**Library Management System:**

Admin and Members are the two role based classes under User.

User have to login first with their username and password. User’s are of two types: Admin and Members.

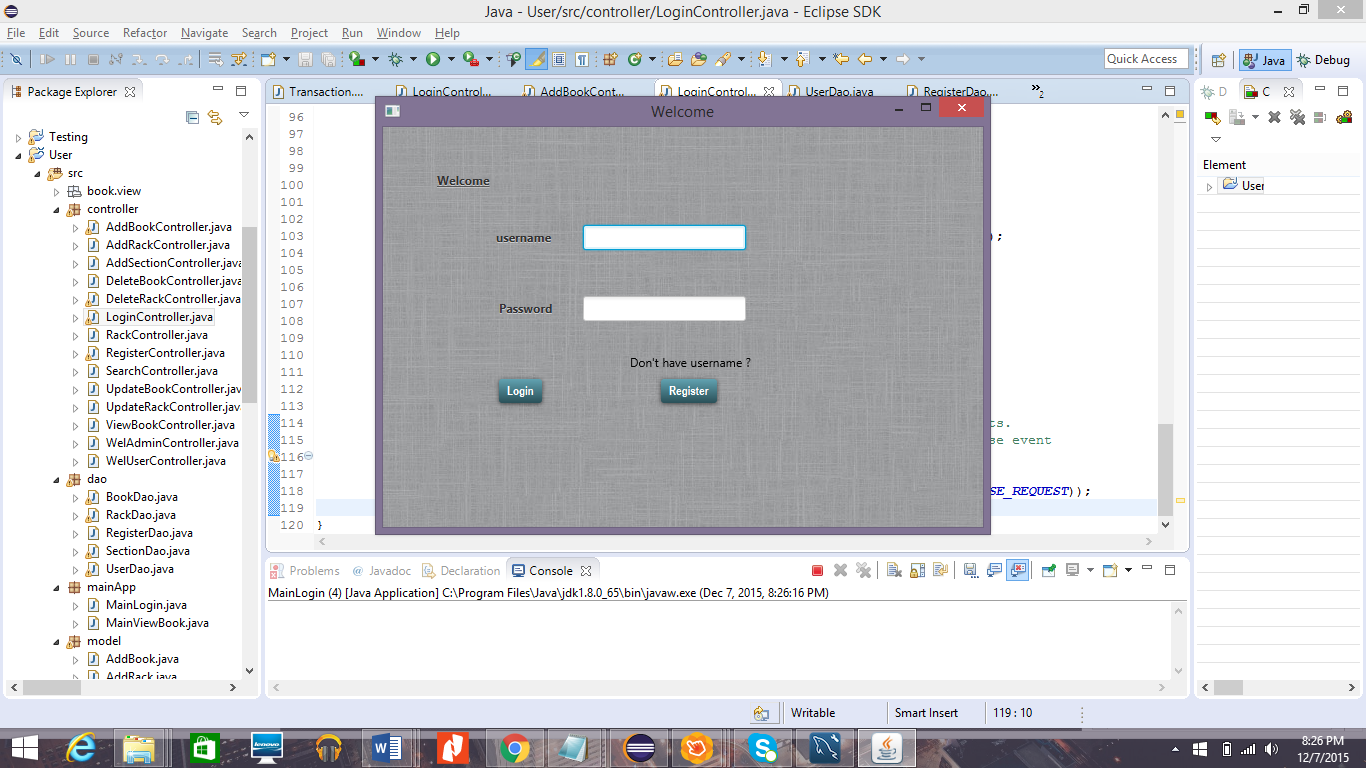
Admin and Members inherit the properties of User. Admin handles and controls the management of library. He can add books, sections and rack. He search them as well. This includes adding new books if the latest edition is arrived or newly available in the market. He can delete the old version of books or the book which are no longer in use. Admin can search books if it’s available or not. Admin can create section, rack and delete the section and rack as well.

Members can access library anytime. Similarly like admin, members have to login first with it’s username and password. Username is the ID of members. Members can see the books available and not available. Members can view the list of books they have issued and when it is due. Members can search the books.

There are different sections, rack to keep the books. Each Section will have different Racks and Different Racks will have different books.

Searchable is used as an interface which will be implemented by the Section, Rack and Book to search the book among these.

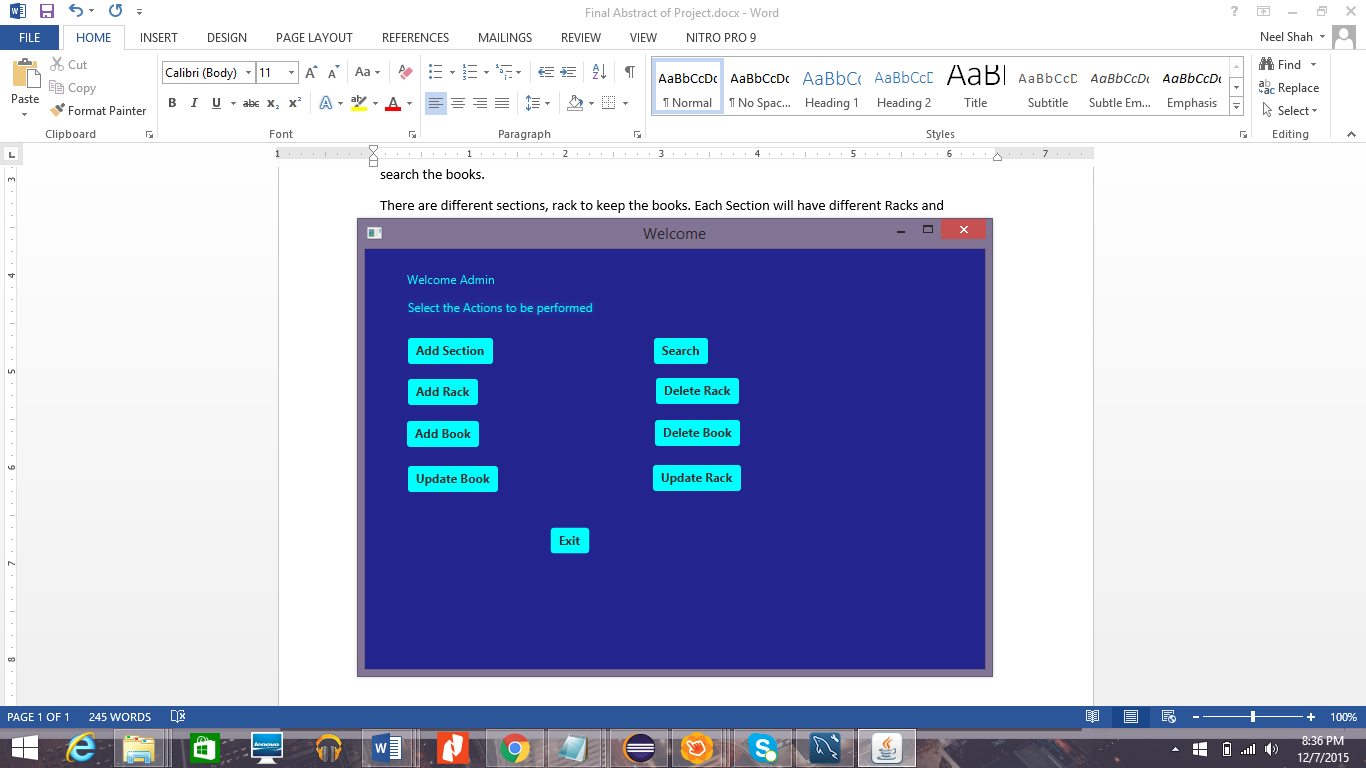
Login Screen:



It’s a Welcome Screen for user and admin.

Admin will login first with the username “admin” and password “admin”

CSS is applied here with background image and having button effect.



This is welcome Screen for the Admin.

Admin can do crud functionality for the following:

**Book**: Create Update Delete

**Rack**: Create Update Delete

**Section**: Create

Search:

It’s an **interface** for the Section, Rack and Book with the search functionality.

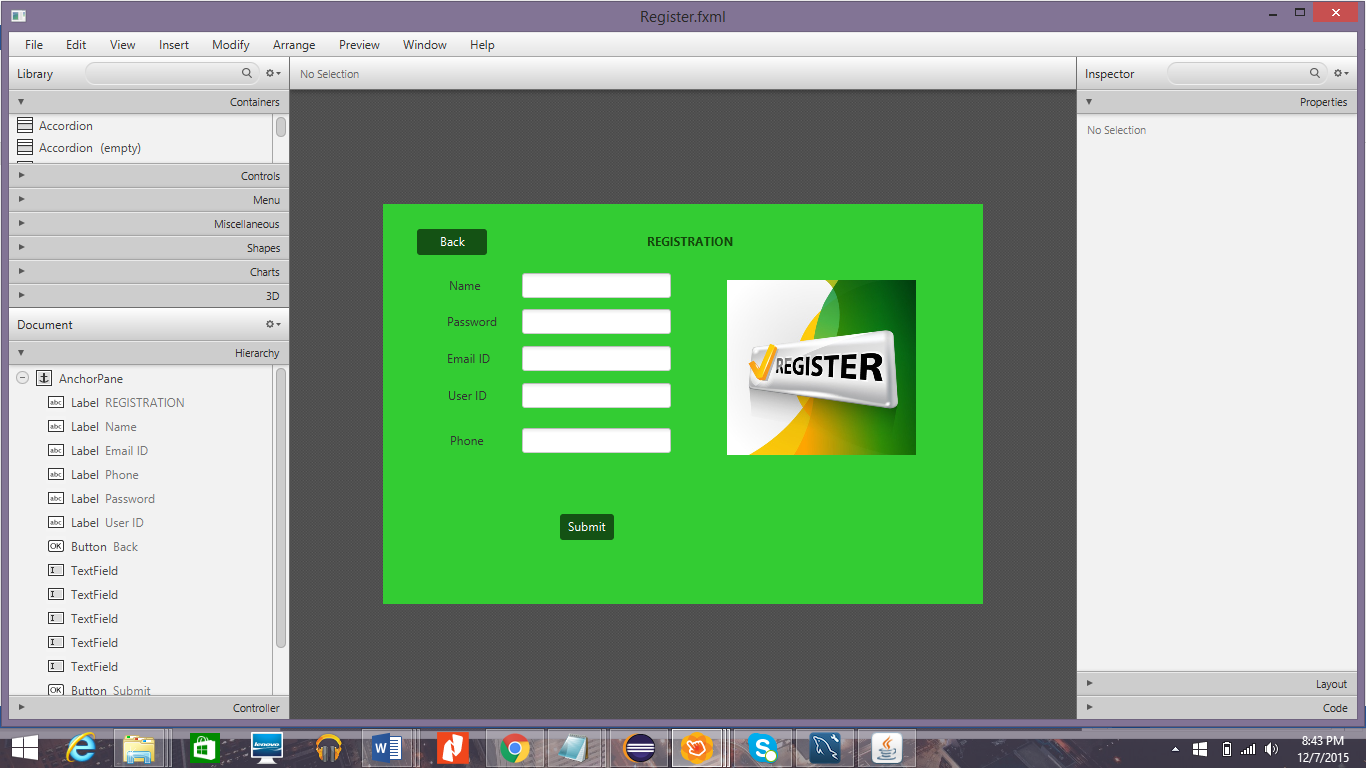
This search will let the admin

***To View Sections***

***To View Rack***

***To View Book***

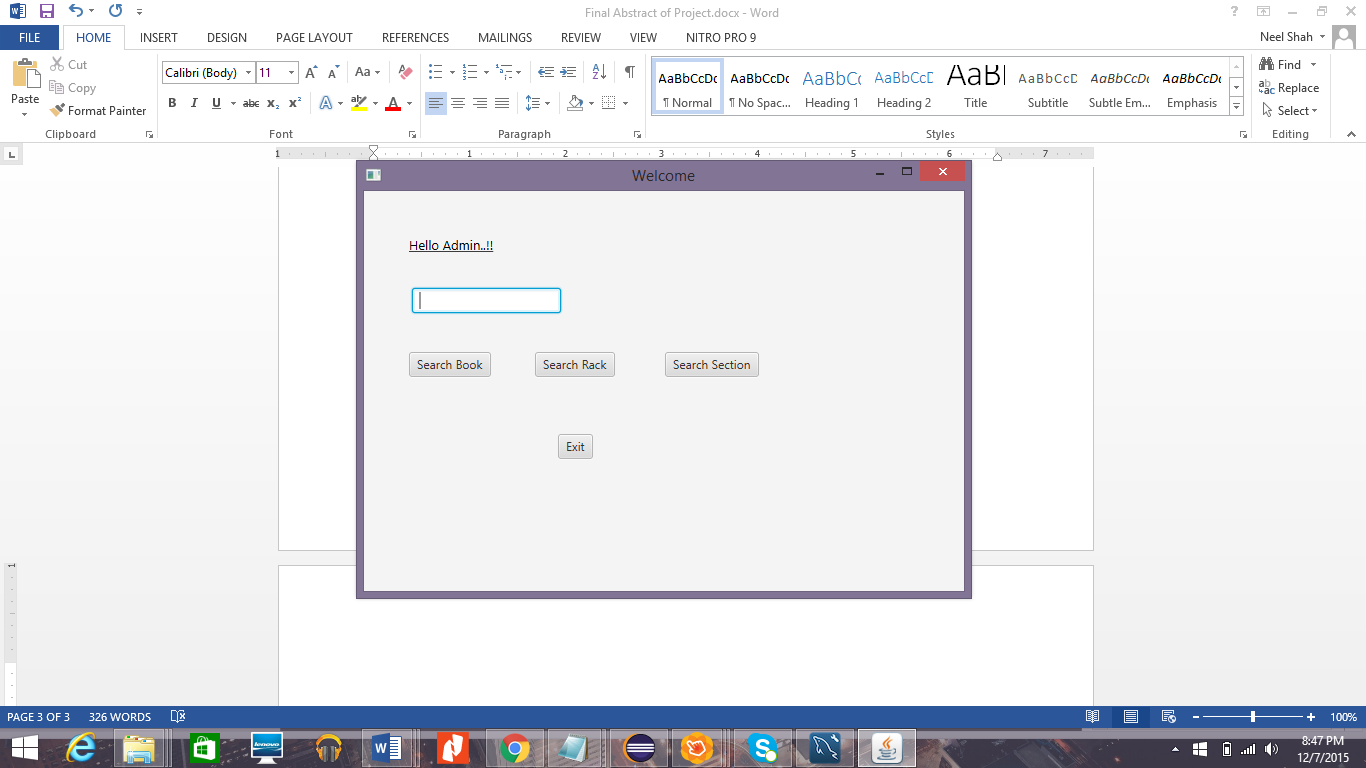
**Registration:**



Inline CSS file is applied in this.

If the user is not registered , he/she can register themselves and that id can be used for login.

Search Interface UI:



Search function is implemented by the section entity, rack entity, book entity.

Database Tables:

1. n\_books;
2. n\_rack;
3. n\_sections;
4. n\_members;
5. n\_users;

Database Tables (DDL Statement):

1. CREATE TABLE n\_book

(

rack\_ID Number(10),

book\_id NUMBER(10,0) NOT NULL,

book\_name VARCHAR2 (50),

author VARCHAR2 (50),

edition NUMBER(10),

ISBN NUMBER(10),

no\_of\_books NUMBER(10),

CONSTRAINT tblBook\_pk PRIMARY KEY(book\_id));

1. CREATE TABLE n\_rack

(

section\_id Number(10),

rack\_id Number(10),

rack\_name VARCHAR2 (50),

no\_of\_rack Number(10),

CONSTRAINT Rack\_pk PRIMARY KEY(rack\_ID),

CONSTRAINT Racks\_pk FOREIGN KEY(section\_ID) references Section (section\_ID ));

1. CREATE TABLE n\_sections

(

section\_id Number(10),

section\_name VARCHAR2 (50),

no\_of\_section Number(10),

CONSTRAINT Section\_pk PRIMARY KEY(section\_ID));

1. CREATE TABLE n\_members

(

Mem\_ID NUMBER (5) NOT NULL,

Mem\_name VARCHAR2 (50),

Mem\_phoneno NUMBER (5),

Mem\_emailID VARCHAR2 (50),

CONSTRAINT Members\_pk PRIMARY KEY(MemID),

CONSTRAINT Member\_pk FOREIGN KEY(ID) references n\_users (ID ));

1. CREATE TABLE Transactions

(

T\_ID NUMBER (5) NOT NULL,

MemID NUMBER (5),

T\_Date date,

Book\_ID Number(5),

Return\_date date,

Borrow\_date date,

CONSTRAINT Transaction\_pk PRIMARY KEY(T\_ID) ,

CONSTRAINT Transactions\_pk FOREIGN KEY(MemID) references tblMember (MemID ));

Project is implemented using MVC model.

Dao layer is created for different models wherever necessary.

Different methods are implemented inside those DAO.

I tried implementing issue book but couldn’t do so.