A
PROJECT REPORT
ON

Library Management System

By

SHINGALA SHUBHAM P. (CE-146) (19CEUOS159)
UNAGAR KEVAL V. (CE-167) (19CEUBG010)

B.Tech CE Semester - VI
Subject : SOC

Guided by:

Prof. Prashant M. Jadav

Prof. Ankit P. Vaishnav

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

CERTIFICATE

This is to certify that the practical / term work carried out in subject Of **SOC** and recorded in this journal is the Bonafede work of

SHINGALA SHUBHAM P. (CE 146) (19CEUOS159)

UNAGAR KEVAL V. (CE 167) (19CEUBG010)

Of B. Tech semester **VI** in the branch of **Computer Engineering** During the academic year **2021-22.**

Prof. Prashant M. Jadav
Associate Professor

Prof. Ankit P. Vaishnav
Assistant Professor

Dept. of Computer Eng.,
Faculty of Technology
Dharmsinh Desai University, Nadiad

Dr. C. K. Bhensdadia,
Head,
Dept. of Computer Eng.,
Faculty of Technology
Dharmsinh Desai University, Nadiad

Contents:

1. Introduction

Library Management System is a dynamic web application which can be used by the Library Owner's & their Staff for storing all the data related to books and users digitally in the database & update books related details on a regular basis like no. of copies available, no. of borrowed copies, etc.

The First Module is Registration. In this module all the People that are working at the library can register in the website. To register as a user, it requires details like Id, name, email, password, DOB, etc. After Registration the next module is Login. In this module User can login to the website using id & password. After Login, the next module is Books. In this module user can add any new books (first time introduced), by providing all the required details such as name of the book, Author, Edition, No of Total Copies, No of Available Copies, Category, etc. User can also update & delete the existing books data from the library database. So, the main objective of this project is at any point of time user of library can fetch all the data related to books.

➢ Tools/Technologies Used

❖ Technologies:

- .Net Framework
- WCF
- HTML
- CSS
- Bootstrap

❖ Tools

- Visual Studio
- Git

❖ Platform

- Local development server

2. Software Requirement Specifications

➢ Functional Requirements:

1. Login / Registration

R.1.1: Create New Account

Description :- If user does not have an account on website then he/she has to create an Account.

Input:- Enter details like name, id, password, etc... to create an account.

Output:- Display generated id as a message.

R.1.2: Login

Description :- User can access this website by login using id and password.

Input:- Enter id and password.

Output:- If id and password are correct then all functionalities will be display, otherwise message will be display as "**Invalid credentials**".

2. Books

R.2.1: Add Book

Description :- User can add new book in the system.

Input:- Enter details like name of the book, author, category, edition, no. of available copies, no. of total copies, etc.

Output:- Display success message and book add in library.

R.2.2: Update Book

Description :- User can Update existing book in the system.

Input:- Choose id of the book & Update the necessary details.

Output:- Display success message and book update in library.

R.2.3: Get Book Data

Description :- User can see all the available data for the existing book in the system.

Input:- Choose id of the book.

Output:- All the book related data will be shown.

R.2.4: Delete Book

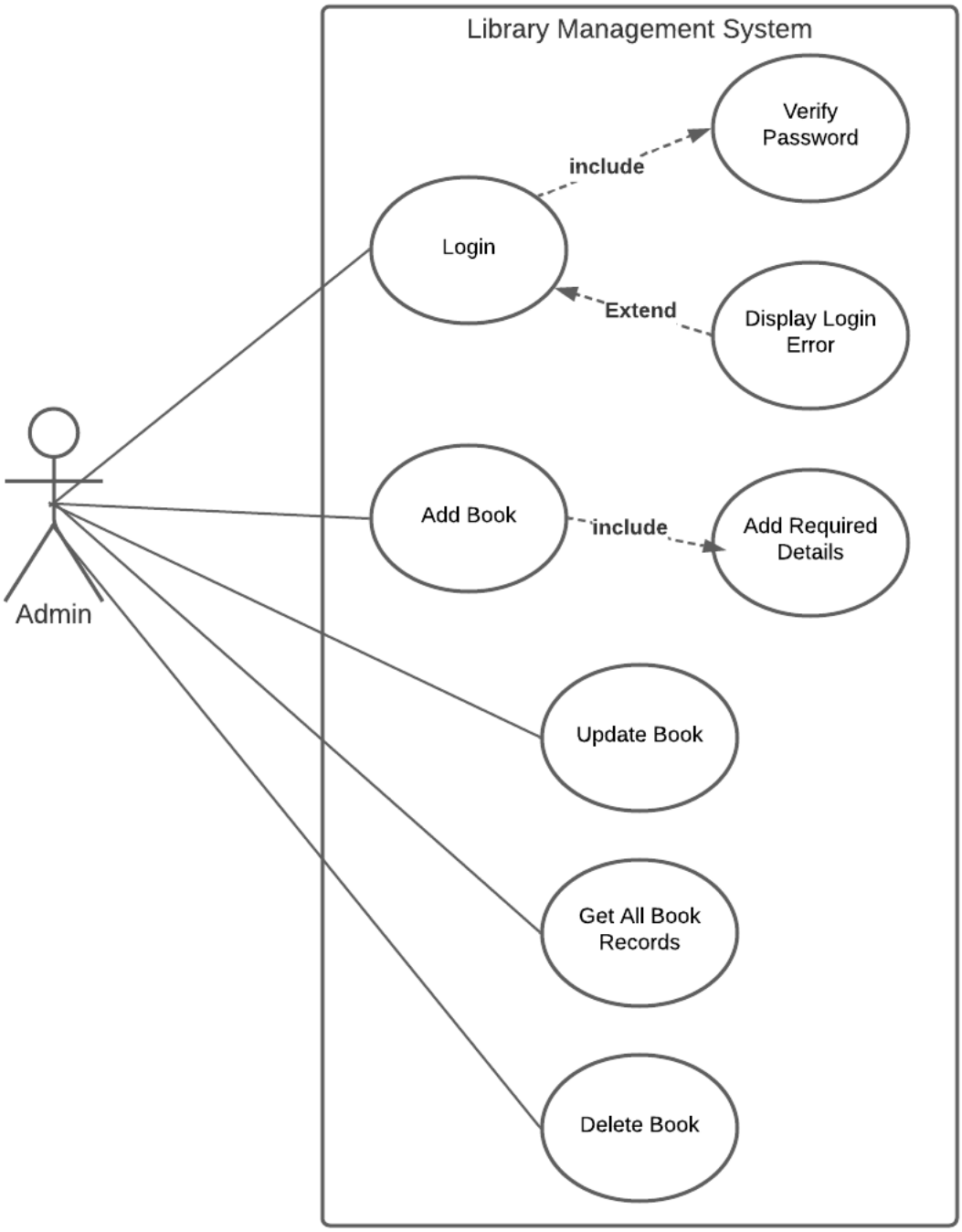
Description :- User can delete existing book in the system.

Input:- Choose id of the book.

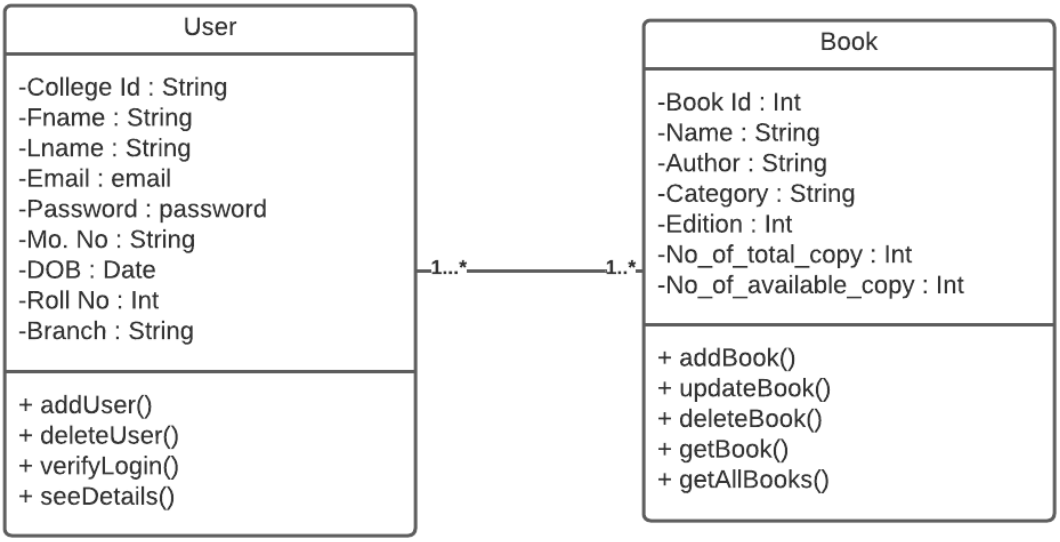
Output:- Display success message and book delete in library.

3. Design

3.1 Use Case Diagram



3.2 Class Diagram



3.3 Data Dictionary

dbo.Students [Data]

Max Rows: 1000

Id	CollegeId	FirstName	LastName	EmailId	DOB	MobileNum	Password	Salt	RollNo	Branch
1	2022ComputerS...	Shubham	Shingala	abc@abc.com	09/06/2009 00:...	2356483164	e276a94536e30...	dgsje	146	Computer
2	2022Computer...	DDU	Nadiad	ddu@ddu.ac.in	03/05/1999 00:...	9452365472	e276a94536e30...	dgsje	112	Computer
3	2022Computer6...	Keval	Unagar	keval@gmail.co...	21/01/2015 00:...	9632154786	e276a94536e30...	dgsje	167	Computer
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

dbo.Books [Data]

Max Rows: 1000

Id	Name	Author	Category	Edition	No_of_Total_C...	No_of_Availabl...
1	Wings of Fire	A P J Abdul Kal...	autobiography	1	50	50
3	Deep Learning	Yoshua Bengio	Computer	5	60	54
4	Mathematics fo...	Marc Peter Dei...	Math	4	45	45
NULL	NULL	NULL	NULL	NULL	NULL	NULL

4. Implementation Details:

4.1 Description of Modules: -

- The system consists of 2 basic modules namely,

  1) User Module

  2) Book Module

- Each module consists of several methods to implement the required functionality.
- Implementation is done using .net framework using C# and services. Database used in these modules is MySQL.

1. User Module :

- This module is the base for authentication and authorization to ensure the security aspect of the user. It also includes profile creation, login and logout using MySQL database.
- **Create account** : Users can create their account and login to the system.

2. Book Module :

- This module is for the adding new books, updating books by providing required details related to books such as id, name, author, edition, category, no. of copies etc.

- **Add Books :** User can add a new book in the system.

- **View Books :** User can view all the books available in the system.

- **Update Books :** User can update the existing book in the system.

- **Delete Books :** User can delete the existing book in the system.

4.2 Major function prototype :-

1. User service:

```
public class StudentService : IStudentService
{
    public StudentDto getUser(int rollno)
    {
        try
        {
            AppDbContext context = new AppDbContext();
            Student student = context.Students.Where(u => u.RollNo==rollno).FirstOrDefault();
            StudentDto studentDto = new StudentDto
            {
                Id = student.Id,
                CollegeId = student.CollegeId,
                FirstName = student.FirstName,
                LastName = student.LastName
            };

            return studentDto;
        }
        catch (Exception e)
        {
            Console.WriteLine(e.Message);
            throw e;
        }
    }

    public StudentDto GetUserByToken(string token)
    {
        try
        {
            AppDbContext context = new AppDbContext();
            DatabaseTokenValidator tokenValidator = new DatabaseTokenValidator(context);
            if (tokenValidator.IsValid(token))
            {
                Student student = context.Tokens.Where(t => t.SecureToken ==
token).FirstOrDefault().Student;
                StudentDto studentDto = new StudentDto
                {
                    Id = student.Id,
                    CollegeId = student.CollegeId,
                    FirstName = student.FirstName,
                    LastName = student.LastName
                };
                return studentDto;
            }
            throw new InvalidCredentialException("Invalid token");
        }
        catch (Exception e)
        {
            throw e;
        }
    }

    public bool isValidToken(string token)
    {
        AppDbContext dbContext = new AppDbContext();
        DatabaseTokenValidator tokenValidator = new DatabaseTokenValidator(dbContext);
        return tokenValidator.IsValid(token);
```

```
        }

        public string Login(Credentials creds)
        {
            try
            {
                AppDbContext context = new AppDbContext();
                ICredentialsValidator validator = new DatabaseCredentialsValidator(context);
                if (validator.IsValid(creds))
                {
                    return new DatabaseTokenBuilder(context).Build(creds);
                }
                throw new InvalidCredentialException("Invalid credentials");
            }
            catch (Exception e)
            {
                throw e;
            }

        }
        private static Random random = new Random();

        public static string RandomString(int length)
        {
            const string chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";
            return new string(Enumerable.Repeat(chars, length)
                .Select(s => s[random.Next(s.Length)]).ToArray());
        }
        public string Register(Student student)
        {
            try
            {
                AppDbContext context = new AppDbContext();
                student.Salt = "dgsje";
                student.CollegeId =
DateTime.Now.Year+student.Branch+RandomString(3)+student.RollNo;
                student.Password = Hash.Get(student.Password + student.Salt,
Hash.DefaultHashType, Hash.DefaultEncoding);
                context.Students.Add(student);
                context.SaveChanges();
                return student.CollegeId;
            }
            catch (System.Data.Entity.Validation.DbEntityValidationException dbEx)
            {
                Exception raise = dbEx;
                foreach (var validationErrors in dbEx.EntityValidationErrors)
                {
                    foreach (var validationError in validationErrors.ValidationErrors)
                    {
                        string message = string.Format("{0}:{1}",
                            validationErrors.Entry.Entity.ToString(),
                            validationError.ErrorMessage);
                        // raise a new exception nesting
                        // the current instance as InnerException
                        raise = new InvalidOperationException(message, raise);
                    }
                }
                throw raise;
            }
        }
    }
```

2. Book service :

```
public class BookService : IBookService
 {
     public void AddBook(Book book)
     {
         AppDbContext dbContext = new AppDbContext();
         try
         {
             dbContext.Books.Add(book);
             dbContext.SaveChanges();
         }
         catch (System.Data.Entity.Validation.DbEntityValidationException dbEx)
         {
             Exception raise = dbEx;
             foreach (var validationErrors in dbEx.EntityValidationErrors)
             {
                 foreach (var validationError in validationErrors.ValidationErrors)
                 {
                     string message = string.Format("{0}:{1}",
                         validationErrors.Entry.Entity.ToString(),
                         validationError.ErrorMessage);
                     raise = new InvalidOperationException(message, raise);
                 }
             }
             throw raise;
         }
     }

     public void DeleteBook(int id)
     {
         AppDbContext dbContext = new AppDbContext();
         Book book = dbContext.Books.Where<Book>(i => i.Id == id).FirstOrDefault();
         if (book != null)
         {
             dbContext.Books.Remove(book);
             dbContext.SaveChanges();
         }
         else
         {
             Exception exception = new Exception("entered id of book is not found");
             throw exception;
         }
     }

     public Book GetBook(int id)
     {
         AppDbContext dbContext = new AppDbContext();
         Book book = dbContext.Books.Where<Book>(i => i.Id == id).FirstOrDefault();
         if (book != null)
         {
             return book;
         }
         else
         {
             Exception exception = new Exception("entered id of book is not found");
             throw exception;
         }
     }

     public IEnumerable<Book> GetBooks()
     {
```

```
        List<Book> books = new List<Book>();
        AppDbContext dbContext = new AppDbContext();
        books = dbContext.Books.ToList<Book>();
        return books;
    }

    public void UpdateBook(Book book)
    {
        AppDbContext dbContext = new AppDbContext();
        Book book1 = dbContext.Books.Where<Book>(i => i.Id == book.Id).FirstOrDefault();
        if (book1 != null)
        {
            book1.Name = book.Name;
            book1.Author = book.Author;
            book1.Category = book.Category;
            book1.Edition = book.Edition;
            book1.No_of_Available_Copy = book.No_of_Available_Copy;
            book1.No_of_Total_Copy = book.No_of_Total_Copy;
            dbContext.SaveChanges();
        }
        else
        {
            Exception exception = new Exception("entered book details is not found");
            throw exception;
        }
    }
    }
```

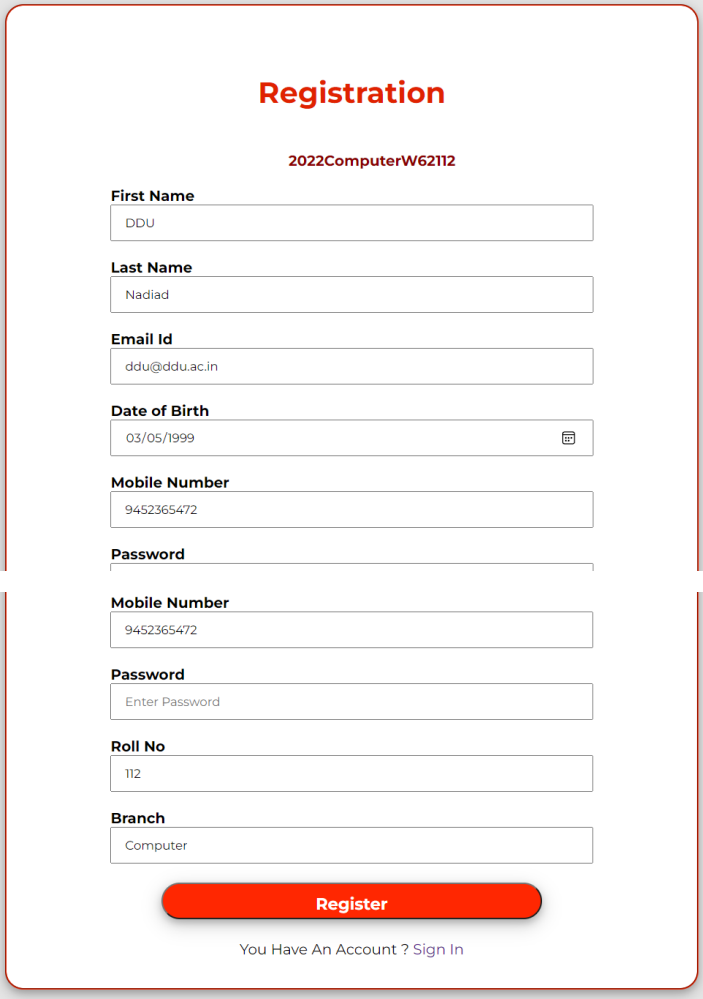
5. Testing

Manual Testing

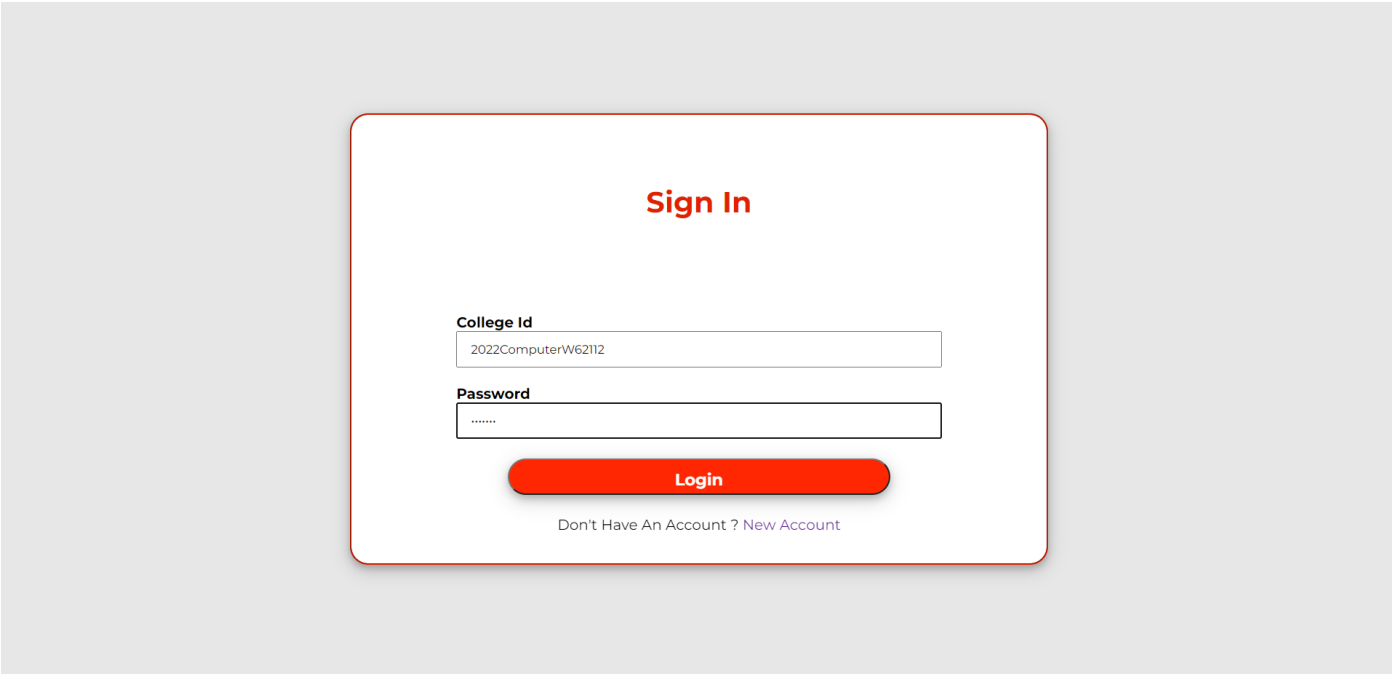
Sr. no	Test Scenario	Expected Result	Actual Result	Status
TC_01	Users Registration and log in with validations.	Users successfully login to the System Users	Successfully login to the system	Success
TC_02	Users can see all the available books in the system.	All Available Books are visible on the main page.	All Available Books are visible on the main page.	Success
TC_03	User can add the new book in the system.	User add all the required details.	User add all the required details	Success
TC_04	New Book Added Successfully.	Visible on the main page.	Visible on the main page.	Success
TC_05	Update Book	Update necessary details.	Update necessary details.	Success
TC_06	Book Updated Successfully.	Updated details shown on the main page.	Updated details shown on the main page.	Success
TC_07	Delete Book	Book deleted, not shown on the main page.	Book Deleted, not shown on the main page.	Success

6. Screenshots

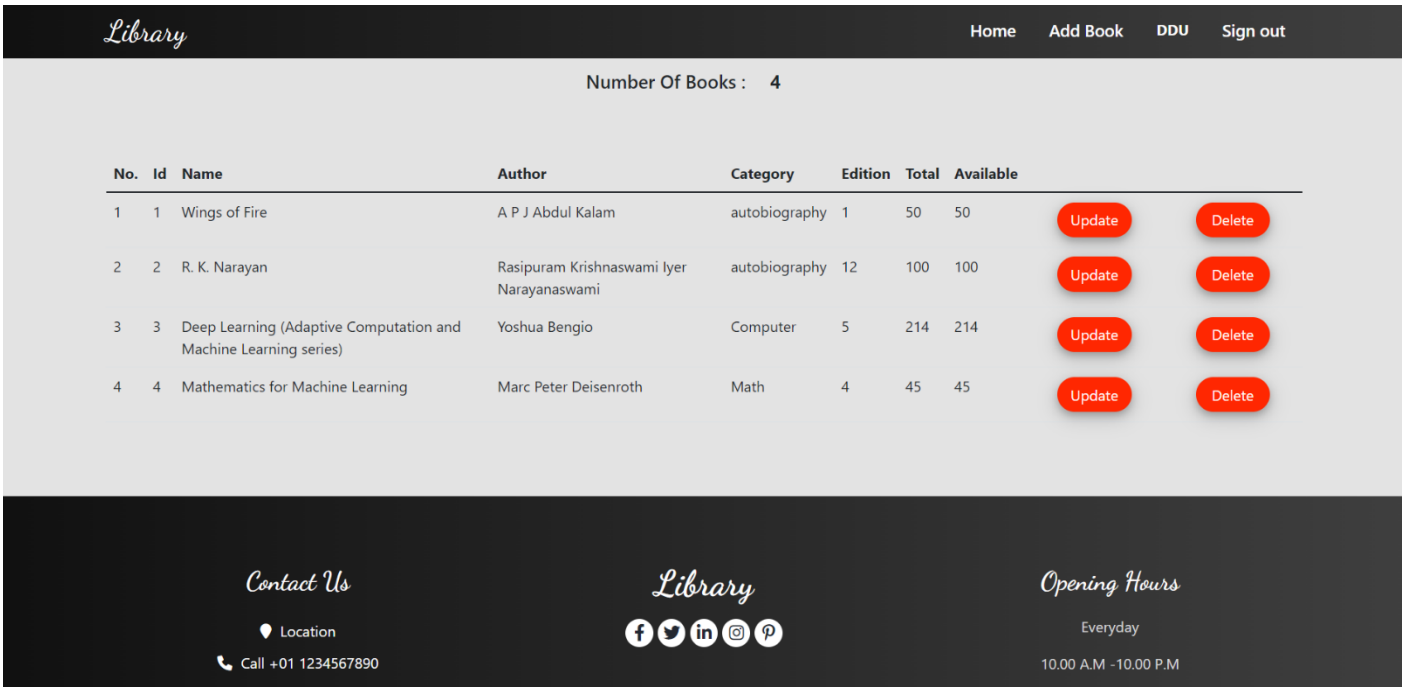
➢ Registration:



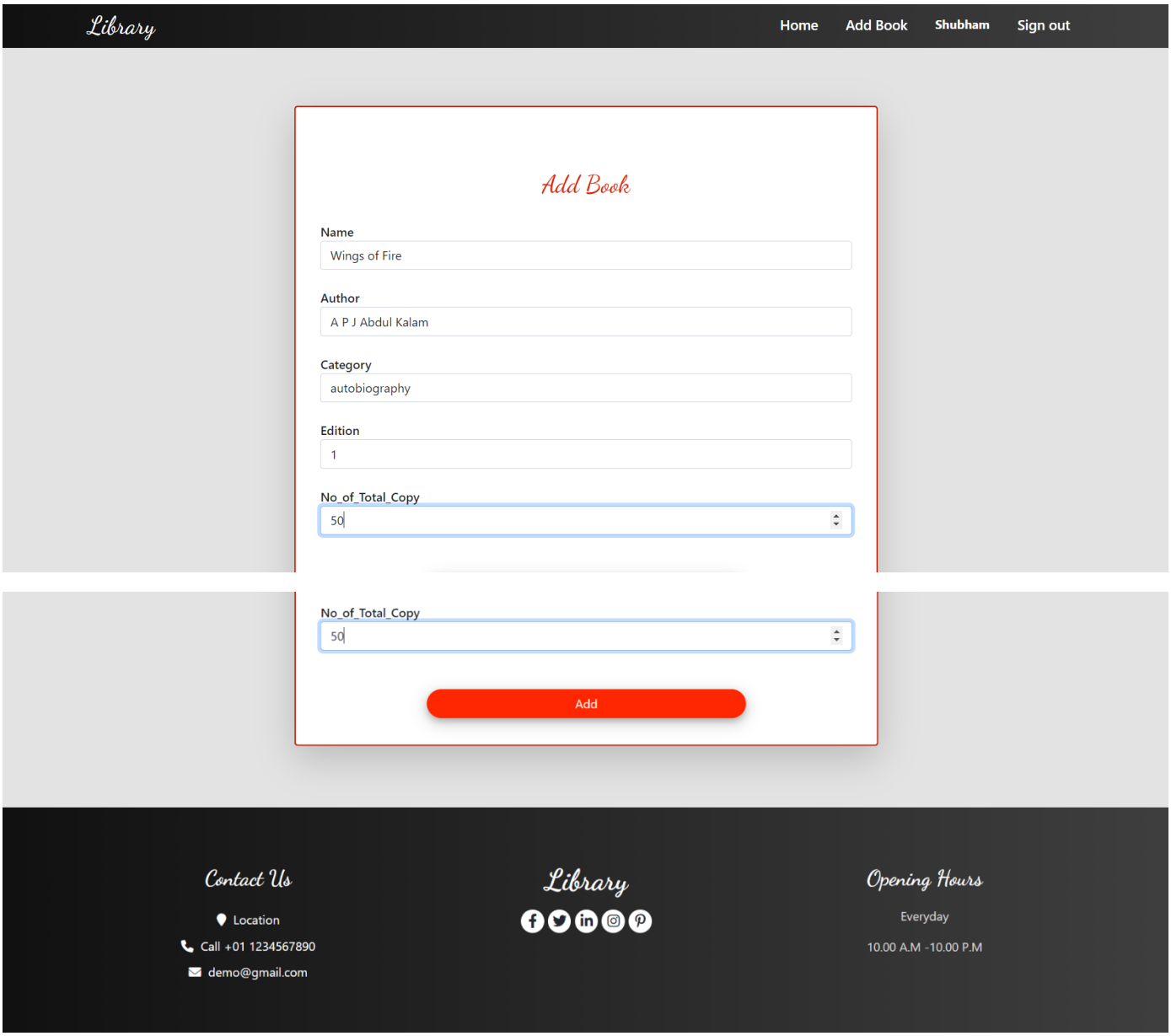
➢ Login:



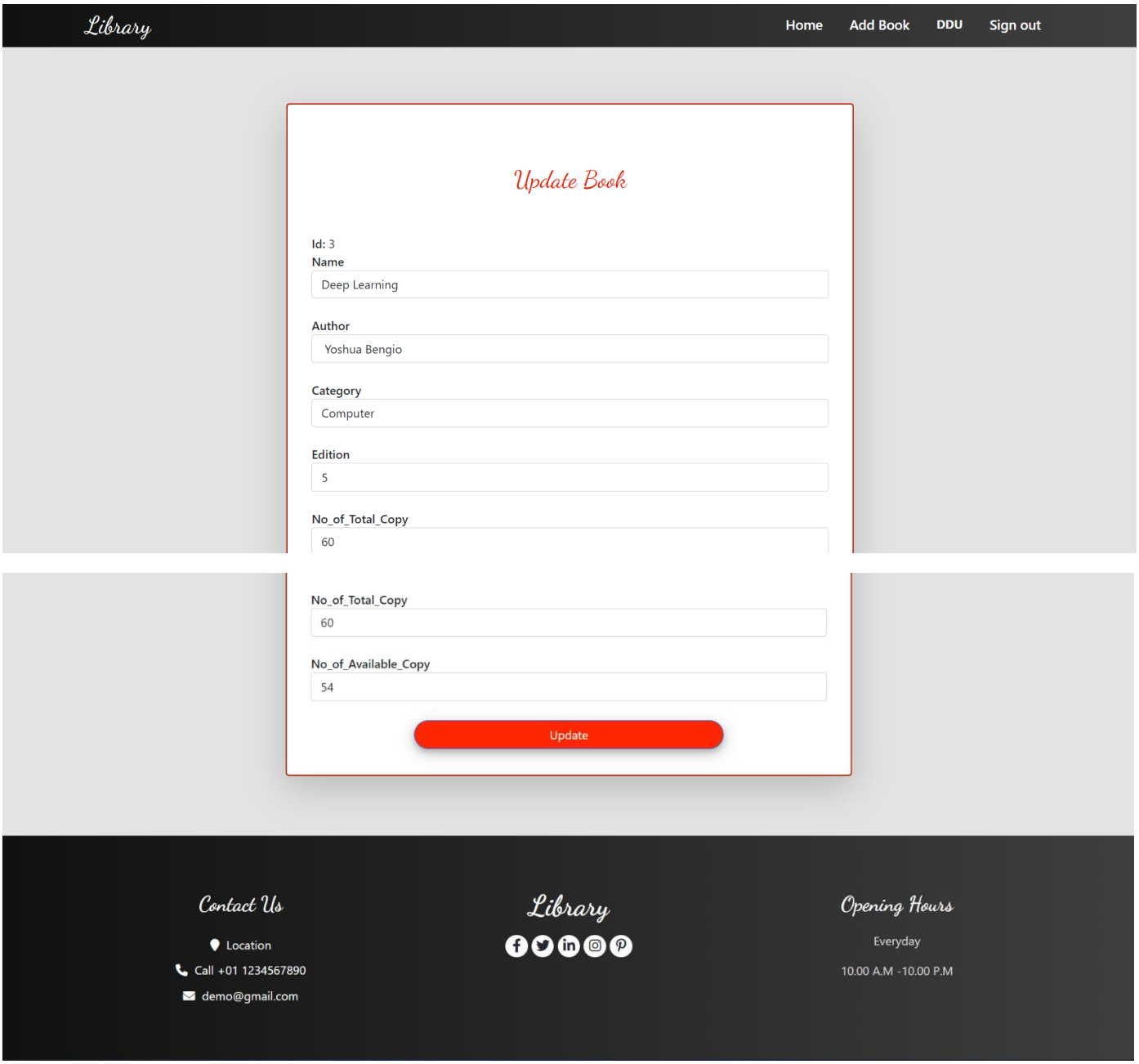
➢ Home Page:



➢ Add Book:


Library
Home
Add Book
Shubham
Sign out
Add Book
Name
Wings of Fire
Author
A P J Abdul Kalam
Category
autobiography
Edition
1
No_of_Total_Copy
50
No_of_Total_Copy
50
Add
Contact Us
Location
Call +01 1234567890
demo@gmail.com
Library
Opening Hours
Everyday
10.00 A.M -10.00 P.M

➢ Update Book:


Library
Home
Add Book
DDU
Sign out
Update Book
Id: 3
Name
Deep Learning
Author
Yoshua Bengio
Category
Computer
Edition
5
No_of_Total_Copy
60
No_of_Total_Copy
60
No_of_Available_Copy
54
Update
Contact Us
Location
Call +01 1234567890
demo@gmail.com
Library
Opening Hours
Everyday
10.00 A.M -10.00 P.M

➢ After Update 3rd Book:

Number Of Books : 4

No.	Id	Name	Author	Category	Edition	Total	Available		
1	1	Wings of Fire	A P J Abdul Kalam	autobiography	1	50	50	Update	Delete
2	2	R. K. Narayan	Rasipuram Krishnaswami Iyer Narayanaswami	autobiography	12	100	100	Update	Delete
3	3	Deep Learning	Yoshua Bengio	Computer	5	60	54	Update	Delete
4	4	Mathematics for Machine Learning	Marc Peter Deisenroth	Math	4	45	45	Update	Delete

➢ After Delete 2nd Book:

Library

Home Add Book DDU Sign out

Number Of Books : 3

No.	Id	Name	Author	Category	Edition	Total	Available		
1	1	Wings of Fire	A P J Abdul Kalam	autobiography	1	50	50	Update	Delete
2	3	Deep Learning	Yoshua Bengio	Computer	5	60	54	Update	Delete
3	4	Mathematics for Machine Learning	Marc Peter Deisenroth	Math	4	45	45	Update	Delete

Contact Us

Location

Call +01 1234567890

demo@gmail.com

Library

Opening Hours

Everyday

10.00 A.M -10.00 P.M

7. Conclusion

The functionalities are implemented in system after understanding all the System modules according to the requirements. Functionalities that are Successfully implemented in the system are:

- User Registration
- Login
- User authentication
- Logout
- Get book
- Add Book
- Update Book
- Delete Book

8. Limitations and Future Enhancements

- As of now this application can be used for library users only for storing books related data. In future we can add more functionalities by creating two different sides, 1) for admin, 2) for end user. In the end user module, user can see all the books borrowed by him/her, date of borrowing, returned books, current books, etc.

9. Reference / Bibliography

Following links and websites were referred during the development of this

Project:

https://github.com/Shubham-Shingala/LibraryManagementSystemWCF

https://docs.microsoft.com/en-us/dotnet/framework/wcf/

https://stackoverflow.com/