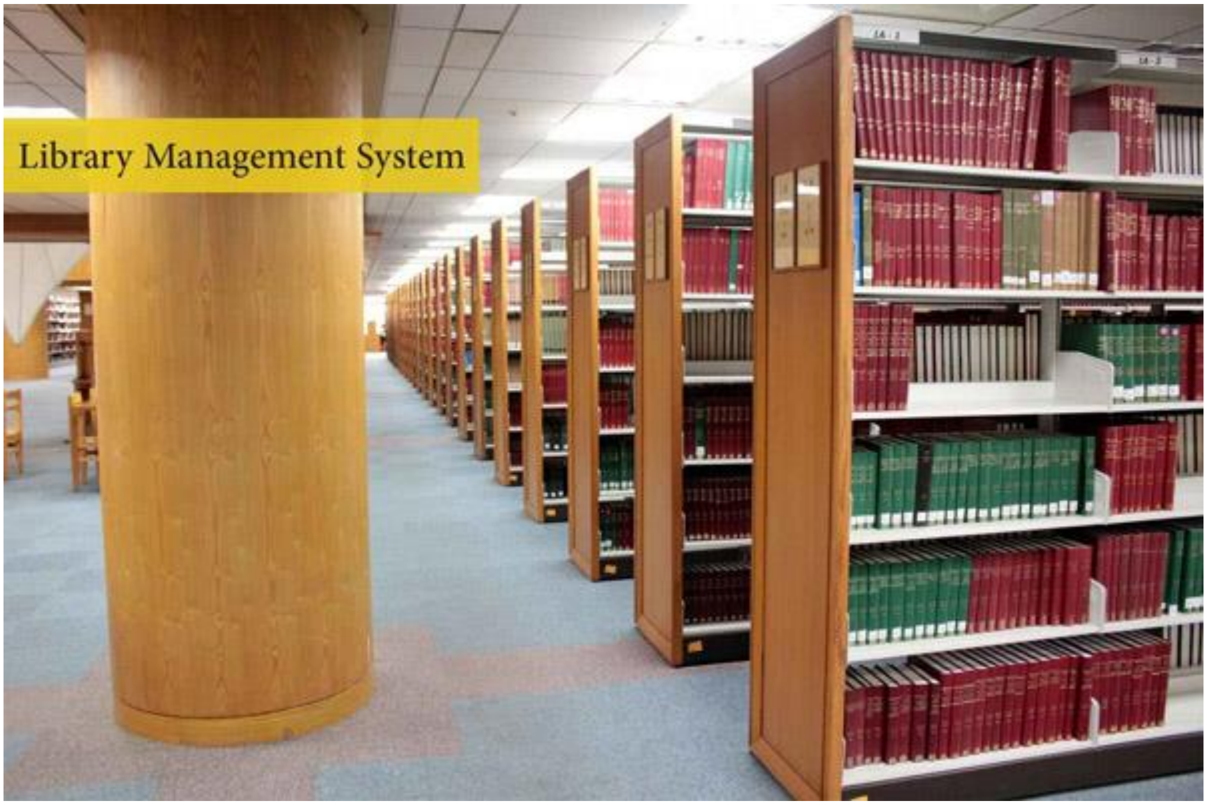
ONLINE LIBRARY MANAGEMENT SYSTEMS



Team:

Utsav Shaileshkumar Chokshi
Preetham Umesh
Gopinath Boyanapally
Grandhe Sai Chetan

Prepared for

CS 687 – Database Systems
Instructor: Dr. Ramazan Aygun

Table of Contents

DESCRIPTION:

The library database system is a web application where the users can search for the books available in the library database and borrow them.

User can register by providing mandatory information like username, firstname, lastname, email, password.

After registering, the user can login to the application with the valid credentials he/she has entered in the register page. After logging in, the user will see a dashboard/home page where he/she can do the following functionalities

- Search for a book in the library database.
- See what books he/she borrowed from the online library and return the books.
- Edit his/her user profile.
- Logout.

PROJECT ENVIRONMENT:

Server:

Server: Amazon AWS Cloud
Operating System: Windows 10
Programming Language: Python
Framework: Django
Database: MySql
Database connector : pymysql

Client:

Operating System: Windows 10
Front end Languages: HTML, JQuery, Bootstrap CSS
Backend Language: Python
Framework: Django
Database: MySql
Testing Environment: PC

Project Plan:

- Topic Submission – January 23rd 2017
- Project Description Submission – January 30th 2107
- Project Formal Description Submission – March 27th 2017
- Server side (Cloud) Implementation – March 31st 2017
- Server and Client Connection, Testing- April 14th 2017
- Source Code submission- April 17th 2017
- Report Submission – April 17th 2017
- Project Demo – April 19th 2017

Responsibilities of each Team Member:

Utsav Chokshi:

- Team Leader
- Server
  - AWS cloud setup
  - Create user relation
  - Insert MySQL method (User Relation)
  - Create
- Client Side
  - Front End Design(Log in page, SignUp page)
- Document the Meeting
- Project Report

Gopinath Boyanapally:

- Server
  - Connect AWS to Database
  - Insert MySQL method (Book Relation)
  - Update MySQL method (Book Relation)
- Client Side
  - Front End Design(Search & return a book pages)
- Project Report

Preetham Umesh:

- Server side:
  - Delete MySQL method(Issue Book Relation)
  - Create Book Relation Table
- Client Side
  - Front End Design(Dashboard, LogOut)
- Project Report

Grandhe Sai Chetan:

- Server side
  - Update MySQL method(Edit User profile Relation)
  - Create Book Relation Table

- Client Side
    - Front End Design(Edit Profile page, LogOut page)
- Document the Meeting
- Project Report

FORMAL DESCRIPTION:

The library database keeps track of the users and books. The users in the library management system have a first name, last name, username, an authentication unique id, Email and Password to access the account.

The book relation contains the following information (i.e) unique Book ISBN, Book name, the author of the book, Publisher and whether the book is issued or not.

The issued by relation will hold the data about the books issued by a specific user, the specific Book ISBN, Book name, issued date and due date.

The book log relation will store the information about the books that were borrowed in the past.

Entity Types:

User:

Following are the attributes of the user entity –

□ ID – This is the primary key of the user relation. It is used to differentiate the users. It gets populated automatically in the database.

□ First name – This attribute stores the first name of the user.

□ Last name – This attribute stores the last name of the user.

□ Username – This is used to access the account. The username should be unique.

□ Password – This attribute stores the password to access the account.

□ Email – This stores the email id of the user.

□ Last login – This stores the recent login by the user. This updated automatically by the Django framework.

□ Superuser - The data type of this attribute is Boolean. The value will be 1 for the library admin since he is the super user and he has the rights to control and manipulate the database. And the value will be 0 for all the normal users.

□ staff – The data type of this attribute is Boolean.This value will be 1 for the library admin as he has all the rights to remove or add the users and the books.

□ active – The data type of this attribute is Boolean. Stores the data of the users who are active as 1 and 0 for non active users. This will be automatically updated by Django framework.

□ Date joined – This will store the date of joining of the user. We do not need to enter this data as it is dynamically inserted by the framework.

Book:

This entity stores the information about the book.
□ Book ISBN: This stores the ISBN number of the book which is unique and used to identify the book. It is also the primary key of the relation.
□ Book name : This stores the name of the book.
□ Author : This attribute stores the name of the author.
□ Publisher: This attribute stores the publisher name of the book.
□ Is_issued: This stores the Boolean value to see if the book is issued or not.

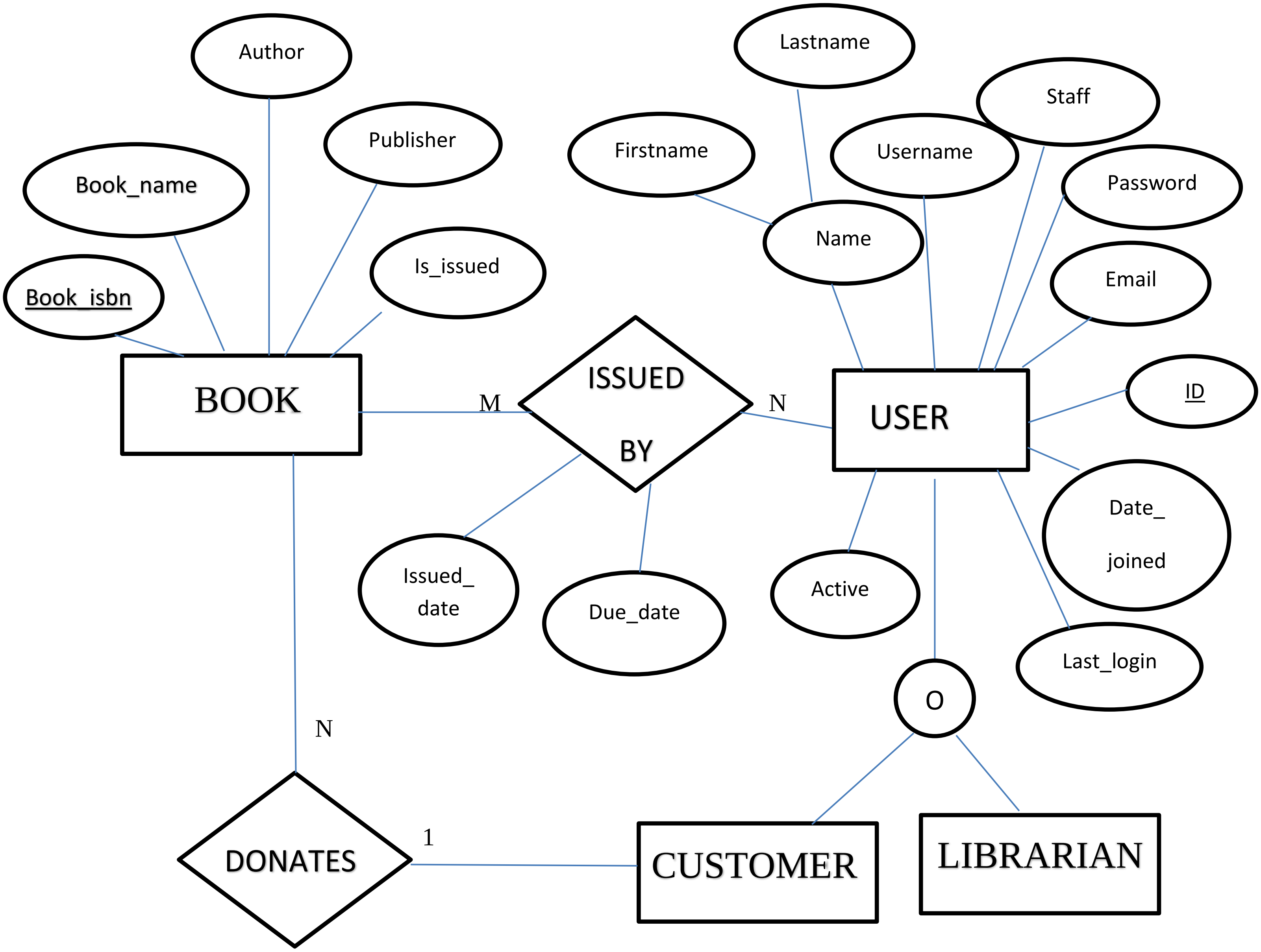
Relationship:

A relationship issue by between two entities user and book associates each user with the books issued by him. This relation contains the data about the books issued by the user. The following are the attributes of this relationship.
□ ID – It is the ID of the user.
□ Book ISBN – This is the ISBN number of the book.
□ Book name – This attribute stores the name of the book.
□ Issued date – This stores the date of the book issued on.
□ Due date – This stores the information of the due date implied on each book.
The primary key of this relation is Book ISBN. There are two foreign keys in this relation.
1.) ID – which refers to the ID in the user relation.
2.) Book ISBN – which refers to the book ISBN attribute of the Book relation.

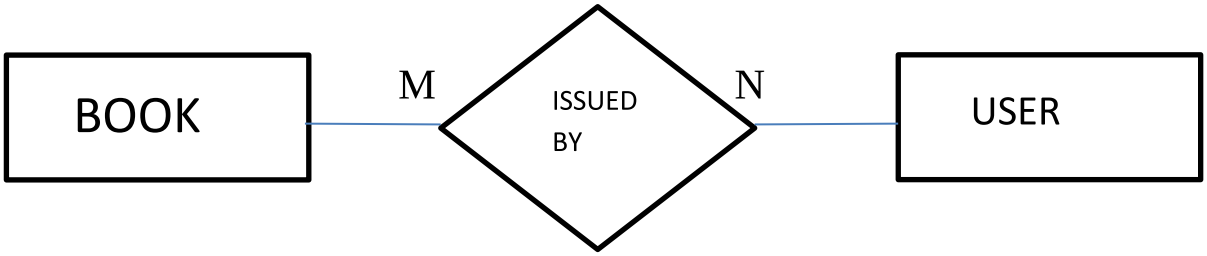
Mapping cardinalities:

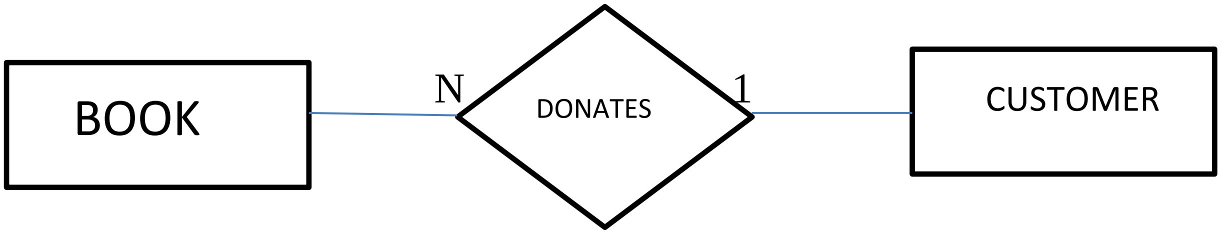
- There is many to many cardinality between the two entities user and book.This means many persons can issue many books or vice versa.
- There is one to many cardinality between the two entities customer and book.This means that one customer can donate many books .

EER DIAGRAM:


Author
Publisher
Book_name
Is_issued
Book_isbn
BOOK
M
ISSUED BY
N
Issued_ date
Due_date
Lastname
Staff
Firstname
Username
Password
Name
Email
USER
ID
Date_ joined
Active
Last_login
O
N
DONATES
1
CUSTOMER
LIBRARIAN

Mapping EER diagram into Object-oriented Database:





1.) Mapping every entity to a relation:
In our database library we have entities user, librarian, customer and book. But customer and librarian are overlapping under user.

Below are the relations of the database.
□ User (ID, Firstname, Lastname, Username, Email, Password, Superuser, Last_login ,Staff, Active, Date_joined)
□ Book(Book ISBN , Book name, Author, Publisher, Is_issued)
We also a separate relation to store the logs of the issued books.
Book_log(ID, Book_ISBN, Book_name, Issued_on, Return_date)

There are two foreign keys in this relation.

1.) ID – which refers to the ID in the user relation.
2.) Book ISBN – which refers to the book ISBN attribute of the Book relation.

2.) Relationships:
Here we have one relationship that is "issued_by". So we will have one relation.
□ Issued_by (ID, Book_ISBN, Book_name, Issued_on, Due_date)
There are two foreign keys in this relation.
1.) ID – which refers to the ID in the user relation.
2.) Book ISBN – which refers to the book ISBN attribute of the Book relation.

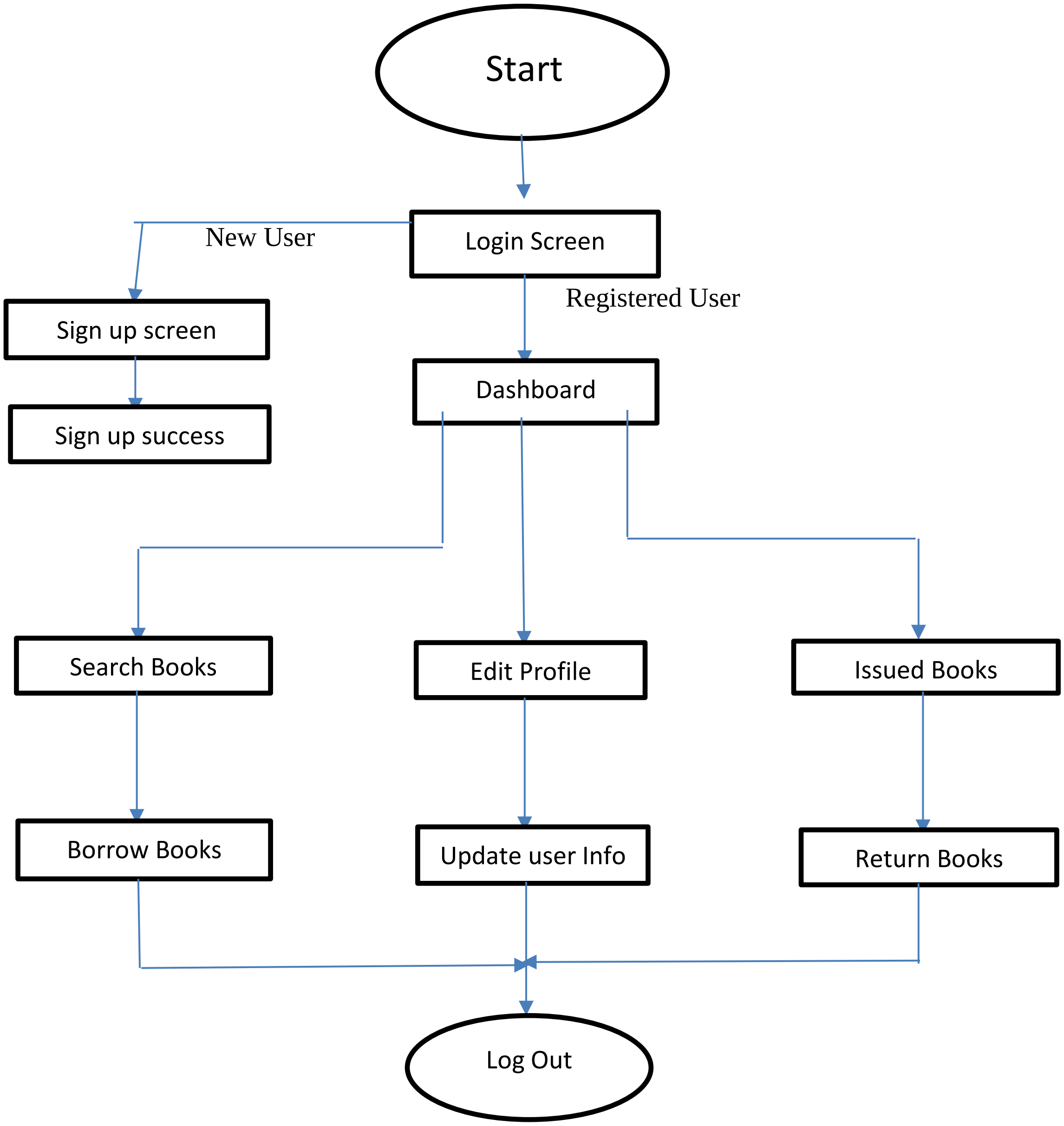
Create Table Statements:

```
CREATE TABLE USER(
        ID      int     NOT NULL PRIMARY KEY,
        Password        varchar(20)     NOT NULL,
        Last_login      date,
        Superuser       boolean         NOT NULL,
        Username        varchar(25) NOT NULL,
        First_name      varchar(25) NOT NULL,
        Last_name       varchar(25) NOT NULL,
        Email           varchar(25) NOT NULL,
        Staff           boolean                 NOT NULL,
        Active                  boolean                 NOT NULL,
        Date_joined date                NOT NULL
);
CREATE TABLE Bookrelation(
        Book_isbn       varchar(9) PRIMARY KEY,
        Book_name varchar(50) NOT NULL,
        Author                  varchar(50)     NOT NULL,
        Publisher       varchar(50)     NOT NULL,
        Donor           varchar(50),
        Is_issued       Boolean                 NOT NULL
);

CREATE TABLE Book_issued(
        User_id                 int                     NOT NULL,
        Book_isbn       varchar(9)      NOT NULL PRIMARY KEY,
        Book            varchar(50) NOT NULL,
        Issued_ON       date            NOT NULL,
        Due_date        date            NOT NULL,
        FOREIGN KEY (Book_isbn) references Bookrelation(Book_isbn),
        FOREIGN KEY (User) references User(ID),
);
```

```
CREATE TABLE Book_log(
        User_id              int                     NOT NULL,
        Book_isbn   varchar(9)   NOT NULL PRIMARY KEY,
        Book            varchar(50) NOT NULL,
        Issued_ON   date             NOT NULL,
        Return_date date             NOT NULL,
        FOREIGN KEY (Book_isbn) references Bookrelation(Book_isbn),
        FOREIGN KEY (User) references User(ID),
);
```

System Architecture:



Component Description:

Sign-Up: This module gets the sign up information from the user and sends the information to the server and displays the response from the server to the user.

Login Screen: This module gets username and password from the user and sends them to the server to get information about the user.

Dashboard: This module is divided into two sub modules: Search a book and Issued books.

Search a book:

- This module downloads the books data from the server and displays them to the user.
- This module has the functionality of issuing the required books to the user one at a time.

Issued books:

- This module downloads the books that the user has borrowed previously.
- This module has the functionality of returning the book.

Edit Profile:

This module helps in updating user information i.e Username, Email address, first name and last name.

Issue Management:

1. Is the application a single-user or multi-user application?
Answer-It is a multiuser Application.

2. Is the application read-only (only retrieval) or read-write application?
Answer - It is a read-write application.

3. How is concurrency handled? (i.e., are you aware of the problems that may occur if concurrency is not handled properly?)
Answer - Concurrency is properly handled by MYSQL RDBMS allowing multiple users to update the same entity at the same time without causing any problems.

4. Is indexing or fast-retrieval an important part of your application?
Answer - Yes

5. What are the steps taken if the system or a transaction fails?
Answer - We have considered various exception handling situations.

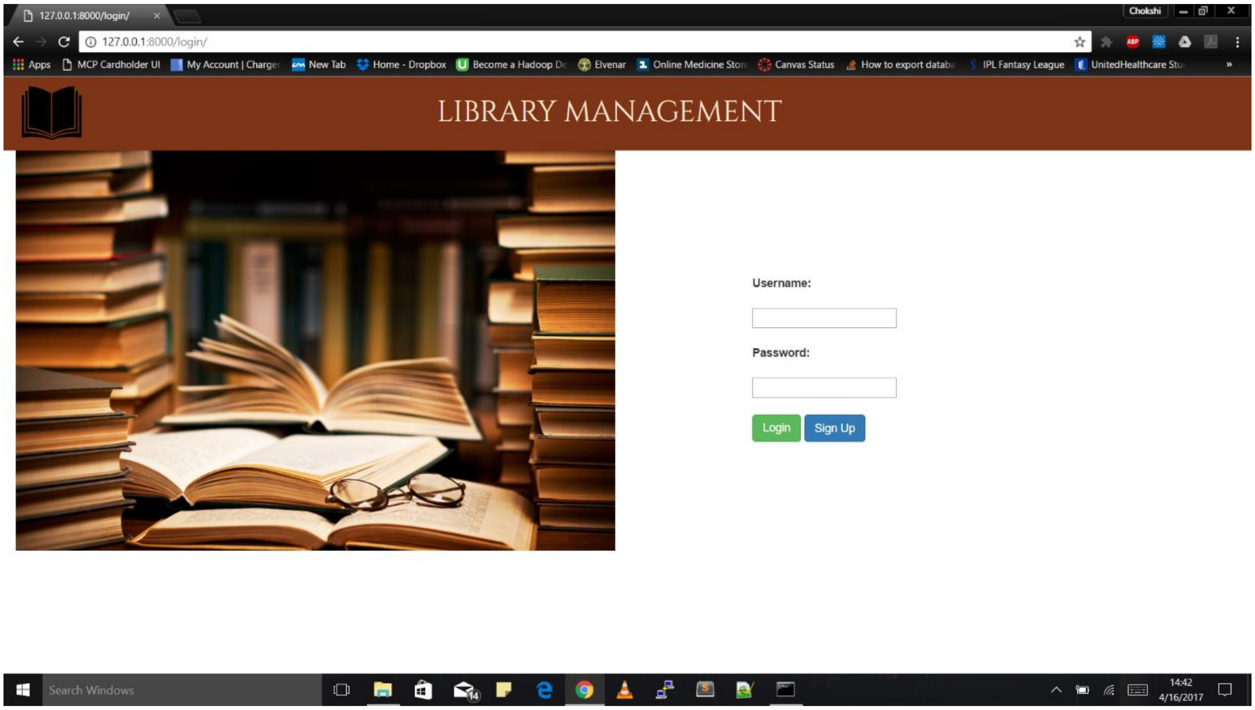
6. Is the mobility of application a good feature? How about deploying front end app on an Android, iOS, or Windows 8 system? Consider whether the database will be stored at the client or at a server.
Answer - The mobility of application is good feature. The application's can be freely excessed as a website on an android, ios or windows system. The Database will be stored on cloud based server.

7. Is your application data stored on a cloud database?
Answer - Yes it is we have hosted our application on cloud database with Amazon web service. We have used MYSQL RDBMS as Server Management provider.

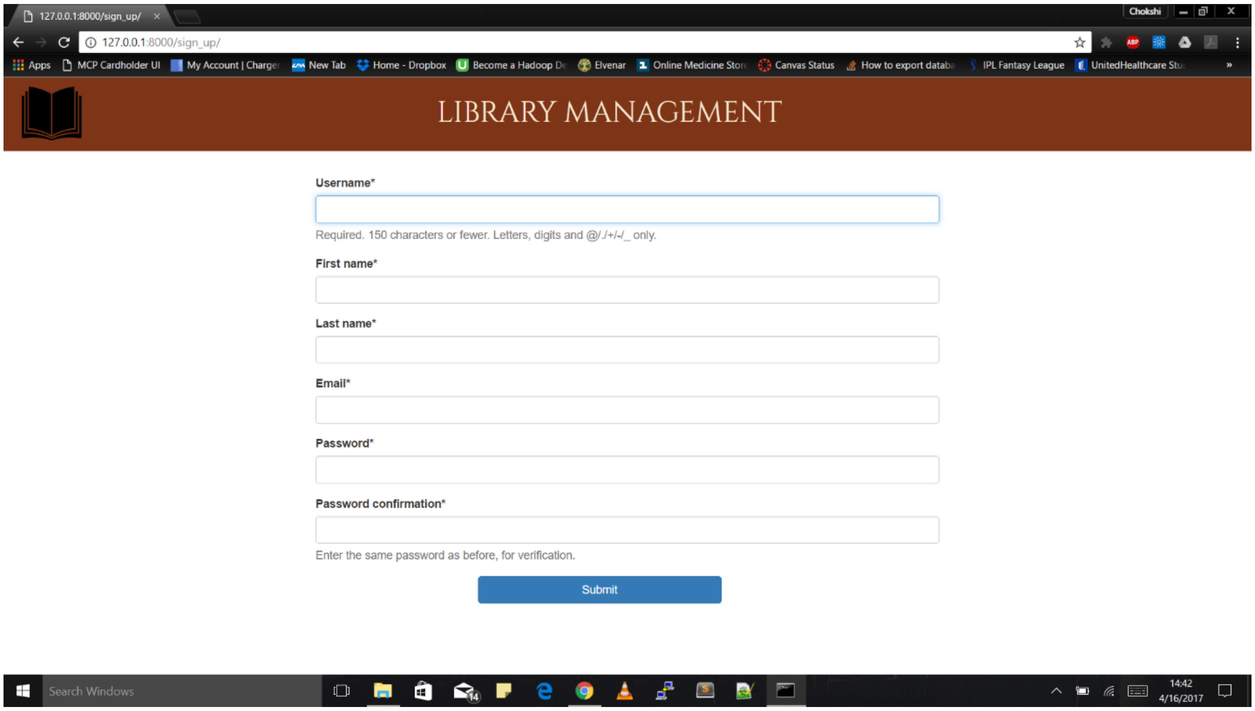
8. What is the size of your database? Are you able to populate your database with realistic data?
Answer – As of now the size of our database is around 20 MB and it will increase as the number of users increases. Yes we have filled the database with realistic data.

UI SNAPSHOTS:

- **LOGIN**



- **SIGNUP**



LIBRARY MANAGEMENT

Username*

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

First name*

Last name*

Email*

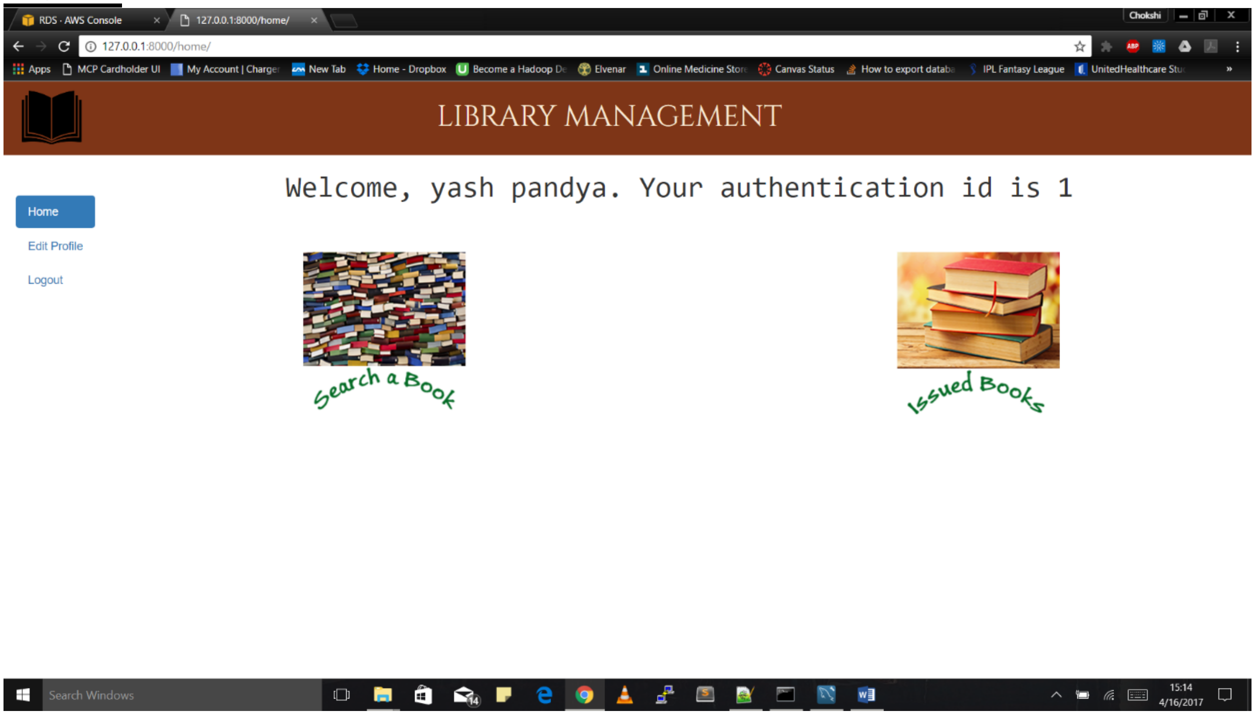
Password*

Password confirmation*

Enter the same password as before, for verification.

Submit

- **HOME**



LIBRARY MANAGEMENT

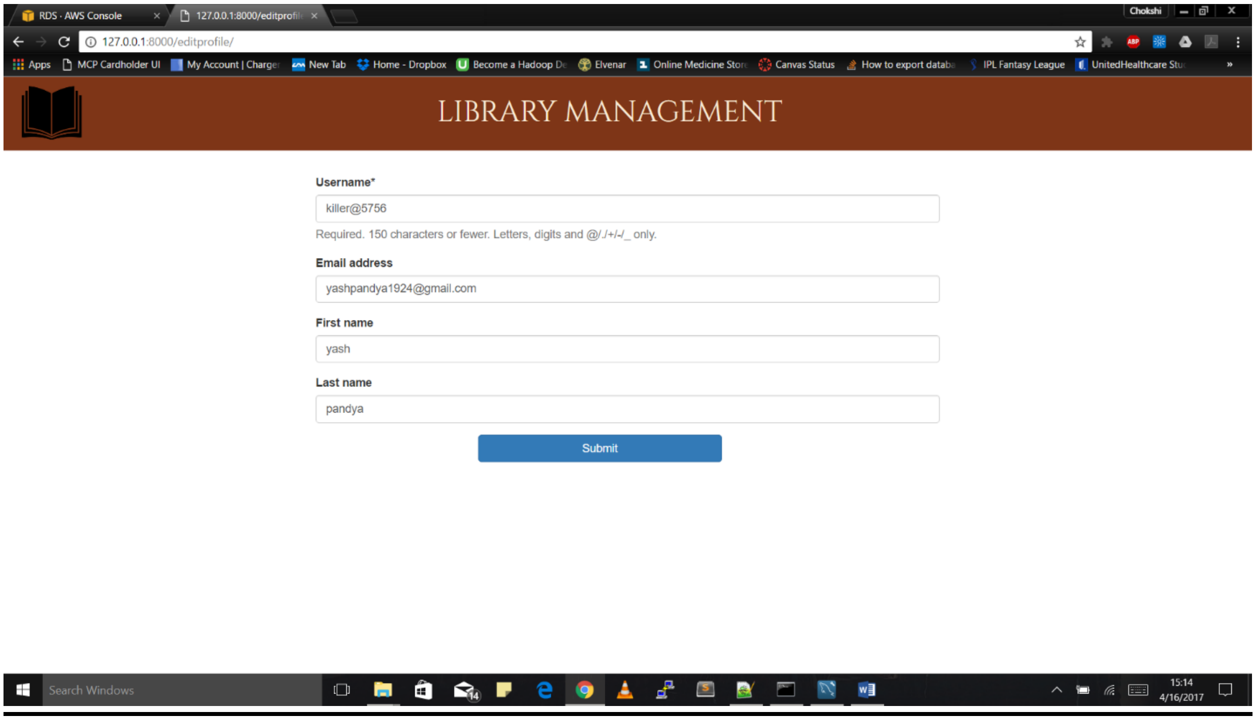
Welcome, yash pandya. Your authentication id is 1

Home

Edit Profile

Logout

- **EDIT PROFILE**



LIBRARY MANAGEMENT

Username*

killer@5756

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Email address

yashpandya1924@gmail.com

First name

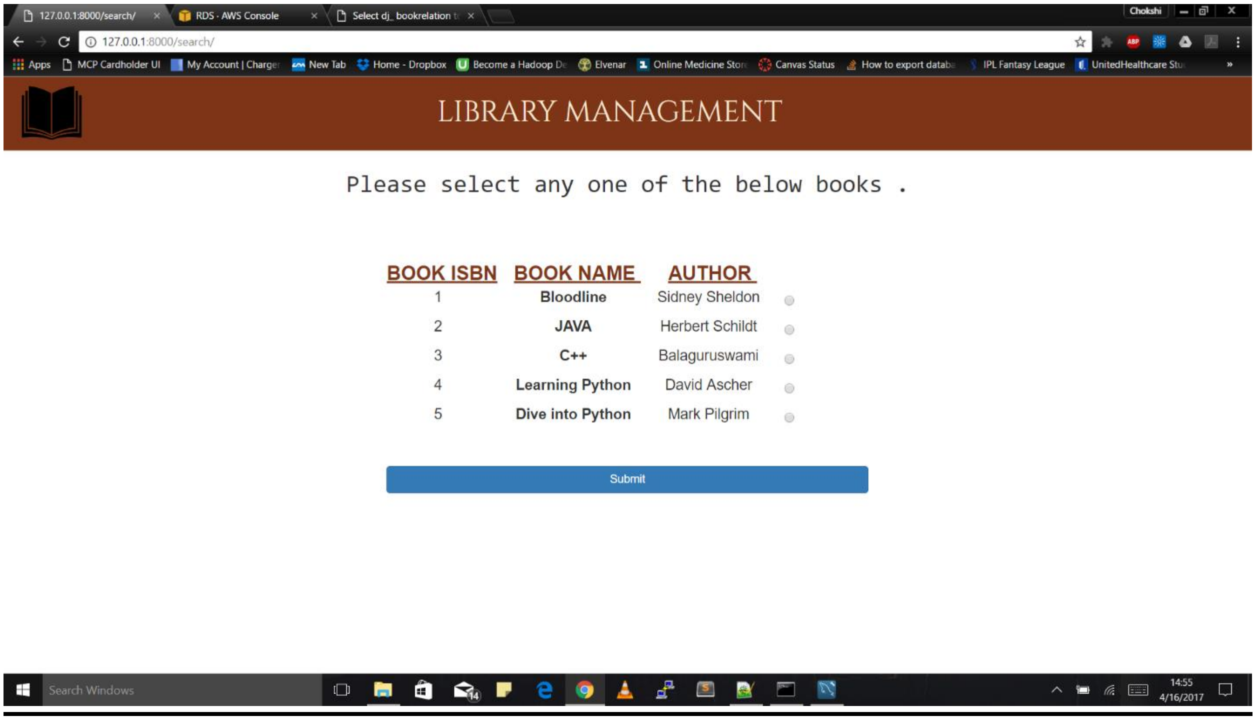
yash

Last name

pandya

Submit

- **SEARCH A BOOK**



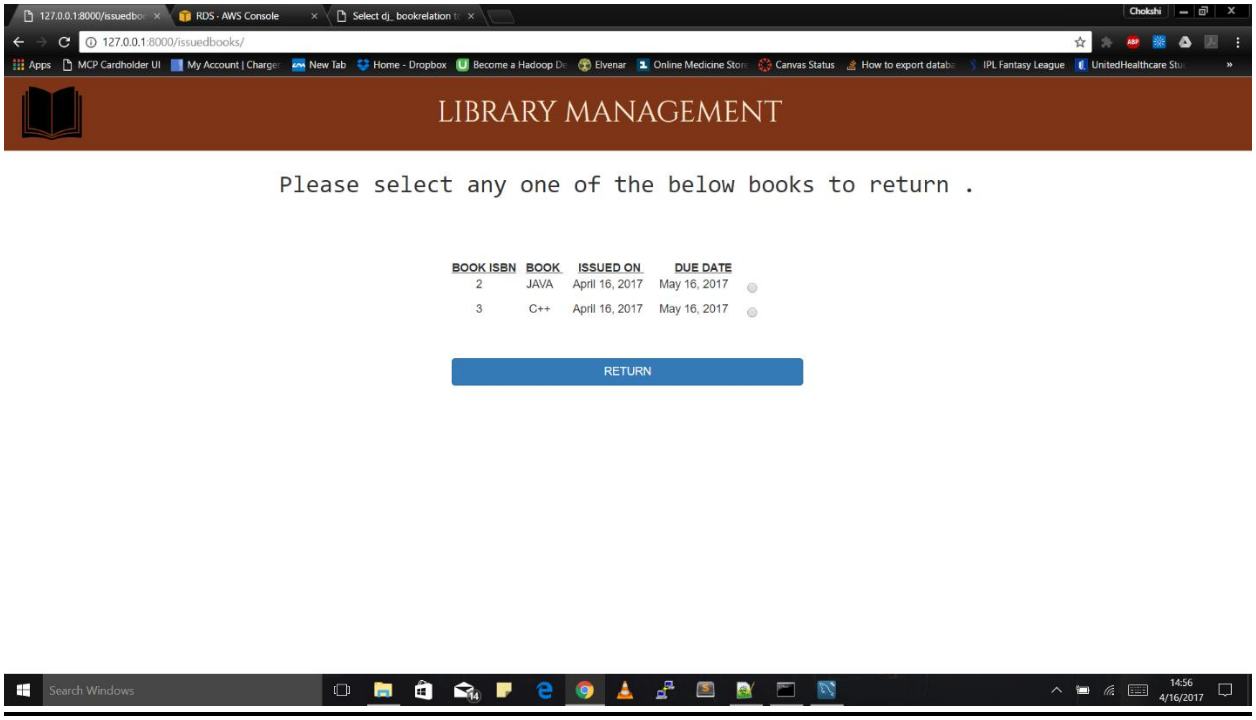
LIBRARY MANAGEMENT

Please select any one of the below books .

BOOK ISBN	BOOK NAME	AUTHOR	
1	Bloodline	Sidney Sheldon	○
2	JAVA	Herbert Schildt	○
3	C++	Balaguruswami	○
4	Learning Python	David Ascher	○
5	Dive into Python	Mark Pilgrim	○

Submit

- ## RETURN A BOOK

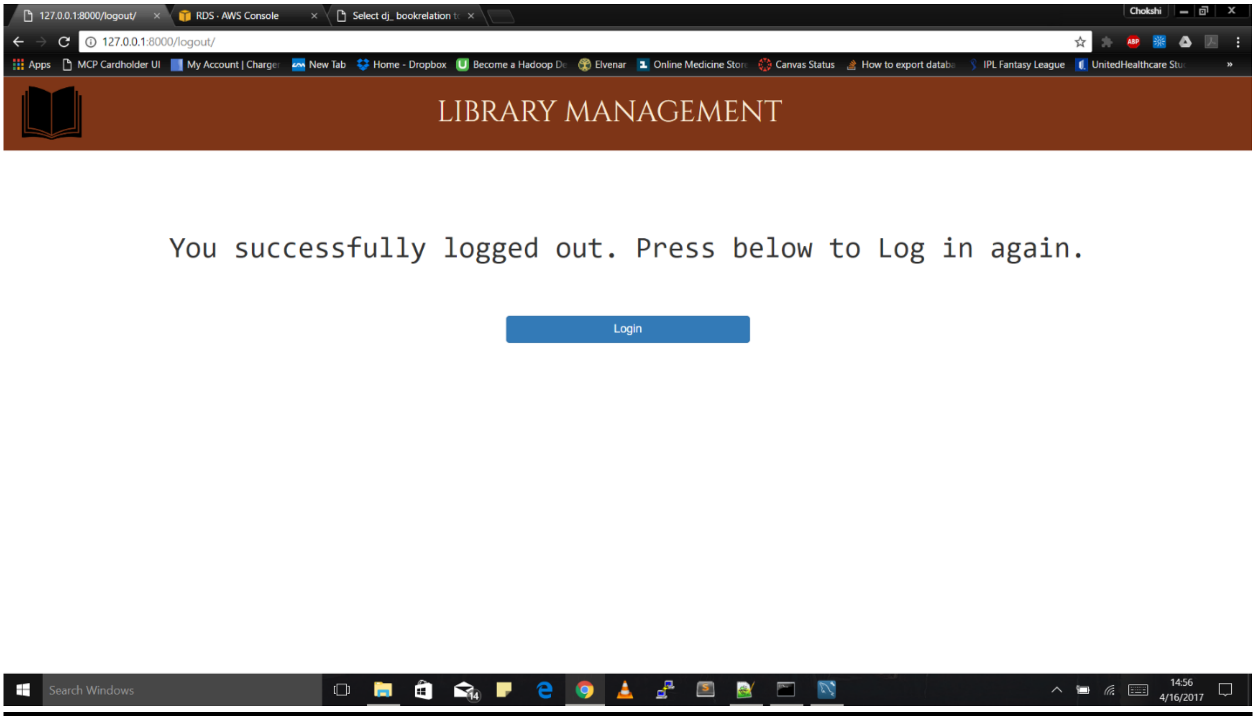


Please select any one of the below books to return .

BOOK ISBN	BOOK	ISSUED ON	DUE DATE	
2	JAVA	April 16, 2017	May 16, 2017	○
3	C++	April 16, 2017	May 16, 2017	○

RETURN

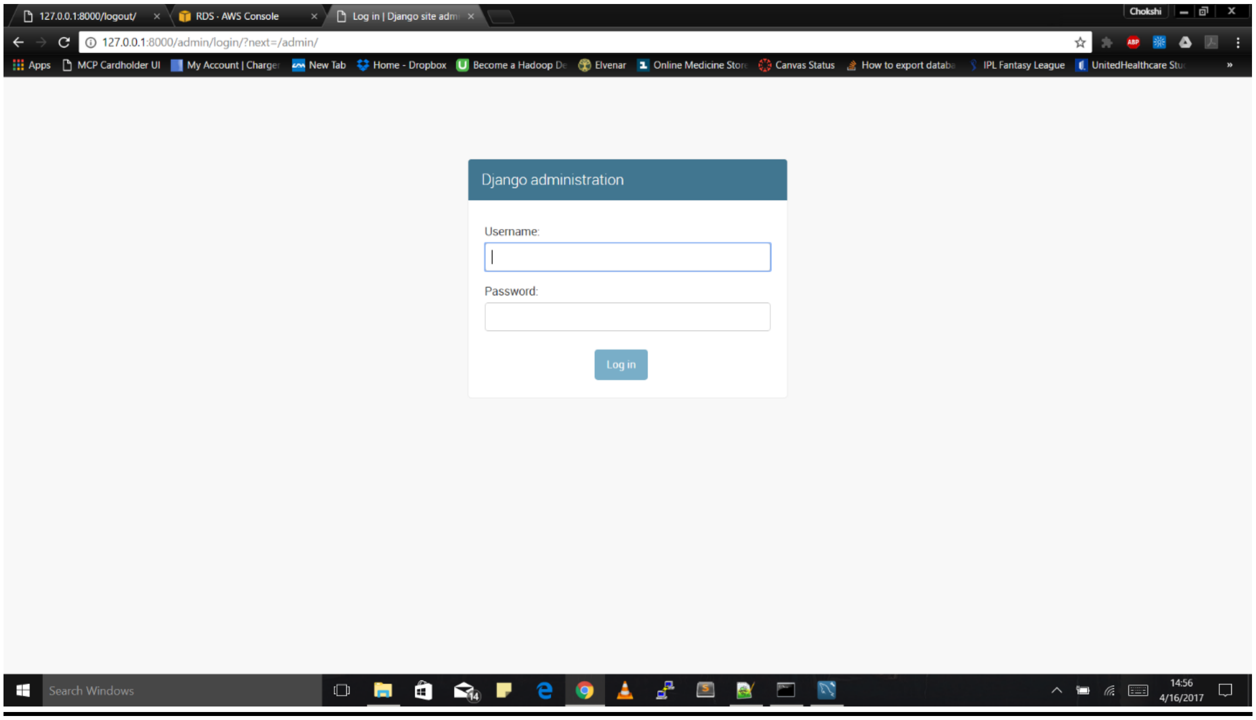
- ## LOGOUT



You successfully logged out. Press below to Log in again.

Login

- **ADMIN LOGIN**



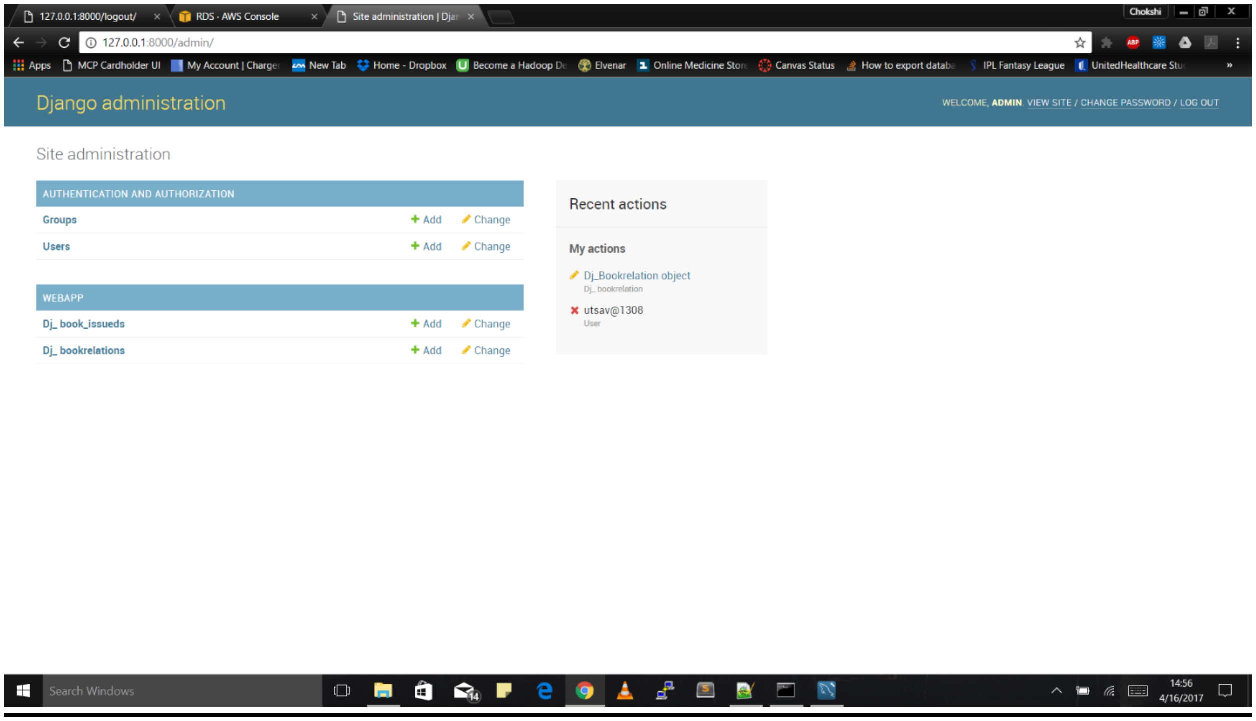
Django administration

Username:

Password:

Log in

- **ADMIN HOME**



Django administration

WELCOME, **ADMIN**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change

WEBAPP		
Dj_ book_issueds	+ Add	Change
Dj_ bookrelations	+ Add	Change

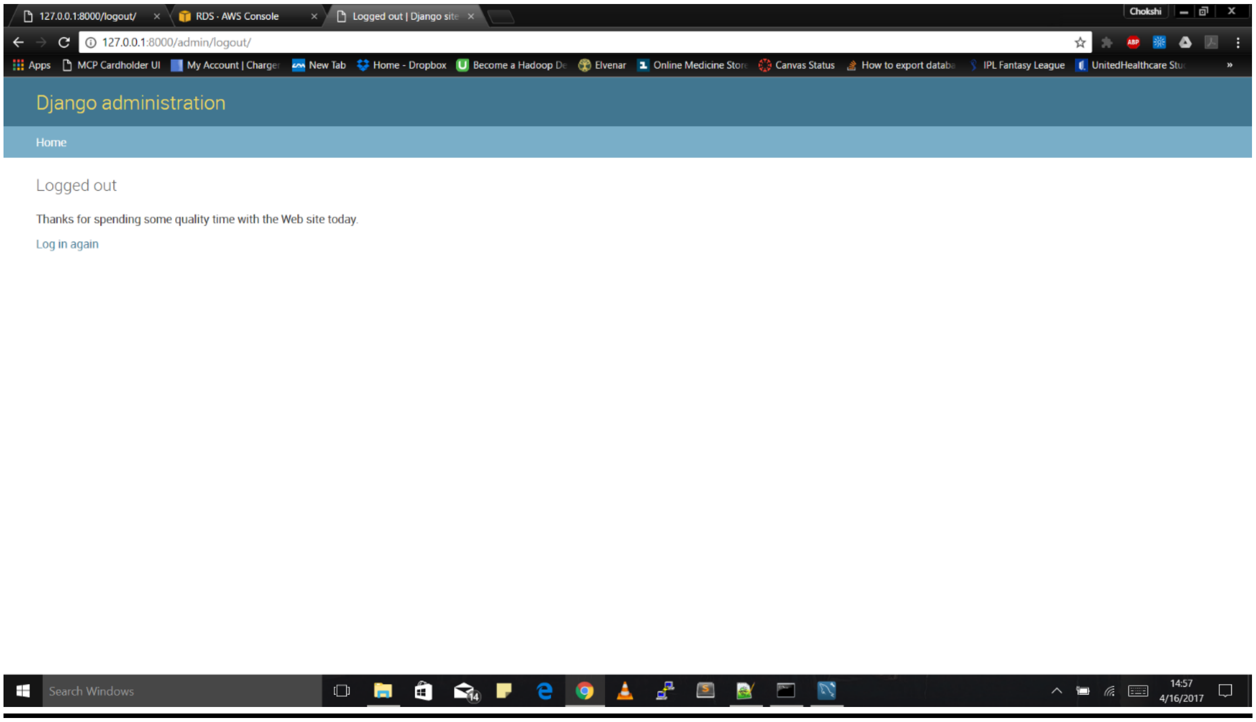
Recent actions

My actions

Dj_Bookrelation object
Dj_ bookrelation

utsav@1308
User

- **ADMIN LOGOUT**



Django administration

Home

Logged out

Thanks for spending some quality time with the Web site today.

Log in again

MINUTES OF MEETING:

First Meeting:

Date of meeting:	January 21st 2017	Time:	11a.m	
Location:	Charger Union	Duration:	45minutes.	
Meeting Objective:	Topic Selection			
Detailed Description:	Discussed on various topics and finalized on making an application on library management system.			
Attendees:	ALL			

Second Meeting:

Date of meeting:	January 24th 2017		Time:	6p.m
Location:	Library		Duration:	30minutes
Meeting Objective:	Project Environment Selection			
Detailed Description:	As this application is used daily, we decided to make an web application and the backend on AWS cloud.			
Attendees:	All			

Third Meeting:

Date of meeting:	January 28th 2017		Time:	12 p.m
Location:	The Overlook		Duration:	1.5hr.
Meeting Objective:	Responsibility Meetings and working on informal description			
Detailed Description:	We decided to split both the server and client side code equally. We also designed the ER diagram and drafted the informal description.			
Attendees:	All			

Fourth Meeting:

Date of meeting:	March 24th 2017	Time:	3p.m	
Location:	The Overlook	Duration:	1.5hrs	
Meeting Objective:	Formal Description			
Detailed Description:	We drafted the formal description and designed the database schema			
Attendees:	All			

Fifth Meeting:

Date of meeting:	March 31st 2017	Time:	4p.m	
Location:	Charger Union	Duration:	2hrs.	
Meeting Objective:	Server Side Implementation			
Detailed Description:	We used AWS as server and MySQL, Python(Django Framework)			
Attendees:	All			

Sixth Meeting:

Date of meeting:	April 2nd 2017	Time:	11a.m	
Location:	Charger Union	Duration:	2hrs.	
Meeting Objective:	Designing the UI			
Detailed Description:	Initially we hand drew the web pages then we used HTML, Bootstrap CSS.			
Attendees:	All			

Seventh Meeting:

Date of meeting:	April 8th 2017	Time:	11a.m	
Location:	Charger Union	Duration:	1.5hrs	
Meeting Objective:	Continuing the server side implementation			
Detailed Description:	Completed the server side implementation.			
Attendees:	All			

Eight Meeting:

Date of meeting:	April 14th 2017	Time:	5p.m	
Location:	The Overlook	Duration:	1hrs.	
Meeting Objective:	Application Testing.			
Detailed Description:	Tested the application in real time and they worked correctly			
Attendees:	All			

Ninth Meeting:

Date of meeting:	April 15th 2017	Time:	11a.m	
Location:	The Overlook	Duration:	2hrs	
Meeting Objective:	Project Report			
Detailed Description:	The final documentation of project report was drafted.			
Attendees:	All			