LIBRARY MANAGEMENT SYSTEM

Submitted By
Priti Patel (0897217)
Vaishnavi Bhambure (1156082)

December 13th, 2017

Presented to
Department of Computer Science
New York Institute of Technology

Database Systems
CSCI_760-M02-2017 FALL
Dr. H. Gu

Table of Contents

ACKNOWLEDGEMENT

Foremost, we would like to express our sincere gratitude to our advisor Prof. Dr. Gu for the continuous support of our study and research, for her patience, motivation, enthusiasm, and immense knowledge. Her guidance helped us in all the time of designing and implementing this project. We could not have imagined having a better advisor and mentor for our Course.

We are also thankful to all the faculty and staff members of the Department of Computer Science of New York Institute of Technology for their valuable time, support, comments, suggestions and persuasion. We would also like to thank the institute for providing the required facilities, Internet access and important books.

Priti Patel

Vaishnavi Bhambure

ABSTRACT

The Library Management System is gaining more importance as the number of its users are increasing rapidly. As the number is rising there is a need of effective management of library, one such effective system is our Library Management System it's designed using AngularJS, HTML and CSS as front end and Java application communicating with SQL as back-end.

The transactions like log in, register, add, search, issue is provided. The Library Management System stores the details like name, address, reader ID number, of users who come to library. The details of books like book name, book number, author, year of publication, the total number of books that are present in the library etc. are also stored.

Modules

Basically, there are two main modules, they are

1. Admin

2. Reader

Admin Module

The Admin module stores the details of books in the library, the details of library branch, updates the status of books, adds newly purchased book details to the database, issues the books, collects fine etc. The admin can add a book copy, search the book copy and check its status. The admin can add a new reader into the system and calculate it average fine paid per reader.

Reader Module

The readers have the functionalists like accessing the books, search for a book, borrow a book, reserve a book, return the book without fine if the book is returned in time and calculate the fine if the book is returned late.

The Library Management System provides exact information about the books that are borrowed and reserved by the users, the books that are available for borrowing or reserving and the books that are already borrowed, date of issue and the due date of return, etc and it even calculates the fine automatically, thereby reducing the risk present in the manual management of libraries.

1. Introduction:

There are two access levels in this library file system, which are ‘Administrator’ and ‘Reader’. The admin can be librarian which is managing all the books in the library or an IT professional at library managing operations on the library system.

Library management system offers many flexible and convenient features, allowing librarian/admin and library readers to maximize time and efficiency. Library System gives the all detailed information about readers and books. It will track how many books are available in the library branch and books borrowed or reserved by the reader. It will provide the total book count in a library branch. It keeps the record of the publishers and authors. Our software is customizing for any library requirement.

Features of library management system:

- Only basic knowledge of computers is required for operation of Library Management System. As it has user-friendly application interface.
- Library Management System is customizable and User Configurable.
- An inbuilt Settings module makes Library Management System flexibility to cater to diverse organizational needs.
- Library Management System brings information to the user’s desktop through integration across all modules.
  - Reader record is maintained
  - Book record is maintained
  - Automatic fine fees calculation
  - Keeps record of publisher’s and author’s
  - Configurable as per user’s requirements

Why do we need it ?

- Improved customer service through greater access to accurate information.
- Increased productivity and it eliminates duplication of effort.
- More economical and safer means of storing and keeping track of information.
- Easier access to Information like management reports and stock etc, as well as more accurate and faster results from statistical analyses.
- Reduces errors and eliminating the long and repetitive manual processing.
- Greater accountability and transparency in operations.
- Improved efficiency and effectiveness in administration and management as it has unprecedented access to real-time information.

2. ER data model design:

- List of Entities and their attributes:

ENTITIES	**ATTRIBUTES**
Reader	Reader_ID, Name, Phone, Address, Number_Of_Books, Card_Number
Book	Book_ID, ISBN, Title, Publisher, Publication_Date
Branch_Book_Copies	Book_ID, Library_ID, Num_Copy
Author	Author_ID, Author_Name
Publisher	Publisher_ID, Publisher_Name, Address
Branch	Library_ID, Library_location, Library_Name
Admin	Admin_ID, User name, Password
Branch_Book	Library_ID, Book_ID, Copy_ID, Status
Book_Reservation	Library_ID, Book_ID, Copy_ID, Reader_ID, Borrowed_Date_Time, Pickedup, Reader_Date_Time
Book_Author	Book_ID, Author_ID

- Properties of each attribute(key)

The properties of each attribute by the entity are shown in the below details:

1. Reader- reader_ID -> Primary Key, (Name, Phone, Address, Number_Of_Books, Card_Number) -> Simple attributes

2. Book - Book_ID -> Primary Key, Publisher_ID-> Foreign Key (ISBN, Title, Publisher, Publication_Date) -> Simple attributes

3. Branch_Book_Copies- {Book_ID, Library_ID} -> Primary Key, {Book_ID, Library_ID} -> Foreign key, (Num_Copy) -> Simple attribute

4. Author- Author_ID -> Primary Key, (Author_Name) -> Simple attributes

5. Publisher- Publisher_ID -> Primary Key, (Publisher_Name, Address) -> Simple attributes

6. Branch- Library_ID -> Primary Key, (Library_location, Library_Name) -> Simple attributes

7. Admin- Admin_ID -> Primary Key, (Username, Password) -> simple attribute

8. Branch_Book- {Library_ID, Book_ID, Copy_ID}-> Primary Key, {Library_ID, Book_ID}-> Foreign Key, Status->simple attribute

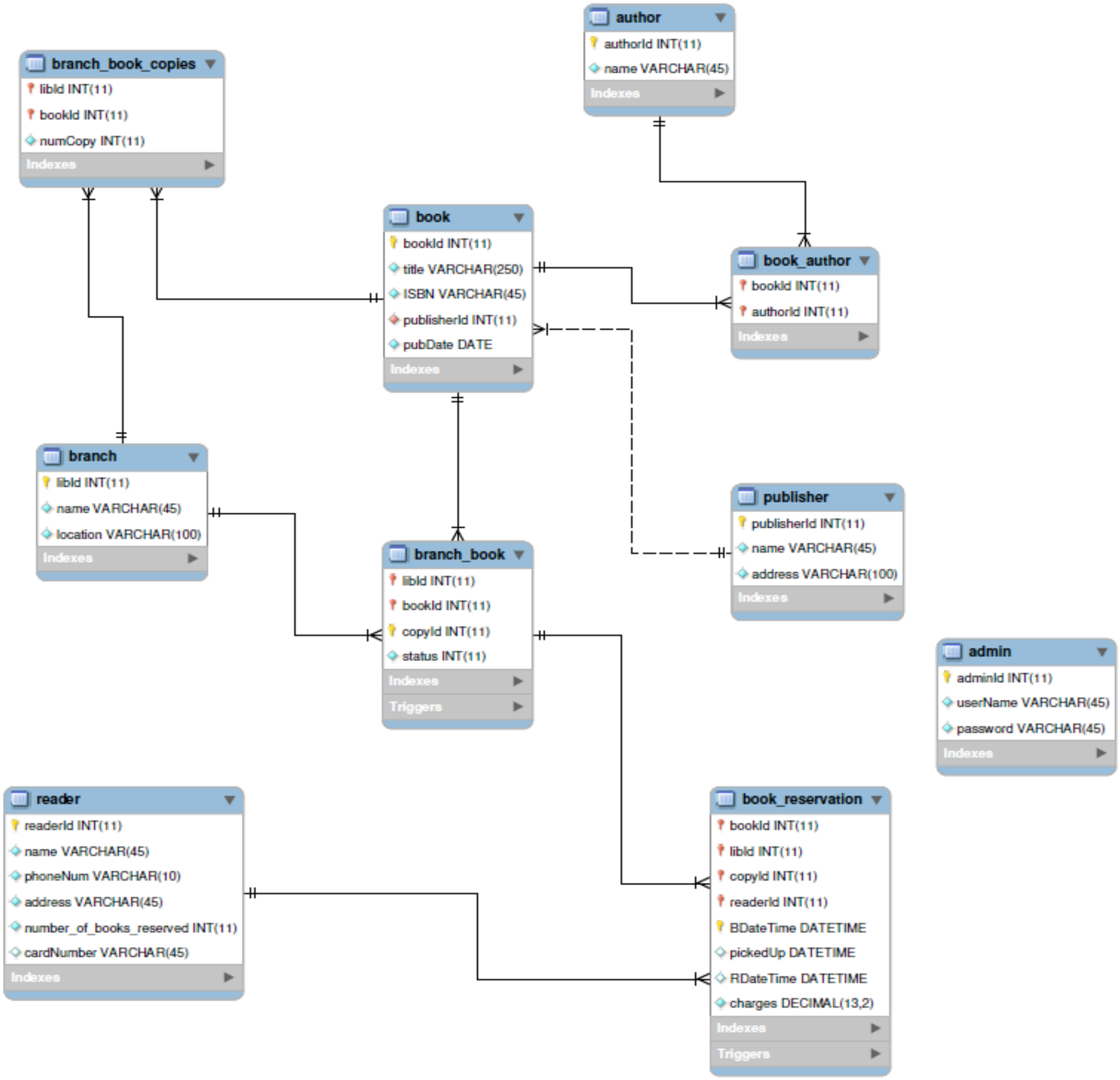
9. Book_Reservation- {Library_ID, Book_ID, Copy_ID, Reader_ID, B_Date_Time}-> Primary Key, {Library_ID, Book_ID, Copy_ID, Reader_ID}-> Foreign Key, (Pickedup, Reader_Date_Time)-> simple attributes

10. Book_Author- { Book_ID, Author_ID }->Primary Key, { Book_ID, Author_ID }-> Foreign Key

- Relationships and their attributes

Relationship	Its Attributes
Write	No attributes
Publish	No attributes
Borrow	Library_ID, Book_ID, Copy_ID, Reader_ID
Reserve	Library_ID, Book_ID, Copy_ID, Reader_ID
Return	Library_ID, Book_ID, Copy_ID, Reader_ID
Has	No attributes
Holds	No attributes
Maintain	No attributes
Handles	No attributes

Following is the ER (Entity Relation) diagram:



3. Logical Design of the Database

Step 1: Mapping of Regular Entities

- Reader (Reader_ID, Name, Phone, Address, Number_Of_Books, Card_Number)

  PK -> Reader_ID

- Book (Book_ID, ISBN, Title, Publisher, Publication_Date)

  PK -> Book_ID

- Branch (Library_ID, Library_location, Library_Name)

  PK -> Library_ID

- Author (Author_ID, Author_Name)

  PK -> Author_ID

- Publisher (Publisher_ID, Publisher_Name, Address)

  PK -> Publisher_ID

Step 2: Mapping of Weak entities

- Branch_Book_Copies(Book_ID, Library_ID, Num_Copy)

  PK->{Book_ID, Library_ID} ; FK -> {Book_ID, Library_ID}

Step 3: Mapping 1:1 Relation types

- No 1:1 relationship.

Step 4: Mapping 1:N Relation types

- Publisher(Publisher_ID, Publisher_Name, Book_ID, Address)

  PK-> Publisher_ID ; FK -> Book_ID

- Reader(Reader_ID, Name, Address, Phone, Book_ID, Library_ID)

  PK-> Reader_ID, FK -> {Library_ID, Book_ID}

Step 5: Mapping M:N Relation type

- Book_Author- (Book_ID, Author_ID)

  { Book_ID, Author_ID }->Primary Key, { Book_ID, Author_ID }-> Foreign Key
- Branch_Book_Copies- (Book_ID, Library_ID, Num_Copy)

  {Book_ID, Library_ID} -> Primary Key, {Book_ID, Library_ID} -> Foreign key,

Final Schema:

- Reader(Reader_ID, Name, Address, Phone, Book_ID, Library_ID)
- Author(Author_ID_ Author_Name)
- Publisher(Publisher_ID, Publisher_Name, Address)
- Branch_Book_Copies (Book_ID, Library_ID, Num_Copy)
- Book(Book_ID, ISBN, Title, Publisher, Publication_Date)
- Branch (Library_ID, Library_location, Library_Name)
- Book_Author- (Book_ID, Author_ID)
- Book_Reservation- (Library_ID, Book_ID, Copy_ID, Reader_ID, Borrowed_Date_Time, Pickedup, Reader_Date_Time)
- Branch_Book- (Library_ID, Book_ID, Copy_ID, Status)

Entity	**Primary/Candidate Key**	**Foreign Key**
Reader	Reader_ID	-
Author	Author_ID	
Publisher	Publisher_ID	
Branch_Book_Copies	{Book_ID, Library_ID}	{Book_ID, Library_ID}
Book	Book_ID	---
Branch	Library_ID	---
Book_Author	{Author_ID, Book_ID}	{Author_ID, Book_ID}

• Specify the entity and referential integrity constraints

4. Relational Database Design

- ✧ First Normal Form (1NF)

The relations are already in first normal form as all the attributes are atomic. Thus, there is no need of flattening or decomposing the tables.

- ✧ Second Normal Form (2NF)

The relations are in second normal form if are all non primary attributes are fully dependent on the primary key.

ENTITIES	**ATTRIBUTES**
Reader	Reader_ID, Name, Phone, Address, Number_Of_Books, Card_Number
Book	Book_ID, ISBN, Title, Publisher, Publication_Date
Author	Author_ID, Author_Name
Publisher	Publisher_ID, Publisher_Name, Address
Branch	Library_ID, Library_location, Library_Name
Admin	Admin_ID, User name, Password
Book_Reservation	Library_ID, Book_ID, Copy_ID, Reader_ID, Borrowed_Date_Time, Pickedup, Reader_Date_Time

- o Third Normal Form (3NF)

ENTITIES	**ATTRIBUTES**
Reader	Reader_ID, Name, Phone, Address, Number_Of_Books, Card_Number
Book	Book_ID, ISBN, Title, Publisher, Publication_Date
Branch_Book_Copies	Book_ID, Library_ID, Num_Copy
Author	Author_ID, Author_Name
Publisher	Publisher_ID, Publisher_Name, Address
Branch	Library_ID, Library_location, Library_Name
Admin	Admin_ID, User name, Password
Branch_Book	Library_ID, Book_ID, Copy_ID, Status
Book_Reservation	Library_ID, Book_ID, Copy_ID, Reader_ID, Borrowed_Date_Time, Pickedup, Reader_Date_Time
Book_Author	Book_ID, Author_ID

- BCNF
  All the relations are already in BCNF.

- Final Normalized Relations:

ENTITIES	**ATTRIBUTES**
Reader	Reader_ID, Name, Phone, Address, Number_Of_Books, Card_Number
Book	Book_ID, ISBN, Title, Publisher, Publication_Date
Branch_Book_Copies	Book_ID, Library_ID, Num_Copy
Author	Author_ID, Author_Name
Publisher	Publisher_ID, Publisher_Name, Address
Branch	Library_ID, Library_location, Library_Name
Admin	Admin_ID, User name, Password
Branch_Book	Library_ID, Book_ID, Copy_ID, Status
Book_Reservation	Library_ID, Book_ID, Copy_ID, Reader_ID, Borrowed_Date_Time, Pickedup, Reader_Date_Time
Book_Author	Book_ID, Author_ID

5. Implementation of Database and SQL Query

- A description of the creation of the database schema and instance.

Database:

- All the tables from 3NF are designed in Microsoft SQL Server.
- Primary Keys and Foreign Keys are assigned to each table.

Database Queries

```
CREATE TABLE `admin` (
  `adminId` int(11) NOT NULL AUTO_INCREMENT,
  `userName` varchar(45) NOT NULL,
  `password` varchar(45) NOT NULL,
  PRIMARY KEY (`adminId`),
  UNIQUE KEY `userName_UNIQUE` (`userName`)
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=latin1;

CREATE TABLE `author` (
  `authorId` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  PRIMARY KEY (`authorId`)
) ENGINE=InnoDB AUTO_INCREMENT=154 DEFAULT CHARSET=latin1;
```

```
CREATE TABLE `book` (
  `bookId` int(11) NOT NULL AUTO_INCREMENT,
  `title` varchar(250) NOT NULL,
  `ISBN` varchar(45) NOT NULL,
  `publisherId` int(11) NOT NULL,
  `pubDate` date NOT NULL,
  PRIMARY KEY (`bookId`),
  UNIQUE KEY `ISBN_UNIQUE` (`ISBN`),
  KEY `fk_book_1_idx` (`publisherId`),
  CONSTRAINT `fk_book_1` FOREIGN KEY (`publisherId`) REFERENCES `publisher`
(`publisherId`) ON DELETE NO ACTION ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=383 DEFAULT CHARSET=latin1;

CREATE TABLE `book_author` (
  `bookId` int(11) NOT NULL,
  `authorId` int(11) NOT NULL,
  PRIMARY KEY (`bookId`,`authorId`),
  KEY `fk_book_author_2_idx` (`authorId`),
  CONSTRAINT `fk_book_author_1` FOREIGN KEY (`bookId`) REFERENCES `book`
(`bookId`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_book_author_2` FOREIGN KEY (`authorId`) REFERENCES `author`
(`authorId`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `book_reservation` (
  `bookId` int(11) NOT NULL,
  `libId` int(11) NOT NULL,
  `copyId` int(11) NOT NULL,
  `readerId` int(11) NOT NULL,
  `BDateTime` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `pickedUp` datetime DEFAULT NULL,
  `RDateTime` datetime DEFAULT NULL,
  `charges` decimal(13,2) NOT NULL DEFAULT '0.00',
  PRIMARY KEY (`bookId`,`libId`,`copyId`,`readerId`,`BDateTime`),
  KEY `fk_book_reservation_2_idx` (`readerId`),
  CONSTRAINT `fk_book_reservation_1` FOREIGN KEY (`bookId`, `libId`, `copyId`)
REFERENCES `branch_book` (`bookId`, `libId`, `copyId`) ON DELETE NO ACTION ON
UPDATE NO ACTION,
  CONSTRAINT `fk_book_reservation_2` FOREIGN KEY (`readerId`) REFERENCES `reader`
(`readerId`) ON DELETE NO ACTION ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `branch` (
  `libId` int(11) NOT NULL,
  `name` varchar(45) NOT NULL,
  `location` varchar(100) NOT NULL,
   PRIMARY KEY (`libId`)
 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

 CREATE TABLE `branch_book` (
   `libId` int(11) NOT NULL,
   `bookId` int(11) NOT NULL,
   `copyId` int(11) NOT NULL,
   `status` int(11) NOT NULL DEFAULT '111',
   PRIMARY KEY (`libId`,`bookId`,`copyId`),
   KEY `fk_branch_book_1_idx` (`bookId`),
   CONSTRAINT `fk_branch_book_1` FOREIGN KEY (`bookId`) REFERENCES `book`
 (`bookId`) ON DELETE CASCADE ON UPDATE CASCADE,
   CONSTRAINT `fk_branch_book_2` FOREIGN KEY (`libId`) REFERENCES `branch` (`libId`)
 ON DELETE CASCADE ON UPDATE CASCADE
 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `branch_book_copies` (
  `libId` int(11) NOT NULL,
  `bookId` int(11) NOT NULL,
  `numCopy` int(11) NOT NULL,
  PRIMARY KEY (`libId`,`bookId`),
  KEY `fk_branch_book_2_1_idx` (`bookId`),
  CONSTRAINT `fk_branch_book_2_1` FOREIGN KEY (`bookId`) REFERENCES `book`
(`bookId`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `fk_branch_book_2_2` FOREIGN KEY (`libId`) REFERENCES `branch`
(`libId`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `publisher` (
  `publisherId` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `address` varchar(100) NOT NULL,
  PRIMARY KEY (`publisherId`)
) ENGINE=InnoDB AUTO_INCREMENT=52 DEFAULT CHARSET=latin1;

CREATE TABLE `reader` (
  `readerId` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(45) NOT NULL,
  `phoneNum` varchar(10) NOT NULL,
  `address` varchar(45) NOT NULL,
  `number_of_books_reserved` int(11) NOT NULL DEFAULT '0',
  `cardNumber` varchar(45) DEFAULT NULL,
  PRIMARY KEY (`readerId`),
  UNIQUE KEY `cardNumber_UNIQUE` (`cardNumber`)
) ENGINE=InnoDB AUTO_INCREMENT=207 DEFAULT CHARSET=latin1;
```

Database Triggers

1. **To maintain number of book copies in each branch**

```
TRIGGER `libsys`.`branch_book_after_insert`
AFTER INSERT ON `libsys`.`branch_book`
FOR EACH ROW
BEGIN
DECLARE numCopyV int;
   SELECT count(copyId) FROM branch_book AS BB WHERE BB.libId = NEW.libId AND
BB.bookId = NEW.bookId INTO numCopyV;
DELETE FROM branch_book_copies WHERE bookId = NEW.bookId AND libId = NEW.libId;
   INSERT INTO branch_book_copies
  ( bookId,
    libId,
    numCopy)
  VALUES
  ( NEW.bookId,
    NEW.libId,
    numCopyV );
END$$
```

2. **To generate and maintain unique copy id of each book at a library location**

```
TRIGGER `libsys`.`branch_book_before_insert`
BEFORE INSERT ON `libsys`.`branch_book`
FOR EACH ROW
BEGIN
DECLARE copyId int;
   SET copyId = 0;
   SELECT BB.copyId FROM branch_book AS BB WHERE BB.libId = NEW.libId AND
BB.bookId = NEW.bookId ORDER BY BB.copyId DESC LIMIT 1 INTO copyId;
   SET copyId = copyId + 1 ;
   SET NEW.copyId = copyId;
END$$
```

3. **After book is reserved it’s status is changed to ‘222’ which means reserved**

```
TRIGGER `libsys`.`book_reservation_AFTER_INSERT`
AFTER INSERT ON `libsys`.`book_reservation`
FOR EACH ROW
BEGIN
UPDATE branch_book SET status = 222 WHERE bookId = NEW.bookId AND NEW.libId =
libId AND NEW.copyId = copyId;
END$$
```

4. **To calculate the fine**

```
TRIGGER `libsys`.`book_reservation_BEFORE_update`
BEFORE UPDATE ON `libsys`.`book_reservation`
FOR EACH ROW
BEGIN
DECLARE numDays int;

IF OLD.RDateTime IS NULL AND NEW.RDateTime IS NOT NULL THEN
UPDATE branch_book SET status = 111 WHERE NEW.bookId = bookId AND NEW.libId =
libId AND NEW.copyId = copyId;
SET numDays = 0;
SELECT TIMESTAMPDIFF(DAY, NEW.BDateTime, NEW.RDateTime) INTO numDays;
IF numDays>20 THEN
SET NEW.charges = ((numDays-20) * 20.00) / 100.00;
END IF;
ELSE
UPDATE branch_book SET status = 333 WHERE NEW.bookId = bookId AND NEW.libId =
libId AND NEW.copyId = copyId;
END IF;
END$$
```

5. **Reader can’t reserve same book title and copy more than once if not returned. Reader can’t have more than 10 books reserved at once**

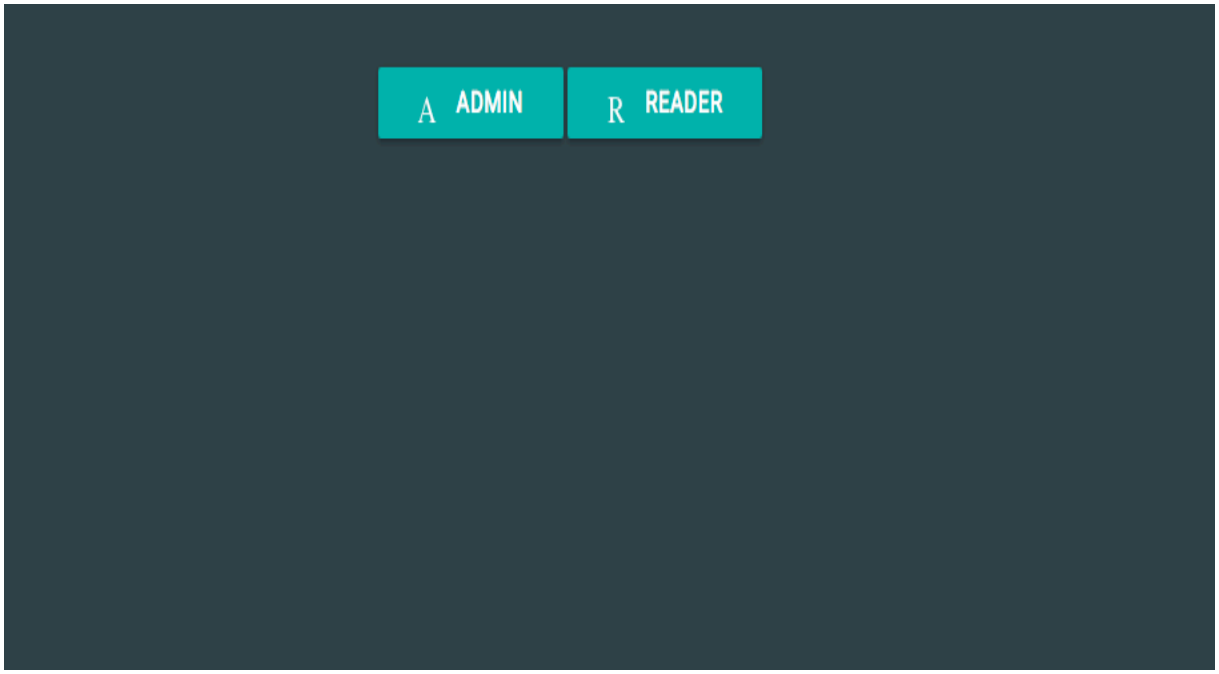
```
TRIGGER `libsys`.`book_servation_before_insert`
BEFORE INSERT ON `libsys`.`book_reservation`
FOR EACH ROW
BEGIN
DECLARE numR int;
   SET numR = 0;
SELECT count(*) FROM book_reservation AS BR WHERE BR.bookId = NEW.bookID AND
BR.libId = NEW.libID AND BR.copyId = NEW.copyId AND BR.readerId = NEW.readerId AND
BR.RDateTime IS NULL LIMIT 1 INTO numR;
   IF numR>0 THEN
signal sqlstate ‘45000’ set message_text = ‘You can’t reserve same book title and copy more than
once if not returned’;
   END IF;
   SET numR = 0;
SELECT count(*) FROM book_reservation AS BR WHERE BR.bookId = NEW.bookID AND
BR.libId = NEW.libID AND BR.copyId = NEW.copyId AND BR.RdateTime IS NULL LIMIT 1
INTO numR;
   IF numR>0 THEN
signal sqlstate ‘45001’ set message_text = ‘This book is reserved’;
   END IF;
   SET numR = 0;
```

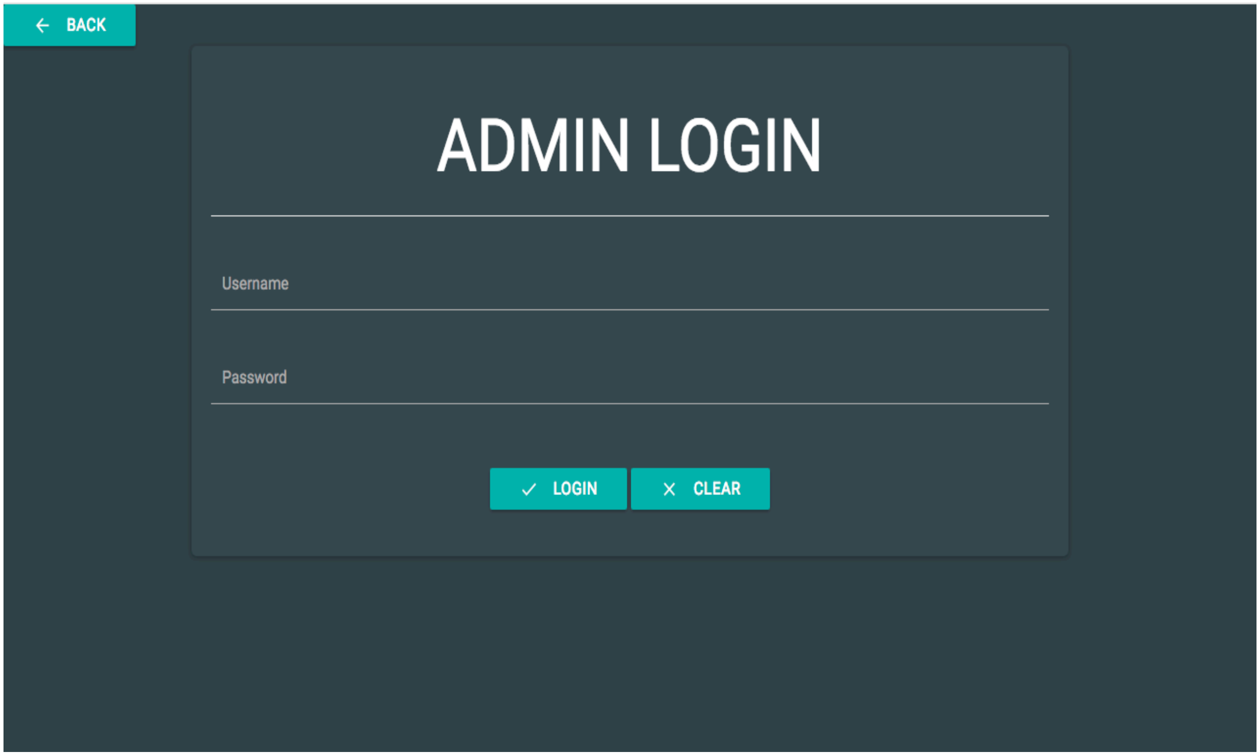
```
SELECT count(*) FROM book_reservation AS BR WHERE BR.readerId = NEW.readerId AND
BR.RdateTime IS NULL LIMIT 10 INTO numR;
   IF numR>9 THEN
Signal sqlstate ‘45002’ set message_text = ‘You can’t have more than 10 book at same time';
   END IF;
END$$
```

6. Application Design AND Snapshot of Application

- A description of the application programs and all tasks.

Application Program and all tasks:

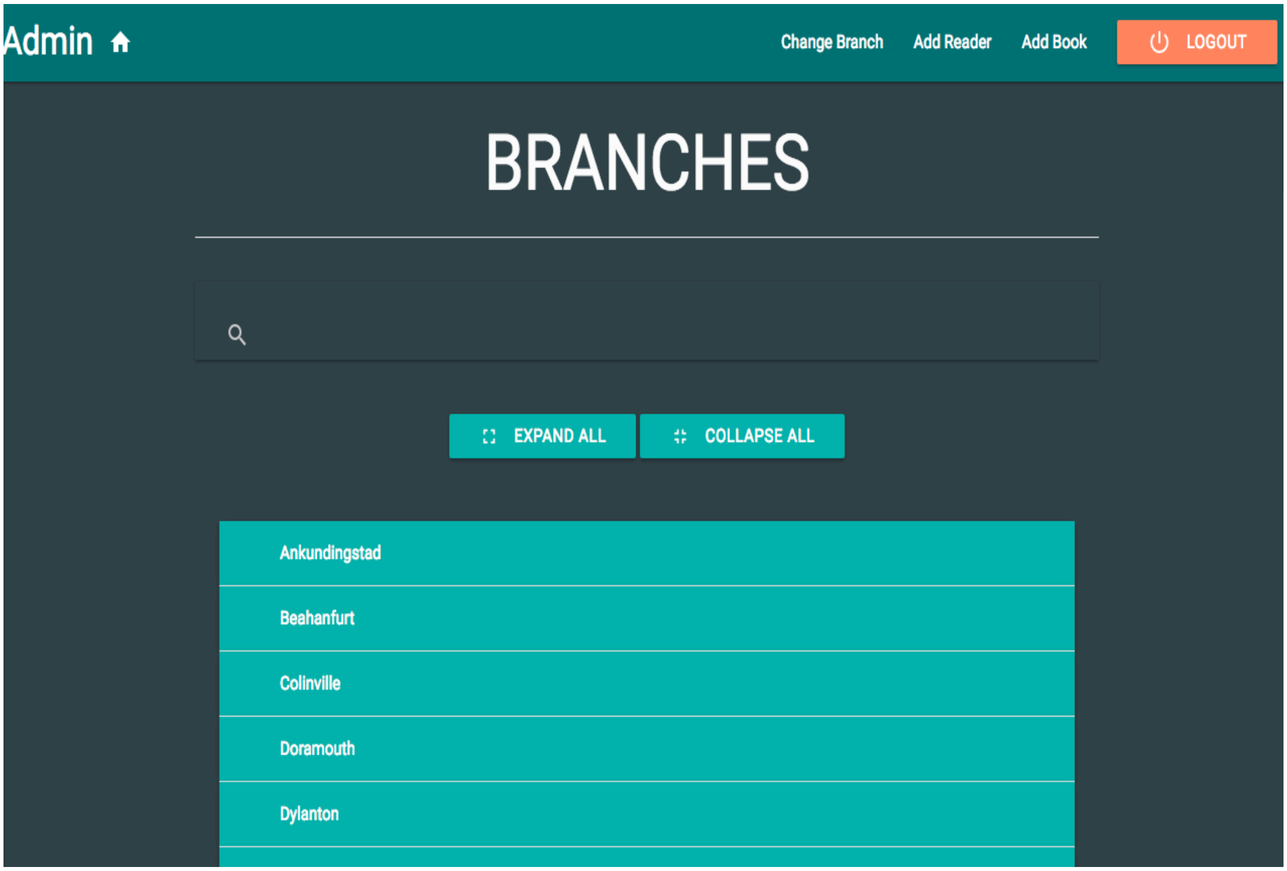




ADMIN LOGIN

Username

Password



BRANCHES

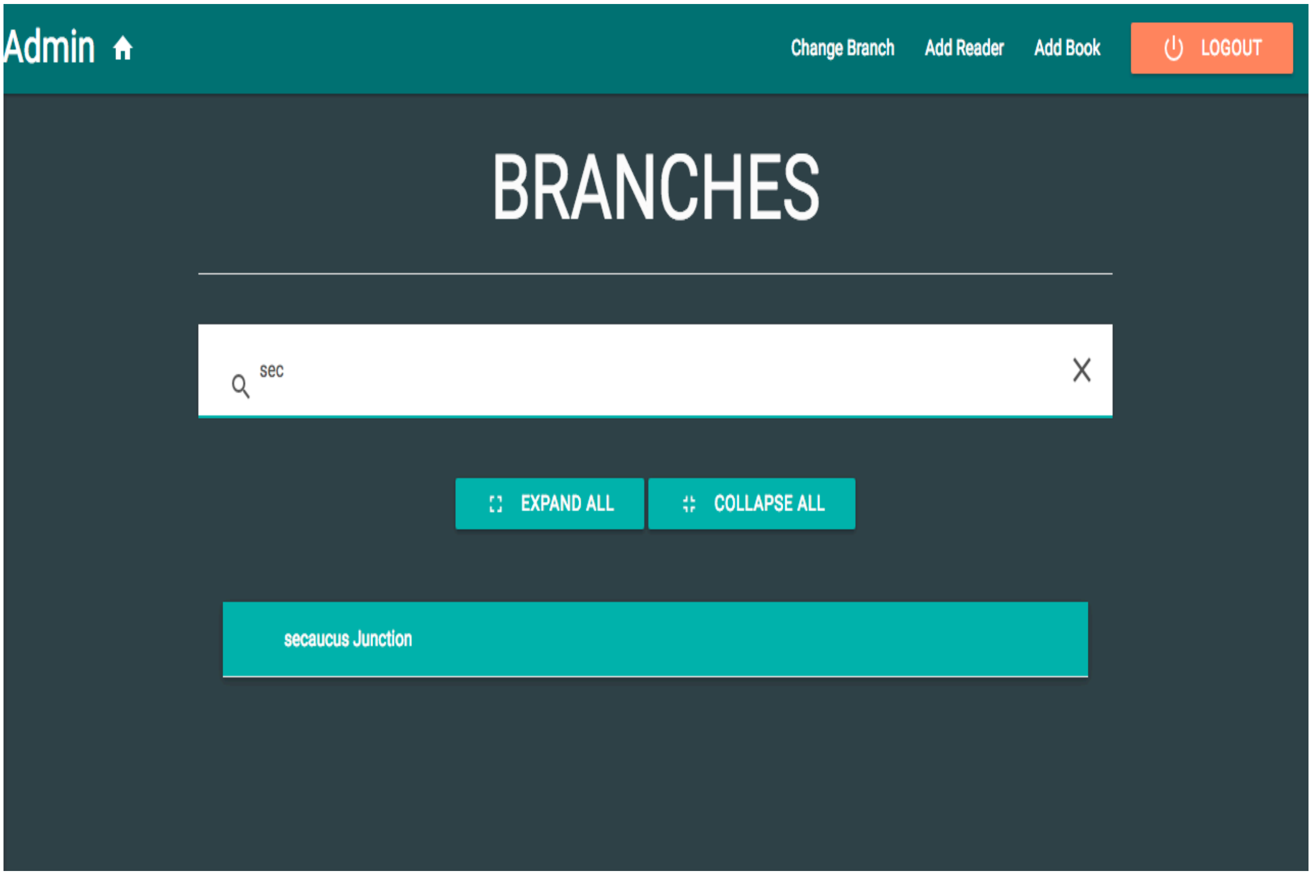
Ankundingstad

Beahanfurt

Colinville

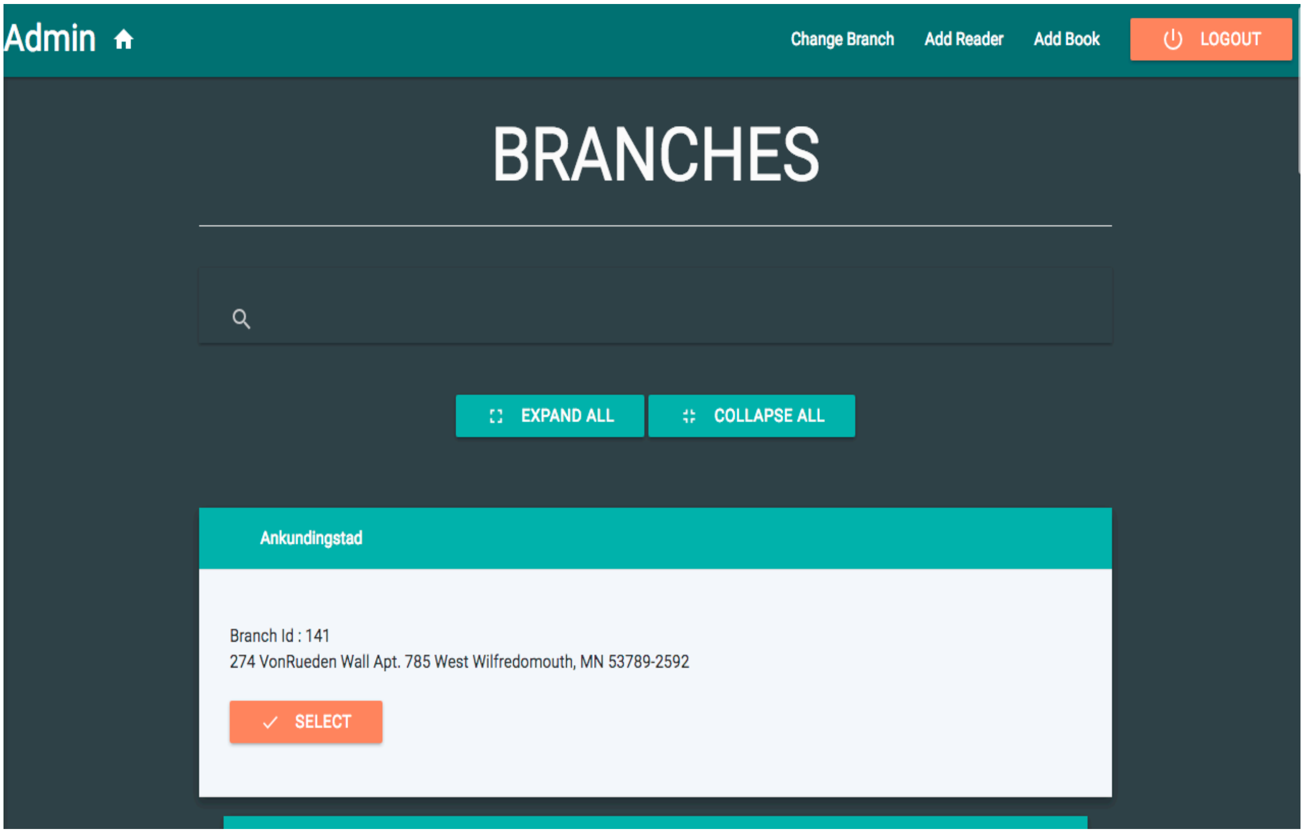
Doramouth

Dylanton



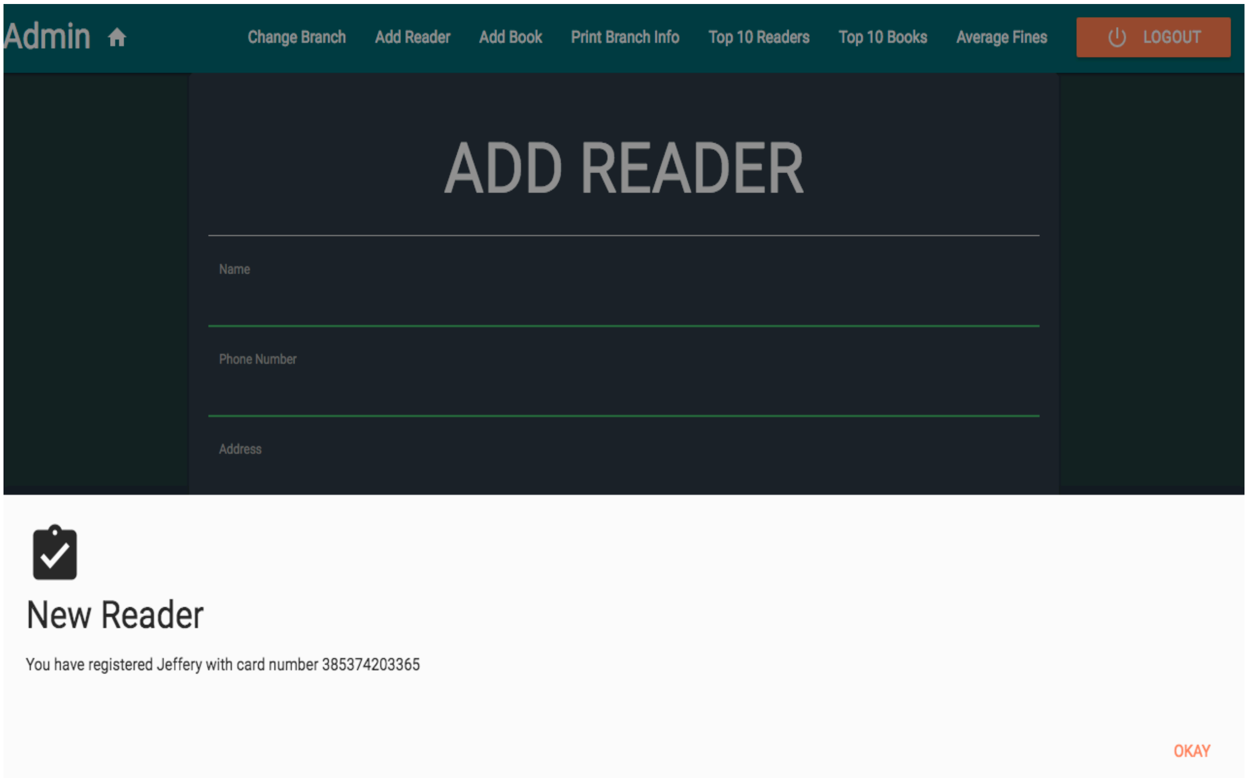
sec

secaucus Junction



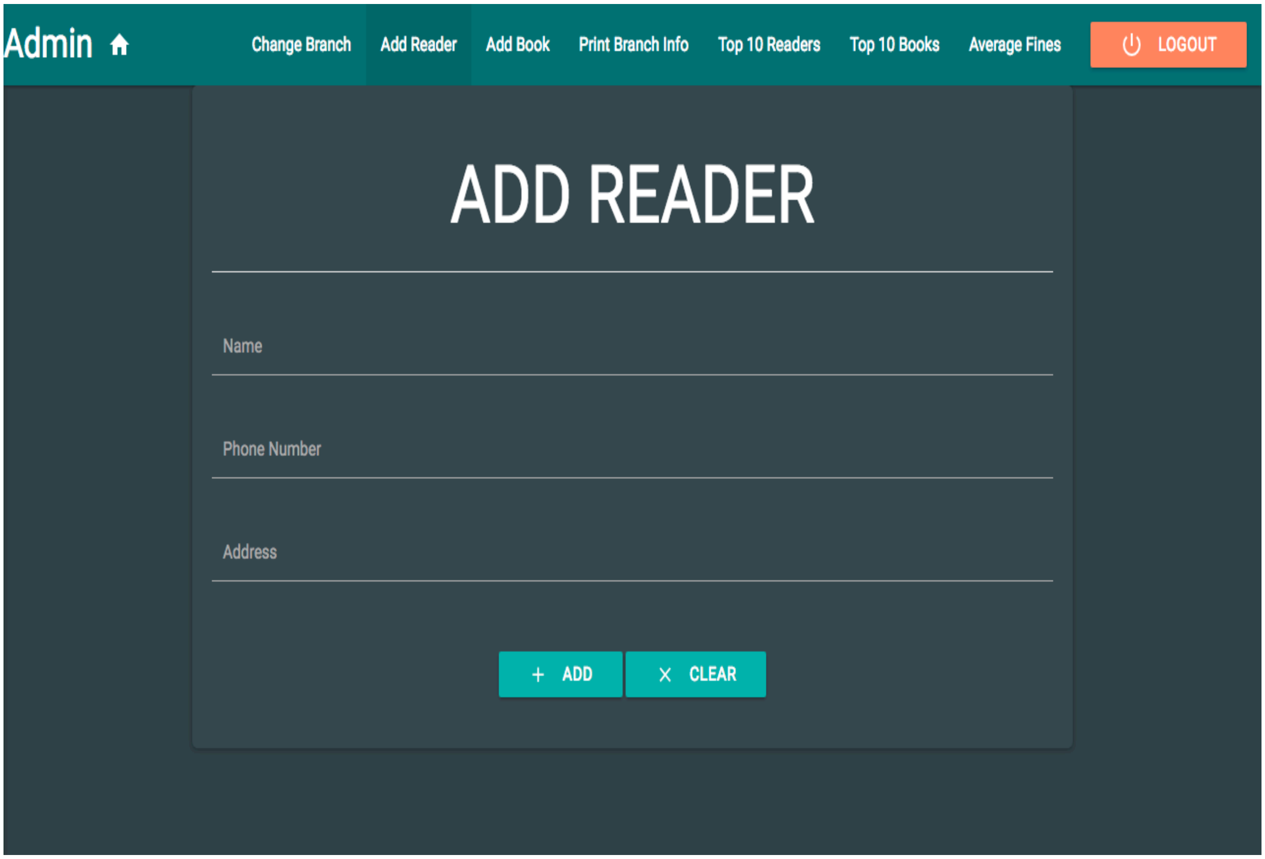
Ankundingstad

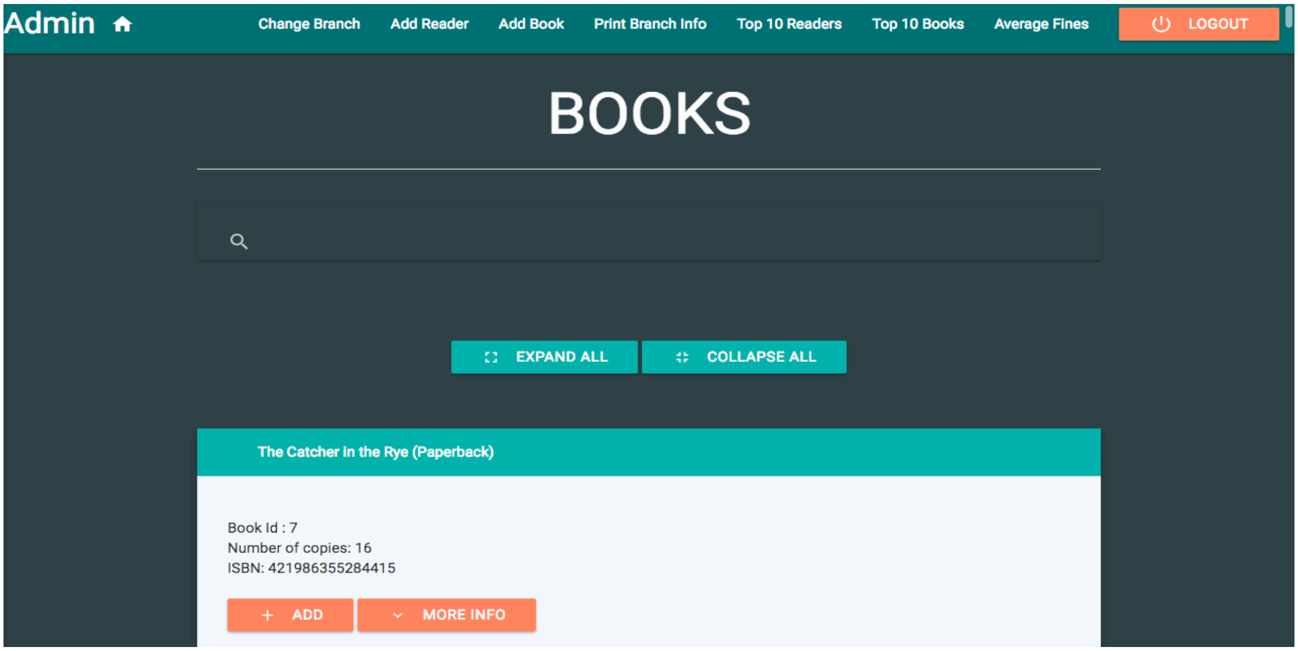
Branch Id : 141
274 VonRueden Wall Apt. 785 West Wilfredomouth, MN 53789-2592



New Reader

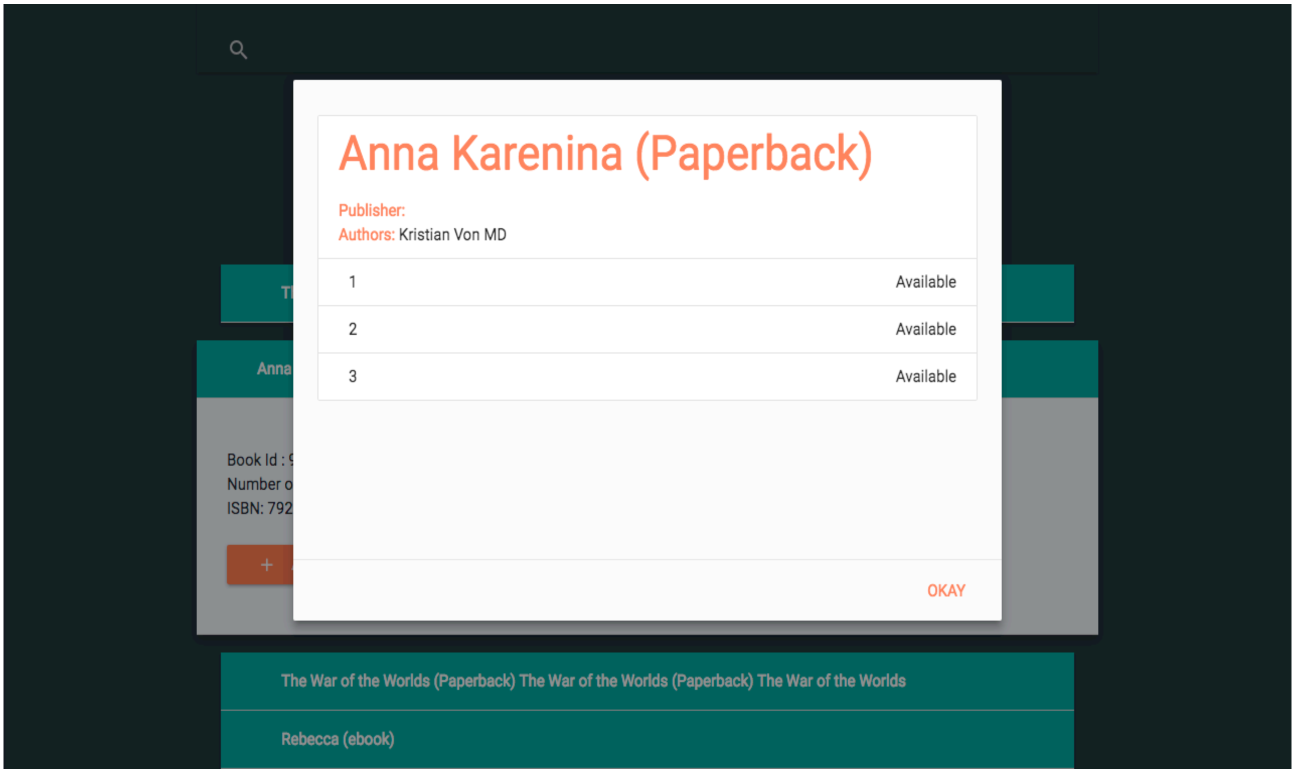
You have registered Jeffery with card number 385374203365





The Catcher in the Rye (Paperback)

Book Id : 7
Number of copies: 16
ISBN: 421986355284415



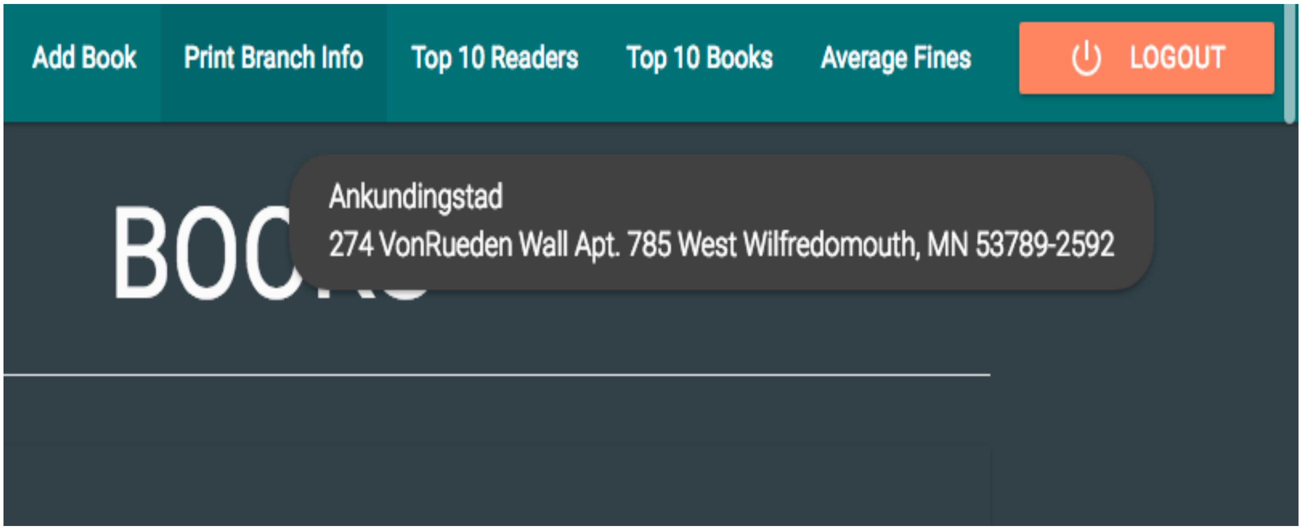
Anna Karenina (Paperback)

Publisher:
Authors: Kristian Von MD

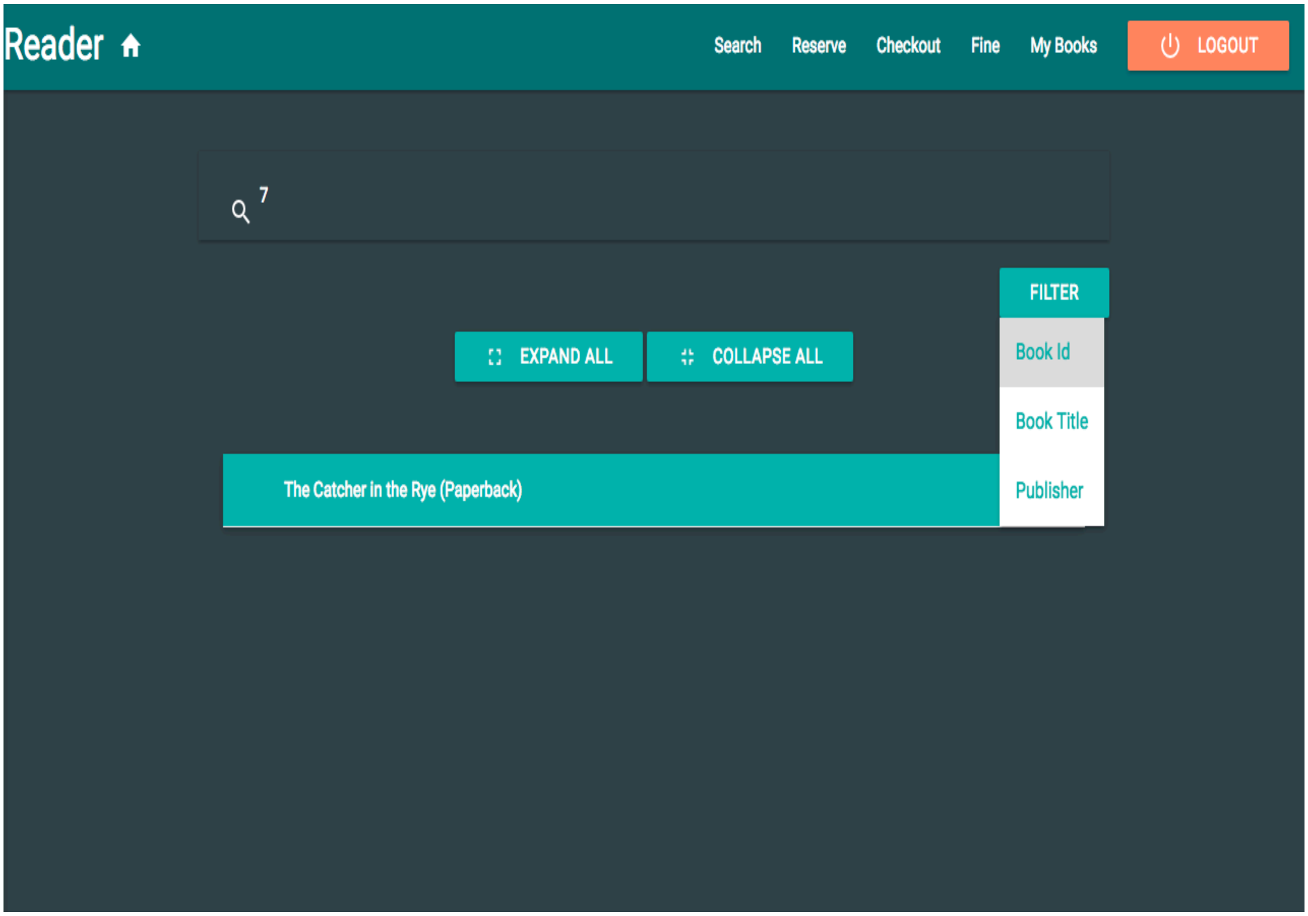
1	Available
2	Available
3	Available

The War of the Worlds (Paperback) The War of the Worlds (Paperback) The War of the Worlds

Rebecca (ebook)

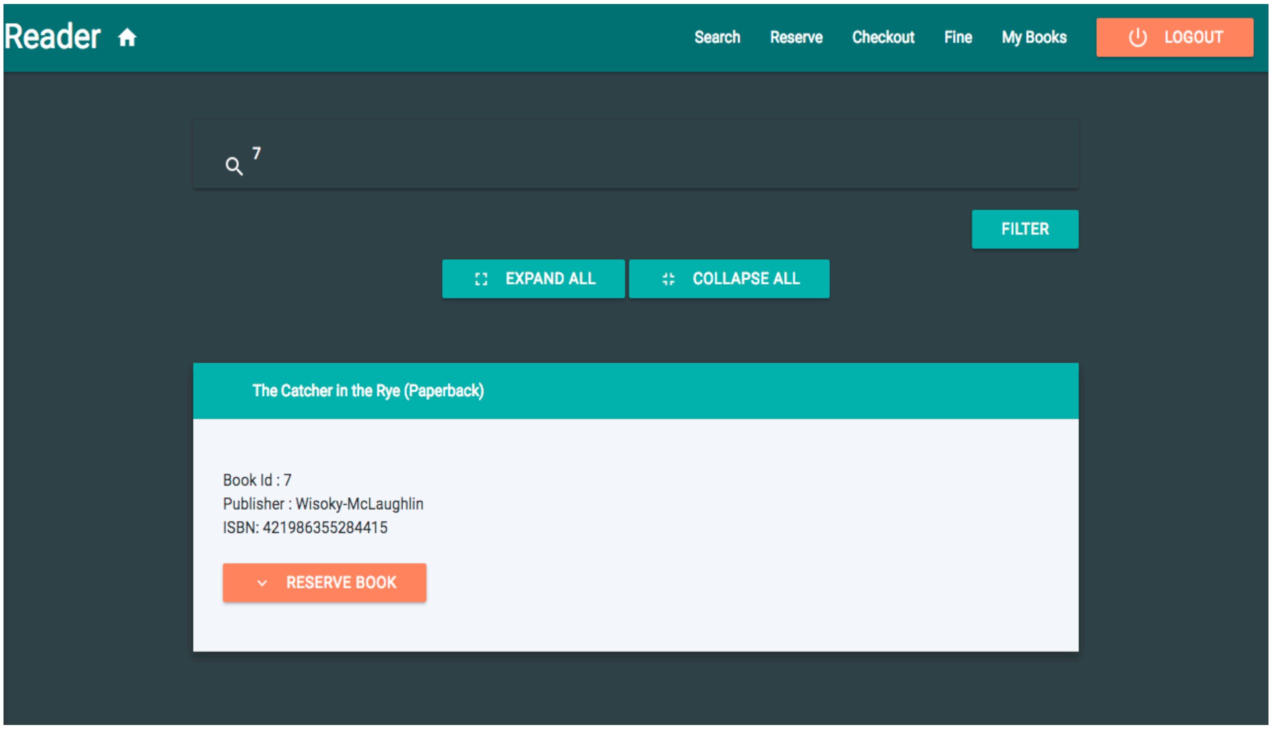


Ankundingstad
274 VonRueden Wall Apt. 785 West Wilfredomouth, MN 53789-2592



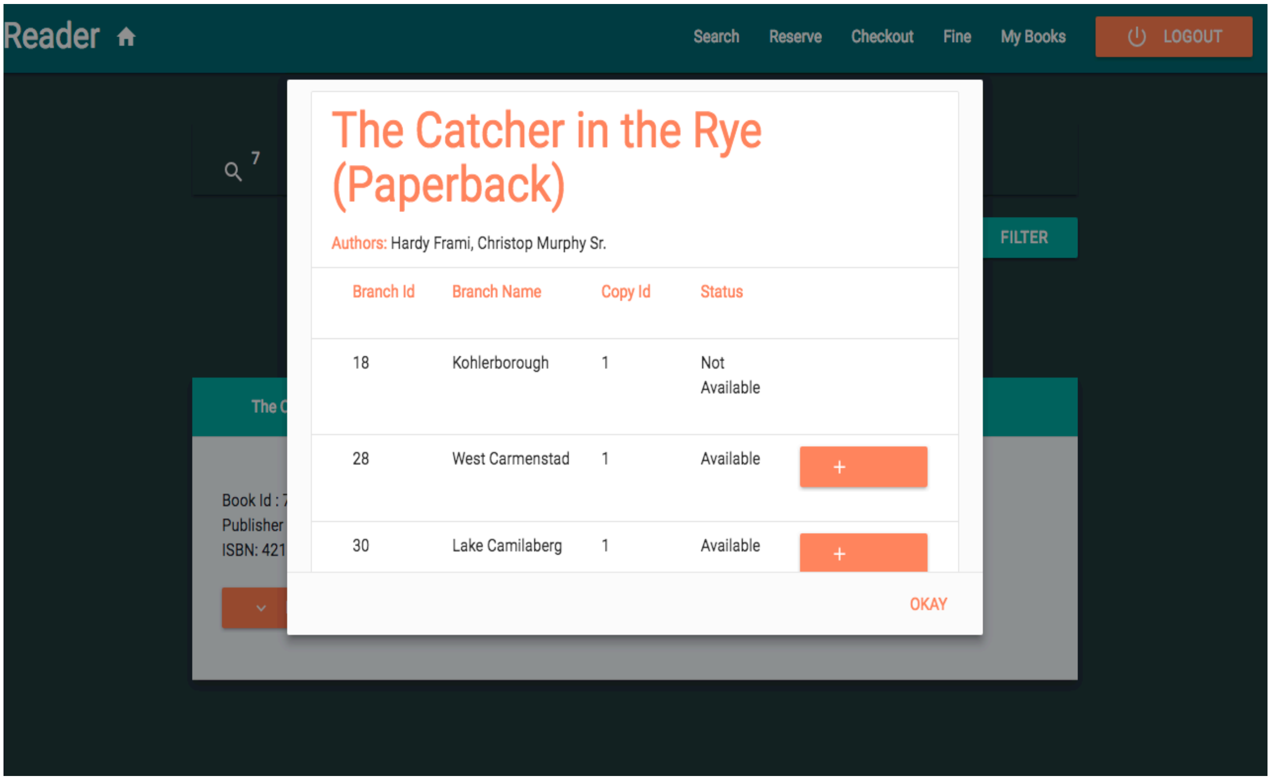
7

The Catcher in the Rye (Paperback)



The Catcher in the Rye (Paperback)

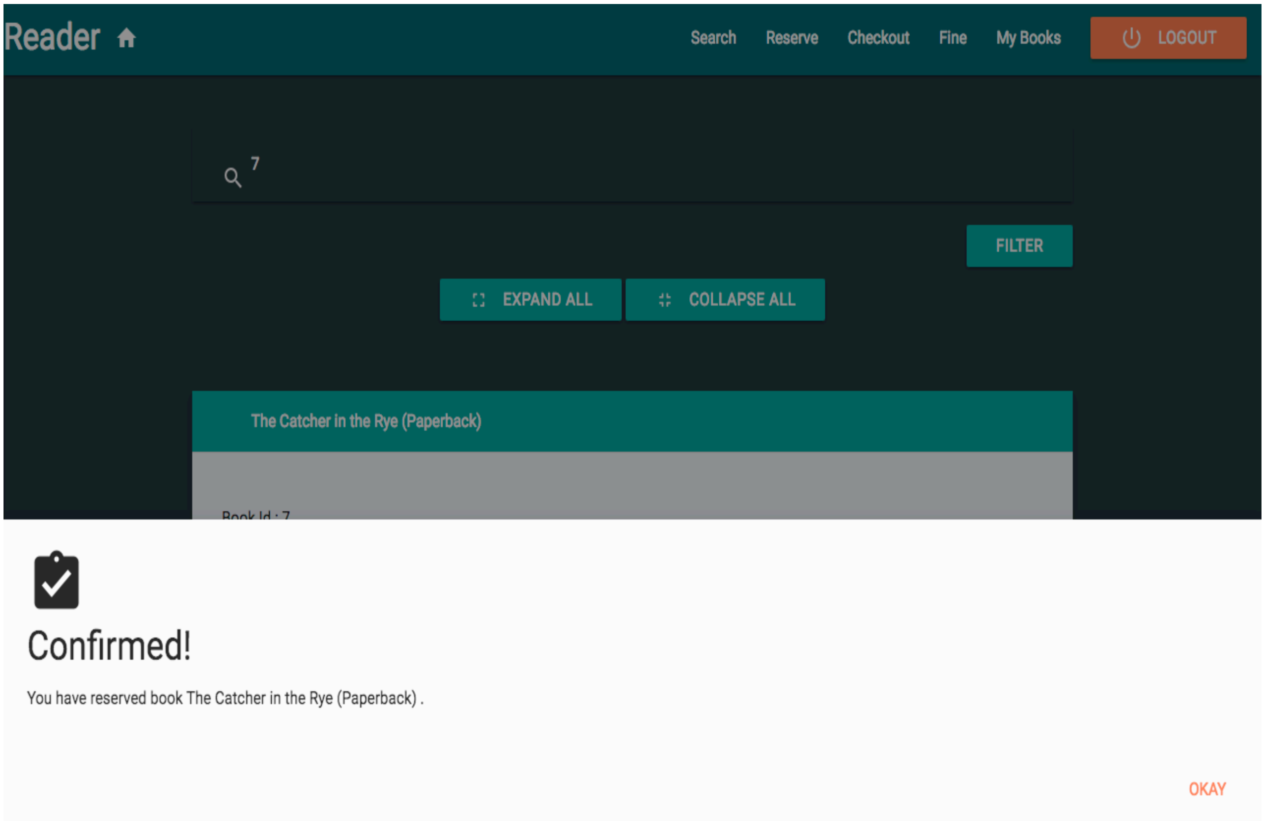
Book Id : 7
Publisher : Wisoky-McLaughlin
ISBN: 421986355284415



The Catcher in the Rye (Paperback)

Authors: Hardy Frami, Christop Murphy Sr.

Branch Id	Branch Name	Copy Id	Status	
18	Kohlerborough	1	Not Available	
28	West Carmenstad	1	Available	+
30	Lake Camilaberg	1	Available	+

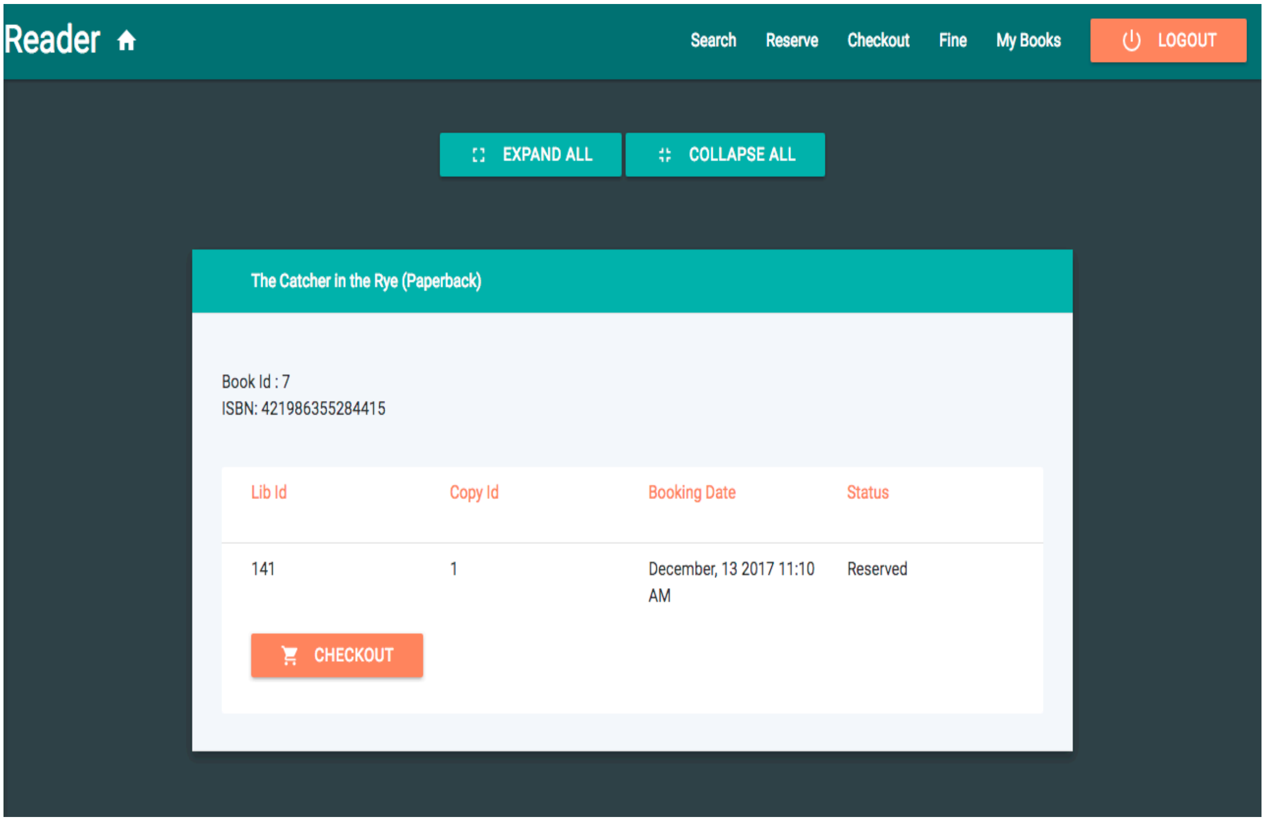


The Catcher in the Rye (Paperback)

Book Id : 7

Confirmed!

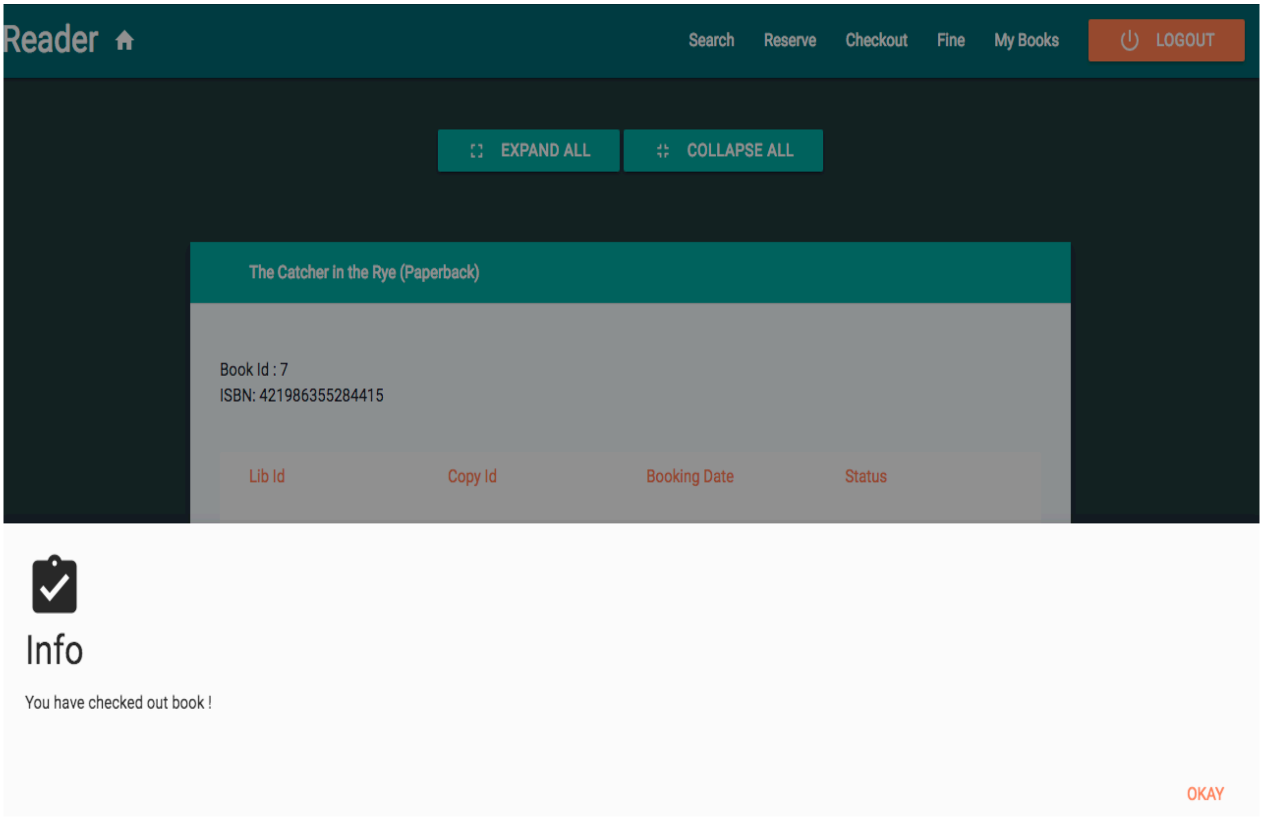
You have reserved book The Catcher in the Rye (Paperback) .



The Catcher in the Rye (Paperback)

Book Id : 7
ISBN: 421986355284415

Lib Id	Copy Id	Booking Date	Status
141	1	December, 13 2017 11:10 AM	Reserved



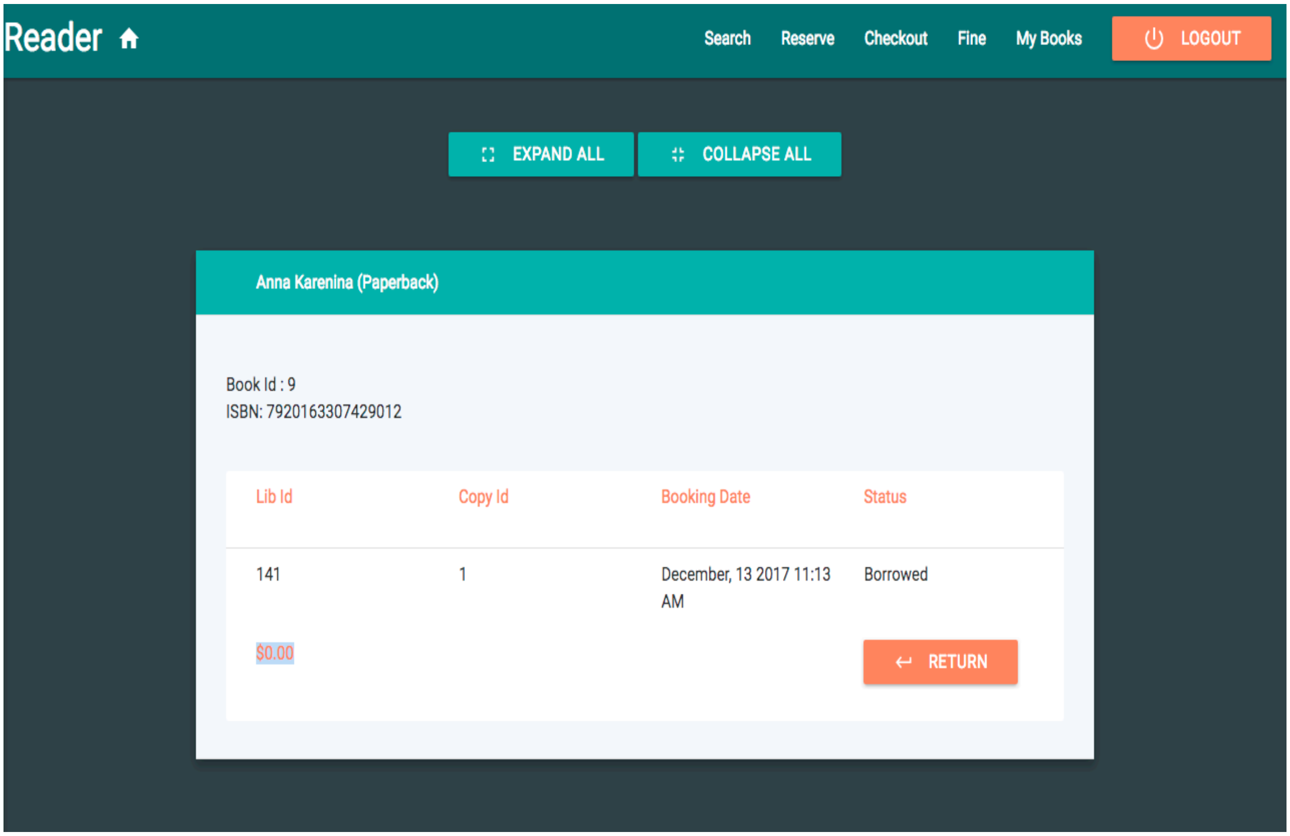
The Catcher in the Rye (Paperback)

Book Id : 7
ISBN: 421986355284415

Lib Id	Copy Id	Booking Date	Status

Info

You have checked out book !

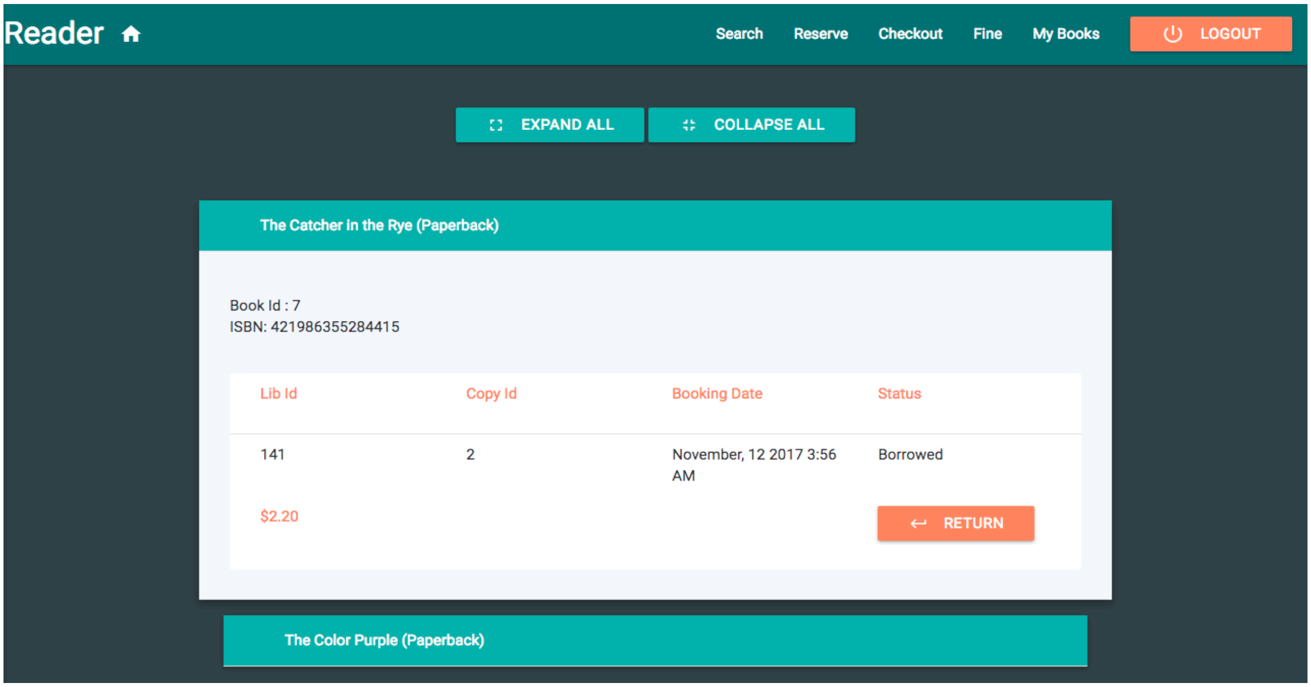


Anna Karenina (Paperback)

Book Id : 9
ISBN: 7920163307429012

Lib Id	Copy Id	Booking Date	Status
141	1	December, 13 2017 11:13 AM	Borrowed

$0.00

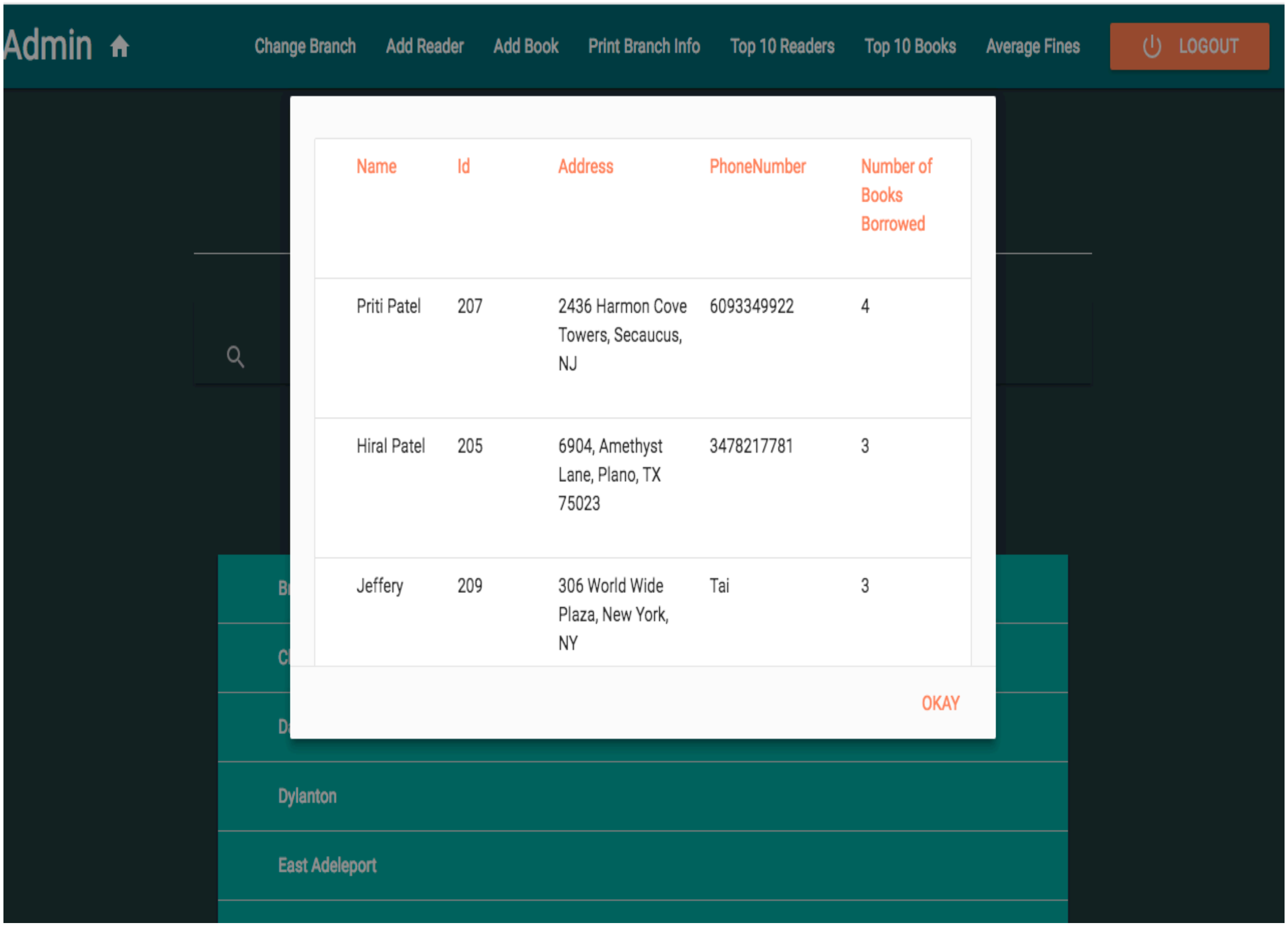


The Catcher in the Rye (Paperback)

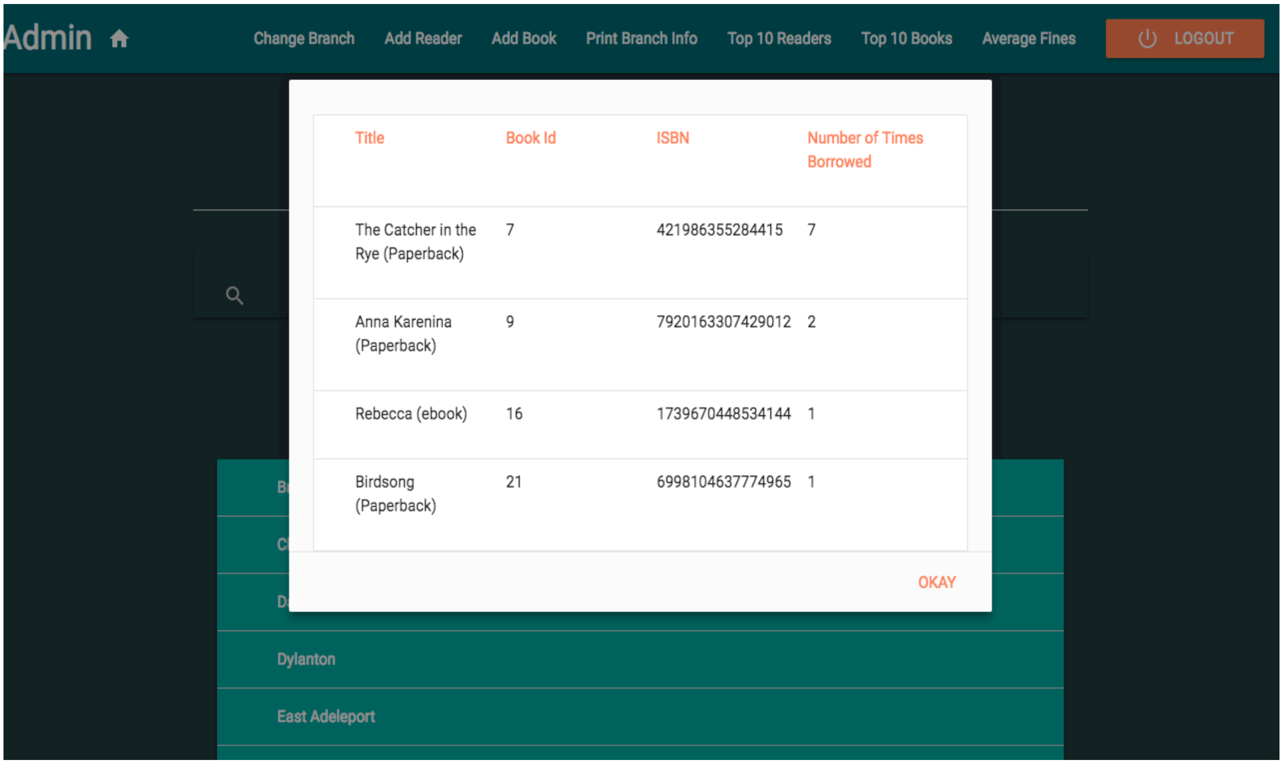
Book Id : 7
ISBN: 421986355284415

Lib Id	Copy Id	Booking Date	Status
141	2	November, 12 2017 3:56 AM	Borrowed
$2.20			RETURN

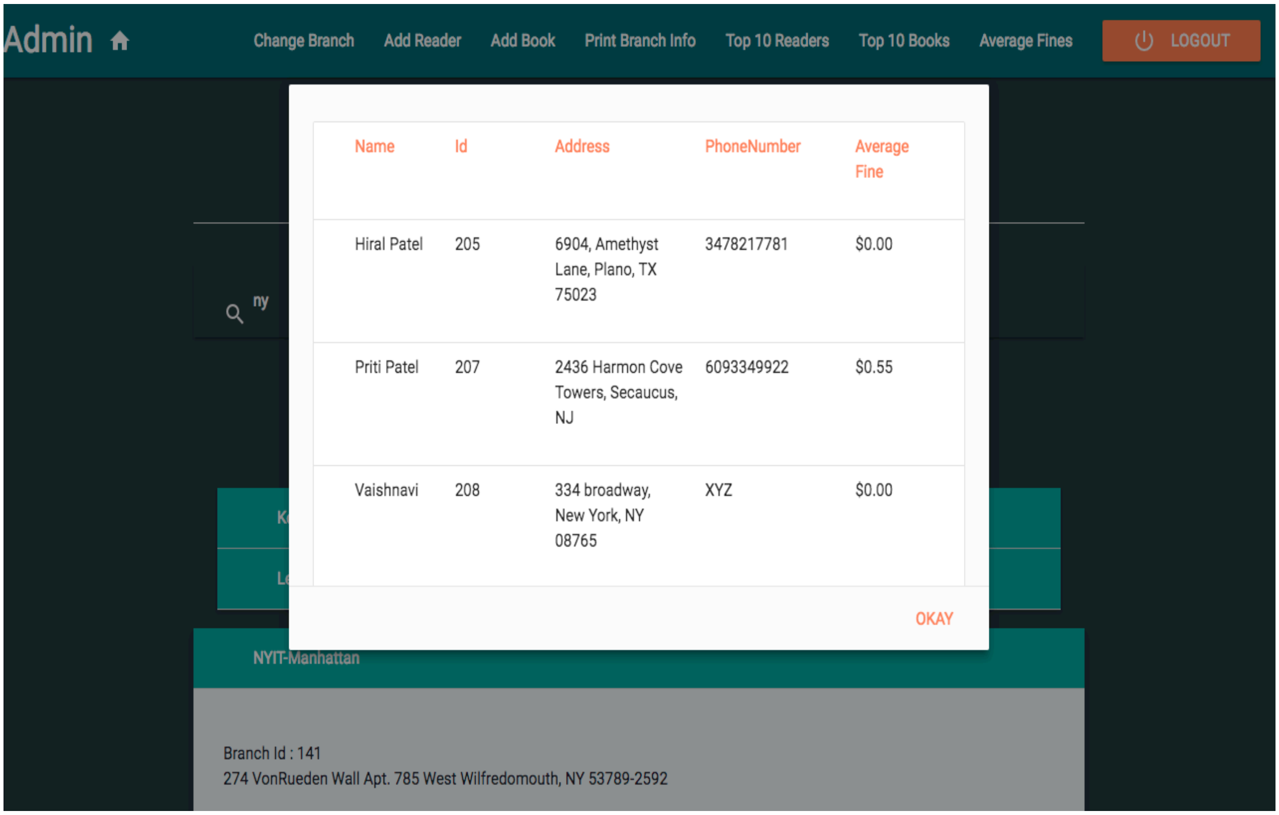
The Color Purple (Paperback)



Name	Id	Address	PhoneNumber	Number of Books Borrowed
Priti Patel	207	2436 Harmon Cove Towers, Secaucus, NJ	6093349922	4
Hiral Patel	205	6904, Amethyst Lane, Plano, TX 75023	3478217781	3
Jeffery	209	306 World Wide Plaza, New York, NY	Tai	3



Title	Book Id	ISBN	Number of Times Borrowed
The Catcher in the Rye (Paperback)	7	421986355284415	7
Anna Karenina (Paperback)	9	7920163307429012	2
Rebecca (ebook)	16	1739670448534144	1
Birdsong (Paperback)	21	6998104637774965	1



Name	Id	Address	PhoneNumber	Average Fine
Hiral Patel	205	6904, Amethyst Lane, Plano, TX 75023	3478217781	$0.00
Priti Patel	207	2436 Harmon Cove Towers, Secaucus, NJ	6093349922	$0.55
Vaishnavi	208	334 broadway, New York, NY 08765	XYZ	$0.00

Branch Id : 141
274 VonRueden Wall Apt. 785 West Wilfredomouth, NY 53789-2592

7. User Guide

Library Management System allows users to easily search for a book, checkout a book, check-in a book. Add new borrowers to the system and computes fine for the borrowed books. Above all it authenticates the librarian/admin to the library database system to prevent any kind of misuse of the system.

Following is a brief description about the working of the system:-

- Homepage - Authenticates the librarian, if failed no activity can be performed on the system.
- Search a Book - Allows the user to search any book given any combination of Book id and/or Book Title and /or Book Author. This provides flexibility to the librarians in searching the book in any order.

- Checkout a Book – Allows a user to check-out the book from a branch based on its availability and book borrowers credibility. Checkout is restricted for all borrowers who have exceeded 3 book loans or have any past fines due on them.

- Return a Book - Allows a user to check-in the book. This feature first searches for all the book loans the borrower has taken which are not checked-in yet and then allows the user to check in the selected the book. It also intimates the user if the borrower has any fine due on that book.

- Add New Reader - Allows the user to add a new borrower to the library loan system. The system generates a unique card no for each borrower. Uniqueness is defined based on the first name, last name and address details of a borrower.

- Compute Fine - This feature allows two types of fine computation. First – Computes the fine for all the borrowers who have taken a book loan. Second – Computes the fine for an individual book borrower who has taken a book loan.

8. Specify in detail each team member's work:

Modules Team Members Name	Database Design	Project Report	Implementation and coding
Vaishnavi Bhambure (1156082)	-Schema Mapping -Normalization - Design of ER -Diagram	-Introduction -Relational Database Design -ER data Model Design -Logical Design of Database	-Database Connection -UI design -Database Query -Implementation and synchronization with database design
Priti Patel (0897217)	-SQL Queries -Design of Tables (Key Attributes) -SQL Triggers	-Screenshots and queries	-Implementation/ Development of Java Application -AngularJS Development -Database queries and triggers