THESIS

Prepared by: Kiss Gábor, Software Developer
Year: 2024-2025
Class: 13.D

Noszlopy Gáspár Közgazdasági Technikum

Bookstore Management System

Content

---

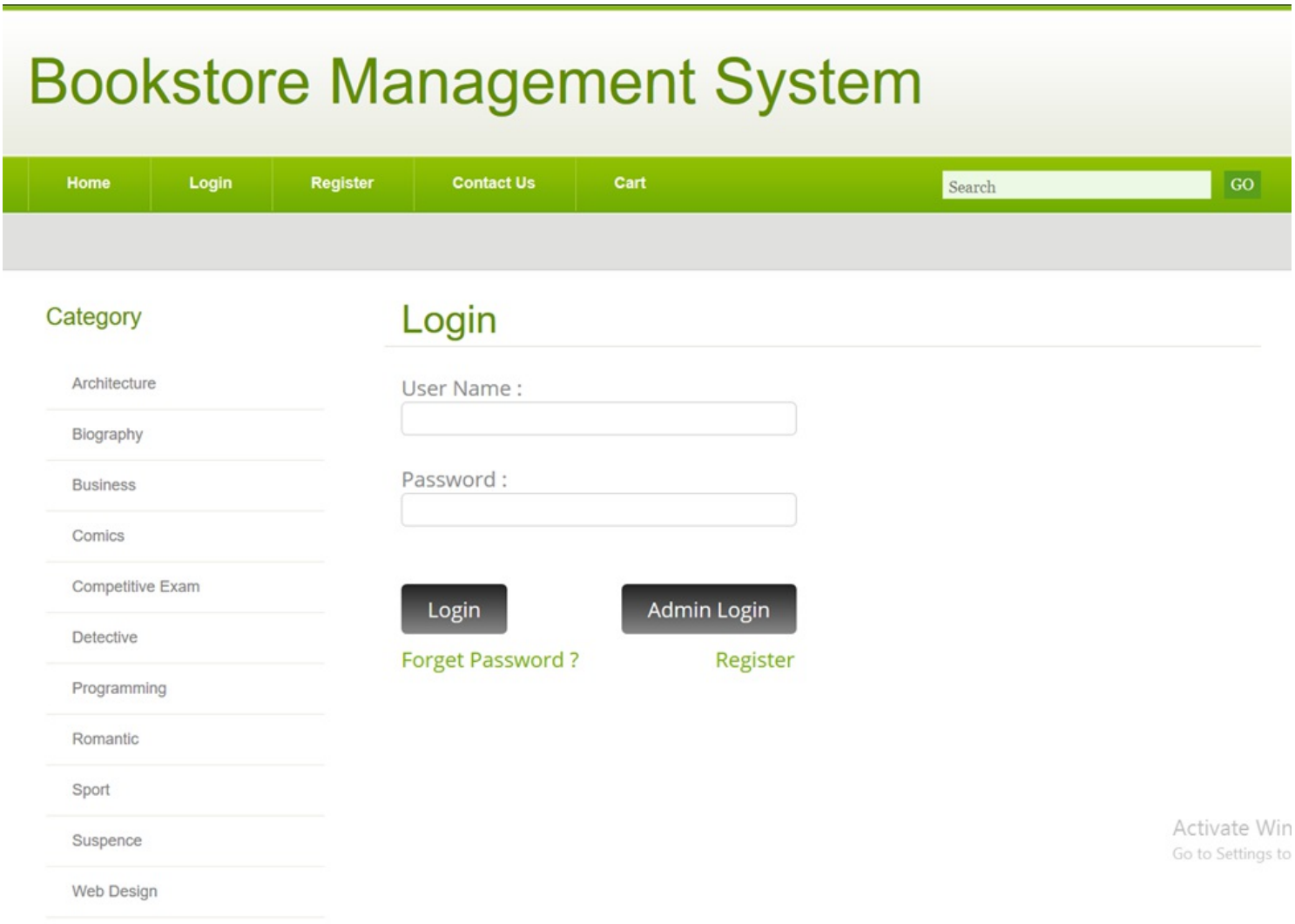
1. Introduction

1.1 Choosing the Thesis Topic

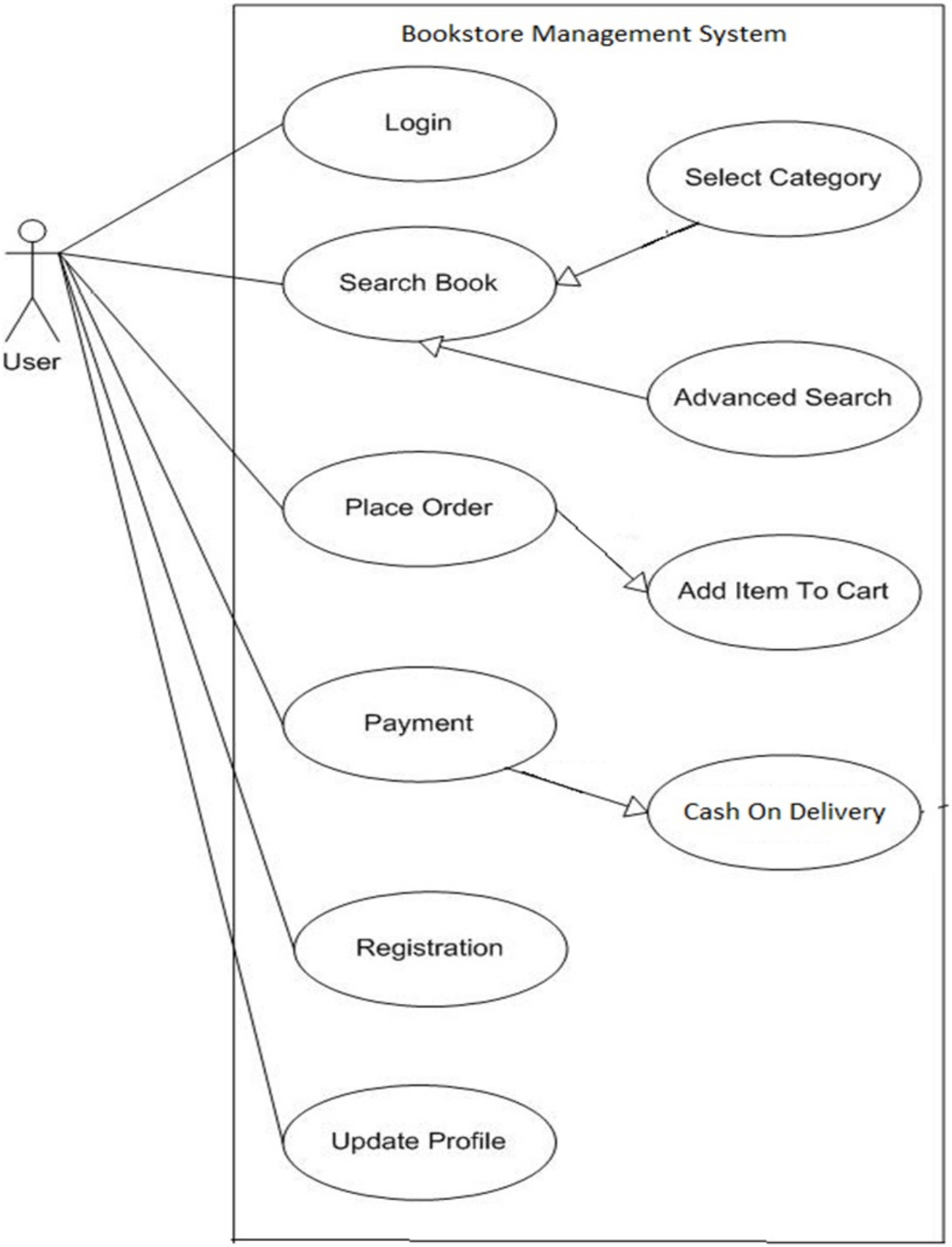
The Bookstore Management System is a web-based application designed to facilitate book trading across various categories, such as biographies, programming, and management. The aim of the project is to reduce the paperwork involved in administration and digitize the purchasing processes. The reason for choosing this topic is to provide bookstores with a simpler, centralized database solution for managing stock and customer data.

1.2 Application Design

After deciding to focus on designing a bookstore management system, I created initial mockups for the application. The first step was developing a basic design that aimed to enhance user experience and make features more accessible.



User use case diagram:



1.3 Choosing the Programming Language, Preparations

Once I decided to create an application, I faced the problem of choosing from the many platforms available for app

development. It took time to review a few of them, but I eventually decided on using 'Visual Studio Code' with HTML, CSS, Java, and Bootstrap.

1.4 Database

To create and manage the database, I used the XAMPP program, which fully met all my needs.

Of course, I also had a plan for the database.

The various tables used in the system are as follows:

1. Admin
2. Book
3. Category
4. Contact
5. Register
6. Order

1. Table Name: Admin

- **Primary Key**: `a_id`
- **Description**: Stores Admin login details.

Field	Type	Description
a_id	int(4)	Stores Admin ID
a_unm	varchar(3)	Stores Admin Username
a_pwd	varchar(30)	Stores Admin Password

2. Table Name: Book

- **Primary Key**: `b_id`
- **Description**: Stores Book details.

Field	Type	Description
b_id	int(10)	Stores Book ID
b_nm	varchar(50)	Stores Book Name
b_cat	int(6)	Stores Book Category ID
b_desc	longtext	Stores Book Description
b_price	int(4)	Stores Book Price
b_img	varchar(50)	Stores Book Image Name
b_time	int(20)	Stores the Time of Book Insertion

3. Table Name: Category

- **Primary Key**: `cat_id`
- **Description**: Stores Category Names.

Field	Type	Description
cat_id	int(10)	Stores Category ID
cat_nm	varchar(50)	Stores Category Name

4. Table Name: Contact

- **Primary Key**: `c_id`
- **Description**: Stores Contact Us details.

Field	Type	Description
c_id	int(4)	Stores Contact ID of Client/User
c_fnm	varchar(100)	Stores User Full Name
c_mno	int(10)	Stores Client/User Mobile Number
c_email	varchar(60)	Stores Client/User E-Mail Address
c_msg	longtext	Stores Client/User Message or Query
c_time	varchar(20)	Stores the Time of Contact Page Data Insertion

5. Table Name: Register

- **Primary Key**: `r_id`
- **Description**: Stores Registration details of Visitors/Users.

Field	Type	Description
r_id	int(8)	Stores User Registration ID
r_fnm	varchar(100)	Stores User Full Name
r_unm	varchar(50)	Stores Username
r_pwd	varchar(30)	Stores User Password
r_cno	varchar(10)	Stores User Contact Number
r_email	varchar(60)	Stores User E-Mail Address
r_time	varchar(20)	Stores User Registration Time

6. Table Name: Order

- **Primary Key**: `o_id`
- **Description**: Stores Order details.

Field	Type	Description
o_id	int(11)	Stores Order ID
o_name	varchar(30)	Stores User Full Name
o_address	varchar(200)	Stores User Address
o_pincode	int(20)	Stores User City Pincode
o_city	varchar(30)	Stores User City Name
o_state	varchar(30)	Stores User State
o_mobile	bigint(20)	Stores User Mobile Number
o_rid	int(8)	Stores User Registration ID

1.5 Acknowledgements

Even though we spent little time in school this year, I received a lot of help and support from Mr. Kovács László, both practically and theoretically.

2. User Documentation

2.1 System Requirements

Hardware Requirements

- System type: 32-bit operating system.
- Windows 7/8/8.1/10
- Linux Ubuntu / Light Ubuntu
- Mac OS
- 350 MB RAM

Software Requirements

- WAMP Server
- MySQL
- Browser
- PHPMyAdmin

Client-side Tools

- Processor: PC with Dual-core or higher processor recommended: 2.20 GHz processor.
- RAM: 512 MB or more recommended.
- Hard Disk: 45 MB of free space on the system drive, or more.

- Operating System: Windows or open-source 32/64-bit operating system, or newer versions. Browsers: Mozilla Firefox 2.0 / Internet Explorer 8.0 or newer / Google Chrome.

2.3 Application Usage

2.4 Frequently Asked Questions

- Will it be available in various online stores?
    - Yes, it's accessible through all browsers on the system.
- How do I contact support?
    - The developer contact information is found at the end of this document.
- What is the return policy for digital products?
    - We provide a 7-day return policy.

2.5 Developer Contact Information

- **Kiss Gábor**
- Email: kissgabor@school.com
- Phone: +36304559752

3. Developer Documentation

3.1 Database

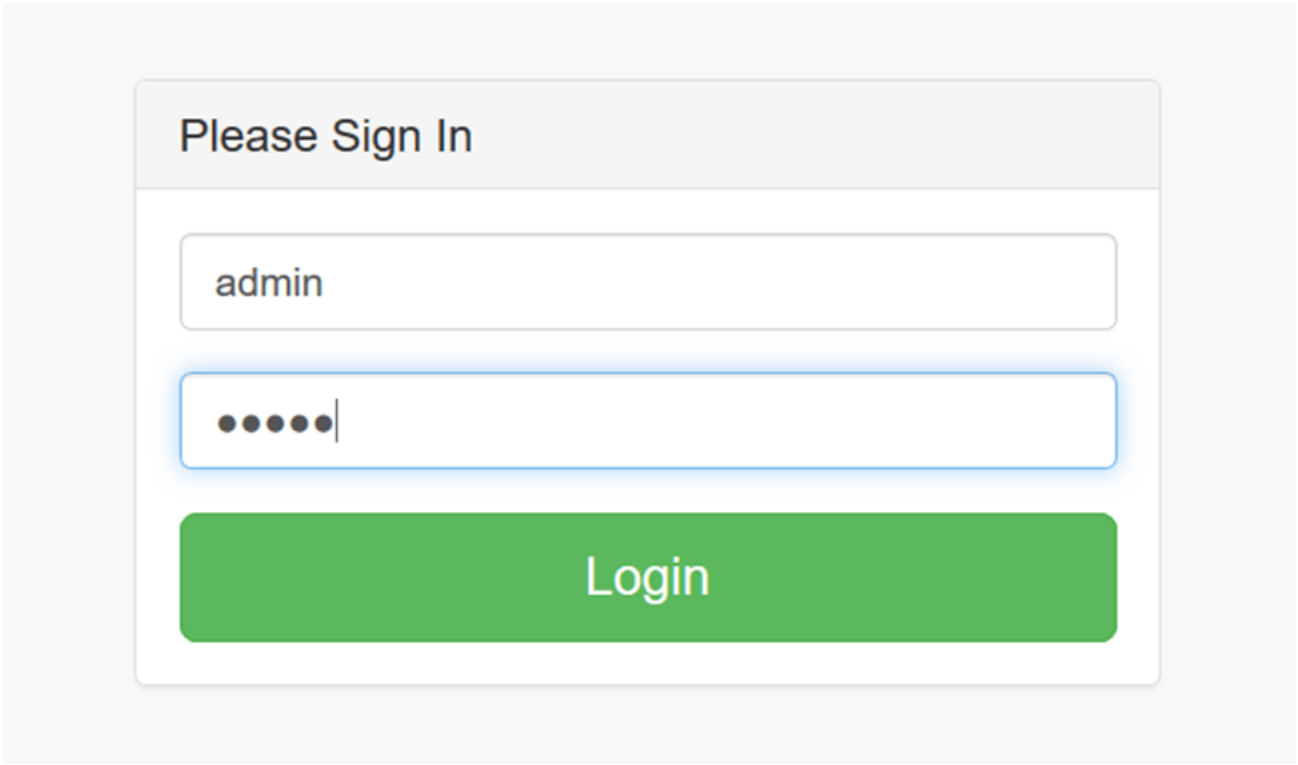
The central database stores user and book data. The database tables include: admin, book, category, customer, and order.

3.6 Admin Interface

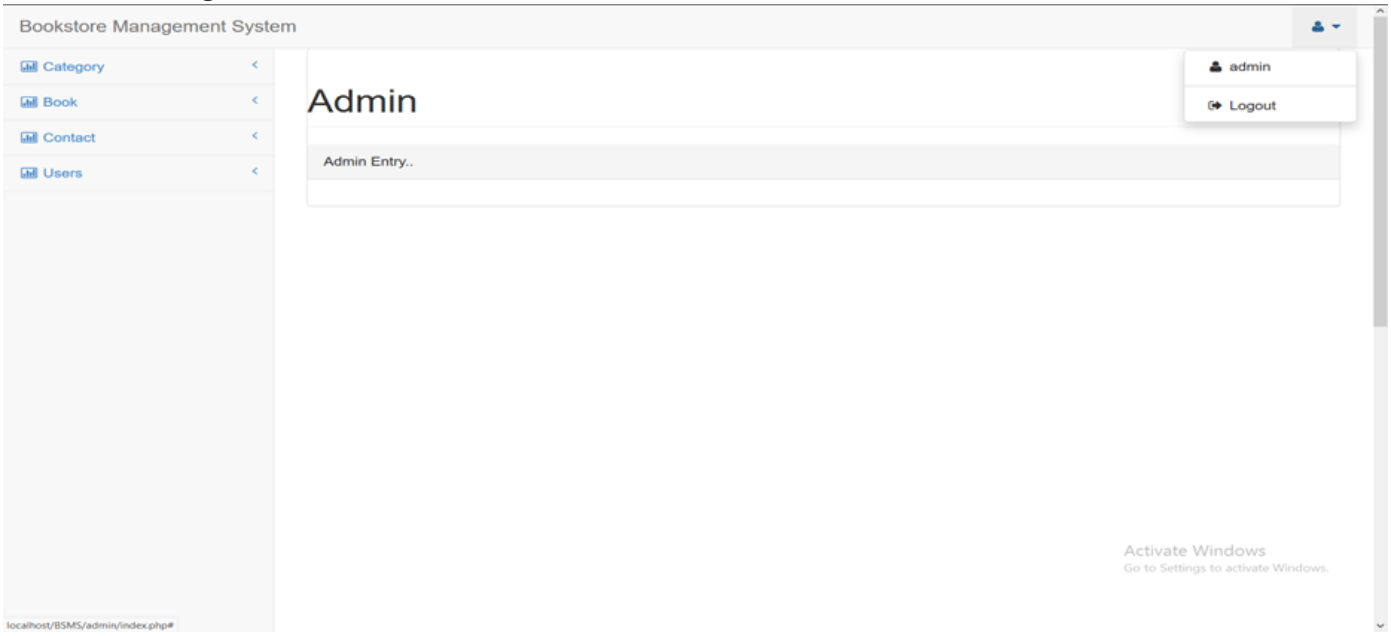
Admin Functions:

- This module primarily includes the following tasks:
    - Add a category.
    - Category list.
    - Add a new book.
    - View books.
    - View messages sent by customers.

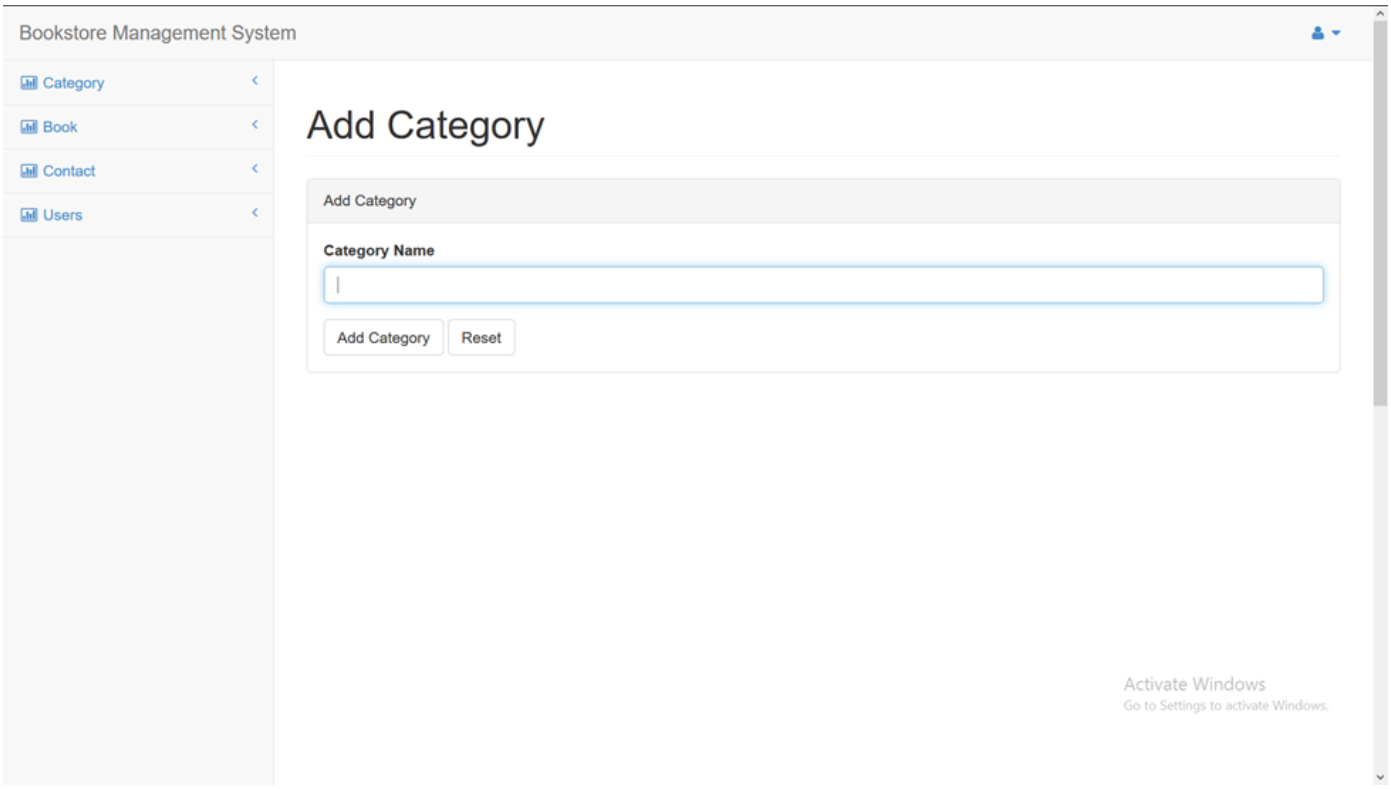
13. Admin Login Page (New Template)



14. Admin Home Page



15. Add Category (Admin)



3.7 Development Opportunities

Help Module

With this module, users can receive help to access the system. It provides descriptions of all the system's functionalities. Users can easily access the entire module with this feature.

Online Payment Module

This feature allows users to pay online. In the future, online payments will be implemented to make transactions easier for users.

Multilingual Support

The system will include multilingual support, allowing users to work in different languages for easier comprehension.

3.8 Challenges Faced

I encountered many challenges. Primarily, it was difficult to start learning a new programming language while attending school, which is evident in my code as I used comments to assist understanding in various places. As a result, the login menu became quite confusing.

After exploring my options, I encountered another obstacle: I couldn't connect the application to the database. Although there were many solutions available online, they varied greatly as the problem is not the same for everyone.

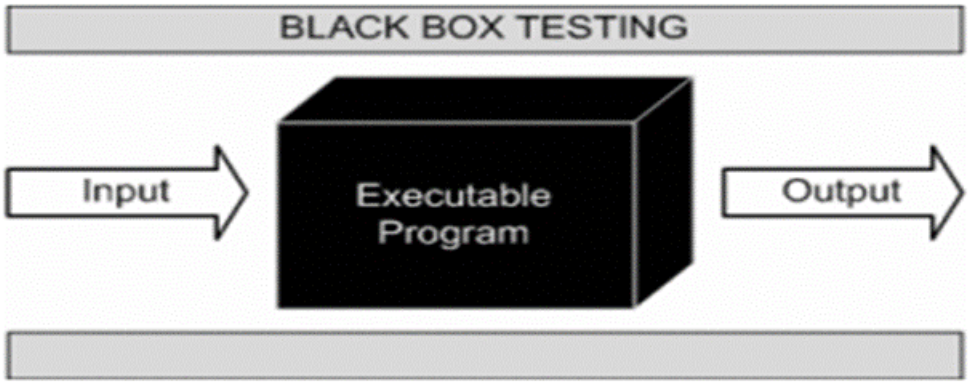
Initially, I didn't consider using PHP for integrating the application. However, my research eventually led me to realize that this was the most convenient solution for me.

3.9 Testing Documentation

Black Box Testing

Black box testing is a software testing method that examines the functionality of an application based on specifications. Also known as specification-based testing, it is conducted by independent testing teams during the software testing lifecycle.

This method can be applied at all levels of software testing, such as unit, integration, system, and acceptance testing.



The term "black box" is used because the software is viewed as a "black box" from the tester's perspective, meaning its internal workings are not visible. This method aims to identify the following types of errors:

- Missing or incorrect functions
- Interface errors
- Errors in data structures or external database access
- Behavioral or performance issues
- Initialization and termination errors

Advantages

- Testing is done from a user's perspective and helps identify deviations from specifications.
- Testers do not need to know programming languages or how the software is implemented.

White Box Testing

White box testing is a technique that examines the structure of a program and derives test data based on the logic/code of the program. It is also known as glass box testing, clear box testing, open box testing, logic-driven testing, or structure testing.

White box testing looks at the code structure. Knowing the product's internal structure ensures that internal operations are performed according to specifications and that all internal components are tested adequately.

White Box Testing Techniques

- **Statement coverage**: This technique aims to test all programming statements with a minimal set of tests.
- **Branch coverage**: This involves running a series of tests to ensure that all branches are tested at least once.
- **Path coverage**: This ensures that all possible paths in the code are tested, meaning all statements and branches

are covered.

Advantages

1. Forces the developer to thoroughly consider the implementation.
2. Helps uncover "hidden" code errors.
3. Supports best coding practices or other issues with the code.

Gray Box Testing

Gray box testing is a technique where the tester has limited knowledge of the system's internal workings. Gray box testers have access to detailed design information or requirements.

Gray box testing is based on state-based models, UML diagrams, or other documentation. This technique tests a software product or application with partial knowledge of its internal functioning.

Admin Login Details

- **Username:** Admin
- **Password:** Admin

Expected Result:

- If fields are empty, an error message prompts the user to fill in the fields.
- If the username or password does not exist, an error message prompts the user to input correct credentials.

Login Details

- **Username:** Dhaval
- **Password:** Dhaval

Expected Result:

- If fields are empty, an error message prompts the user to fill in the fields.
- If the username or password does not exist, an error message prompts the user to input correct credentials.

Registration Details

- **Username:** EMPTY
- **Password:** EMPTY
- **Full Name:** EMPTY
- **Security Question Answer:** EMPTY

Expected Result:

- If fields are empty, an error message prompts the user to fill in the fields.
- If the username or password does not exist, an error message prompts the user to input correct credentials.
- If the password is less than 8 characters, an error message appears.

Order Details

- **Full Name:**
- **Address:**
- **Contact Number:** EMPTY

Expected Result:

- If fields are empty, an error message prompts the user to fill in the fields.
- If the contact number is not numeric, an error message appears.

4 References

- Color selection: **https://www.w3schools.com/colors/colors_picker.asp**
- PHP commands: **https://www.php.net/manual/en/index.php**
- Java help: **https://www.w3schools.com/java/**
- Java reading: Application Development in Android Studio