirements seem to work as intended. Both black box and white box testing has been done and the output came out as expected. Through the statement coverage testing, it is discovered that all statements in the nested if else conditions can be executed correctly without stopping at any statements.

6.6 Conclusion

In chapter 6, all the testing that has been conducted to the system throughout the testing phase has been discussed in detailed. All the test cases are being designed and executed only by the system developer. In test plan, tasks done by tester (system developer), test environment to identify the pre-condition needed for the environment setup and the outline of test schedule are documented in this chapter. As in test result, the result from executed test cases is listed out and the analysis of the result has also been explained.

In chapter 7, it will discuss about the strength and weaknesses of the system. Further improvements from the weaknesses of the system will also be discussed in the next chapter.

CHAPTER 7: CONCLUSION

7.1 Observation on Weakness and Strengths

7.1.1 Strengths

One of the strengths of this system is that this system allows user to directly search for a book by scanning a barcode of a book. By doing so, the system will redirect the user to the appropriate page for the reviews of the book. This can help book readers to quickly get to know what people are saying about a book in a situation like in a book fair or shopping at a bookstore. A user can use their smartphone camera to scan for a barcode.

Another strength of this system is that a user does not have to edit their reviews if they want to make a change. They can just resubmit a review for a book again and it will automatically update the previous review that they had left.

Next, the system also allows user to report inappropriate reviews and give a reason why would they want to report a review. An admin can later review the report and decide whether to ignore or remove the review. A review will be deleted if an admin decided the reason is appropriate.

7.1.2 Weaknesses

The weakness of this system is that it would not allow users to sort books or reviews according to rating. Some users may want to see different ratings so they can review a book based on different point of views.

Lastly, this system is developed only on a web-based application. Browser technologies are too limiting. This can be seen that not everyone is using the same browser. Browser support might be different for each user. Although web-based development and deployment remains the cheapest, fastest route to the market, it is still not enough to ensure the technology used is parallel with current and future technologies.

7.2 Propositions for Improvement

Based on the weakness of the system, the system should allow user to sort the books and reviews based on the ratings score. This can help give readers a different perspective on a book based on the vastly different score and review.

Next, while an author can give feedback to a review, the reviewer would not be notified if an author has replied to their review. So, a notification system can be added to the system to notify a user if their reviews had received replies from the authors themselves.

7.3 Project Contribution

Book Ranks is a web app that provide a way for users to share their opinions on certain books. After completing the implementation and testing, it is believed that the system has achieved its core ideas.

The system can help book readers to find what people are saying about a book quickly. A user can search for a book by its title, ISBN, or even author's names. Furthermore, the system allows for a user to scan barcodes. This will allow even a faster way for users to get to the book page. A user may use their phones to scan a barcode while they are shopping for books, this can save time since it is just a one hand operation.

7.4 Conclusion

The project has been developed by following the methodology of database lifecycle in which it can be concluded that the project is successfully in fulfilling the objectives of project from the planning phase. The project is also managed to solve all the problems stated. Besides, Book Ranks is a web app developed that achieve the functional and non-functional requirements as well as meet its user's expectation.

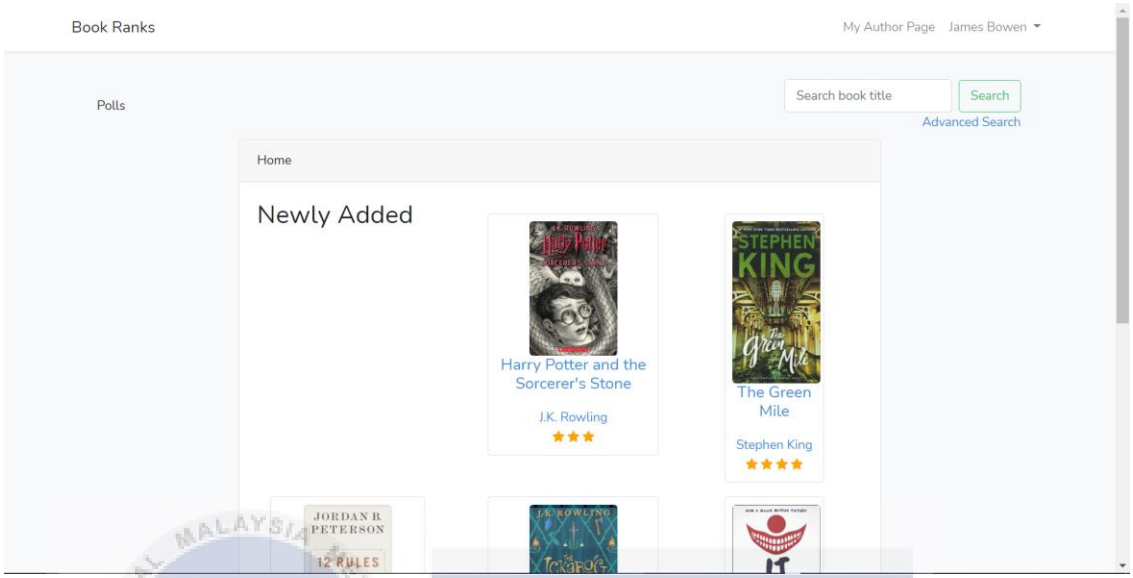
REFERENCES

1. Diamond, J. (2019, September 5). *The Problems With Goodreads*. Literary Hub. https://lithub.com/the-problems-with-goodreads/
2. Otwell, T. (2020). *Meet Laravel*. Laravel. https://laravel.com/docs/8.x#meet-laravel
3. *Composer Documentation*. (2012). Composer. https://getcomposer.org/doc/
4. Way, J. (2021, April). *Laravel 8 From Scratch*. Laracasts. https://laracasts.com/series/laravel-8-from-scratch/
5. About Goodreads. (2021). GoodReads. https://www.goodreads.com/about/usG
6. Book Marks. (2020). Literary Hub. https://lithub.com/author/book-marks/
7. Google Play Books. (2010). Google Play Books & Audiobooks APKs. https://www.apkmirror.com/apk/google-inc/google-play-books/

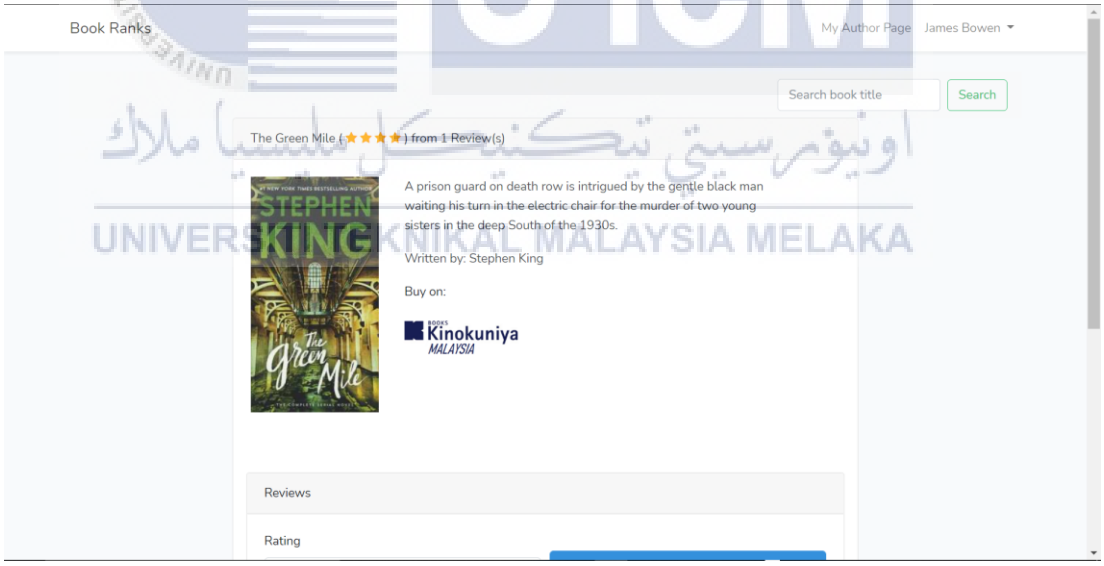
APPENDIX A

User Interfaces

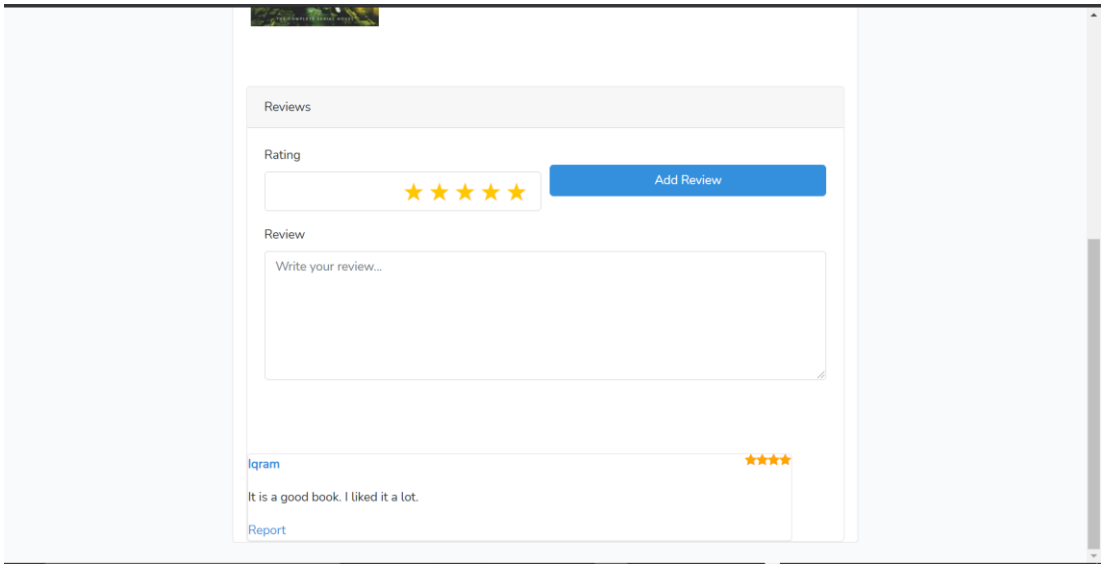
Attachment 4.1: Home Page



Attachment 4.2: Book Review Page

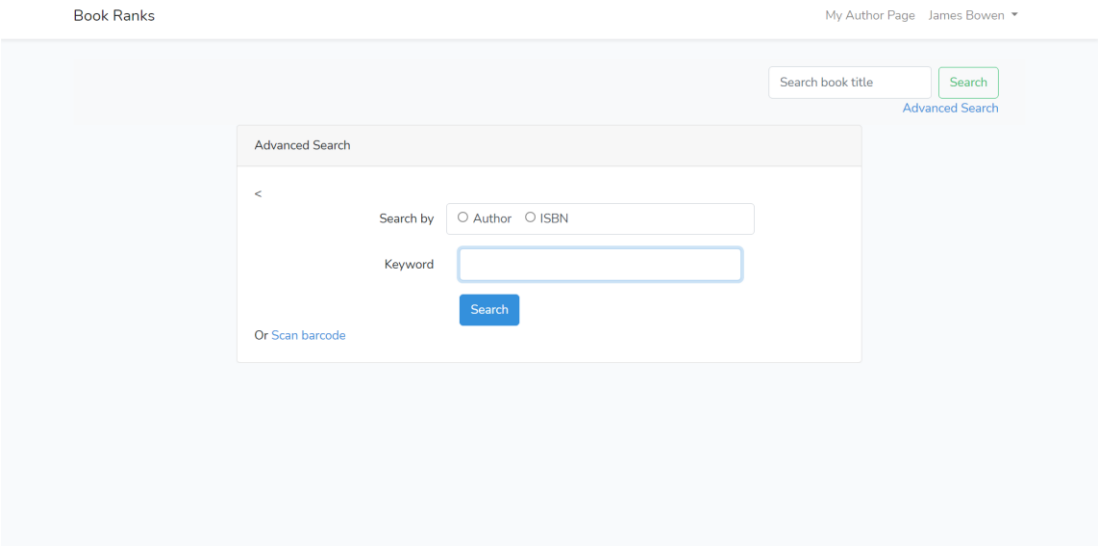


Attachment 4.3: Book Review

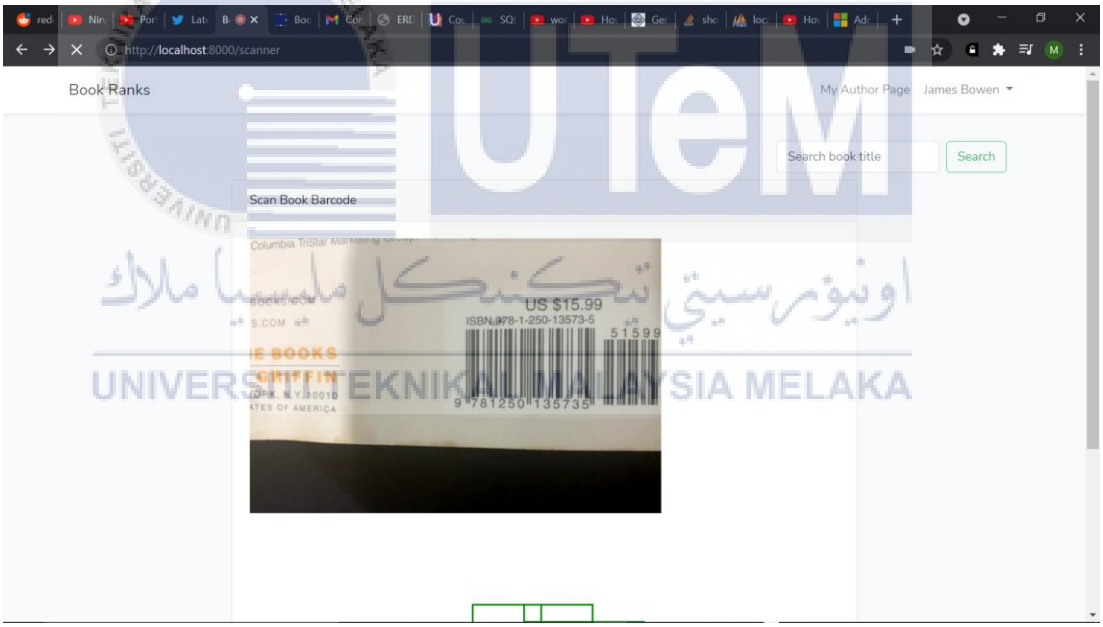


Attachment 4.4: Search Result

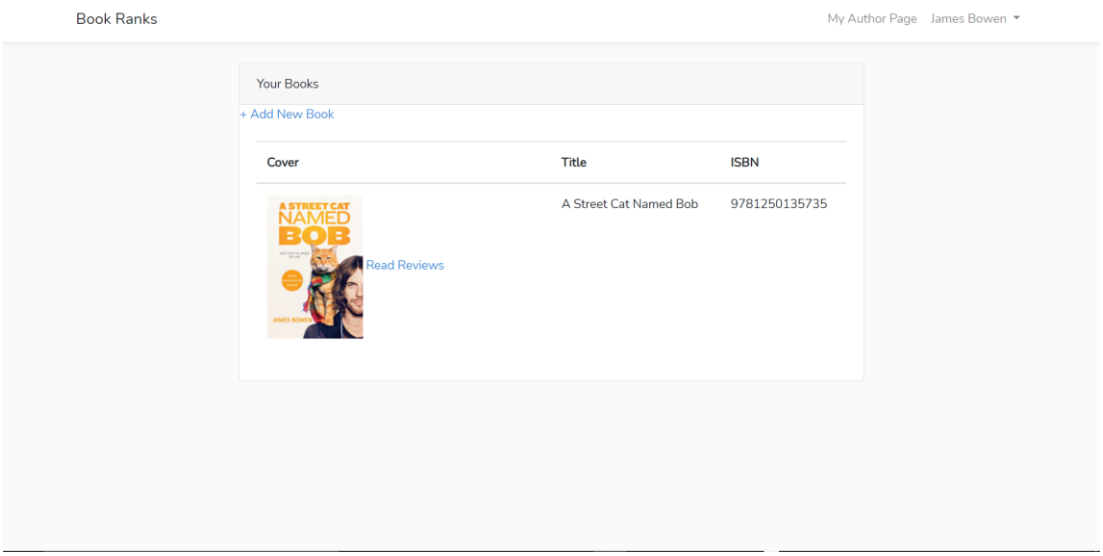
Attachment 4.5: Advanced Search Option



Attachment 4.6: Scan Barcode

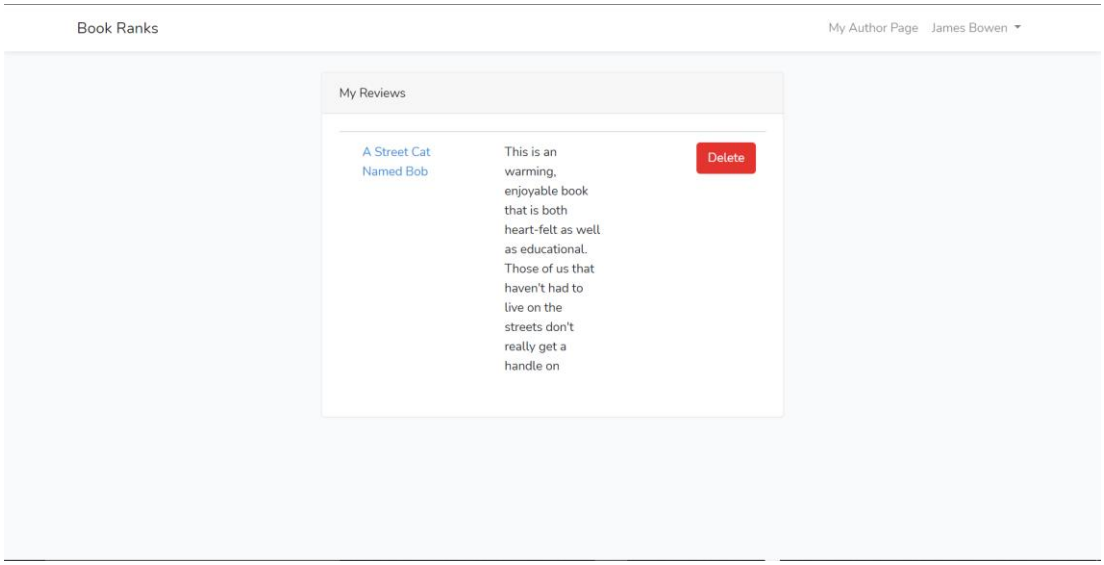


Attachment 4.7: Author Page

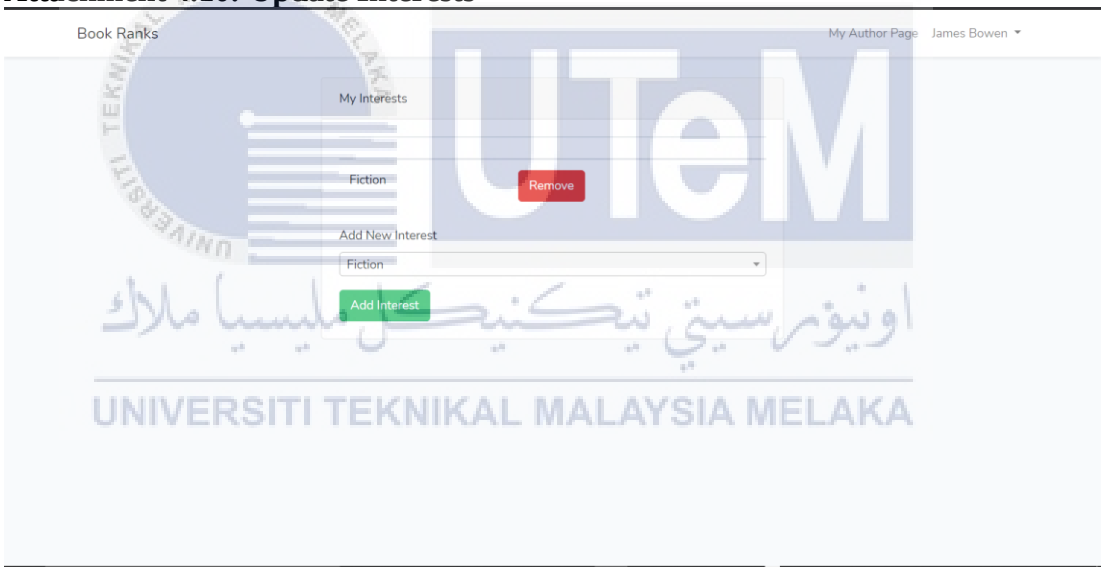


Attachment 4.8: Give Feedback

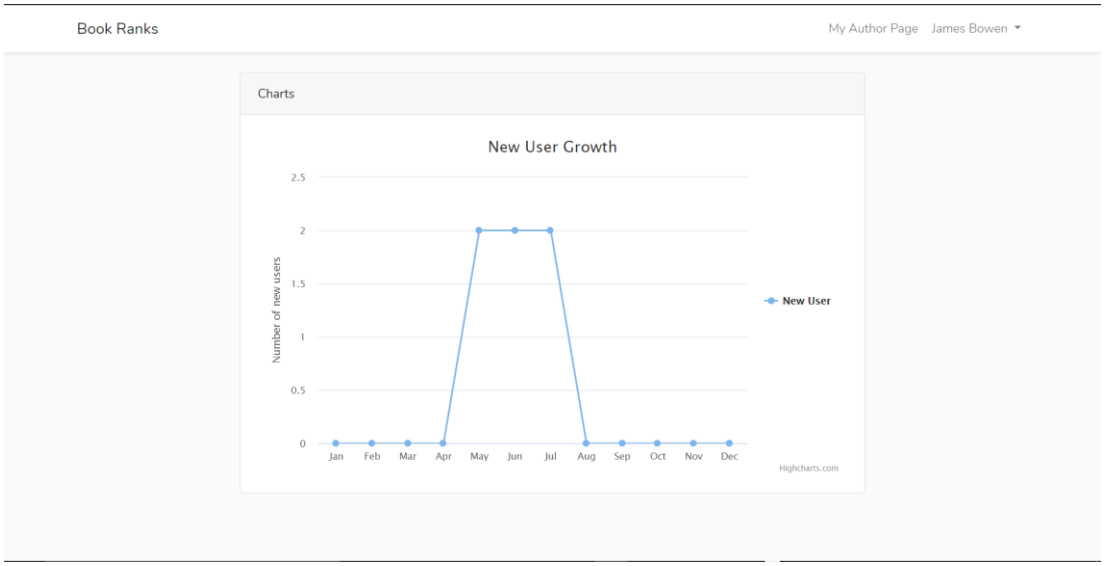
Attachment 4.9: Logged In users reviews



Attachment 4.10: Update Interests



Attachment 4.11: Chart

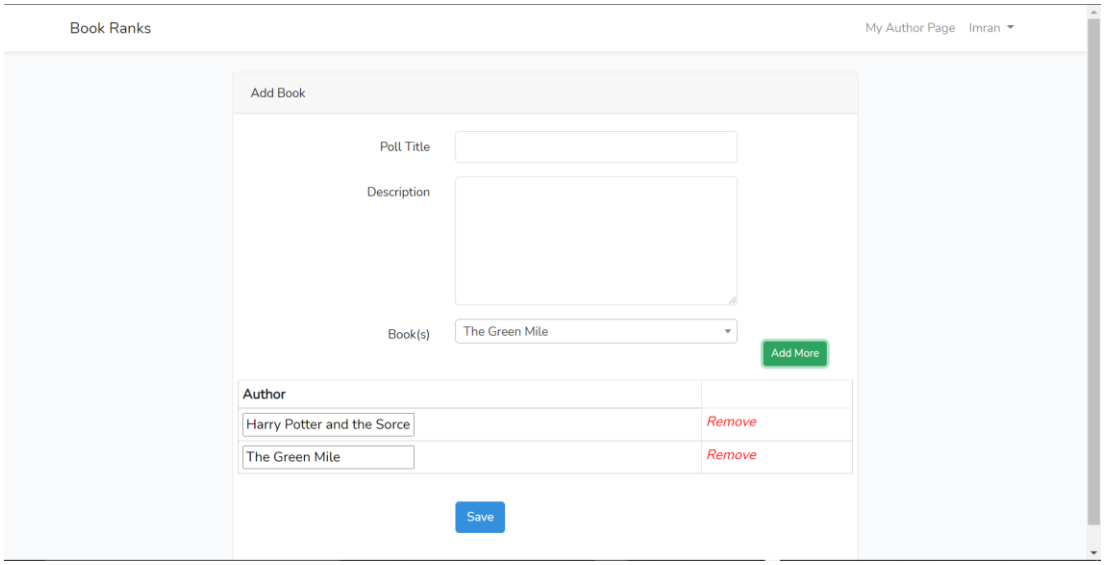


Attachment 4.12: Report Review

Attachment 4.13: Review Reports

Attachment 4.14: Manage Polls

Attachment 4.15: Create Polls



Attachment 4.16: Poll Voting

APPENDIX B

Sample Source Codes

- **Book Controller**

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Author;
use App\Models\Book;
use App\Models\User;
use App\Models\Book_Author;
use App\Models\Review;

class BookController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        //
        $books = Book::paginate(2);
        return view('admin.book.index', compact('books'));
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
```

```
     */
    public function create()
    {
        $authors = Author::all();
        return view('admin.book.create', compact('authors'));
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return \Illuminate\Http\Response
     */

    public function store(Request $request)
    {
        //$user = Auth::user();

        $request->validate([
            'title' => 'required',
            'isbn' => 'required|integer|digits:13',
            'description' => 'required',
            'image' => 'required'
        ]);


        $book = new Book();
        $book->Book_Title = $request->title;
        $book->Book_ISBN = $request->isbn;
        $book ->Book_Description = $request->description;
        $book->user_id = Auth()->id();

        //insert image
```

```
        $newImageName = time().'-'.$request->image-
>getClientOriginalName();
        $request->image->move(public_path('BookCovers'), $newImageName);

        $book->book_picture=$newImageName;

        //dd($newImageName);
        if($book->save()){
            $count = count($request->author_id);

            for ($i=0; $i < $count; $i++) {
                // $book_auth = new Book_Author();
                // $book_auth->book_id = $book->id;
                // $book_auth->author_id = $request->author_id[$i];
                // $book_auth->save();

                // $category = Category::find($request->cat);
                // $user = auth()->user();
                // $user->categories()->attach($category);

                $author = Author::find($request->author_id[$i]);
                $book->authors()->attach($author);
            }
            return redirect()->route('books.index');
            //dd('book inserted');
        }
    }

    /**
     * Display the specified resource.
     *
     * @param  string  $slug
     * @return \Illuminate\Http\Response
```

```
     */
    public function show($isbn)
    {
        $book = Book::where('Book_ISBN', $isbn)->firstOrFail();
        $reviews = Review::where('book_id', $book->id)->paginate(5);

        return view('book', compact('book', 'reviews'));
    }

    /**
     * Show the form for editing the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function edit($id)
    {
        //
    }

    /**
     * Update the specified resource in storage.
     *
     * @param  \Illuminate\Http\Request  $request
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {
        //
    }

    /**
```

```
    * Remove the specified resource from storage.
    *
    * @param  int  $id
    * @return \Illuminate\Http\Response
    */
   public function destroy($id)
   {
      //
   }
}
```

BookController is responsible for fetching, inserting, and updating book records.

- Home view

```
@extends('layouts.app')

@section('content')

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
<style>
.checked {
  color: orange;
}

.books{
      display: grid;
      grid-template-columns: auto auto auto;
      margin: auto;
   }
</style>

<div class="container">
<nav class="navbar navbar-light bg-light justify-content-between">

<a class="btn" href="{{url('vote')}}" role="button">Polls</a>
  <a class="navbar-brand"></a>
  <div class="col-md-4">
    <div class="float-right">
      <form action={{route('search') }} method="GET" class="form-inline">
```

```
            <input class="form-control mr-sm-
2" name="query" type="search" placeholder="Search book title" ari
a-label="Search">
            <button class="btn btn-outline-success my-2 my-sm-
0" type="submit">Search</button>
        </form>
    </div>
    <a href="{{route('advanced-search')}}" class="float-
right">Advanced Search</a>
  </div>
</nav>
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header">{{ __('Home') }}</div>

                <div class="card-body">
                    @if (session('status'))
                        <div class="alert alert-success" role="alert">
                            {{ session('status') }}
                        </div>
                    @endif
                    <div class="books">
                        <h2>Newly Added</h2>
                        @foreach($books as $book)
                        <div class="">
                        <a href="{{route('books.show', $book-
>Book_ISBN)}}">
                            <div class="col-md-10">
                                <div class="card mt-3">
                                    <div class="align-items-center p-2 text-
center">
                                        <img src="{{asset('BookCovers/'.$book-
>book_picture)}}" alt="" class="rounded" width="100">
                                        <h5>{{$book->Book_Title}}</h5>
                                        <div class="mt-3 info">
                                            <span class="text1 d-block">
                                                @foreach($book-
>authors as $author)
                                                    {{$author->F_Name}} {{$author-
>L_Name}}
                                                @endforeach
                                            </span>

                                            @for($i = 0; $i < $book->reviews-
>avg('Rating'); $i++)
                                            <span class="fa fa-
star checked"></span>
                                        @endfor
```

```
                                    </div>

                                </div>
                            </div>
                        </a>
                    </div>
                    @endforeach

                </div>

                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

Home View display books fetched from the controller.

- Show Poll View

```
@extends('layouts.app')

@section('content')
@if (\Session::has('success'))
    <div class="alert alert-success">
        <ul>
            <li>{!! \Session::get('success') !!}</li>
        </ul>
    </div>
@endif
<div class="container">
<nav class="navbar navbar-light bg-light justify-content-between">
  <a class="navbar-brand"></a>
  <form action={{route('search') }} method="GET" class="form-
inline">
    <input class="form-control mr-sm-
2" name="query" type="search" placeholder="Search book title" ari
a-label="Search">
    <button class="btn btn-outline-success my-2 my-sm-
0" type="submit">Search</button>
  </form>
</nav>
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header">{{$poll->poll_title}}</div>
```

```
                <div class="card-body">
                    <form action="{{ route('vote.store') }}" method="POS
T">
                        <input type="hidden" name="poll_id" value="{{$po
ll->id}}">
                        @CSRF
                        <table class="table">
                            @foreach($poll->books as $book)
                            <tr>
                                <td><input name="b_id" type="radio" value="
{{$book->id}}"/></td>
                                <td><img style='max-
width:25%;'src="{{asset('BookCovers/'.$book-
>book_picture)}}"/></td>
                                <td> {{$book->Book_Title}}</td>
                            </tr>
                            @endforeach
                        </table>
                        <div class="form-group row mb-0">
                            <div class="col-md-8 offset-md-4">
                                <button type="submit" class="btn btn-
primary">

                                    {{ __('Vote') }}
                                </button>
                            </div>
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
</div>

@endsection
```

- Book Model

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Book extends Model
{
    use HasFactory;

   public function authors(){
```

```
        return $this->belongsToMany(Author::class);
    }

    public function polls(){
     return $this->belongsToMany(Poll::class);
}

    public function reviews(){
        return $this->hasMany(Review::class);
    }
}
```

Book Model represent the Book database in Laravel Framework.

APPENDIX C

- **DDL To Create Users**

```
CREATE TABLE `users` (
 `id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
 `name` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
 `email` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
 `email_verified_at` timestamp NULL DEFAULT NULL,
 `password` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
 `remember_token` varchar(100) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
 `created_at` timestamp NULL DEFAULT NULL,
 `updated_at` timestamp NULL DEFAULT NULL,
 `role_id` bigint(20) unsigned NOT NULL DEFAULT 2,
 `proof` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
 `status` varchar(191) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
 PRIMARY KEY (`id`),
 UNIQUE KEY `users_email_unique` (`email`),
 KEY `users_role_id_foreign` (`role_id`),
 CONSTRAINT `users_role_id_foreign` FOREIGN KEY (`role_id`) REFERENCES `roles` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```

- DDL Create Roles

```
CREATE TABLE `roles` (
 `id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
 `role_name` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
 PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```

- **DDL Create Books**

```
CREATE TABLE `books` (
 `id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
 `created_at` timestamp NULL DEFAULT NULL,
 `updated_at` timestamp NULL DEFAULT NULL,
 `Book_Title` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
 `Book_ISBN` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
 `Book_Description` text COLLATE utf8mb4_unicode_ci NOT NULL,
 `user_id` bigint(20) unsigned NOT NULL,
 `book_picture` varchar(1024) COLLATE utf8mb4_unicode_ci DEFAULT
NULL,
 PRIMARY KEY (`id`),
 KEY `books_user_id_foreign` (`user_id`),
 CONSTRAINT `books_user_id_foreign` FOREIGN KEY (`user_id`)
REFERENCES `users` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=24 DEFAULT
CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```


- DDL Create Authors

```
CREATE TABLE `authors` (
 `id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
 `created_at` timestamp NULL DEFAULT NULL,
 `updated_at` timestamp NULL DEFAULT NULL,
 `F_Name` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
 `L_Name` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
 `user_id` bigint(20) unsigned DEFAULT NULL,
 PRIMARY KEY (`id`),
 KEY `authors_user_id_foreign` (`user_id`),
 CONSTRAINT `authors_user_id_foreign` FOREIGN KEY (`user_id`)
REFERENCES `users` (`id`)
```

```
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```

- DDL Create author_book

```
CREATE TABLE `author_book` (
`id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
`author_id` int(10) unsigned NOT NULL,
`book_id` int(10) unsigned NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```

- DDL Create Book_Poll

```
CREATE TABLE `book_poll` (
`id` int(10) unsigned NOT NULL AUTO_INCREMENT,
`book_id` int(10) unsigned NOT NULL,
`poll_id` int(10) unsigned NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```


- DDL Create Categories

```
CREATE TABLE `categories` (
`id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
`created_at` timestamp NULL DEFAULT NULL,
`updated_at` timestamp NULL DEFAULT NULL,
`category_name` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```

- DDL Create Category_User

```
CREATE TABLE `category_user` (
`id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
`user_id` int(10) unsigned NOT NULL,
```

```
`category_id` int(10) unsigned NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=13 DEFAULT
CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```

- DDL Create Polls

```
CREATE TABLE `polls` (
`id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
`created_at` timestamp NULL DEFAULT NULL,
`updated_at` timestamp NULL DEFAULT NULL,
`poll_title` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL,
`poll_description` text COLLATE utf8mb4_unicode_ci NOT NULL,
`status` int(11) NOT NULL DEFAULT 1,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT
CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```

- DDL Create Reviews

```
CREATE TABLE `reviews` (
`id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
`created_at` timestamp NULL DEFAULT NULL,
`updated_at` timestamp NULL DEFAULT NULL,
`Rating` tinyint(3) unsigned NOT NULL,
`Review` varchar(191) COLLATE utf8mb4_unicode_ci NOT NULL
DEFAULT '1',
`user_id` bigint(20) unsigned NOT NULL,
`book_id` bigint(20) unsigned NOT NULL,
PRIMARY KEY (`id`),
KEY `reviews_user_id_foreign` (`user_id`),
KEY `reviews_book_id_foreign` (`book_id`),
CONSTRAINT `reviews_book_id_foreign` FOREIGN KEY (`book_id`)
REFERENCES `books` (`id`),
CONSTRAINT `reviews_user_id_foreign` FOREIGN KEY (`user_id`)
```

```
REFERENCES `users` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT
CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```

- DDL Create Votes

```
CREATE TABLE `votes` (
`id` bigint(20) unsigned NOT NULL AUTO_INCREMENT,
`created_at` timestamp NULL DEFAULT NULL,
`updated_at` timestamp NULL DEFAULT NULL,
`user_id` bigint(20) unsigned NOT NULL,
`book_id` bigint(20) unsigned NOT NULL,
`poll_id` bigint(20) unsigned NOT NULL,
PRIMARY KEY (`id`),
KEY `votes_user_id_foreign` (`user_id`),
KEY `votes_book_id_foreign` (`book_id`),
KEY `votes_poll_id_foreign` (`poll_id`),
CONSTRAINT `votes_book_id_foreign` FOREIGN KEY (`book_id`)
REFERENCES `books` (`id`),
CONSTRAINT `votes_poll_id_foreign` FOREIGN KEY (`poll_id`)
REFERENCES `polls` (`id`),
CONSTRAINT `votes_user_id_foreign` FOREIGN KEY (`user_id`)
REFERENCES `users` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT
CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci
```