****

**School of Engineering and Applied Science (SEAS), Ahmedabad University**

**B.Tech(CSE) Semester IV: CSE 250 Database Management System**

**Submitted to- Prof. Shefali Naik**

**Group No: 11 LIBRARY MANAGEMENT SYSTEM**

|  |  |  |
| --- | --- | --- |
| **Name** | **Enrolment Number** | **Program** |
| Drashti Soni | AU1940058 | BTech in CSE |
| Shrey Patel | AU1940110 | BTech in CSE |
| Shubham Patel | AU1940155 | BTech in CSE |

1. **Description of Project: Describe the project in detail with proper flow. Explain each functionality which is included in the project. (System Requirement Specification).**

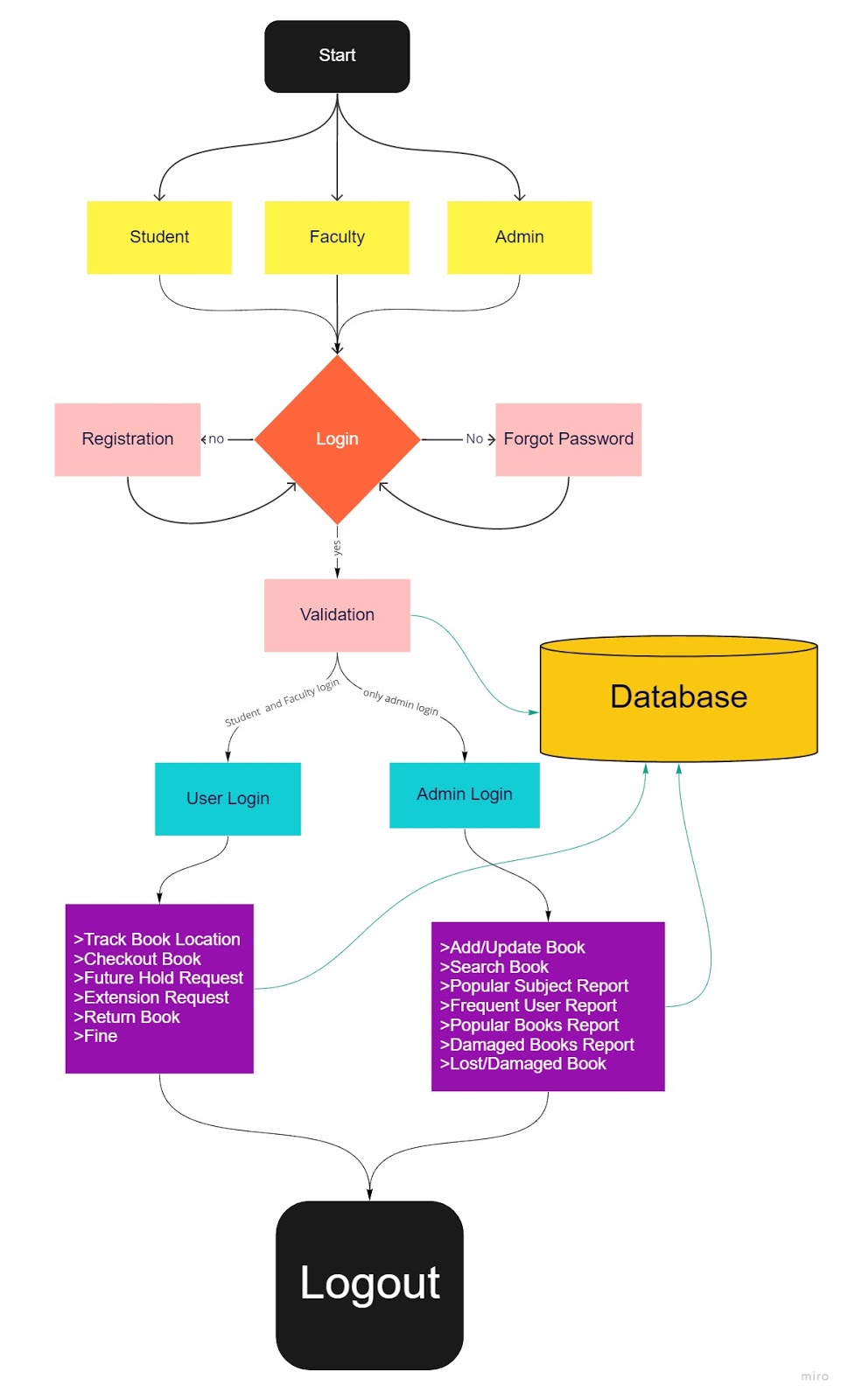
System Requirements:

* My SQL(Workbench) v8.0
* PHP v8.0.3
* XAMPP Control Panel v3.2.4 (for Apache Server)
* Bootstrap v5.0
* CSS v2.1
* HTML v5

The objective of our project is to make the user get easy access to the database of our Library. For this , we have added many functionalities which could help the user fulfill his/her requirements. The user can be of various types i.e. Students, faculty and the administrator. At the starting of the website; the user is asked to enter his username and password which has already been stored in the database. If the user is new, then he can create his new id. Every user has to use an unique username while registration and has to create a password which he/she needs to remember while visiting the website again for easy login. In case the user forgets his password, then there is a functionality of updating the password by entering his/her username.

After the user enters the login details, he gets to the page on the basis of his user type( student, admin or faculty). The student and faculty get the option to register for a book by entering its ISBN code and author name and other details. While if the user is admin then he gets directed to a page where he gets options to see the details of users such as Username, Name, DOB, Email, Penalty, Address etc. of all users. Also the admin can add or  update a new book. He can add the quantity of the book as well.

Apart from this when a student/faculty does the login, he gets options for searching a particular book, trackbook and holding a future request for a book. After doing so, he can check-out. Then the user also gets a button to return the book. After pressing the button of the return book he gets an option to select whether the book is damaged or not. Any user of the system gets the book for 14 days. The user needs to return the book within 14 days and if he fails to return, then he has to pay the fine. The fine is Rs2 per day of the extended time. There is also an option for extension of the book issued. A student can extend his/her return date of the book maximum 2 times (upto 28 days after his/her original return date) and for faculty it is upto maximum 5 times. Also a user can see his/her current fine.

****

**B.** **Entity-Relationship Diagram (Image)**

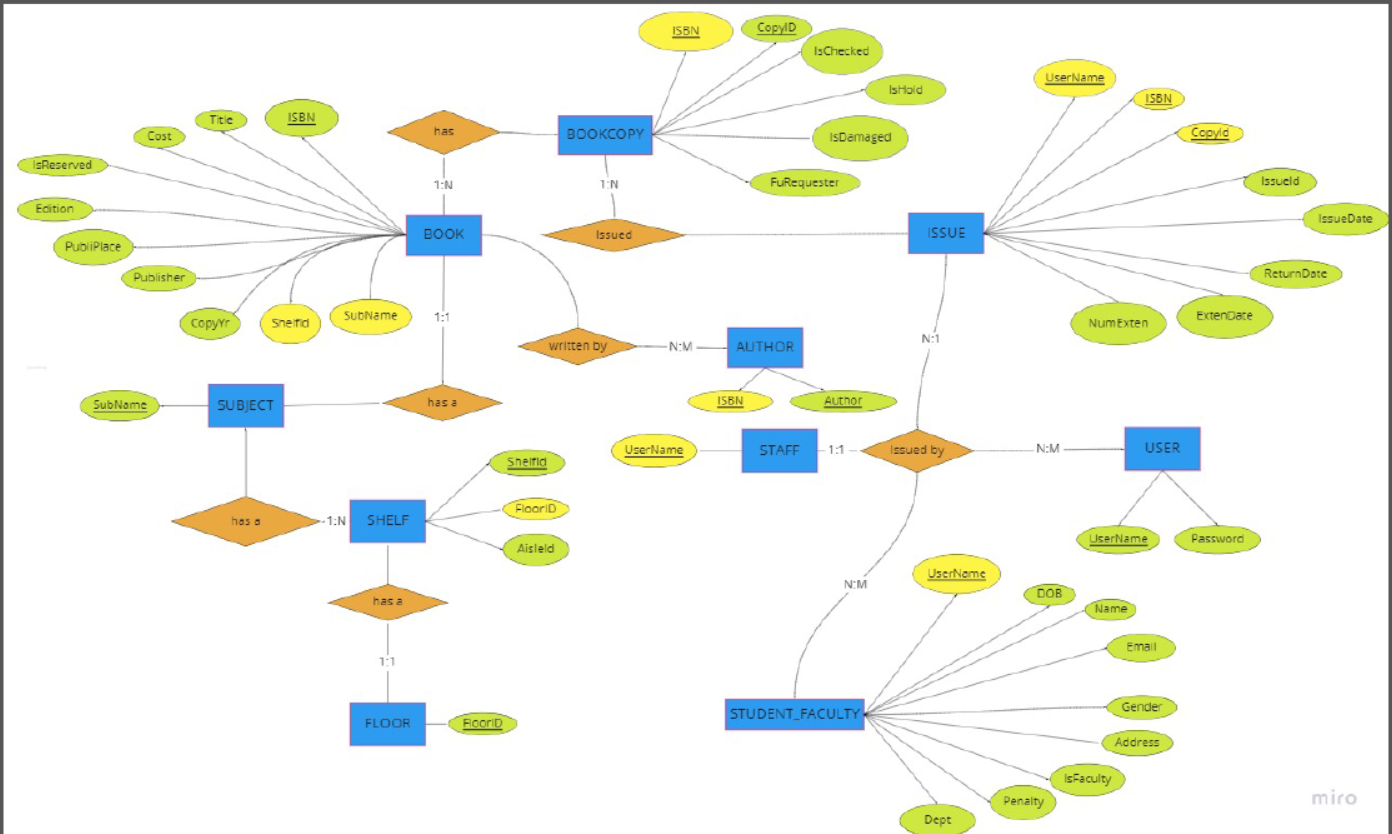
indicates “Foreign Key” attributes

indicates that the attribute is Primary key

indicates  “relationship between two table“

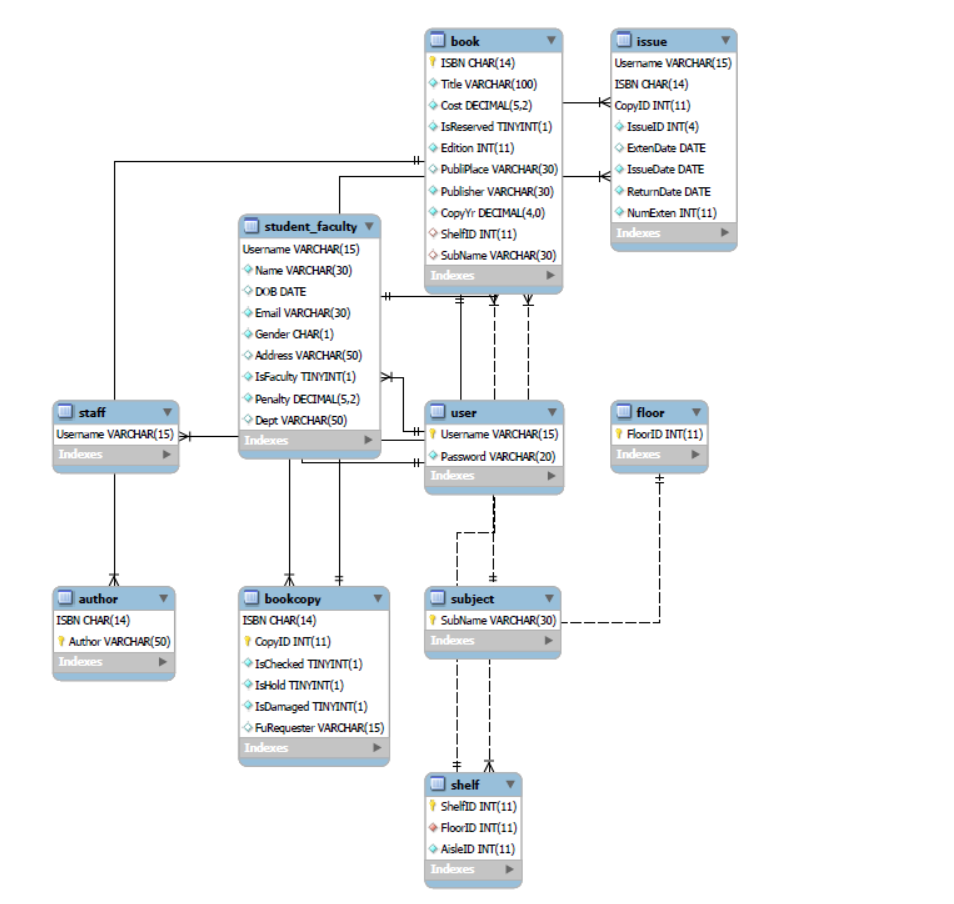
are attributes

are table



**Please refer to attached pdf of ER-Diagram if images is not properly visible here**

**C.** **Table Design (Data Dictionary)**

****

**D.**  **Stored Procedure, Functions and Triggers (With code and statement to call procedure, function written on front-end).**

**Procedure and Functions:**

1. **Listbook**

DELIMITER //

DROP PROCEDURE IF EXISTS `listbooks` //

CREATE PROCEDURE `listbooks` ()

BEGIN

SELECT \* FROM `book`;

END //

DELIMITER ;

$sql\_query = "CALL listbooks()";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

    exit();

}

1. **Addbook**

DELIMITER //

DROP PROCEDURE IF EXISTS `addbook` //

CREATE PROCEDURE `addbook`(IN `\_isbn` char(14), IN `\_q` int(11), IN `\_Author1` varchar(50), IN `\_Author2` varchar(50), IN `\_Title` varchar(100), IN `\_Cost` decimal(5,2), IN `\_IsReserved` tinyint(1), IN `\_Edition` int(11), IN `\_PubliPlace` varchar(30), IN `\_Publisher` varchar(30), IN `\_CopyYr` decimal(4,0), IN `\_ShelfID` int(11), IN `\_SubName` varchar(30))

BEGIN

DECLARE done INT DEFAULT False;

DECLARE `temp\_copyid` int(11) DEFAULT 0;

    DECLARE `temp\_qty` int(11) DEFAULT 1;

    DECLARE `temp\_isbn` char(14) DEFAULT 'NILL';

    SELECT `ISBN` INTO `temp\_isbn` FROM `book` WHERE STRCMP(trim(`ISBN`), trim(`\_isbn`)) = 0;

    IF (STRCMP(`temp\_isbn`, 'NILL') = 0) THEN

-- SET `temp\_isbn` = '1';

INSERT INTO `book` (`ISBN`, `Title`, `Cost`, `IsReserved`, `Edition`, `PubliPlace`, `Publisher`, `CopyYr`, `ShelfID`, `SubName`) VALUES (`\_isbn`, `\_Title`, `\_Cost`, 0, `\_Edition`, `\_PubliPlace`, `\_Publisher`, `\_CopyYr`, `\_ShelfID`, `\_SubName`);

        INSERT INTO `author` (`ISBN`, `Author`) VALUES (`\_isbn`, `\_Author1`);

        IF (STRCMP(`\_Author2`, 'NILL') != 0) THEN

INSERT INTO `author` (`ISBN`, `Author`) VALUES (`\_isbn`, `\_Author2`);

        END IF;

        read\_loop: LOOP

IF `temp\_qty` > `\_q` THEN

LEAVE read\_loop;

ELSE

INSERT INTO `bookcopy` (`ISBN`, `CopyID`, `IsChecked`, `IsHold`, `IsDamaged`, `FuRequester`) VALUES (`\_isbn`, `temp\_qty`, '0', '0', '0', NULL);

set `temp\_qty` = `temp\_qty` + 1;

END IF;

END LOOP;

    ELSE

SET `temp\_copyid` = (SELECT COUNT(`CopyID`) FROM `bookcopy` WHERE STRCMP(trim(`ISBN`), trim(`\_isbn`)) = 0);

read\_loop: LOOP

IF `temp\_qty` > `\_q` THEN

LEAVE read\_loop;

ELSE

INSERT INTO `bookcopy` (`ISBN`, `CopyID`, `IsChecked`, `IsHold`, `IsDamaged`, `FuRequester`) VALUES (`\_isbn`, `temp\_copyid` + `temp\_qty`, '0', '0', '0', NULL);

set `temp\_qty` = `temp\_qty` + 1;

END IF;

END LOOP;

    END IF;

END //

DELIMITER ;

$sql\_query = "CALL addbook ('$isbn', '$quantity', '$author1', '$author2', '$title', '$cost', '$isreserved', '$edition', '$publiplace', '$publisher', '$copyyr', '$shelfid', '$subname')";

        $result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

        if($result\_sqlquery == false){

            ?>

            <p>"Book record not updated</p>

            <?php

            echo 'Query falied';

            exit();

        }

1. **Issuebook**

DELIMITER //

DROP PROCEDURE IF EXISTS `issuebook` //

CREATE PROCEDURE `issuebook` (IN `\_username` varchar(15), IN `\_isbn` char(14), IN `\_copyid` int(11), IN `\_newisssueid` int(4), IN `\_today` date, IN `\_estimate` date)

BEGIN

INSERT INTO issue (Username, ISBN, CopyID, IssueID, ExtenDate, IssueDate, ReturnDate, NumExten) VALUES (`\_username`, `\_isbn`, `\_copyid`, `\_newisssueid`, NULL, `\_today`, `\_estimate`, 0);

UPDATE bookcopy AS BC SET IsHold = 1 Where BC.ISBN = `\_isbn` AND BC.CopyID = `\_copyid`;

END //

DELIMITER ;

$insertissue = "CALL issuebook ('$username', '$isbn', '$copyid', '$newisssueid', '$today', '$estimate')";

    $result = mysqli\_query ($link, $insertissue)  or die(mysqli\_error($link));

    if($result == false)

    {

        echo 'The query by ISBN failed.';

        exit();

    }

1. **Trackbook**

DELIMITER //

DROP PROCEDURE IF EXISTS `trackbook` //

CREATE PROCEDURE `trackbook` (IN `\_isbn` char(14))

BEGIN

SELECT S.ShelfID AS shelfid, S.AisleID AS aisleid, F.FloorID AS floorid, SU.SubName AS subname FROM book AS B, shelf AS S, floor AS F, subject AS SU WHERE B.ISBN = `\_isbn` AND B.ShelfID = S.ShelfID AND B.SubName = SU.SubName AND S.FloorID = F.FloorID;

END //

DELIMITER ;

$sql\_query2 = "CALL trackbook ('$isbn')";

    $result2 = mysqli\_query ($link, $sql\_query2)  or die(mysqli\_error($link));

    if($result2 == false)

    {

        echo 'The query by ISBN failed.';

        exit();

    }

1. **Checkoutbook**

DELIMITER //

DROP PROCEDURE IF EXISTS `checkoutbook` //

CREATE PROCEDURE `checkoutbook` (IN `\_isbn` char(14), IN `\_copyid` int(11), IN `\_issueid` int(4), IN `\_today` date, IN `\_estimate` date)

BEGIN

UPDATE issue AS I SET ExtenDate = `\_today`, IssueDate = `\_today`, ReturnDate = `\_estimate` WHERE I.IssueID = `\_issueid`;

    UPDATE bookcopy AS BC SET IsHold = 0, IsChecked = 1 Where BC.ISBN = `\_isbn` AND BC.CopyID = `\_copyid`;

END //

DELIMITER ;

$sql\_query = "CALL checkoutbook ('$isbn', '$copyid', '$issueid', '$today', '$estimate')";

        $result = mysqli\_query ($link, $sql\_query)  or die(mysqli\_error($link));

        if($result == false)

        {

            echo 'The query by ISBN failed.';

            exit();

        }

1. **Returnbook**

DELIMITER //

DROP PROCEDURE IF EXISTS `returnbook` //

CREATE PROCEDURE `returnbook` (IN `\_username` varchar(15), IN `\_this\_penalty` decimal(5,2))

BEGIN

DECLARE old\_penalty decimal(5,2) DEFAULT '0.00';

    DECLARE new\_penalty decimal(5,2);

SELECT SF.Penalty AS old\_penalty FROM student\_faculty AS SF WHERE SF.Username = `\_username`;

    SET new\_penalty = old\_penalty + `\_this\_penalty`;

    UPDATE student\_faculty AS SF SET Penalty = new\_penalty WHERE SF.Username = `\_username`;

END //

DELIMITER ;

$sql\_query = "CALL returnbook ('$username', '$this\_penalty')";

            $result = mysqli\_query ($link, $sql\_query)  or die(mysqli\_error($link));

            if($result == false)

            {

                echo 'The query failed.';

                exit();

            }

1. **Adduser**

DELIMITER //

DROP PROCEDURE IF EXISTS `adduser` //

CREATE PROCEDURE `adduser` (IN `\_username` varchar(15), IN `\_password` varchar(20))

BEGIN

INSERT INTO `user` (Username, Password) VALUES (`\_username`, `\_password`);

END //

DELIMITER ;

$insertStatement = "CALL adduser ('$username', '$password')";

        $result = mysqli\_query ($link, $insertStatement)  or die(mysqli\_error($link));

        if($result == false) {

            echo 'The query failed.';

            exit();

        }

1. **Createprofile**

DELIMITER //

DROP PROCEDURE IF EXISTS `createprofile` //

CREATE PROCEDURE `createprofile` (IN `\_username` varchar(15), IN `\_name` varchar(30), IN `\_DOB` date, IN `\_email` varchar(30), IN `\_gender` char(1), IN `\_address` varchar(50), IN `\_isfaculty` tinyint(1), IN `\_dept` varchar(50))

BEGIN

INSERT INTO `student\_faculty` (Username, Name, DOB, Email, Gender, Address, IsFaculty, Dept) VALUES (`\_username`, `\_name`, `\_DOB`, `\_email`, `\_gender`, `\_address`, `\_isfaculty`, `\_dept`);

END //

DELIMITER ;

$insertStatement = "CALL createprofile ('$username', '$name', '$DOB', '$email', '$gender', '$address', '$isfaculty', '$dept')";

    $result = mysqli\_query ($link, $insertStatement)  or die(mysqli\_error($link));

    if($result == false) {

        echo 'The query failed.';

        exit();

    }

1. **Showmyfine**

DELIMITER //

DROP PROCEDURE IF EXISTS `showmyfine` //

CREATE PROCEDURE `showmyfine` (IN `\_Username` varchar(15))

BEGIN

SELECT `Username` AS username, `Name` AS name, `Penalty` AS penalty FROM `student\_faculty` WHERE `Username` = `\_Username`;

END //

DELIMITER ;

$sql\_query = "CALL showmyfine('$username')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query failed.';

    exit();

}

1. **Listusers**

DELIMITER //

DROP PROCEDURE IF EXISTS `listusers` //

CREATE PROCEDURE `listusers` ()

BEGIN

SELECT \* FROM `student\_faculty`;

END //

DELIMITER ;

$sql\_query = "CALL listusers()";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

1. **Searchbookauthor**

DELIMITER //

DROP PROCEDURE IF EXISTS `searchbookauthor` //

CREATE PROCEDURE `searchbookauthor` (IN `\_author` varchar(50))

BEGIN

SELECT B.ISBN, Title, Edition, COUNT(CopyId) AS copynum FROM book AS B, bookcopy AS BC, author AS A Where A.Author = `\_author` AND A.ISBN = B.ISBN AND B.ISBN = BC.ISBN AND IsReserved = 0 AND IsHold = 0 AND IsChecked = 0 AND IsDamaged = 0 GROUP BY B.ISBN;

END //

DELIMITER ;

$sql\_query = "$sql\_query1 = "CALL searchbookauthor ('$author')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

1. **Searchbooktitle**

DELIMITER //

DROP PROCEDURE IF EXISTS `searchbooktitle` //

CREATE PROCEDURE `searchbooktitle` (IN `\_title` varchar(100))

BEGIN

SELECT B.ISBN, Title, Edition, COUNT(CopyId) AS copynum FROM book AS B, bookcopy AS BC WHERE Title = `\_title` AND B.ISBN = BC.ISBN AND IsReserved = 0 AND IsHold = 0 AND IsChecked = 0 AND IsDamaged = 0 GROUP BY B.ISBN;

END //

DELIMITER ;

$sql\_query = "$sql\_query1 = "CALL searchbooktitle ('$title')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

1. **Searchbookisbn**

DELIMITER //

DROP PROCEDURE IF EXISTS `searchbookisbn` //

CREATE PROCEDURE `searchbookisbn` (IN `\_isbn` char(14))

BEGIN

SELECT B.ISBN, Title, Edition, COUNT(CopyId) AS copynum FROM book AS B, bookcopy AS BC WHERE B.ISBN = `\_isbn` AND B.ISBN = BC.ISBN AND IsReserved = 0 AND IsHold = 0 AND IsChecked = 0 AND IsDamaged = 0;

END //

DELIMITER ;

$sql\_query = "CALL searchbookisbn ('$isbn')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

1. **Reservebookisbn**

DELIMITER //

DROP PROCEDURE IF EXISTS `reservebookisbn` //

CREATE PROCEDURE `reservebookisbn` (IN `\_isbn` char(14))

BEGIN

SELECT B.ISBN, Title, Edition, COUNT(CopyId) AS copynum FROM book AS B, bookcopy AS BC WHERE B.ISBN = `\_isbn` AND B.ISBN = BC.ISBN AND IsReserved = 1 AND IsHold = 0 AND IsChecked = 0 AND IsDamaged = 0;

END //

DELIMITER ;

$sql\_query = "CALL reservebookisbn ('$isbn')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

1. **Reservebookauthor**

DELIMITER //

DROP PROCEDURE IF EXISTS `reservebookauthor` //

CREATE PROCEDURE `reservebookauthor` (IN `\_author` varchar(50))

BEGIN

SELECT B.ISBN, Title, Edition, COUNT(CopyId) AS copynum FROM book AS B, bookcopy AS BC, author AS A Where A.Author = `\_author` AND A.ISBN = B.ISBN AND B.ISBN = BC.ISBN AND IsReserved = 0 AND IsHold = 0 AND IsChecked = 0 AND IsDamaged = 0 GROUP BY B.ISBN;

END //

DELIMITER ;

$sql\_query = "CALL reservebookauthor ('$author')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

1. **Reservebooktitle**

DELIMITER //

DROP PROCEDURE IF EXISTS `reservebooktitle` //

CREATE PROCEDURE `reservebooktitle` (IN `\_title` varchar(100))

BEGIN

SELECT B.ISBN, Title, Edition, COUNT(CopyId) AS copynum FROM book AS B, bookcopy AS BC WHERE Title = `\_title` AND B.ISBN = BC.ISBN AND IsReserved = 0 AND IsHold = 0 AND IsChecked = 0 AND IsDamaged = 0 GROUP BY B.ISBN;

END //

DELIMITER ;

$sql\_query = "CALL reservebooktitle ('$title')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

1. **Payfine**

DELIMITER //

DROP PROCEDURE IF EXISTS `payfine` //

CREATE PROCEDURE `payfine` (IN `\_Username` varchar(15), IN `\_amount` decimal(5,2))

BEGIN

UPDATE `student\_faculty` SET `Penalty` = `Penalty` - `\_amount` WHERE `Username` = `\_Username`;

END //

DELIMITER ;

/\*

$sql\_query = "CALL payfine('$username', '$amount')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

1. **Futureholdrequest**

/\* DELIMITER //

DROP PROCEDURE IF EXISTS `futureholdrequest` //

CREATE PROCEDURE `futureholdrequest` (IN `\_isbn` char(14))

BEGIN

DECLARE copynum int(11) DEFAULT 0;

SELECT COUNT(CopyId) INTO copynum FROM book AS B, bookcopy AS BC Where B.ISBN = `\_isbn` AND B.ISBN = BC.ISBN AND IsReserved = 0 AND IsHold = 0 AND IsChecked = 0 AND IsDamaged = 0;

if (copynum != 0) THEN

ELSE

    END IF;

END //

DELIMITER ; \*/

$sql\_query = "CALL CALL searchbookisbn ('$isbn')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

**19. Checkoutbook**

/\*

DELIMITER //

DROP PROCEDURE IF EXISTS `checkoutbook` //

CREATE PROCEDURE `checkoutbook` (IN `\_isbn` char(14))

BEGIN

SELECT S.ShelfID AS shelfid, S.AisleID AS aisleid, F.FloorID AS floorid, SU.SubName AS subname FROM book AS B, shelf AS S, floor AS F, subject AS SU WHERE B.ISBN = `\_isbn` AND B.ShelfID = S.ShelfID AND B.SubName = SU.SubName AND S.FloorID = F.FloorID;

END //

DELIMITER ;

\*/

$sql\_query = "CALL CALL searchbookisbn ('$isbn')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

**20. Updatepassword**

DELIMITER //

DROP PROCEDURE IF EXISTS `updatepassword` //

CREATE PROCEDURE `updatepassword` (IN `\_username` varchar(15), IN `\_password` varchar(20))

BEGIN

UPDATE `user` SET `Password` = `\_password` WHERE `Username` = `\_username`;

END //

DELIMITER ;

\*/

$sql\_query = "CALL searchbookisbn ('$isbn')";

$result\_sqlquery = mysqli\_query($link, $sql\_query) or die(mysqli\_error($link));

if($result\_sqlquery == false) {

    echo 'The query by ISBN failed.';

  exit();

}

**Triggers:**

1. **checkpassword**

DELIMITER //

DROP TRIGGER IF EXISTS `checkpassword` //

CREATE TRIGGER `checkpassword` BEFORE INSERT ON `user`

FOR EACH ROW

BEGIN

   DECLARE temp\_password1 varchar(20);

   DECLARE temp\_password2 varchar(20) DEFAULT 'NILL';

   DECLARE temp\_password3 varchar(20) DEFAULT 'NILL';

   SET temp\_password1 = NEW.Password;

   IF (CHAR\_LENGTH(temp\_password1) >= 6 AND CHAR\_LENGTH(temp\_password1) <= 19) THEN

        IF (NOT NEW.Password REGEXP '[a-z]' OR NOT NEW.Password REGEXP '[0-9]') THEN

SIGNAL SQLSTATE '45001'

SET MESSAGE\_TEXT = 'Password must contain atleast one alphabet and atleast one number.';

END IF;

   ELSE

SIGNAL SQLSTATE '45002'

            SET MESSAGE\_TEXT = 'Password must have length between 6 to 19 characters.';

   END IF;

END //

DELIMITER ;

1. **Userregistration**

DELIMITER //

DROP TRIGGER IF EXISTS `userregistration` //

CREATE TRIGGER `userregistration` BEFORE INSERT ON `user`

FOR EACH ROW

BEGIN

    DECLARE `temp\_username2` varchar(15) DEFAULT 'NILL';

    SELECT `Username` INTO `temp\_username2` FROM `user` WHERE STRCMP(trim(`Username`), trim(NEW.Username)) = 0;

    IF (STRCMP(`temp\_username2`, 'NILL') != 0) THEN

SIGNAL SQLSTATE '45003'

            SET MESSAGE\_TEXT = 'This username already exists.';

    END IF;

END //

DELIMITER ;

1. **verifyuser**

DELIMITER //

DROP TRIGGER IF EXISTS `verifyuser` //

CREATE TRIGGER `verifyuser` BEFORE UPDATE ON `user`

FOR EACH ROW

BEGIN

DECLARE count int;

    DECLARE `temp\_username2` varchar(15) DEFAULT 'NILL';

    IF (NOT EXISTS (SELECT \* FROM `user` WHERE OLD.`Username` =  NEW.`Username`)) THEN

SIGNAL SQLSTATE '45004'

            SET MESSAGE\_TEXT = 'Incorrect Username.';

END IF;

END //

DELIMITER ;

1. **chktimeinsertissue**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimeinsertissue` //

CREATE TRIGGER `chktimeinsertissue` BEFORE INSERT ON `issue`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45005'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45006'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

1. **chktimeupdateissue**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimeupdateissue` //

CREATE TRIGGER `chktimeupdateissue` BEFORE UPDATE ON `issue`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45007'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45008'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

1. **chktimedeleteissue**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimedeleteissue` //

CREATE TRIGGER `chktimedeleteissue` BEFORE DELETE ON `issue`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45009'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45010'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

1. **chktimeinsertauthor**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimeinsertauthor` //

CREATE TRIGGER `chktimeinsertauthor` BEFORE INSERT ON `author`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45011'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45012'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

1. **chktimeupdateauthor**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimeupdateauthor` //

CREATE TRIGGER `chktimeupdateauthor` BEFORE UPDATE ON `author`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45013'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45014'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

1. **chktimedeleteauthor**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimedeleteauthor` //

CREATE TRIGGER `chktimedeleteauthor` BEFORE DELETE ON `author`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45015'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45016'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

1. **chktimeinsertbook**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimeinsertbook` //

CREATE TRIGGER `chktimeinsertbook` BEFORE INSERT ON `book`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45017'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45018'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

1. **chktimeupdatebook**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimeupdatebook` //

CREATE TRIGGER `chktimeupdatebook` BEFORE UPDATE ON `book`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45019'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45020'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

**12.chktimedeletebook**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimedeletebook` //

CREATE TRIGGER `chktimedeletebook` BEFORE DELETE ON `book`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45021'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45022'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

**13. chktimeinsertbookcopy**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimeinsertbookcopy` //

CREATE TRIGGER `chktimeinsertbookcopy` BEFORE INSERT ON `bookcopy`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45023'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45024'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

**14.chktimeupdatebookcopy**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimeupdatebookcopy` //

CREATE TRIGGER `chktimeupdatebookcopy` BEFORE UPDATE ON `bookcopy`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45025'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45026'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

DELIMITER ;

**15. chktimedeletebookcopy**

DELIMITER //

DROP TRIGGER IF EXISTS `chktimedeletebookcopy` //

CREATE TRIGGER `chktimedeletebookcopy` BEFORE DELETE ON `bookcopy`

FOR EACH ROW

BEGIN

DECLARE currday varchar(10);

    SET currday = DAYNAME(SYSDATE());

    SET @time = CURRENT\_TIME;

IF (currday = 'Sunday' OR currday = 'Saturday') THEN

SIGNAL SQLSTATE '45027'

            SET MESSAGE\_TEXT = 'You cannot access library on Saturdays and Sundays';

ELSE

            IF (CURRENT\_TIME >= TIME('20:00:01') OR CURRENT\_TIME <= TIME('05:59:59')) THEN

SIGNAL SQLSTATE '45028'

SET MESSAGE\_TEXT = 'You cannot access library between 8 PM  AND 6 AM';

END IF;

END IF;

END //

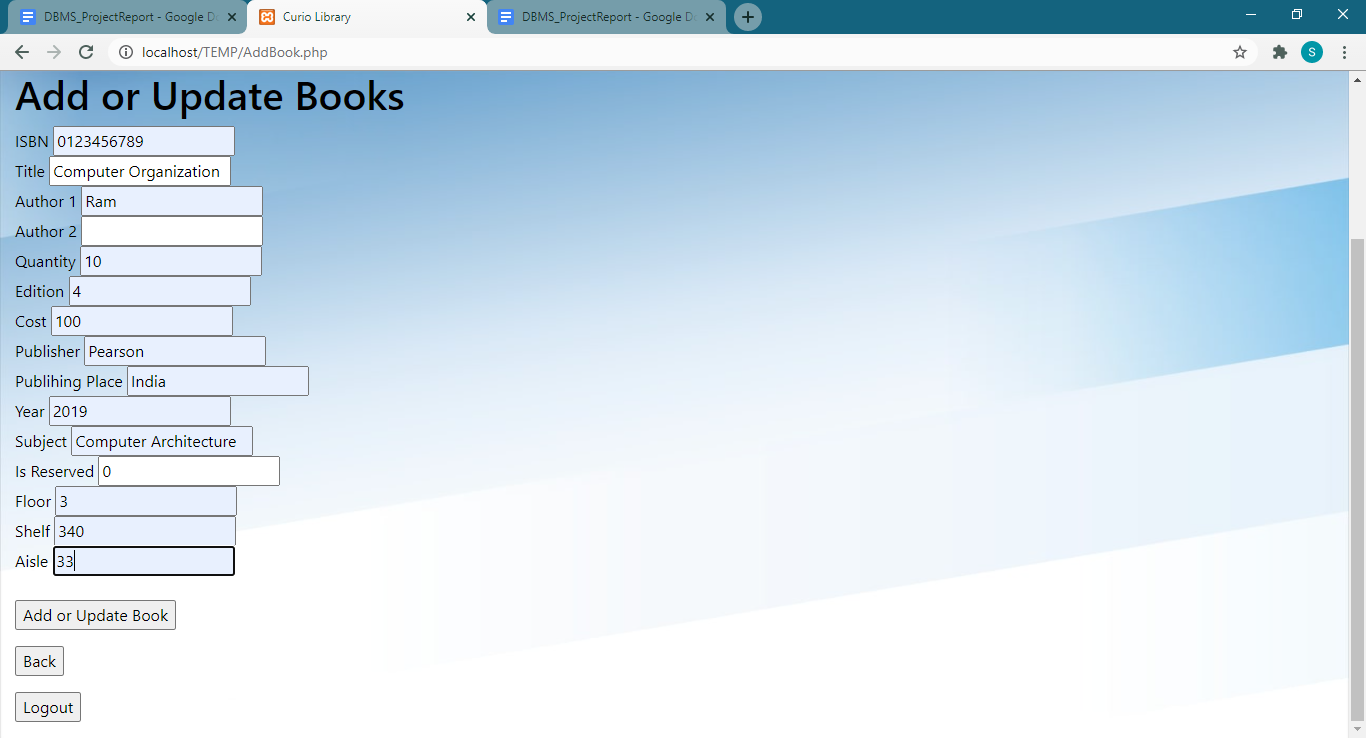
DELIMITER ;

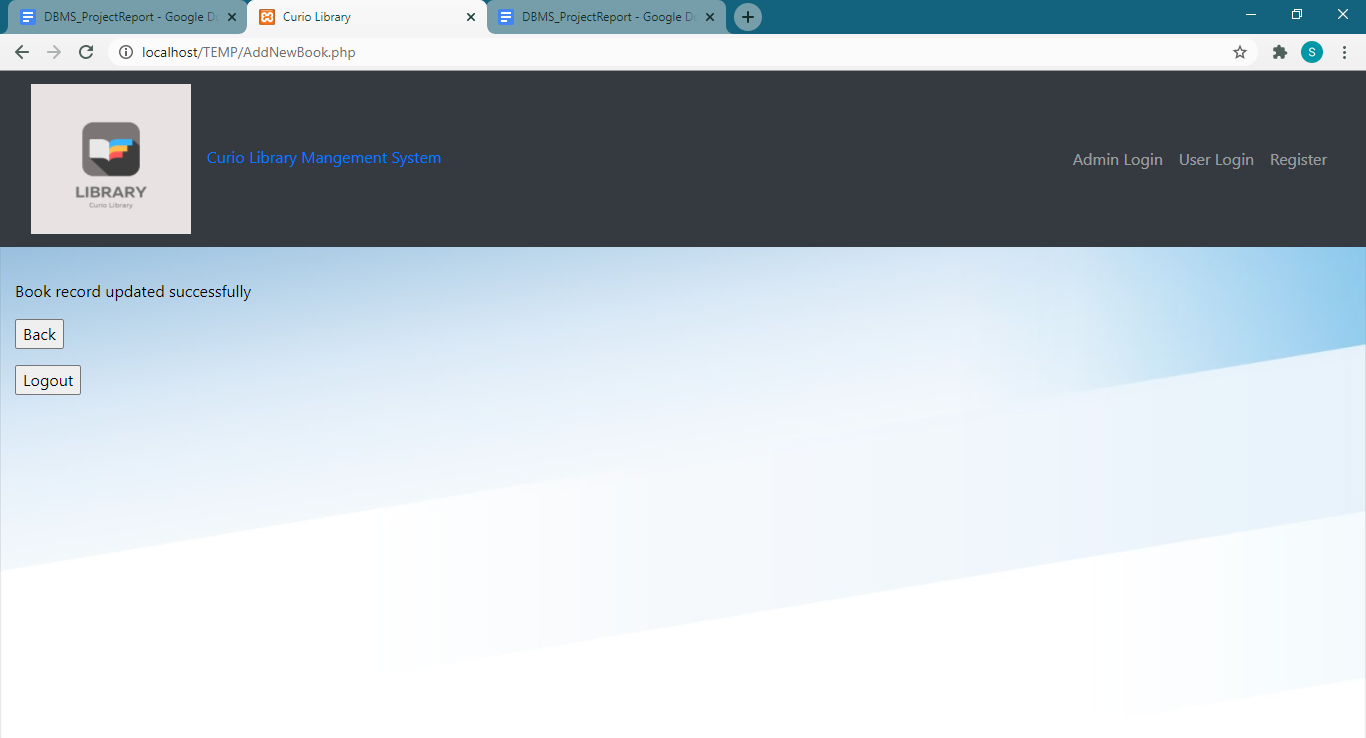
**E.**  **Screen-shots of results generated after procedure and function are called on front-end.**

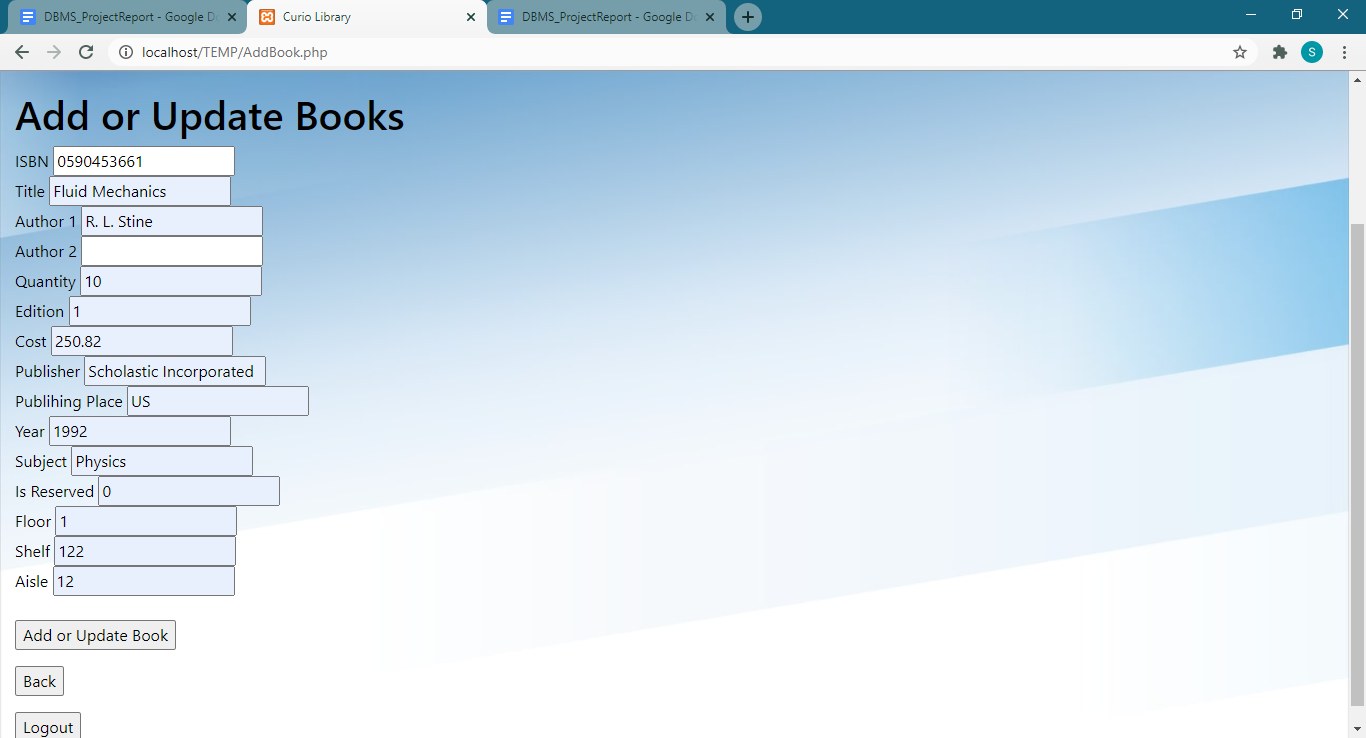
**1. listbook**

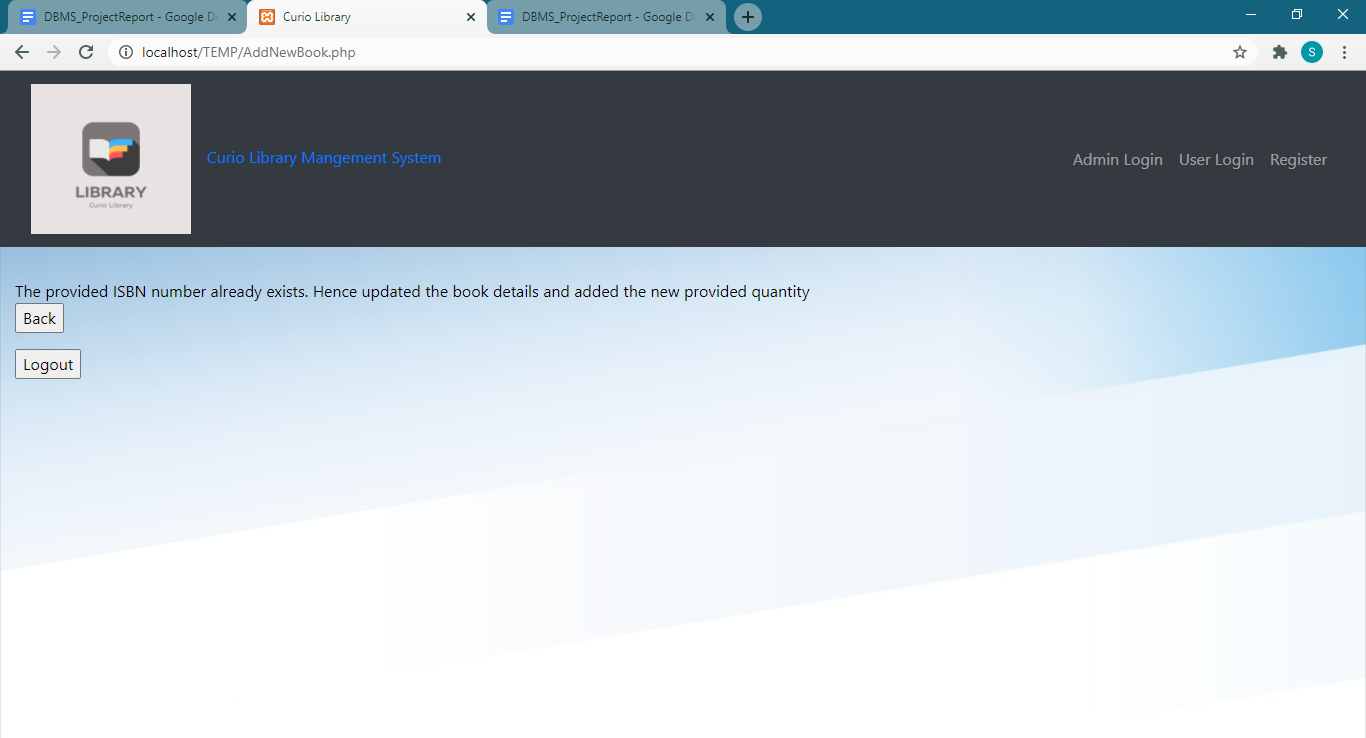
****

**2.  addbook**

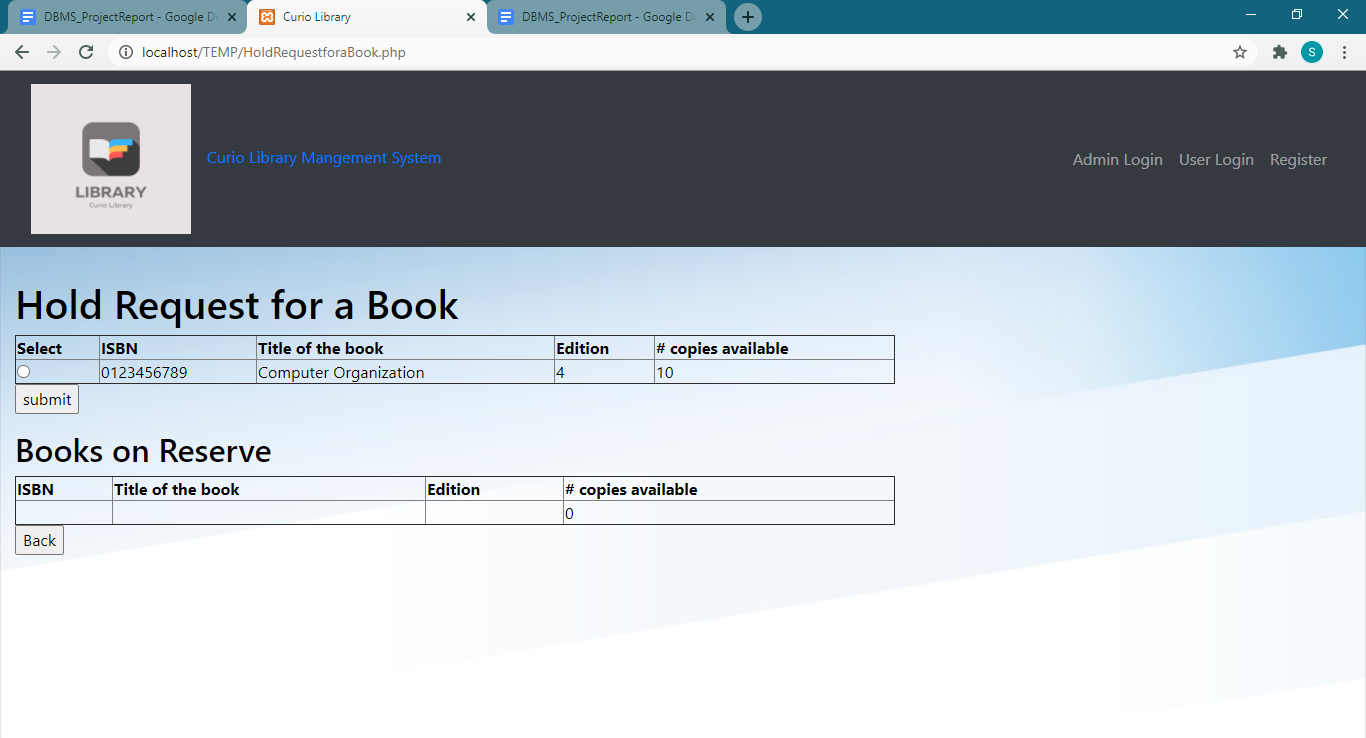
****

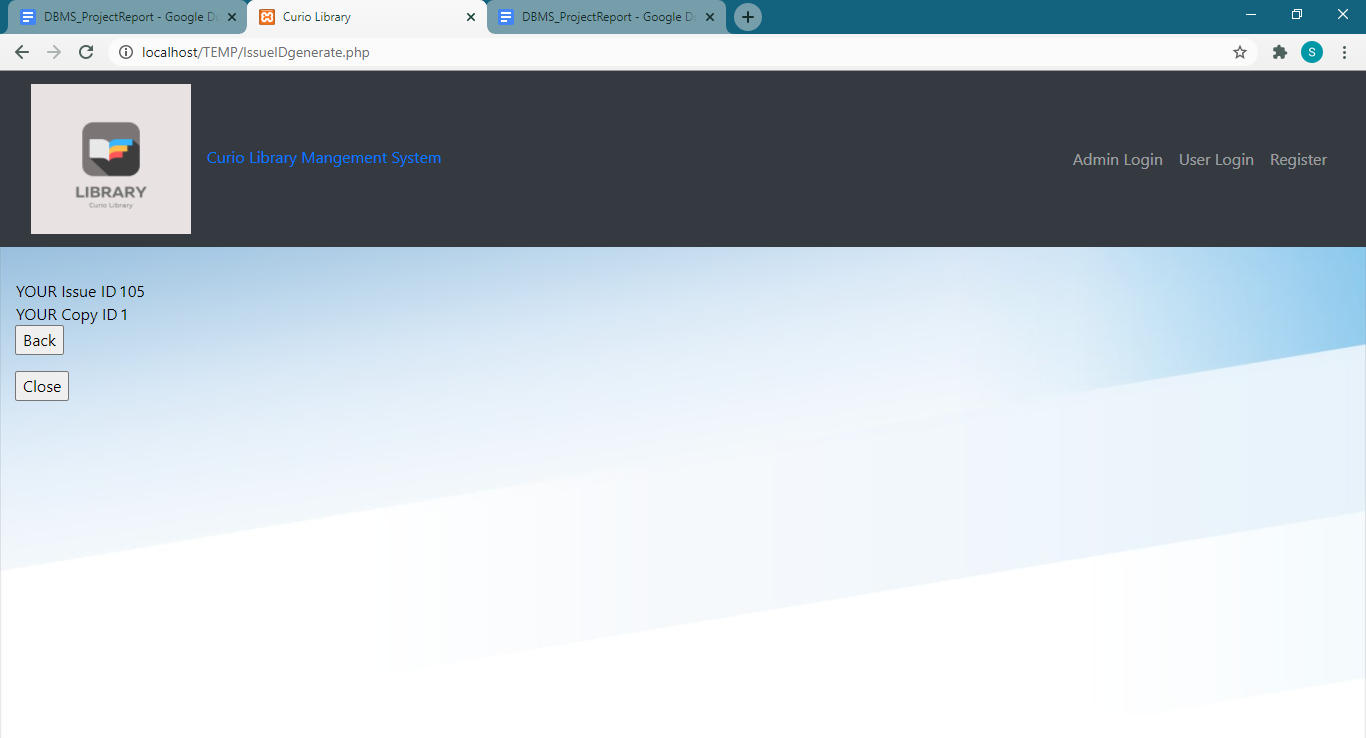
****

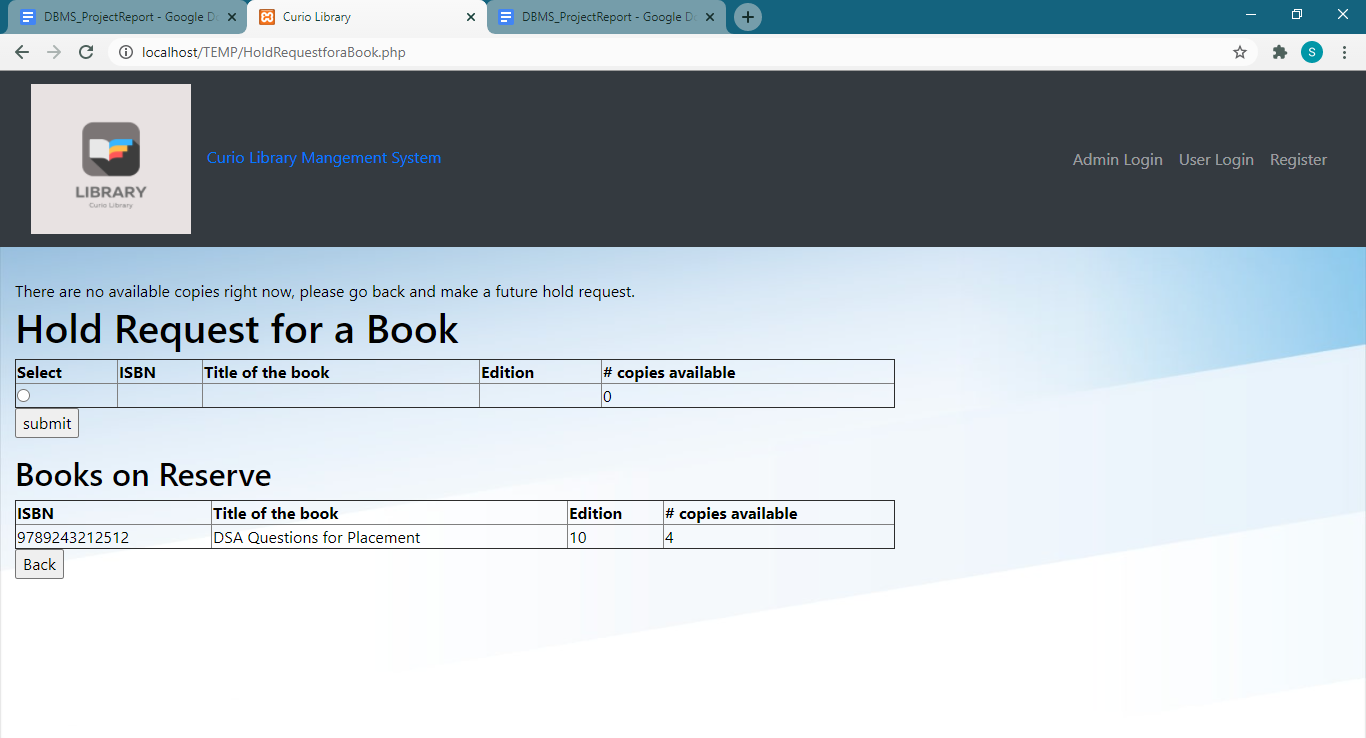
****

****

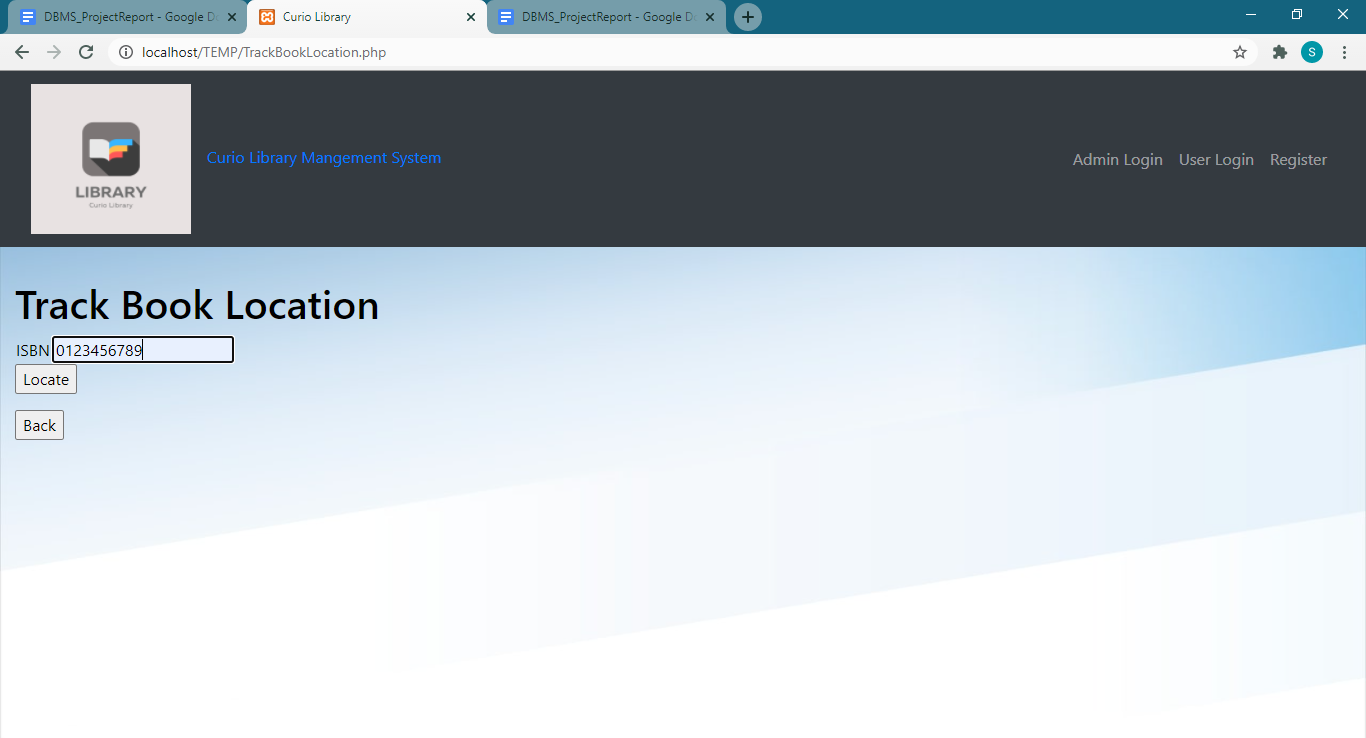
**3. issuebook**

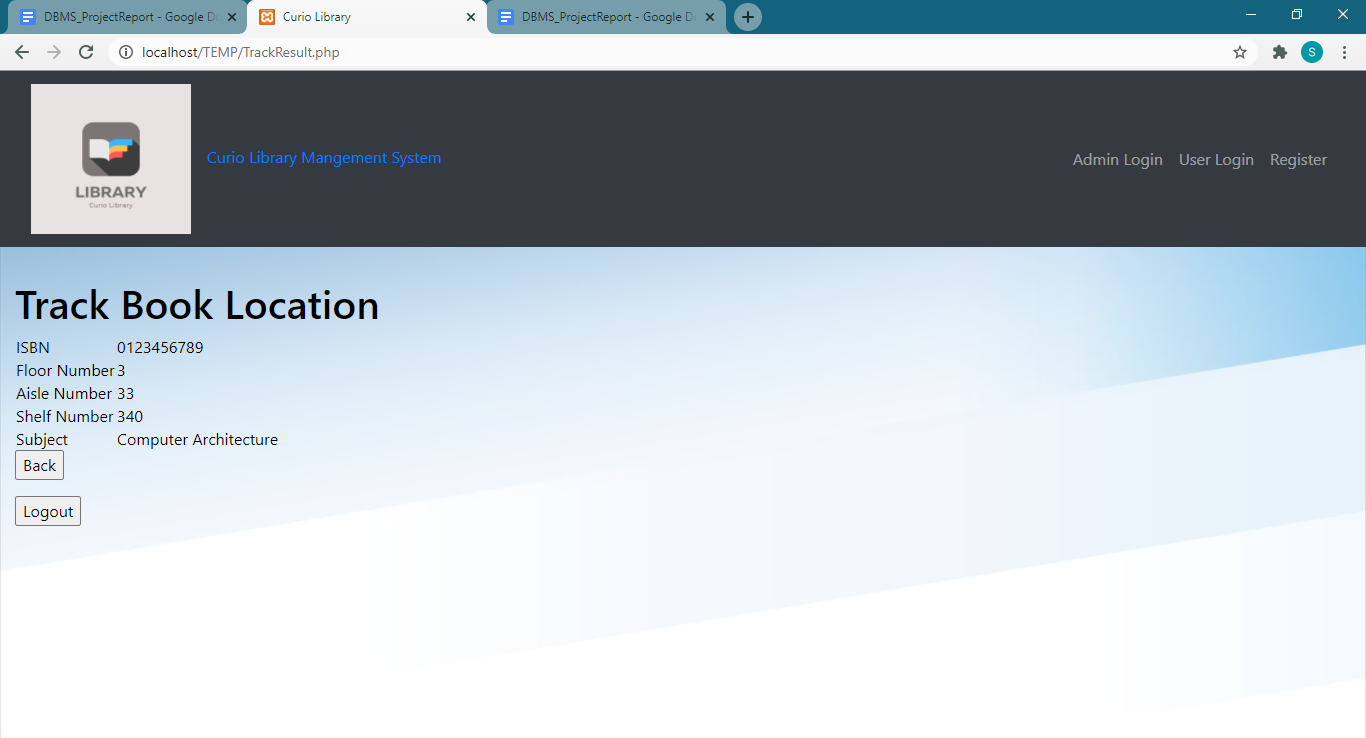
****

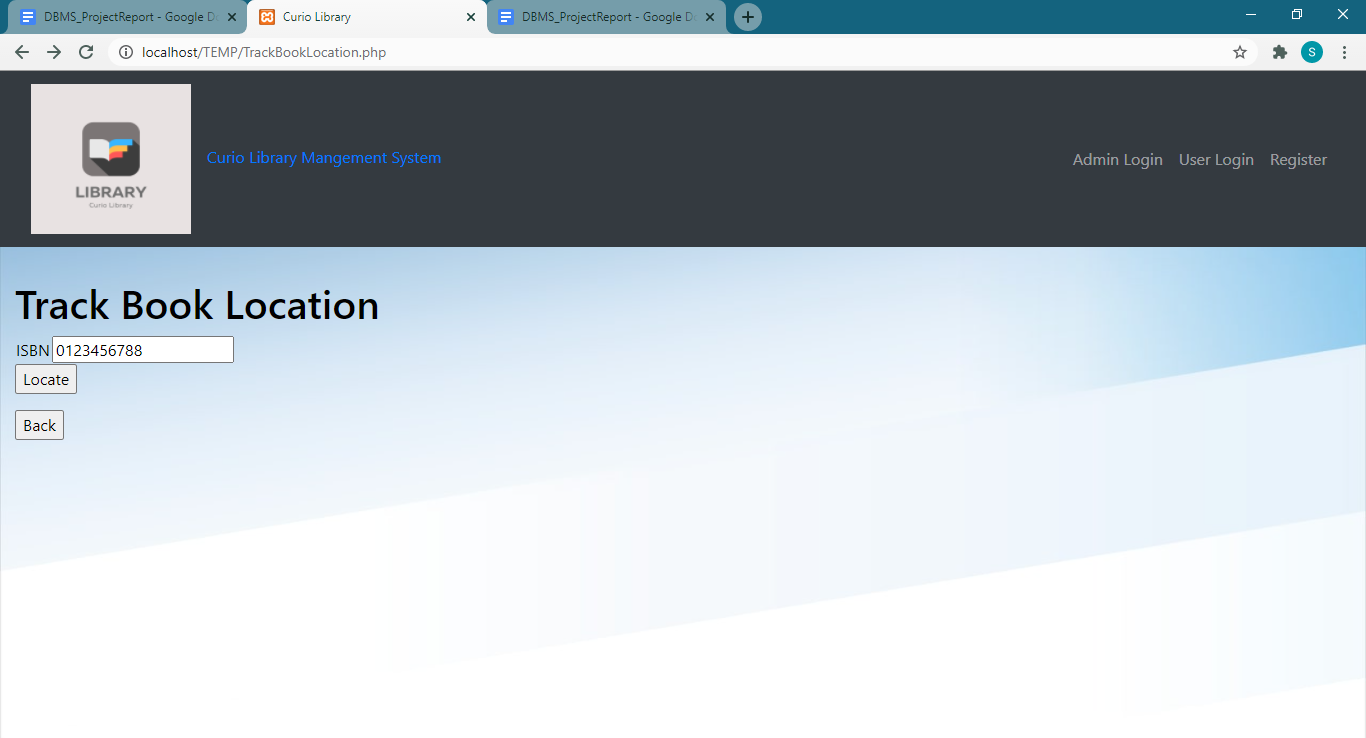
****

****

**4. trackbook**

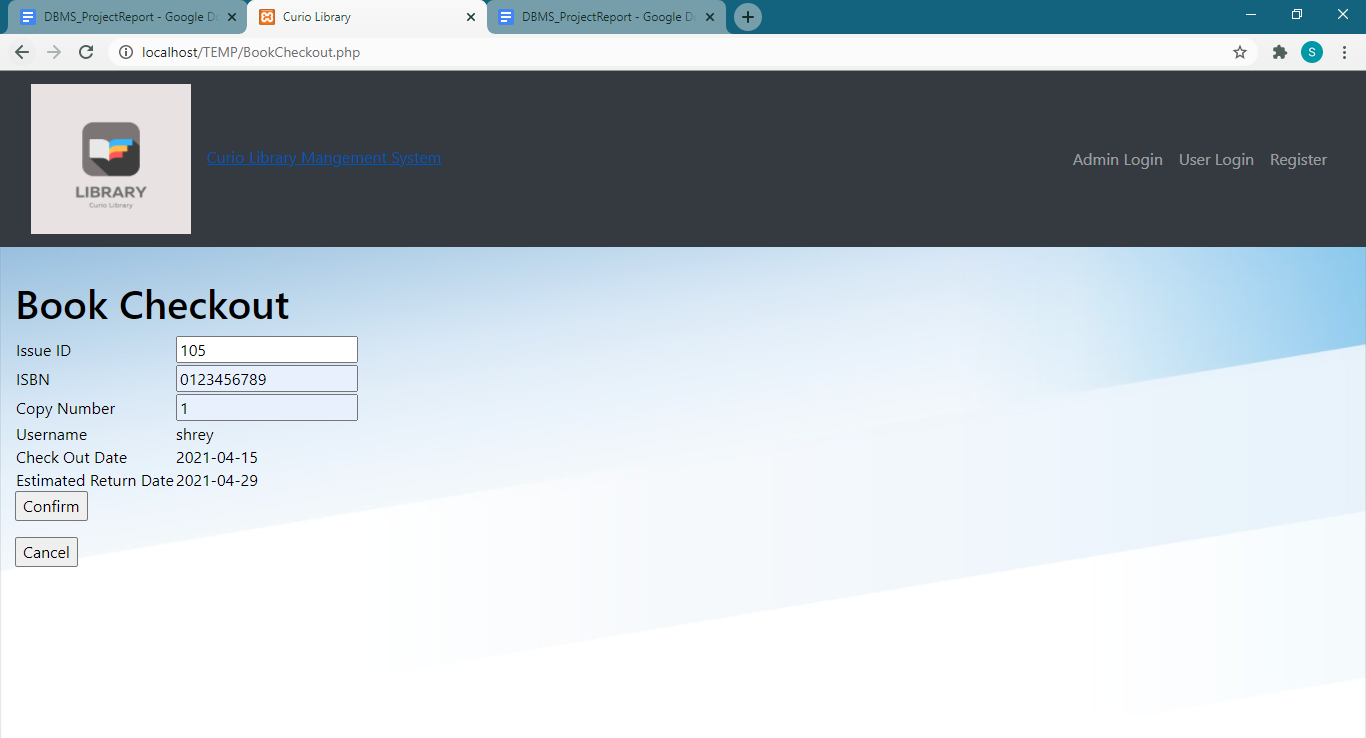
****

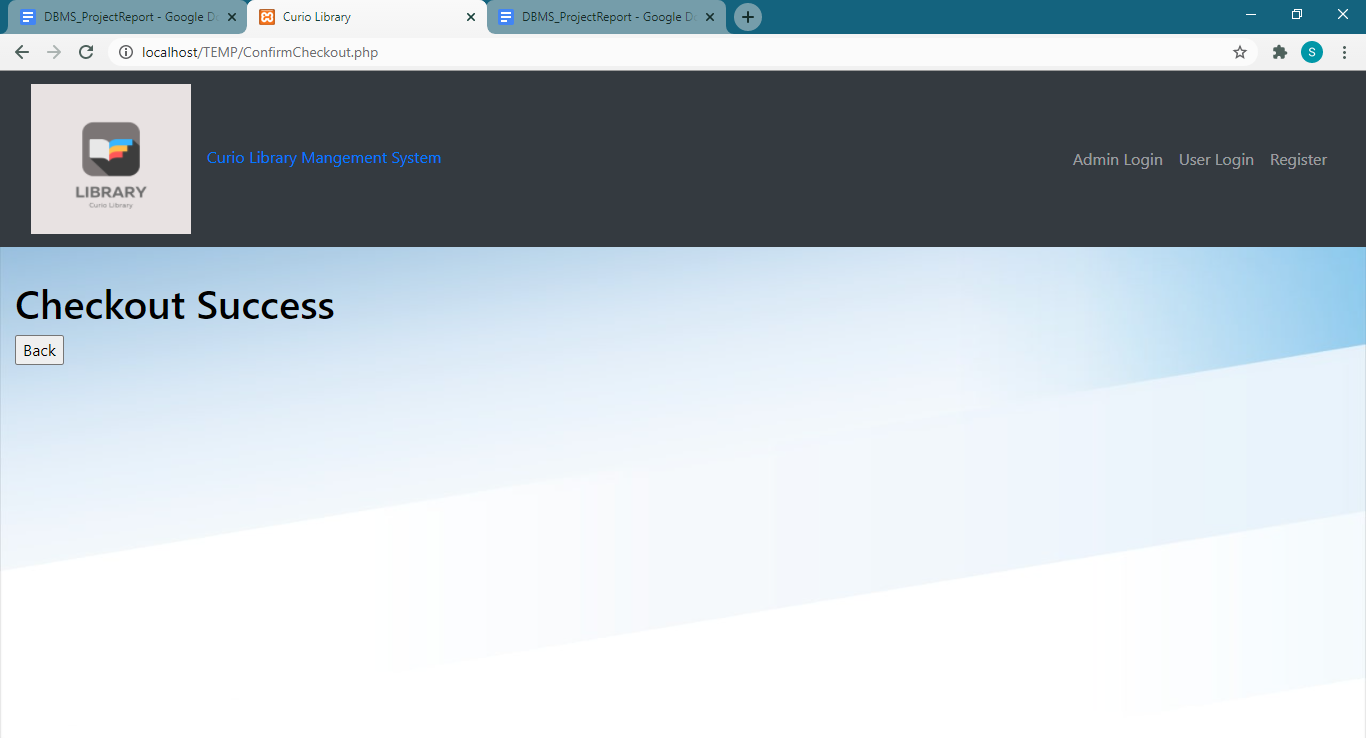
****

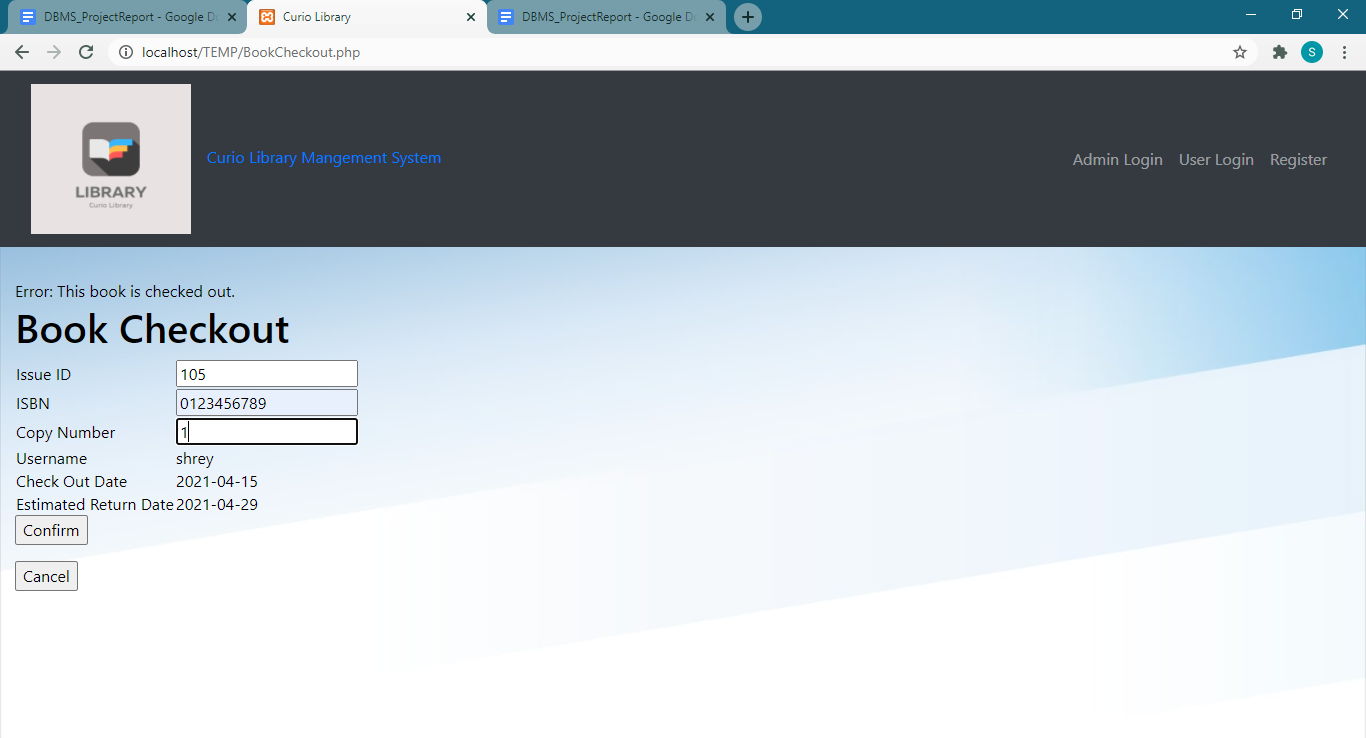
****

****

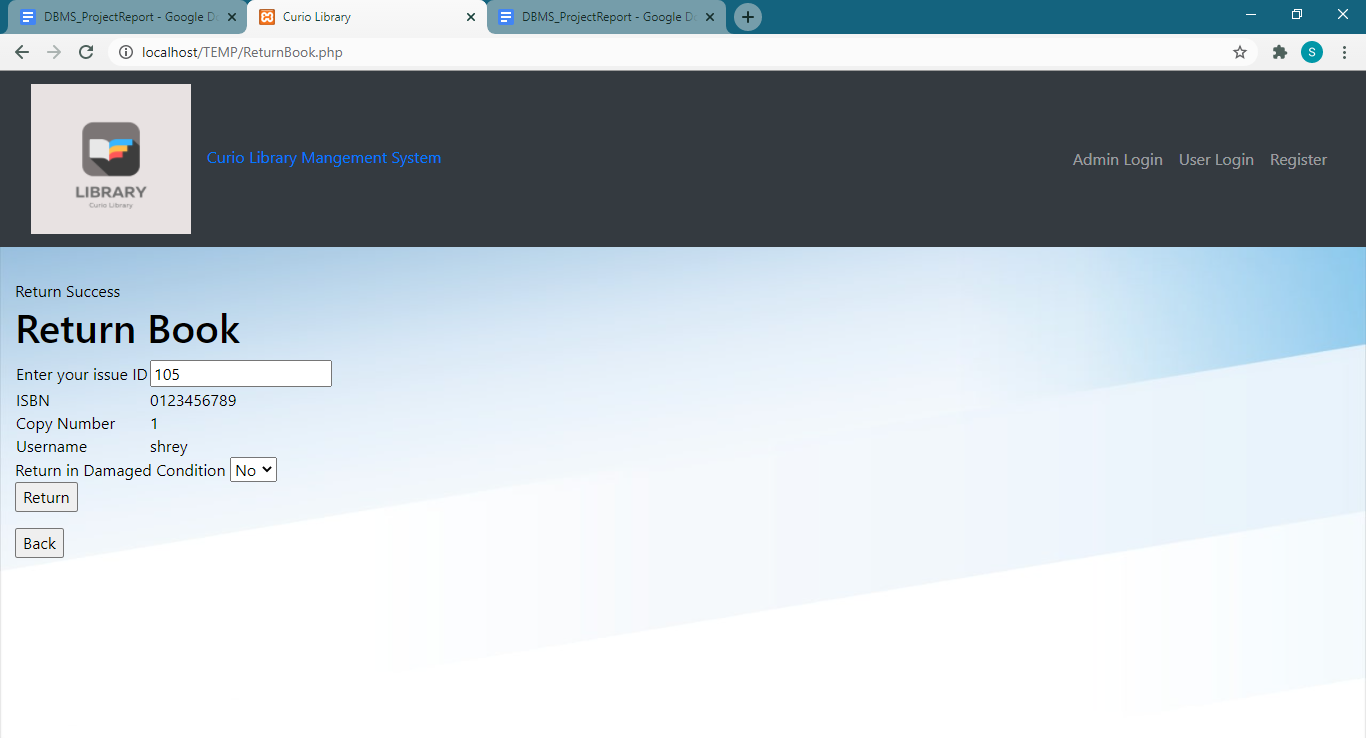
**5. checkoutbook**

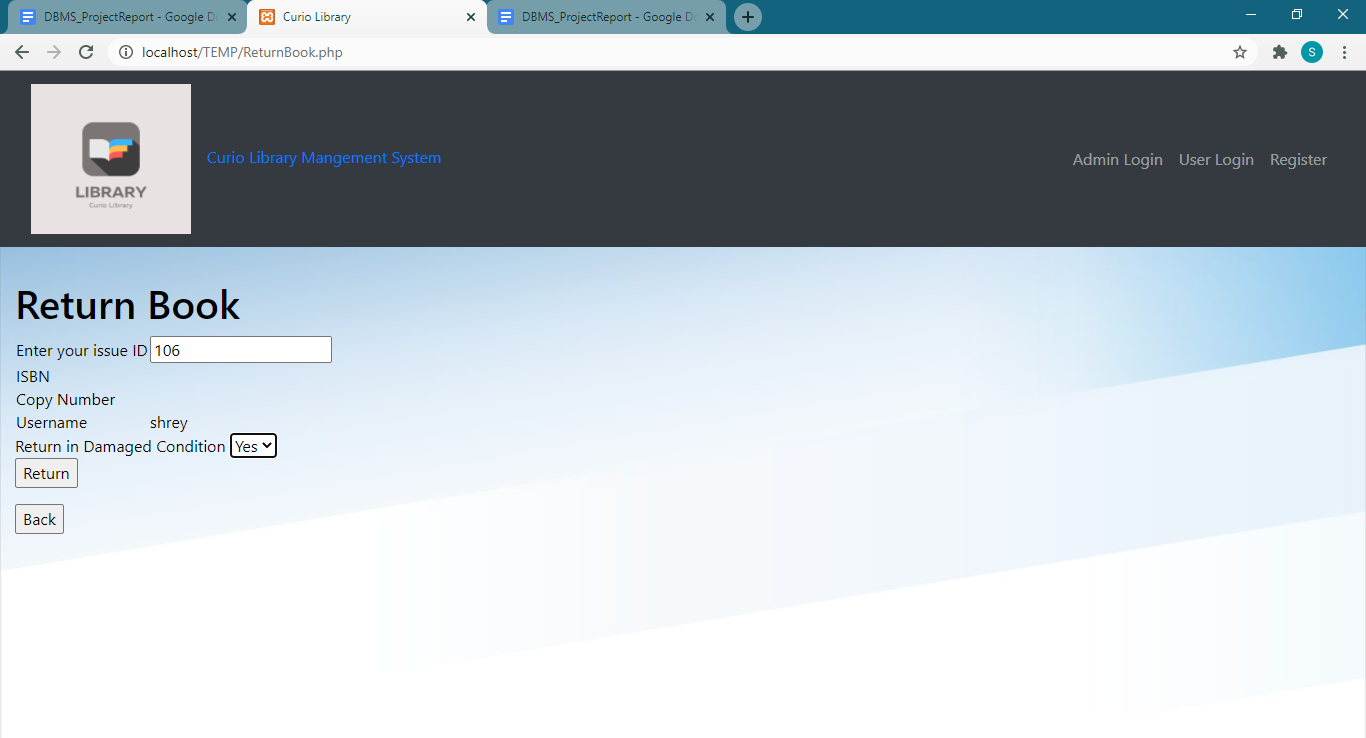
****

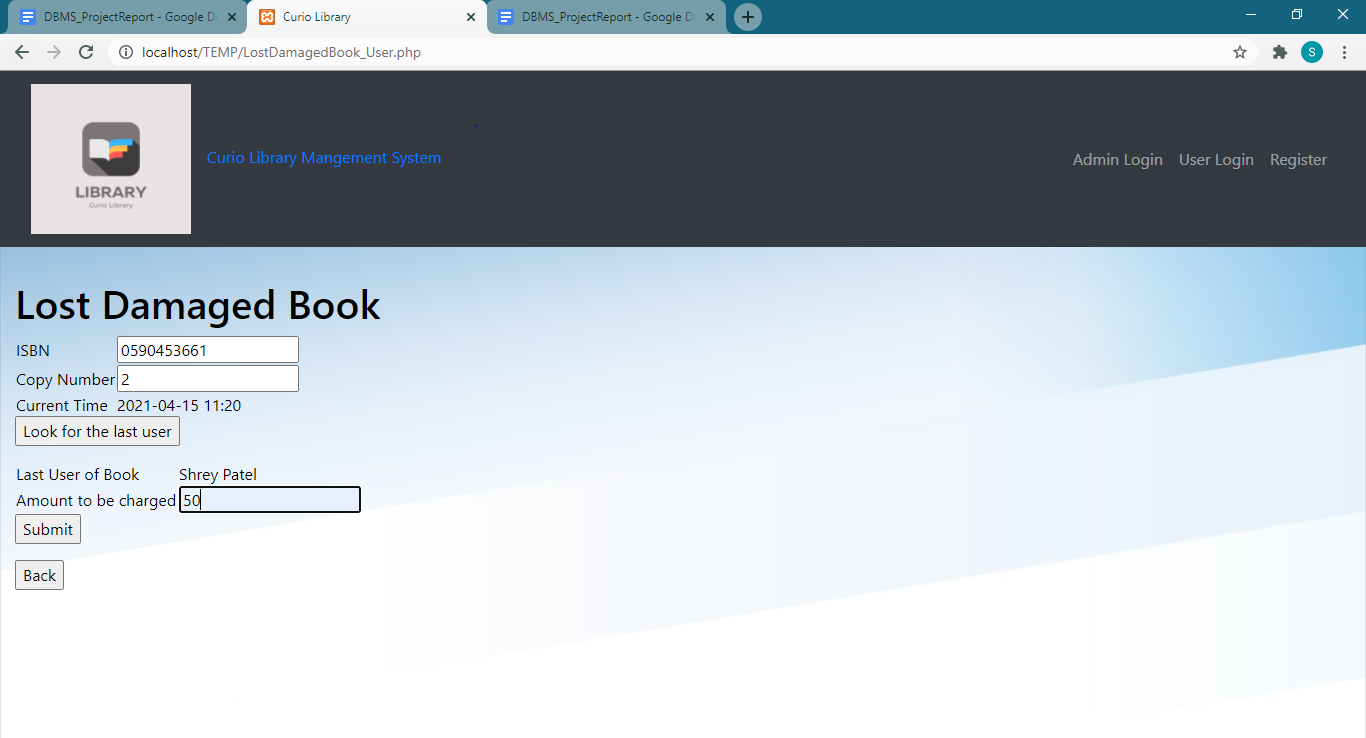
****

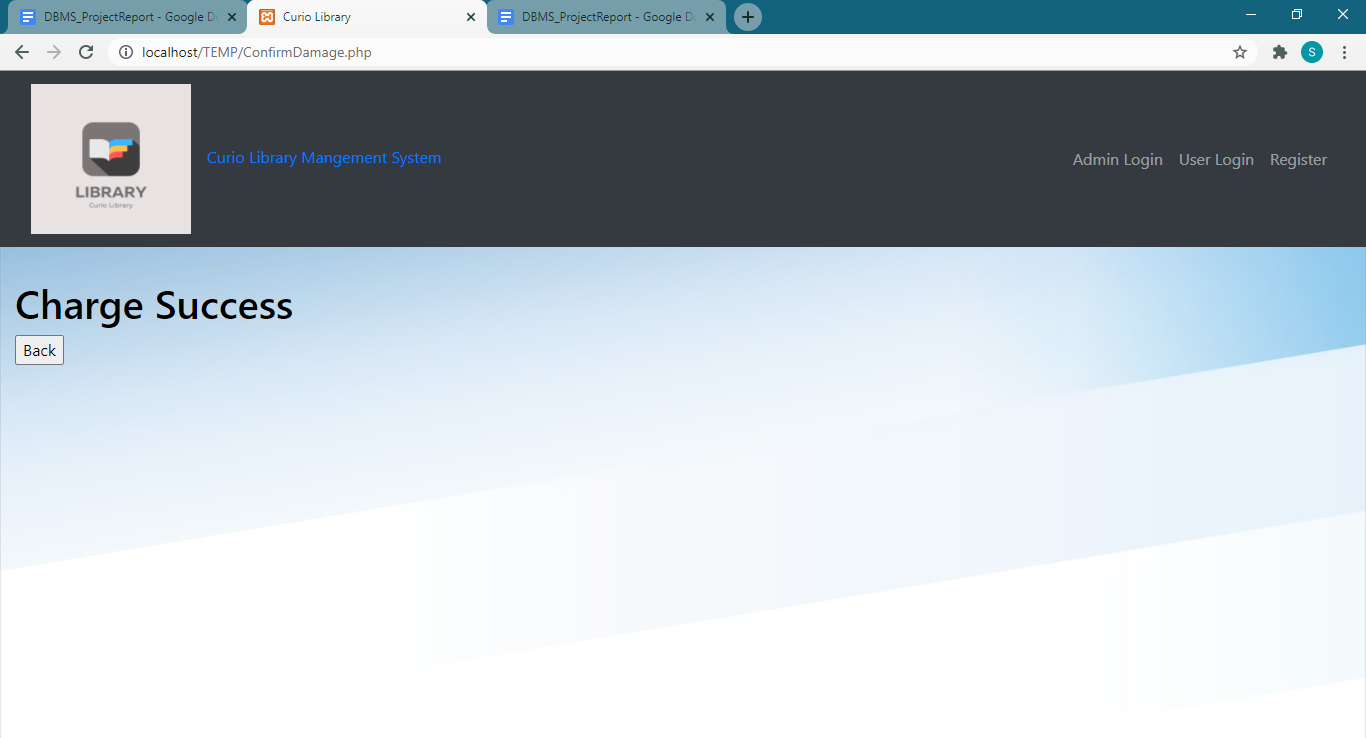
****

**6. returnbook**

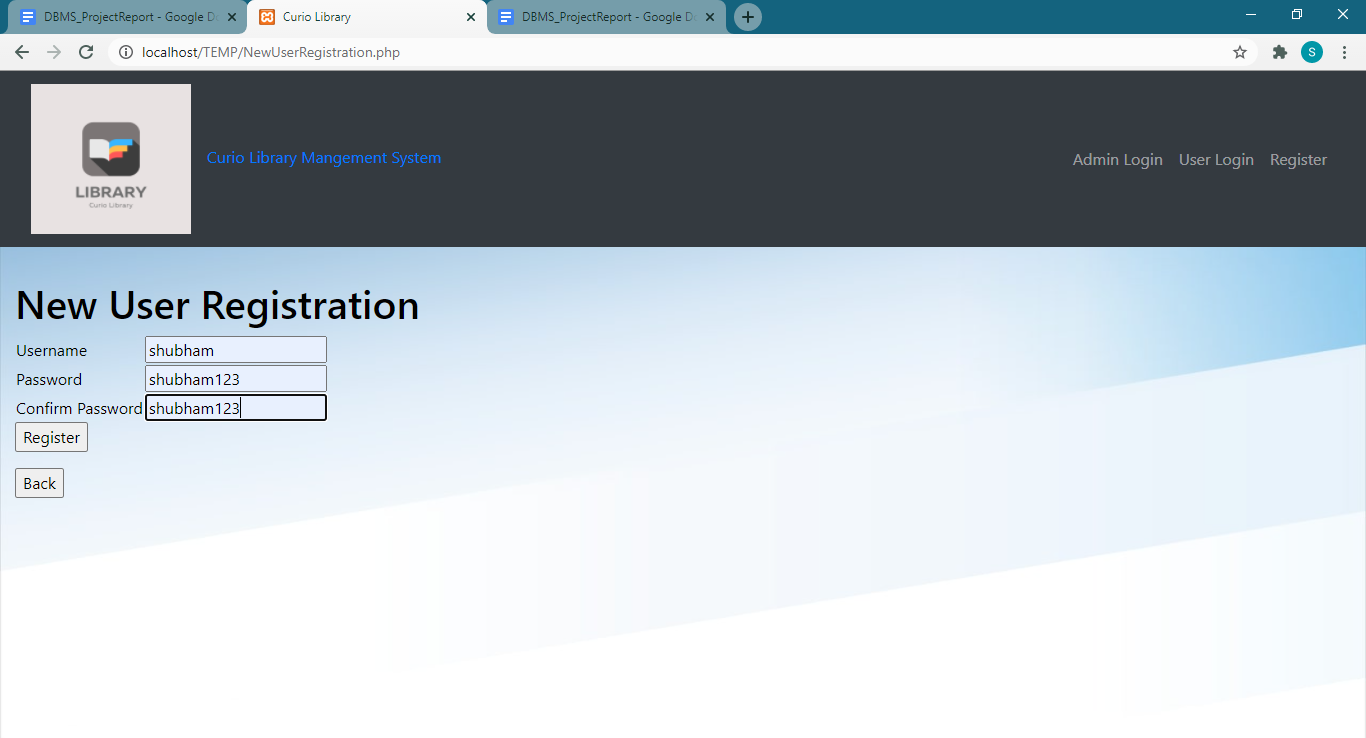
****

****

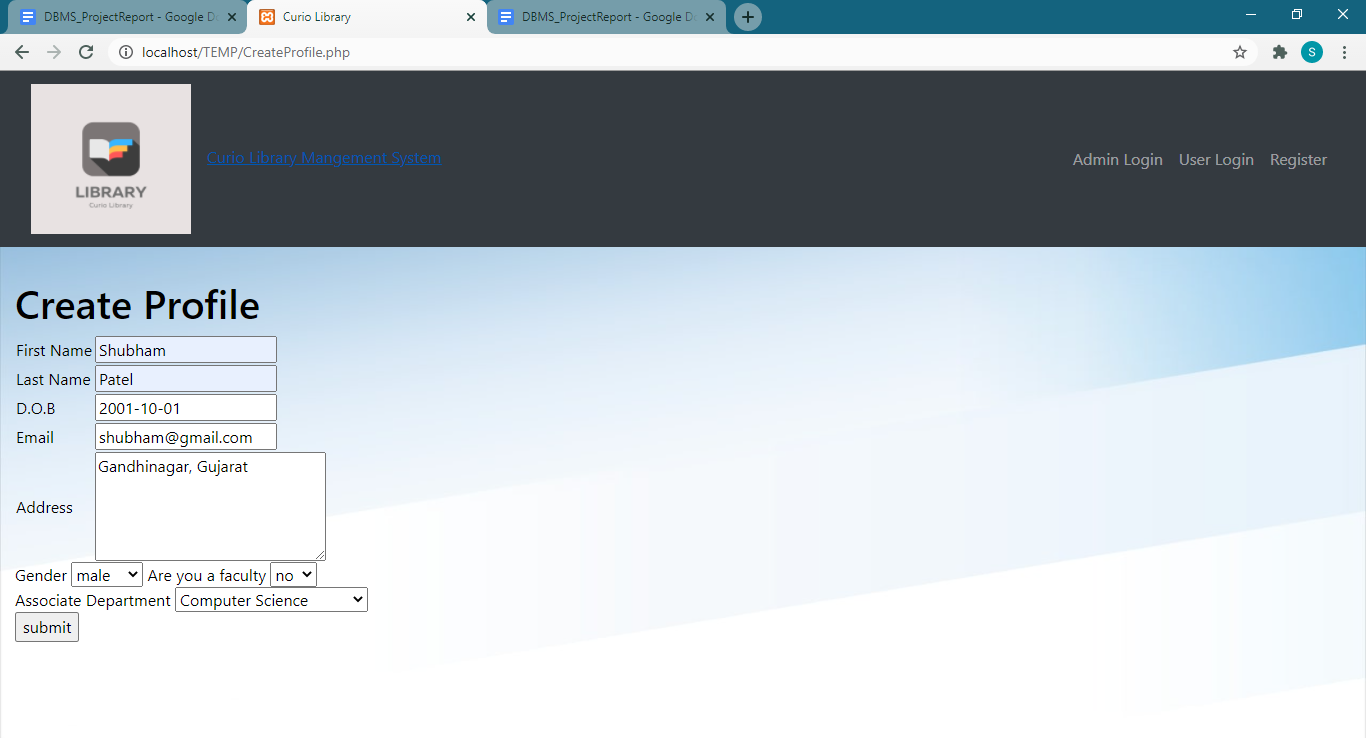
****

****

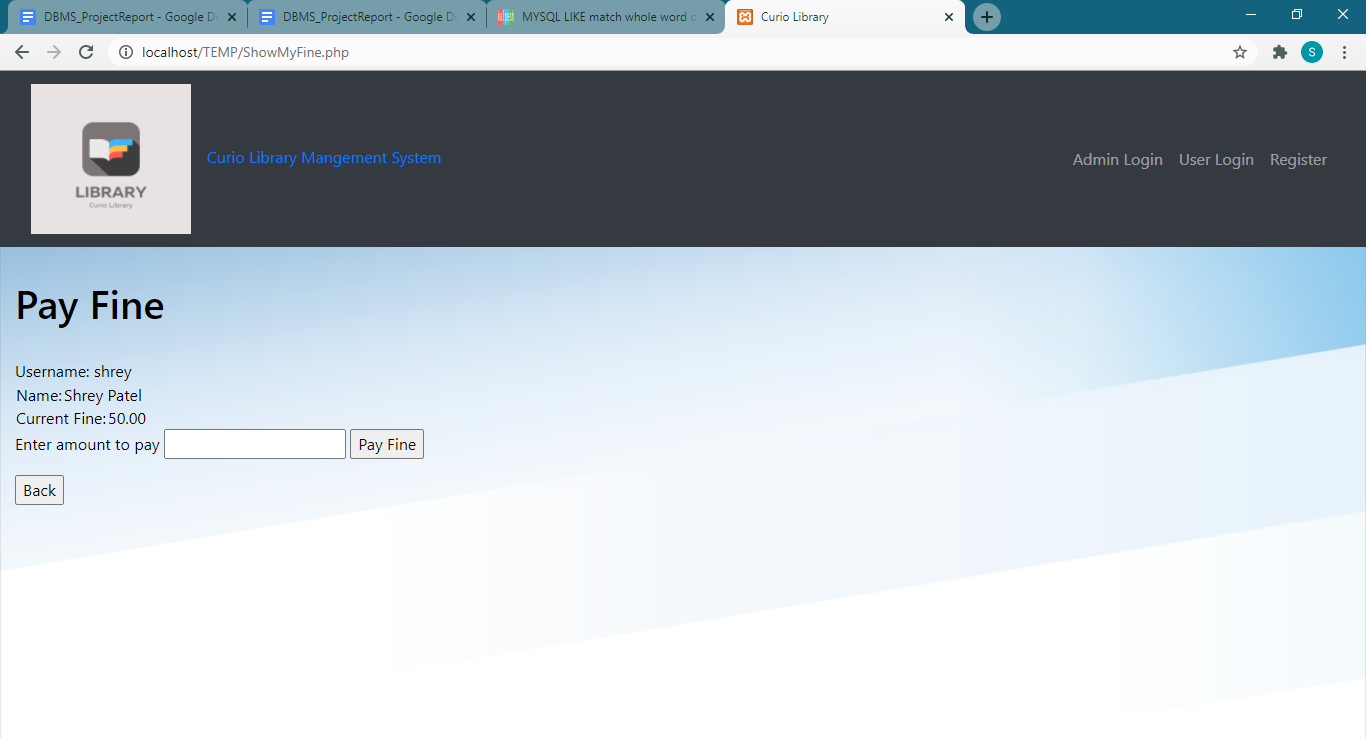
**7. adduser**

****

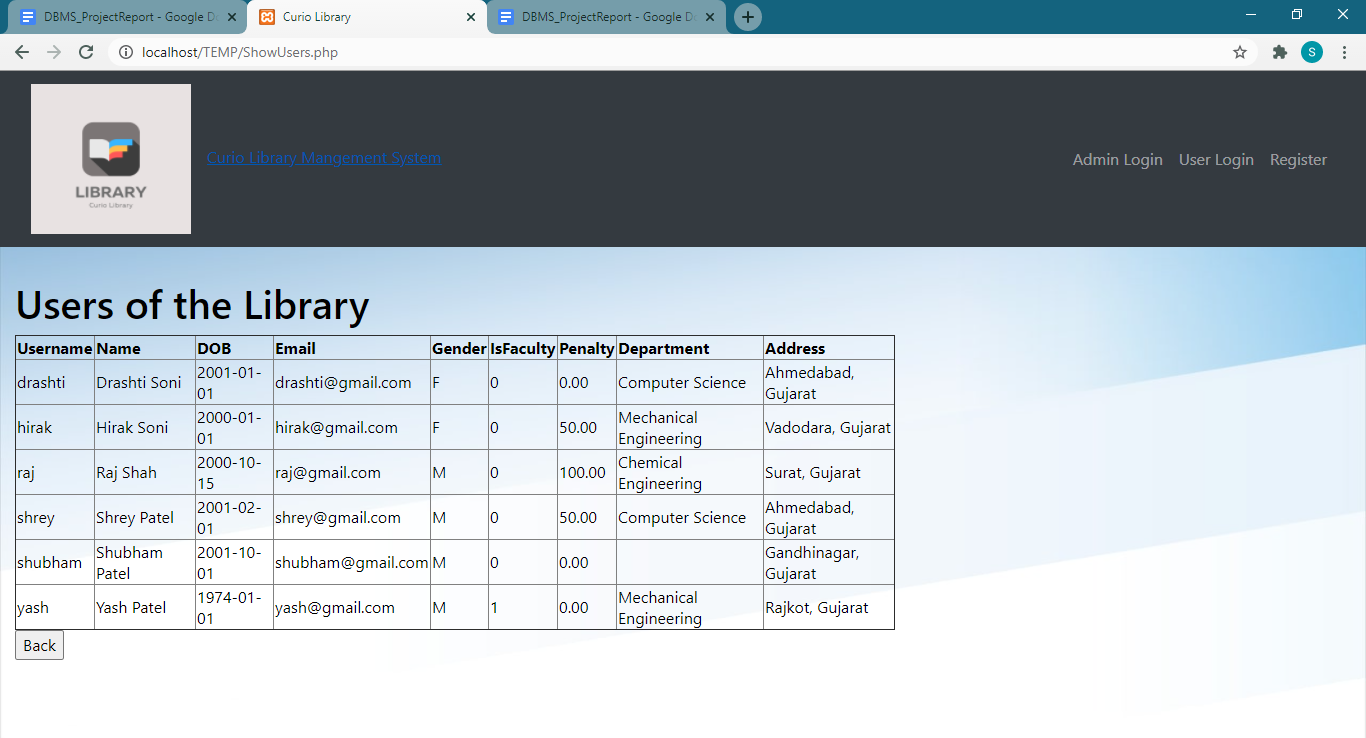
**8. createprofile**

****

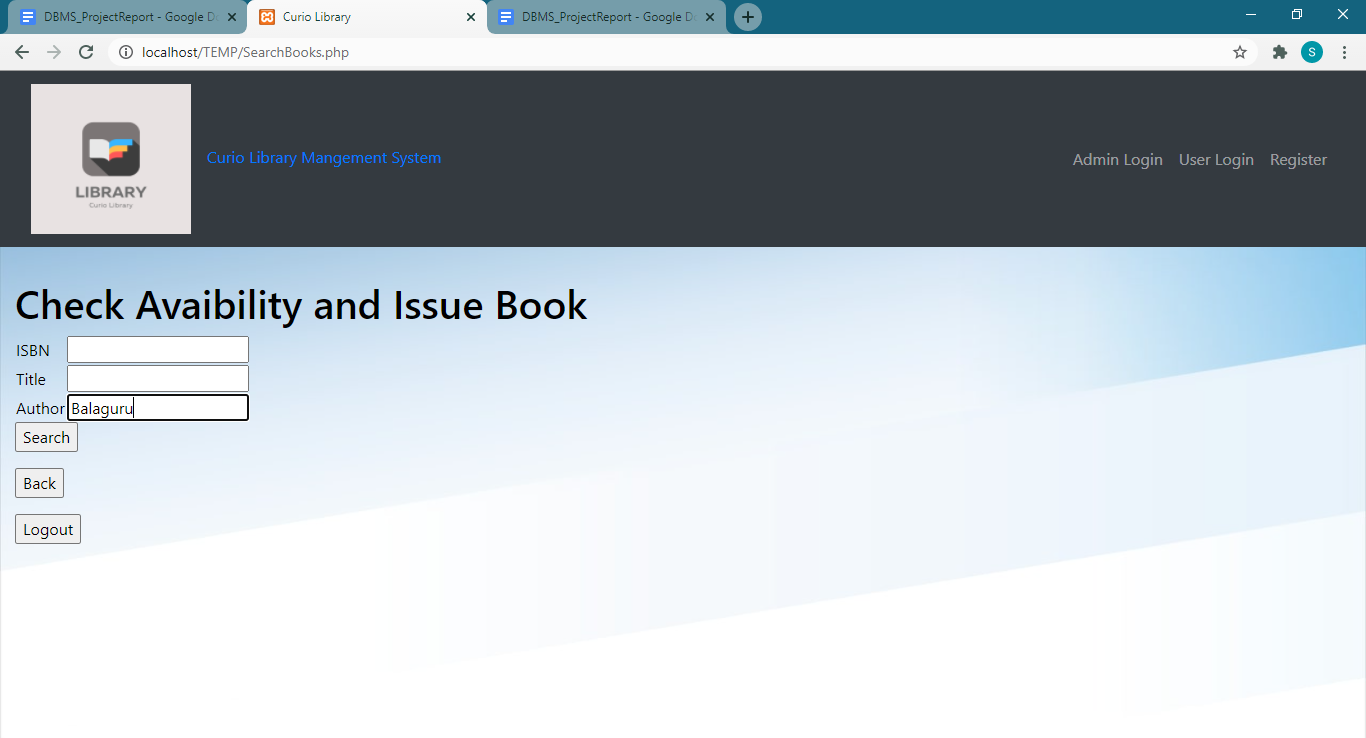
**9. showmyfine**

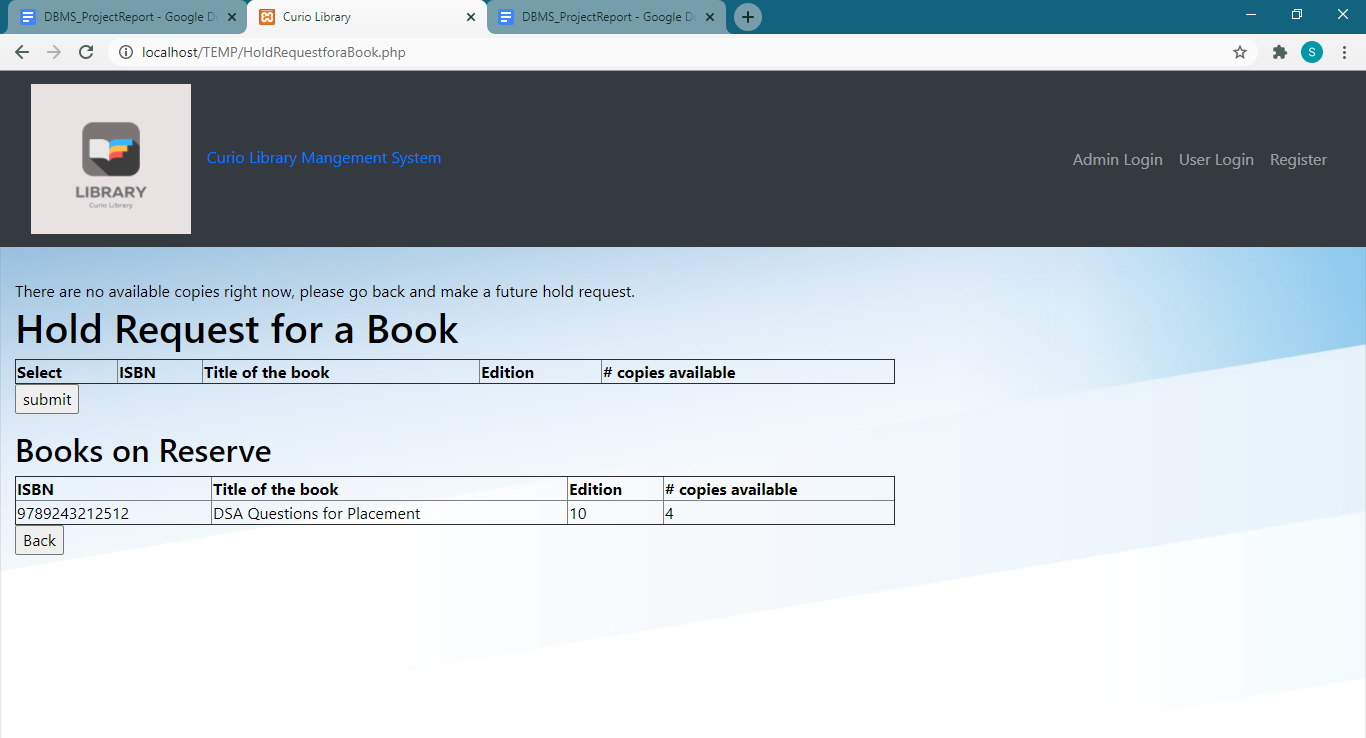
****

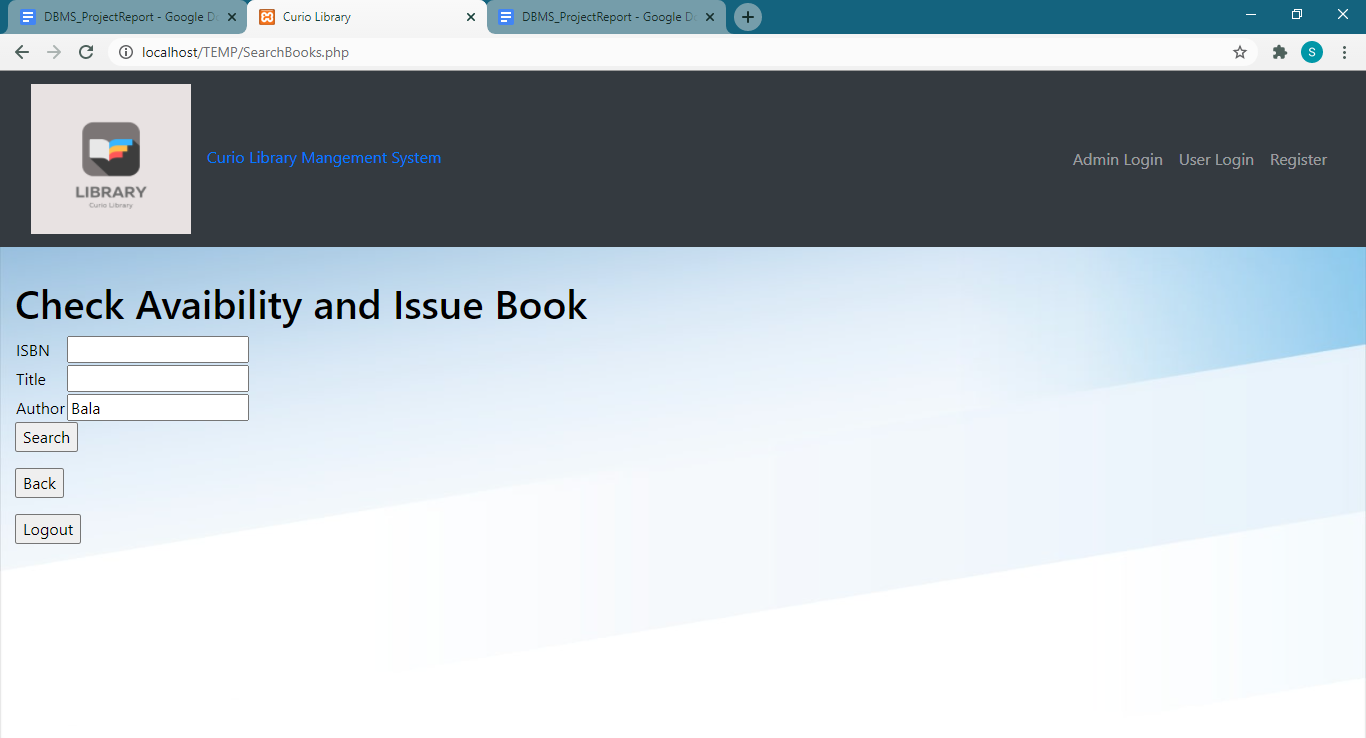
**10. listusers**

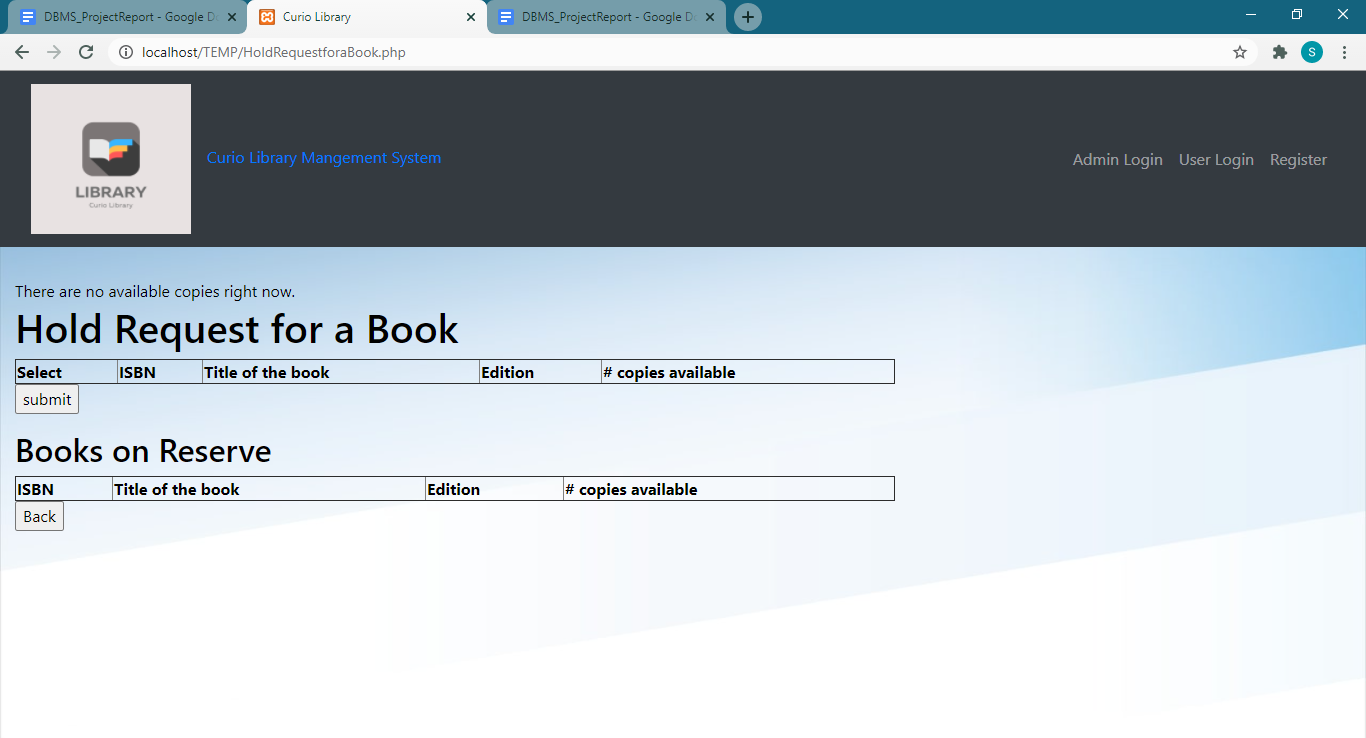
****

**11. searchbookauthor**

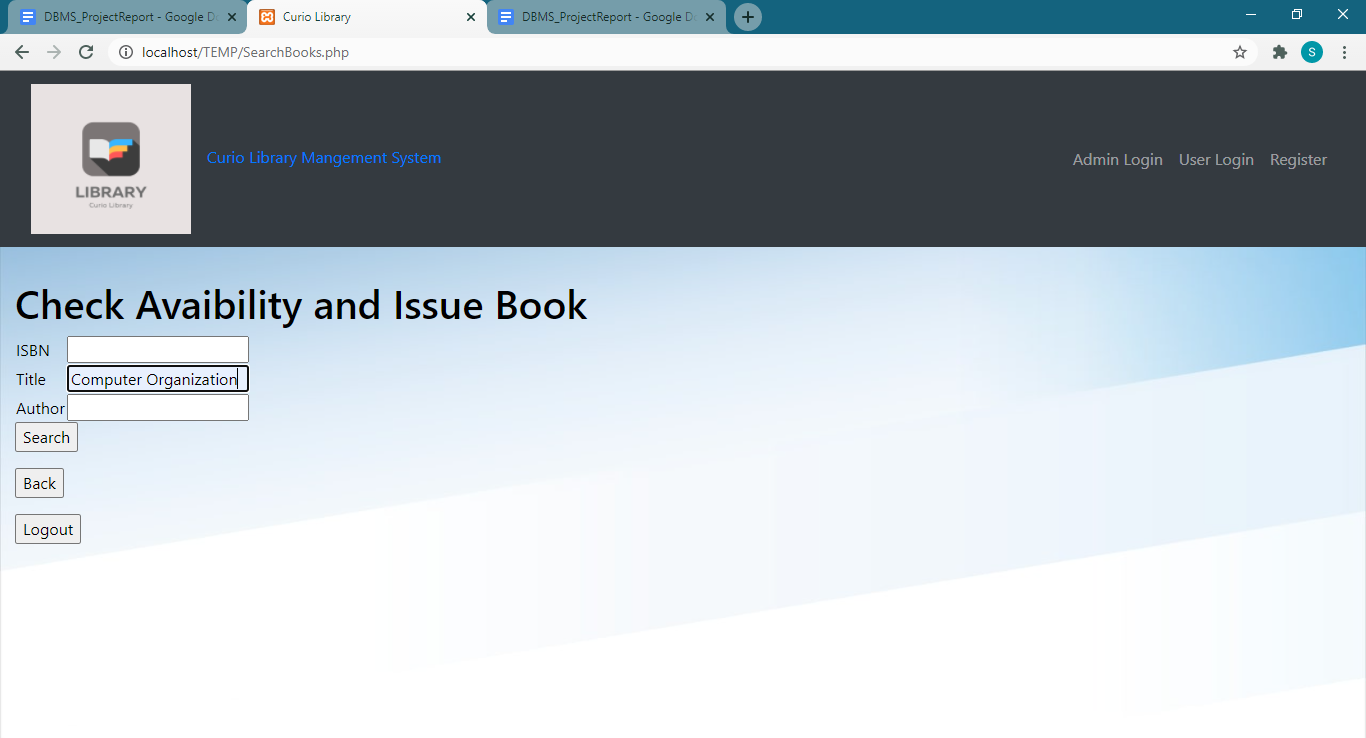
****

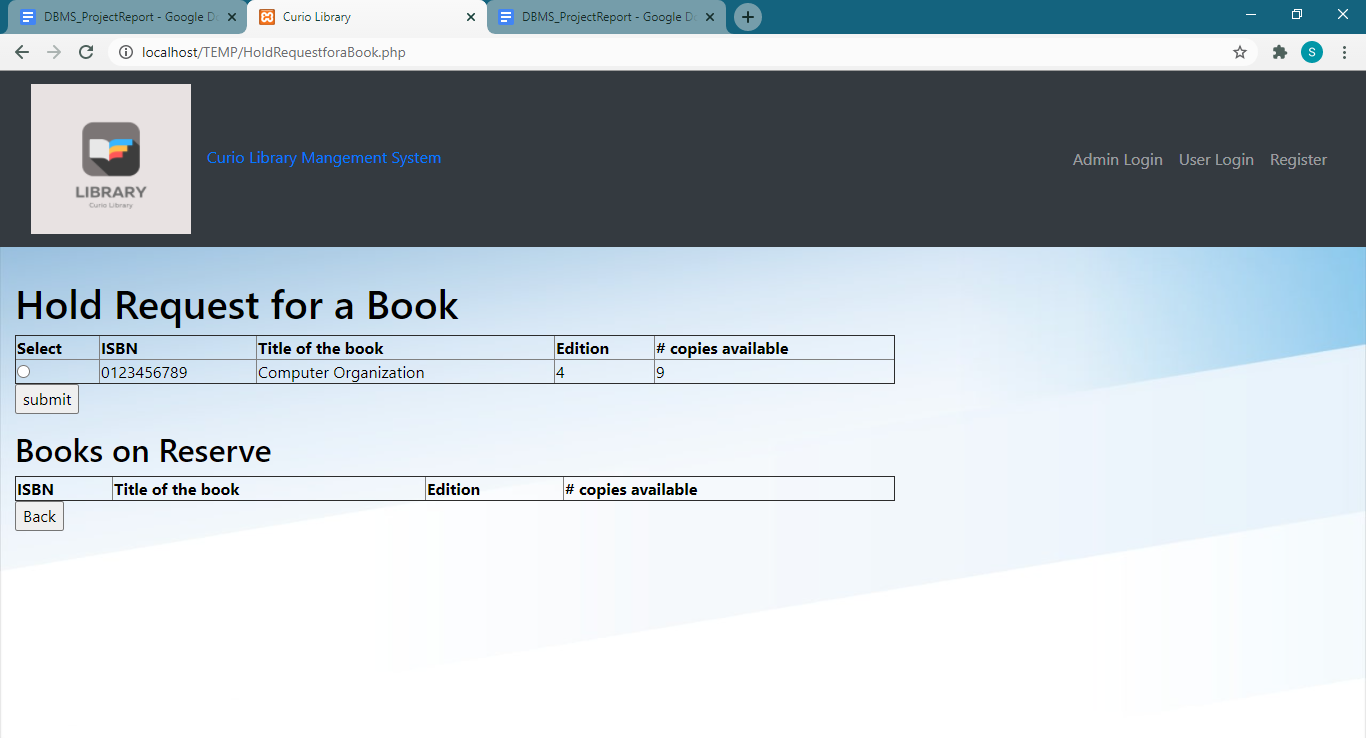
****

****

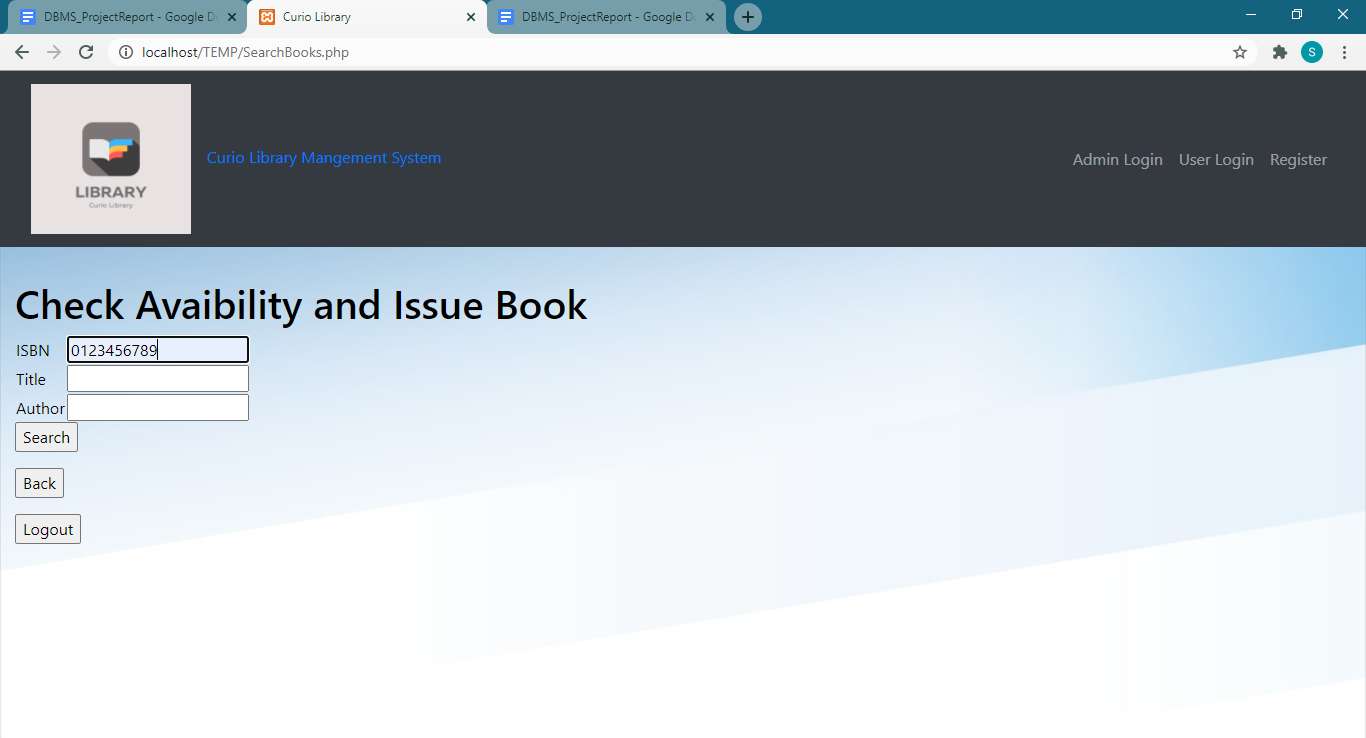
****

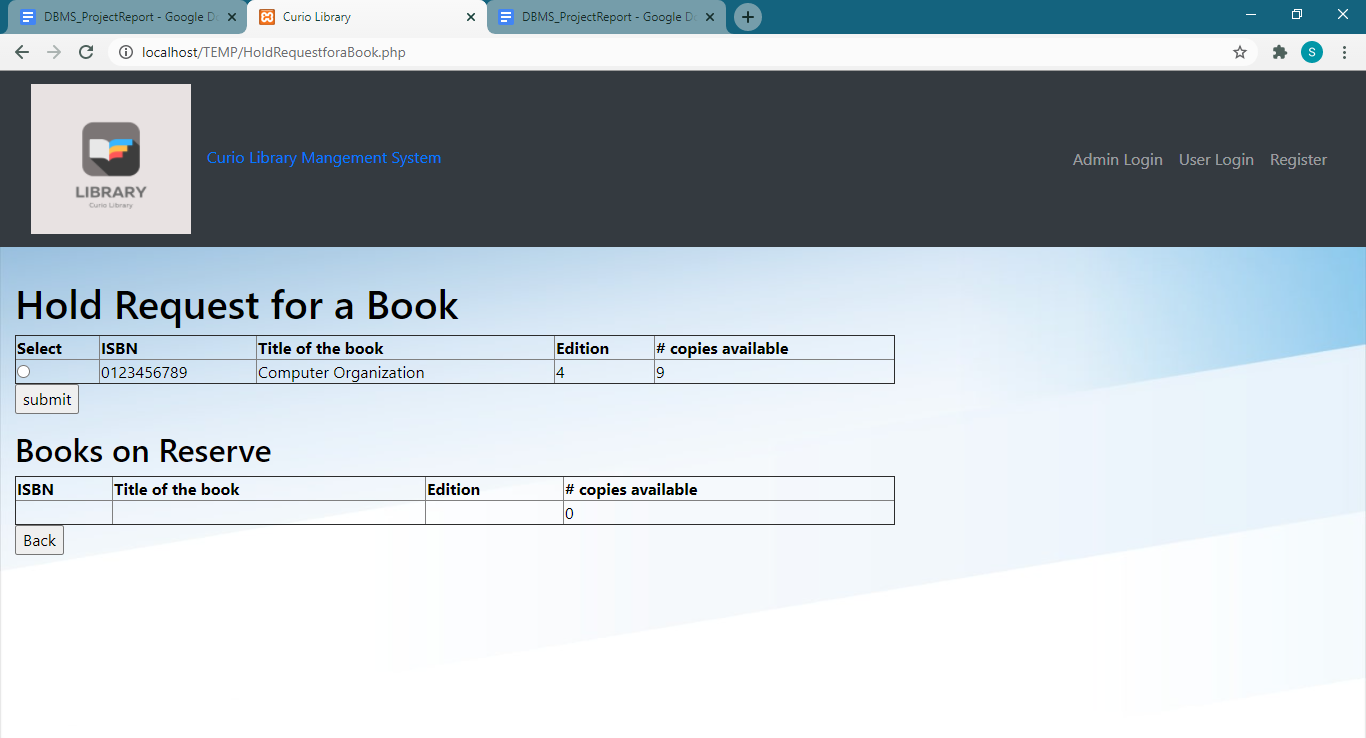
**12. searchbooktitle**

****

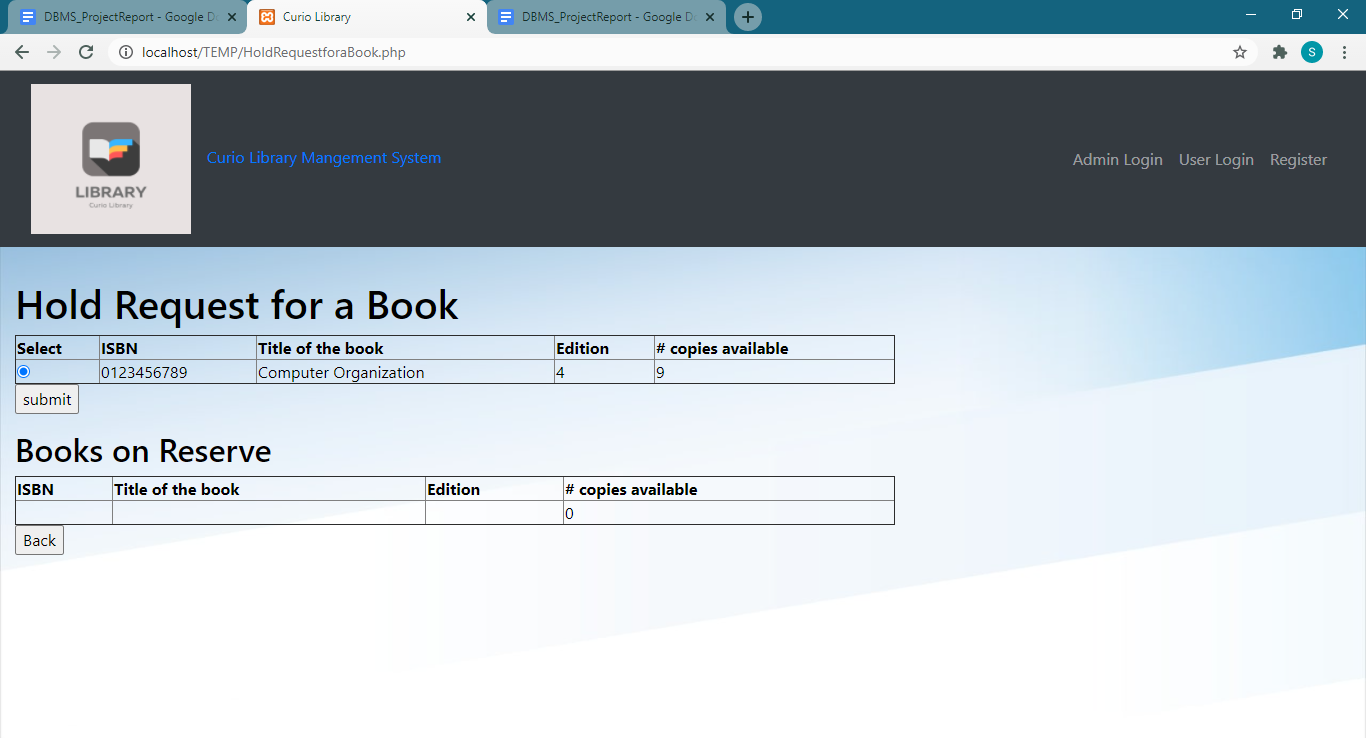
****

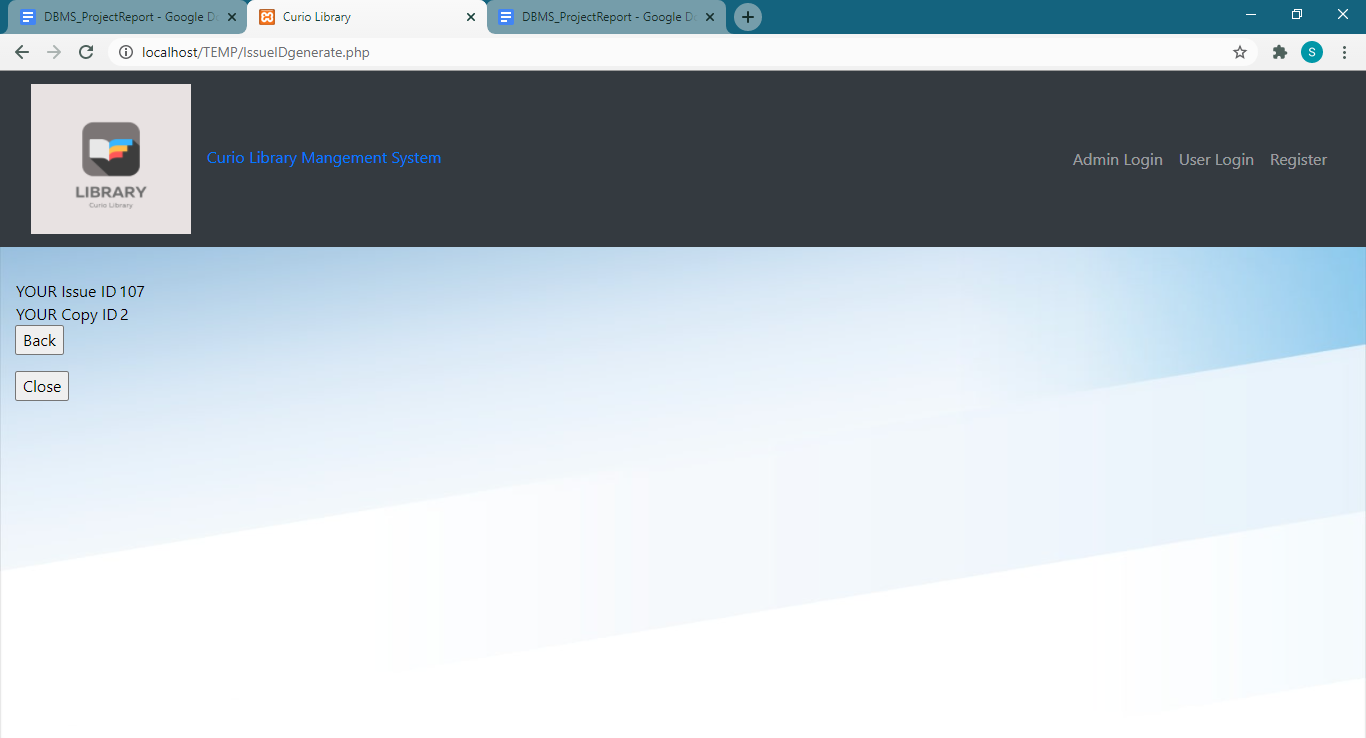
**13. searchbookisbn**

****

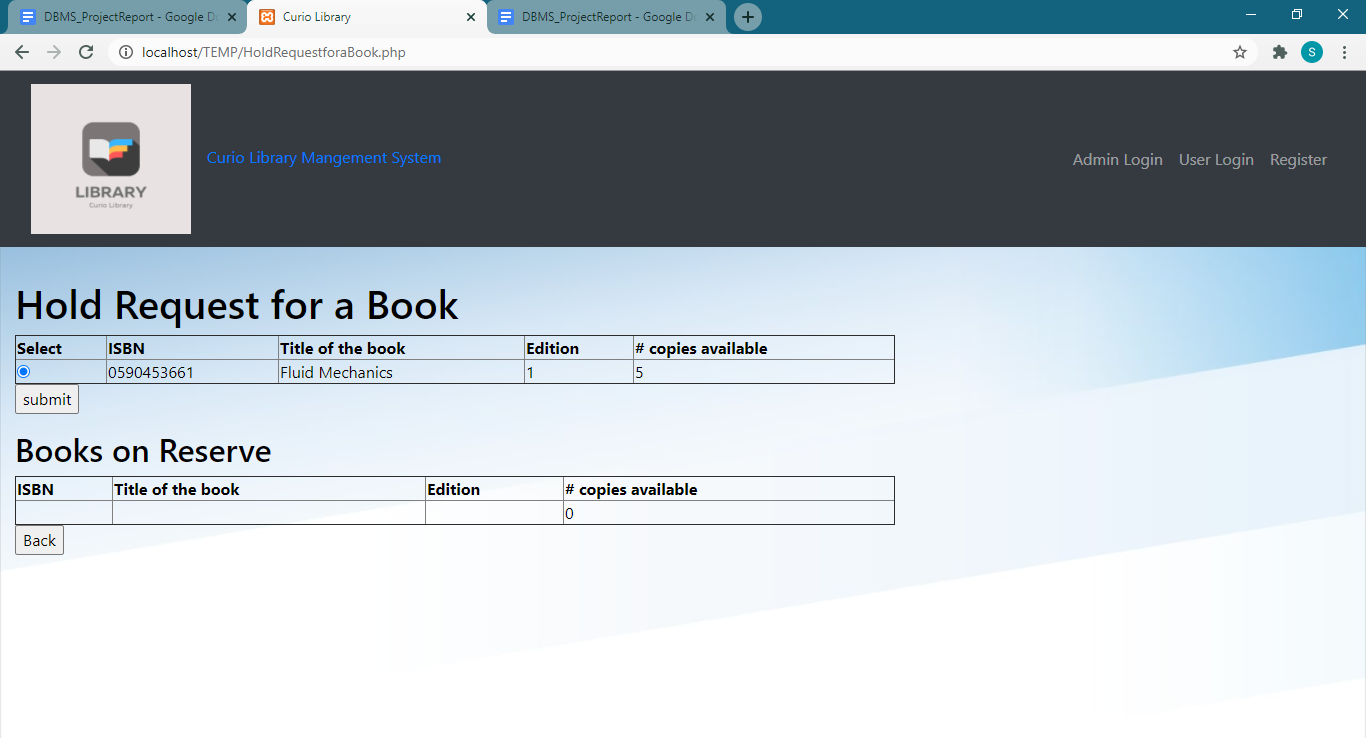
****

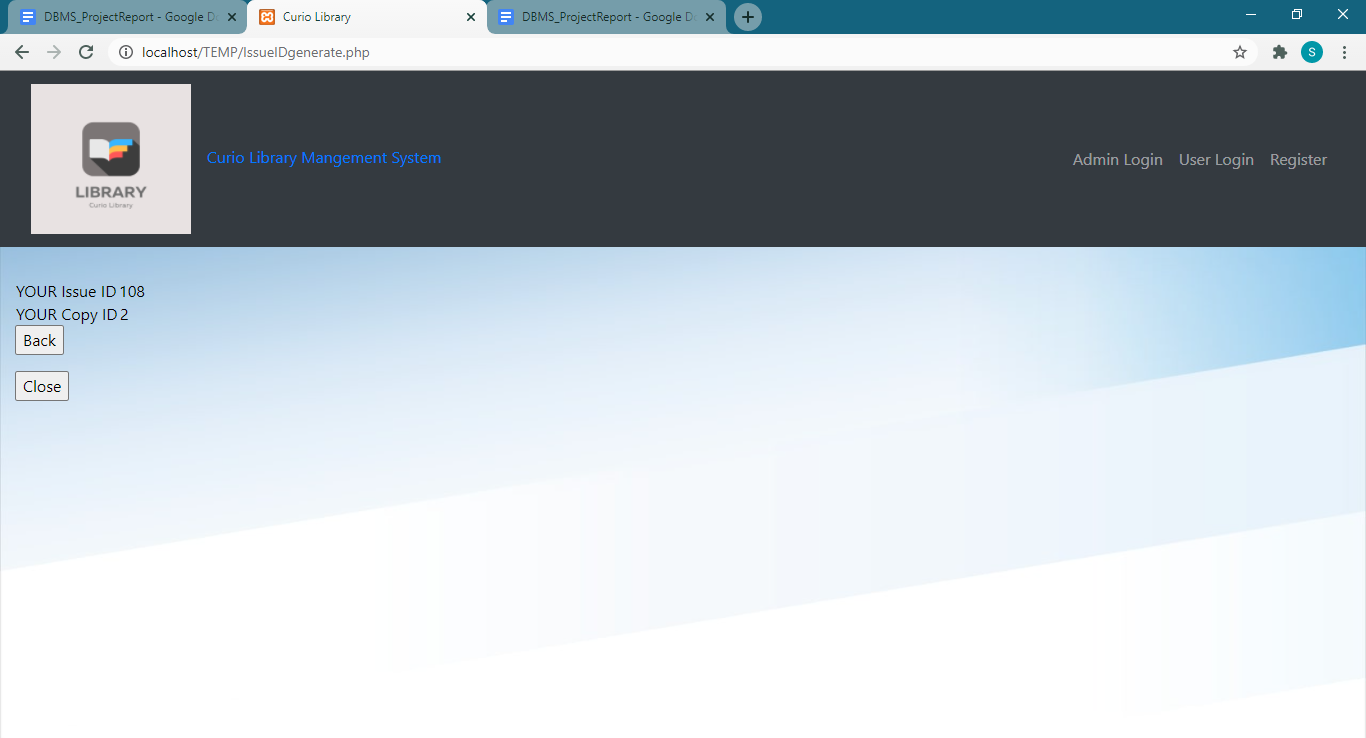
**14. reservebookisbn**

****

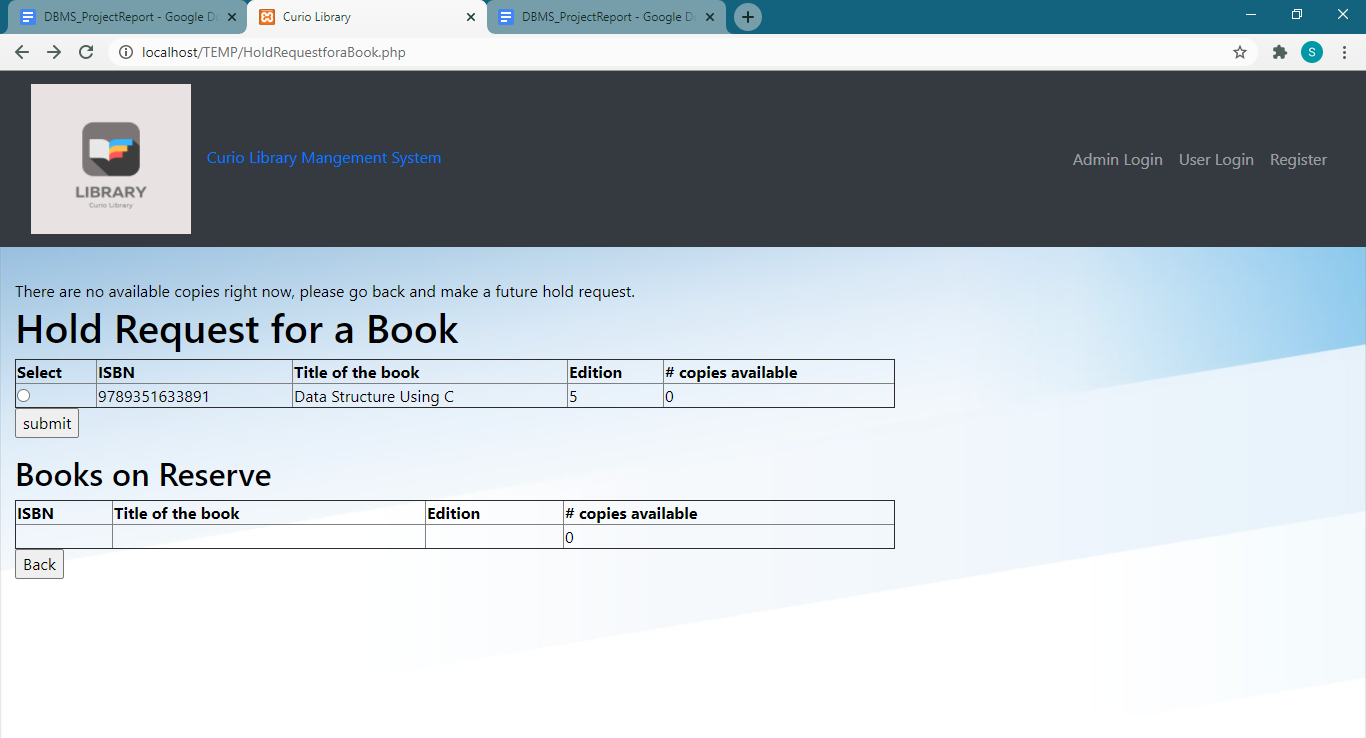
****

**15.  reservebookauthor**

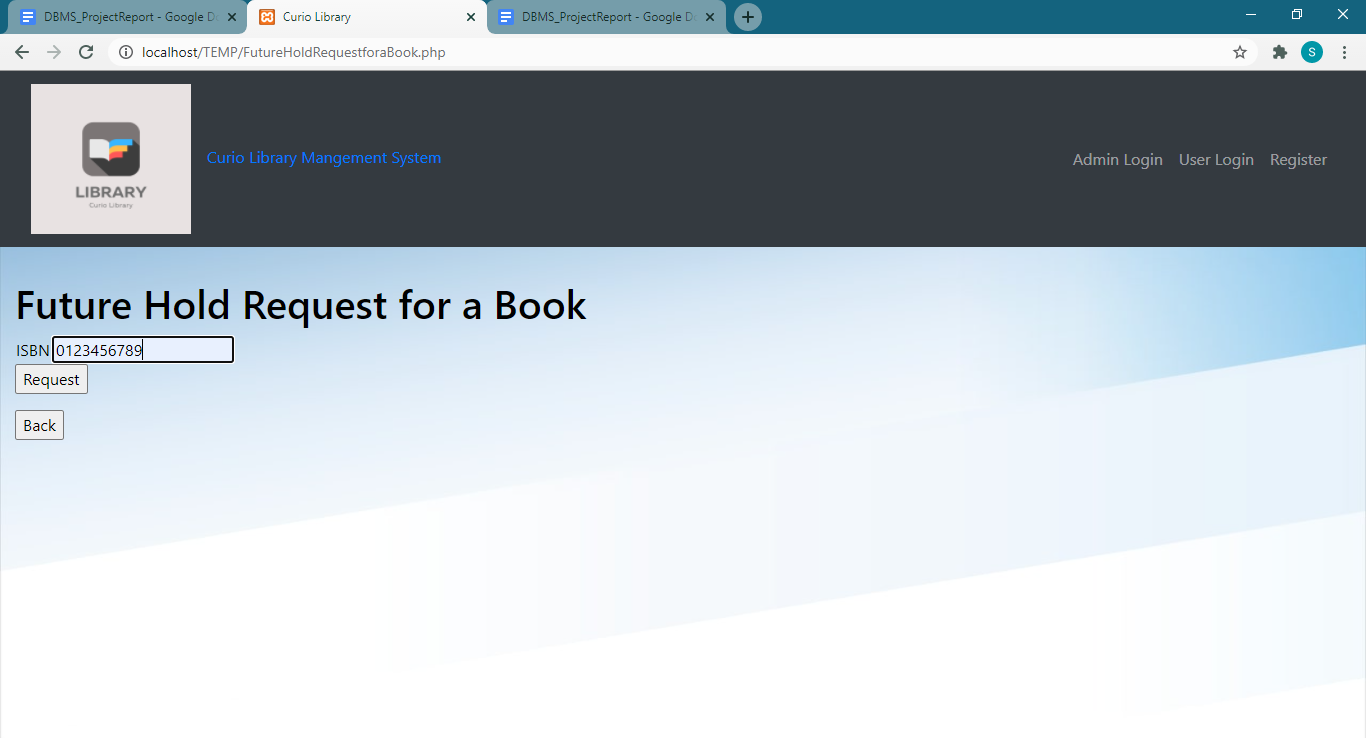
****

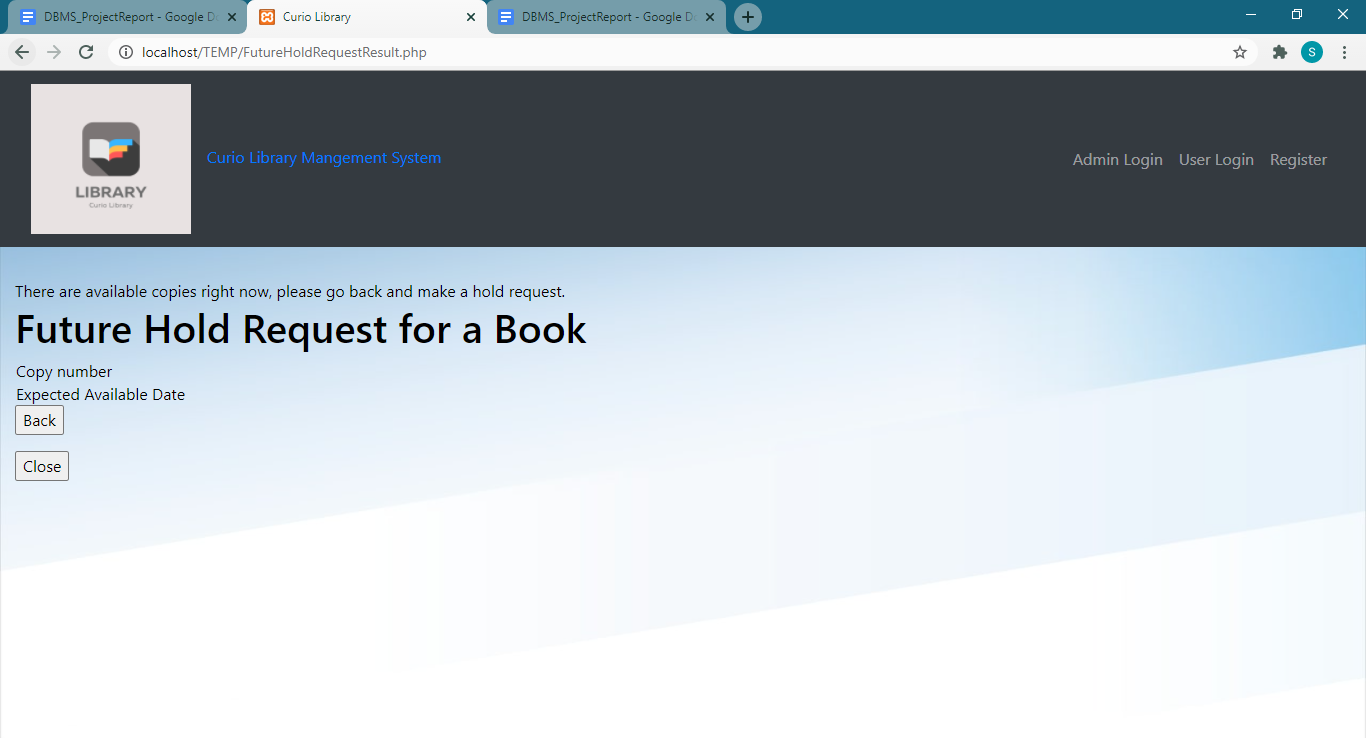
****

**16.  reservebooktitle**

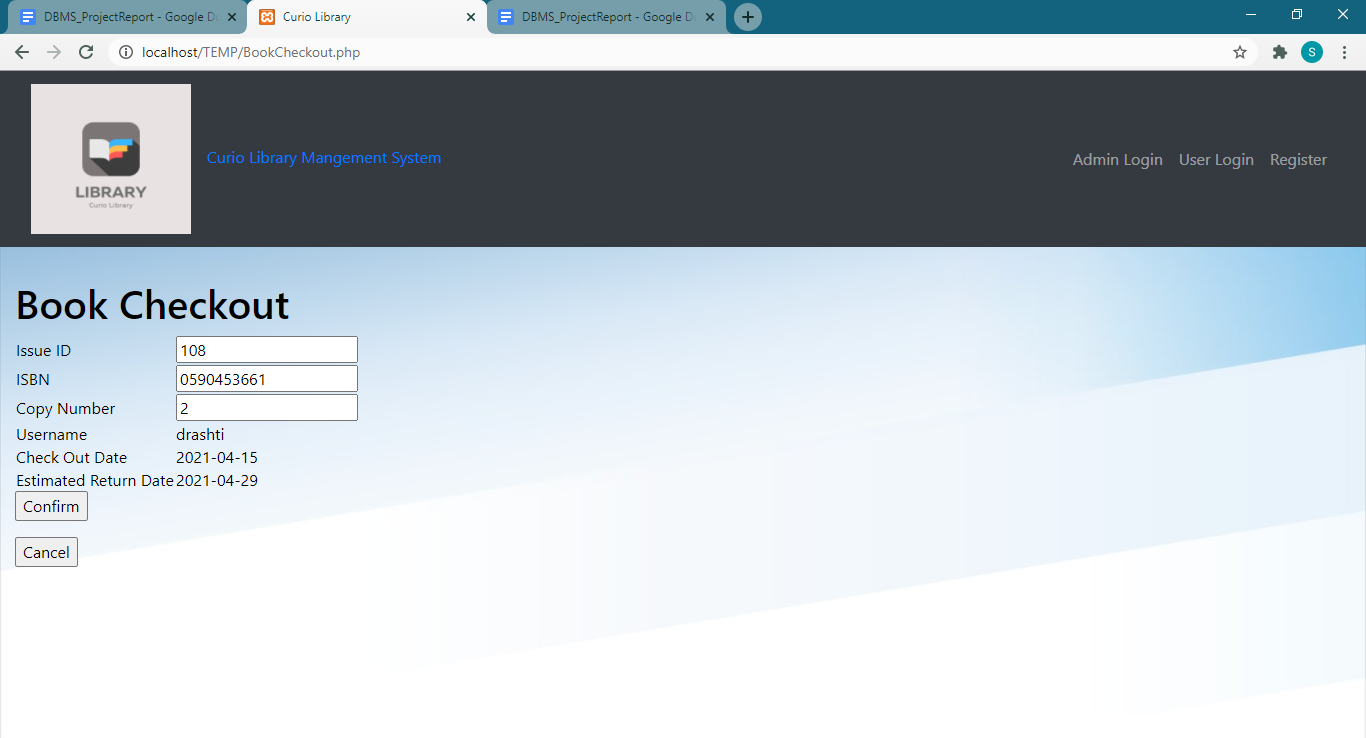
****

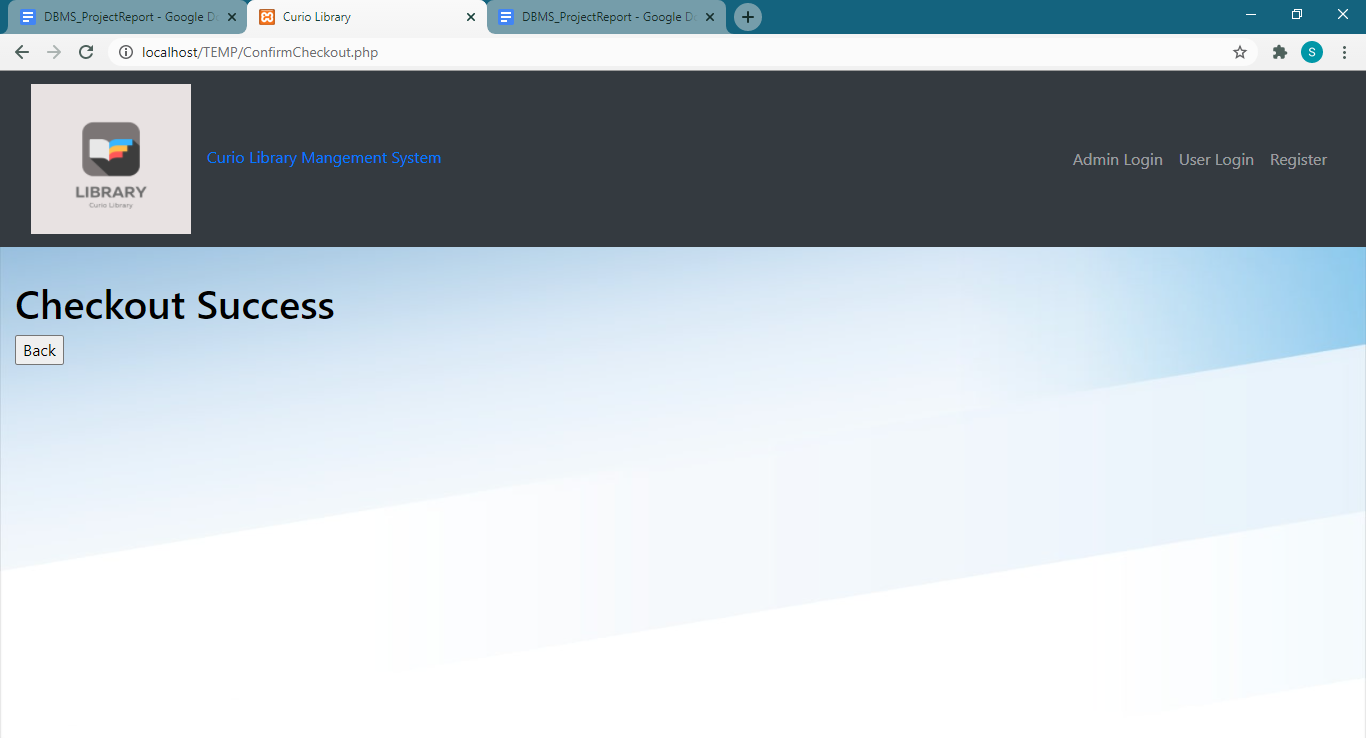
**17. futureholdrequest**

****

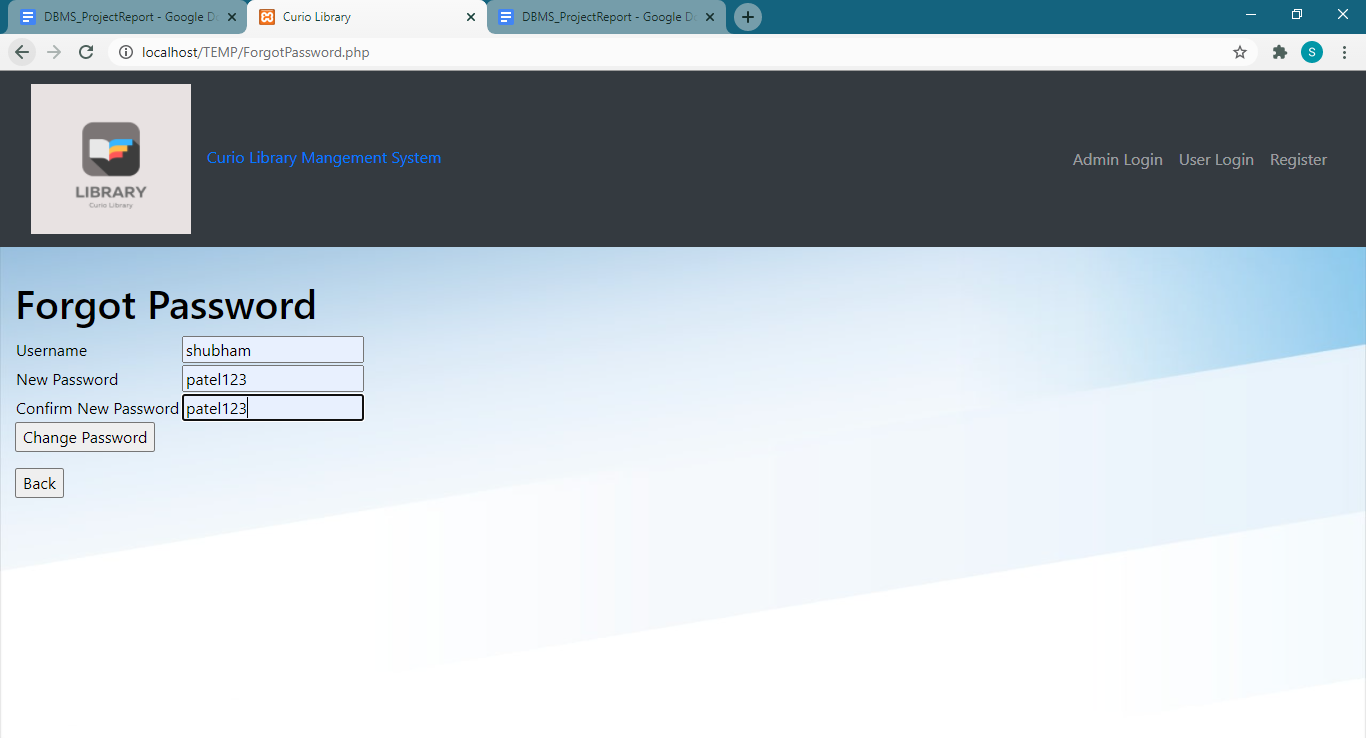
****

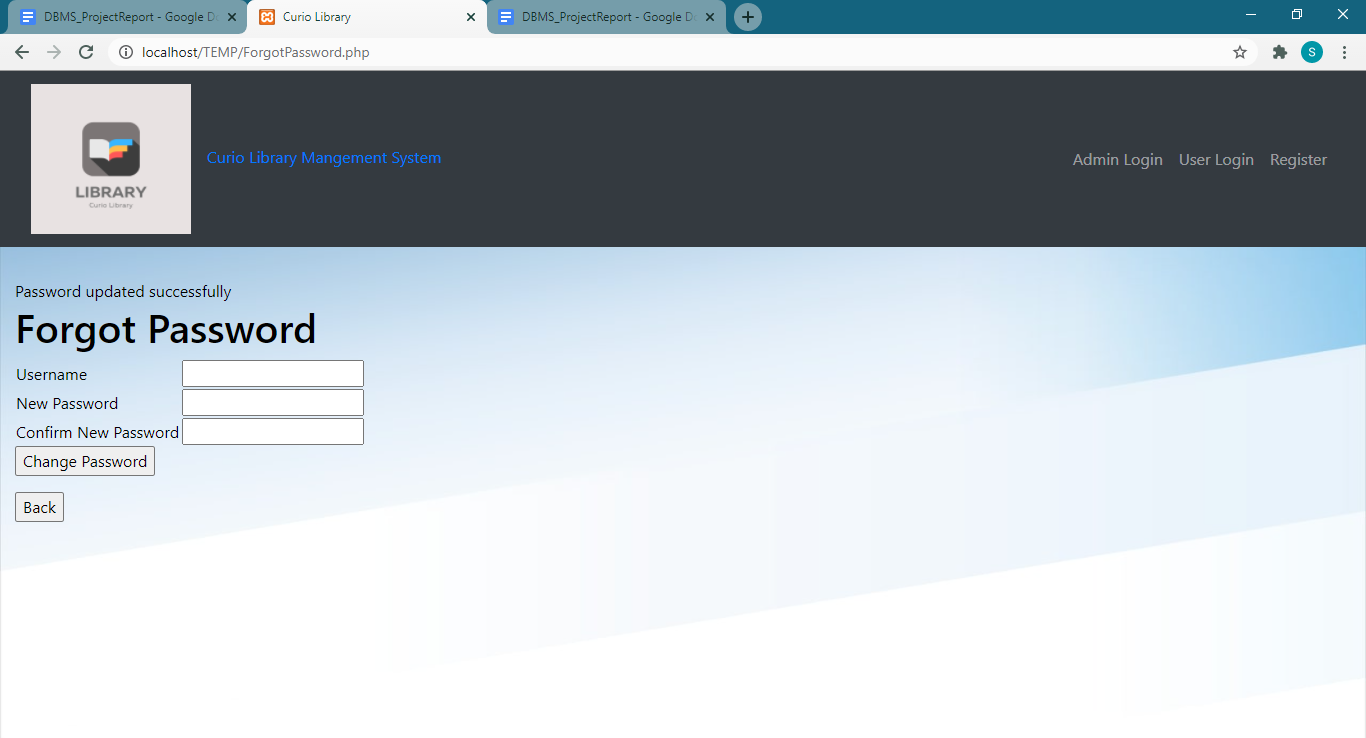
**18. checkoutbook**

****

****

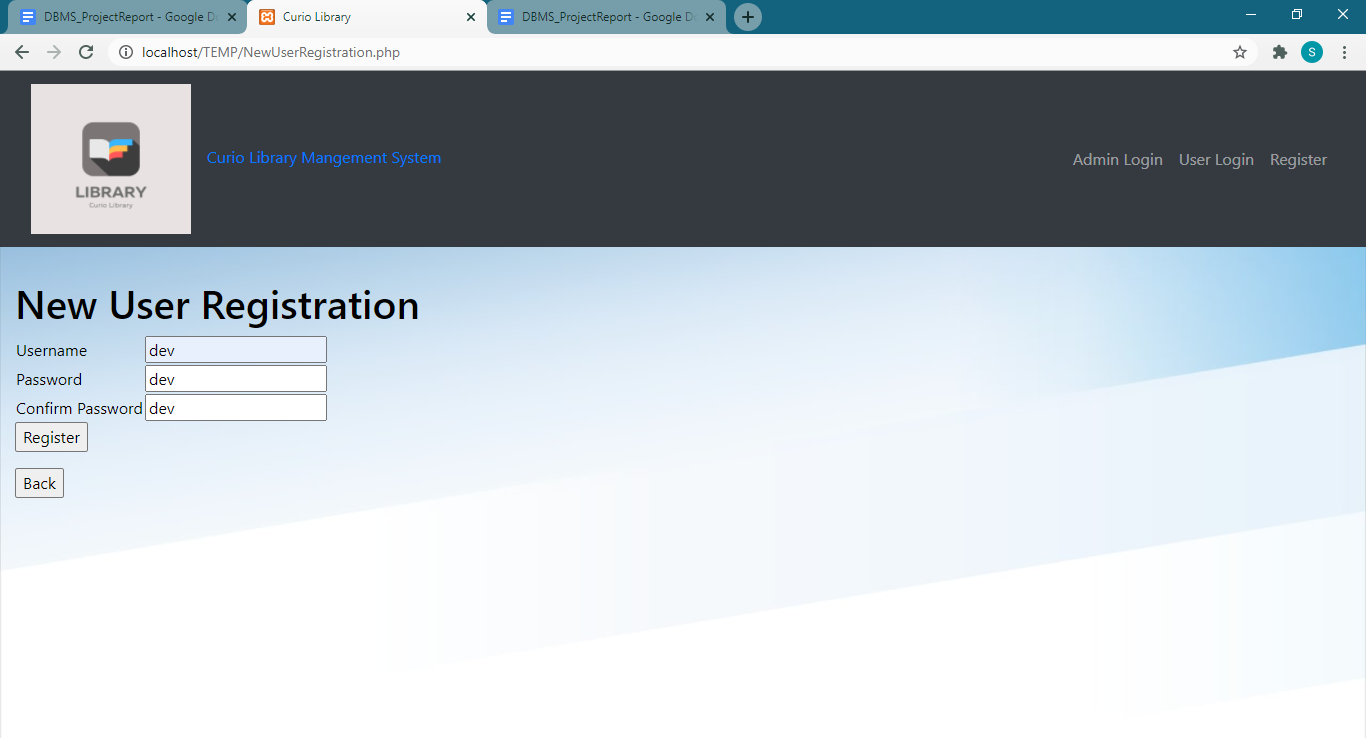
**19. updatepassword**

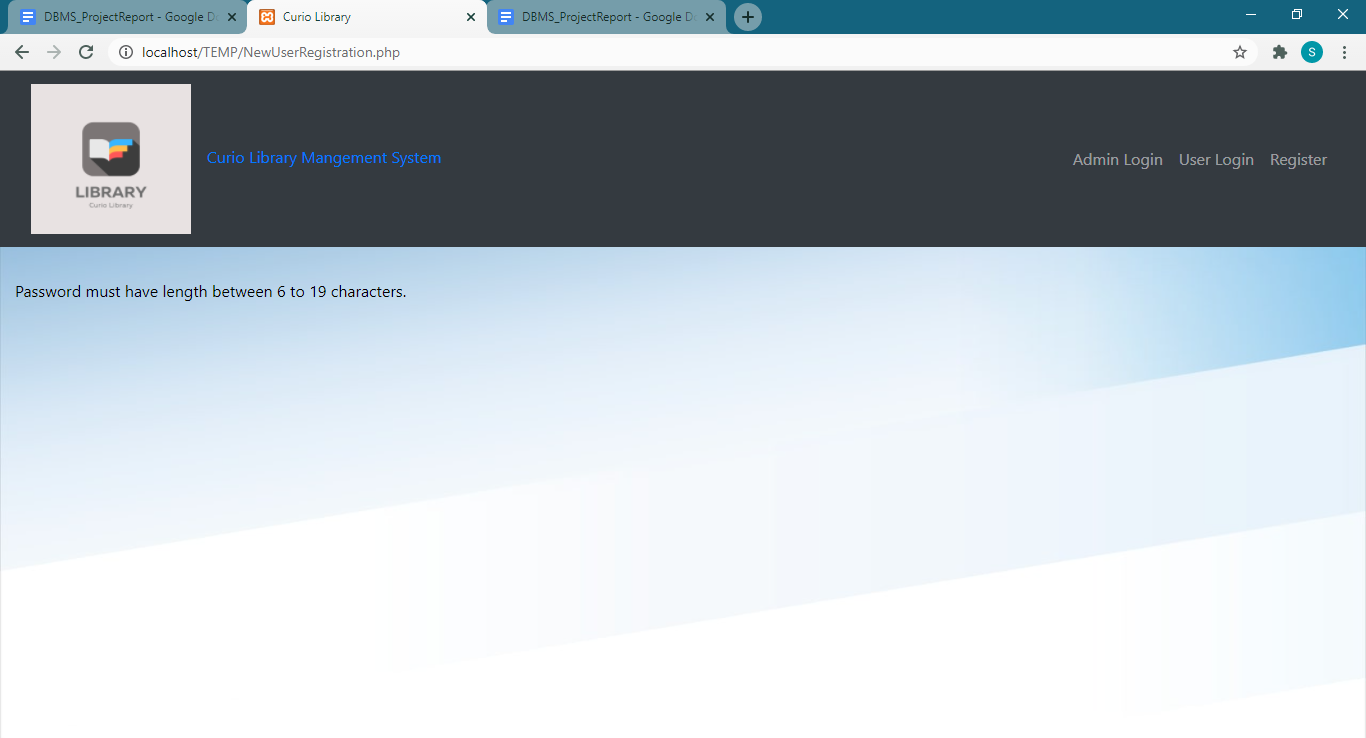
****

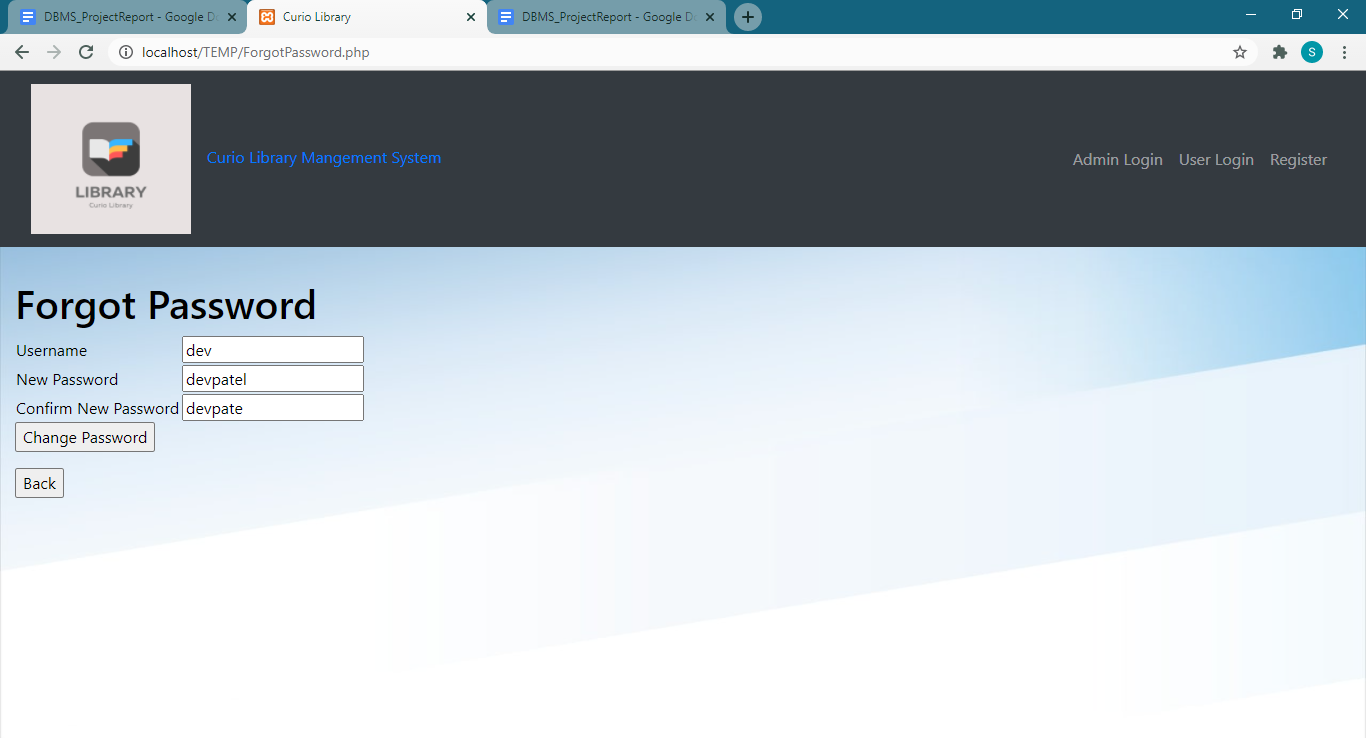
****

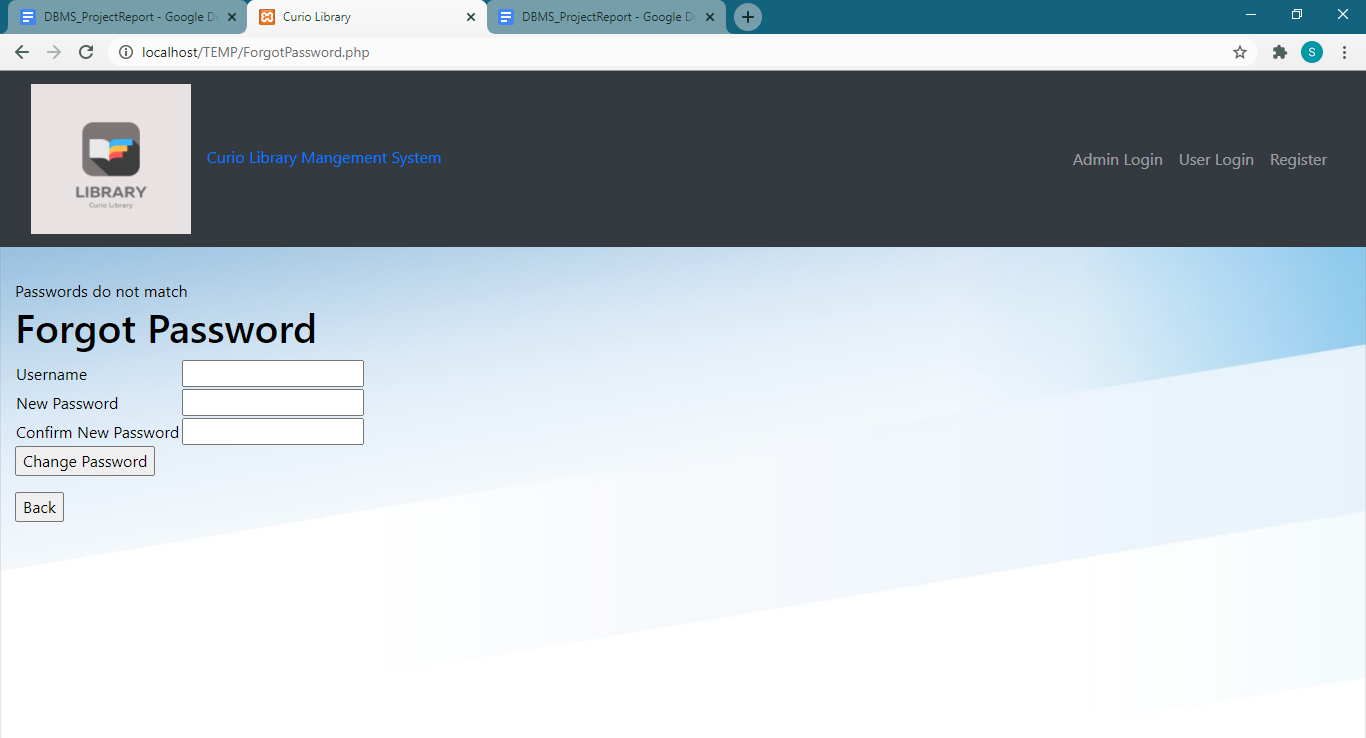
**F.**   **Screen-shots of errors generated on the front-end when the trigger is fired.**

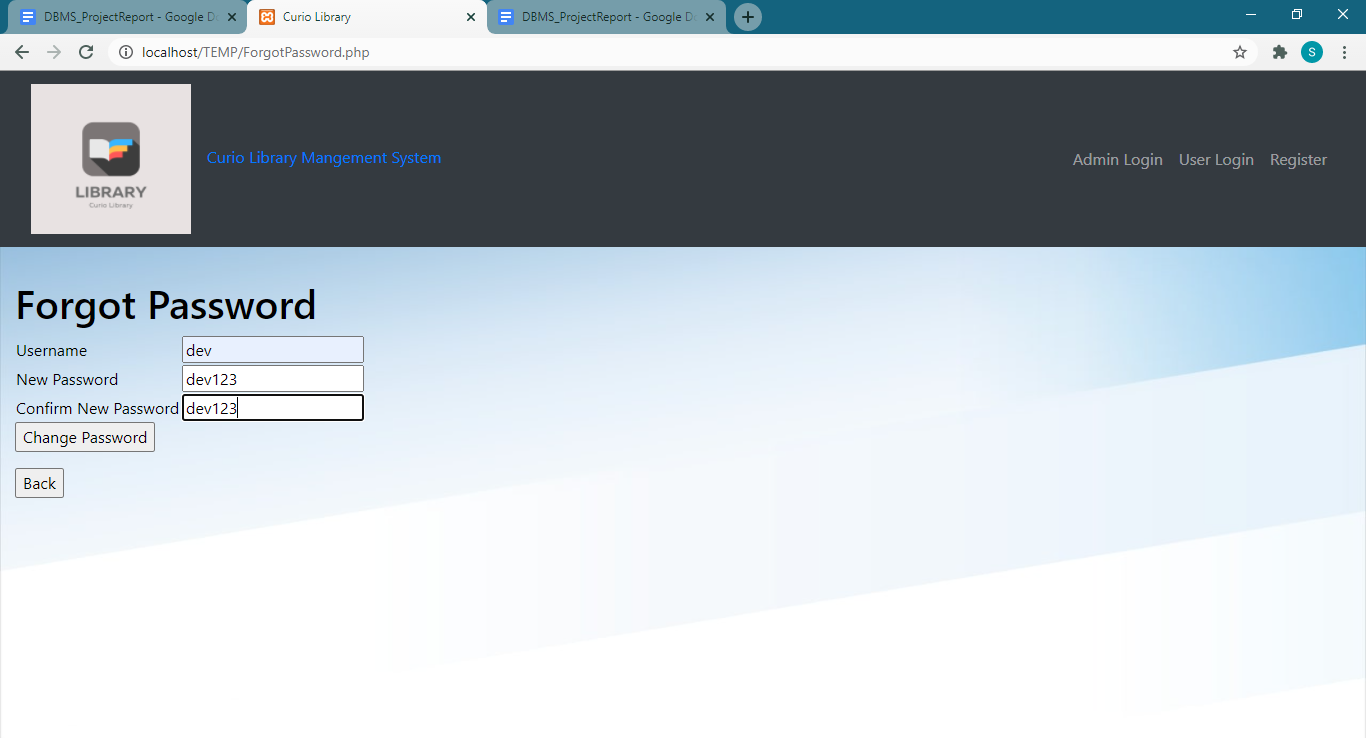
**1. checkpassword**

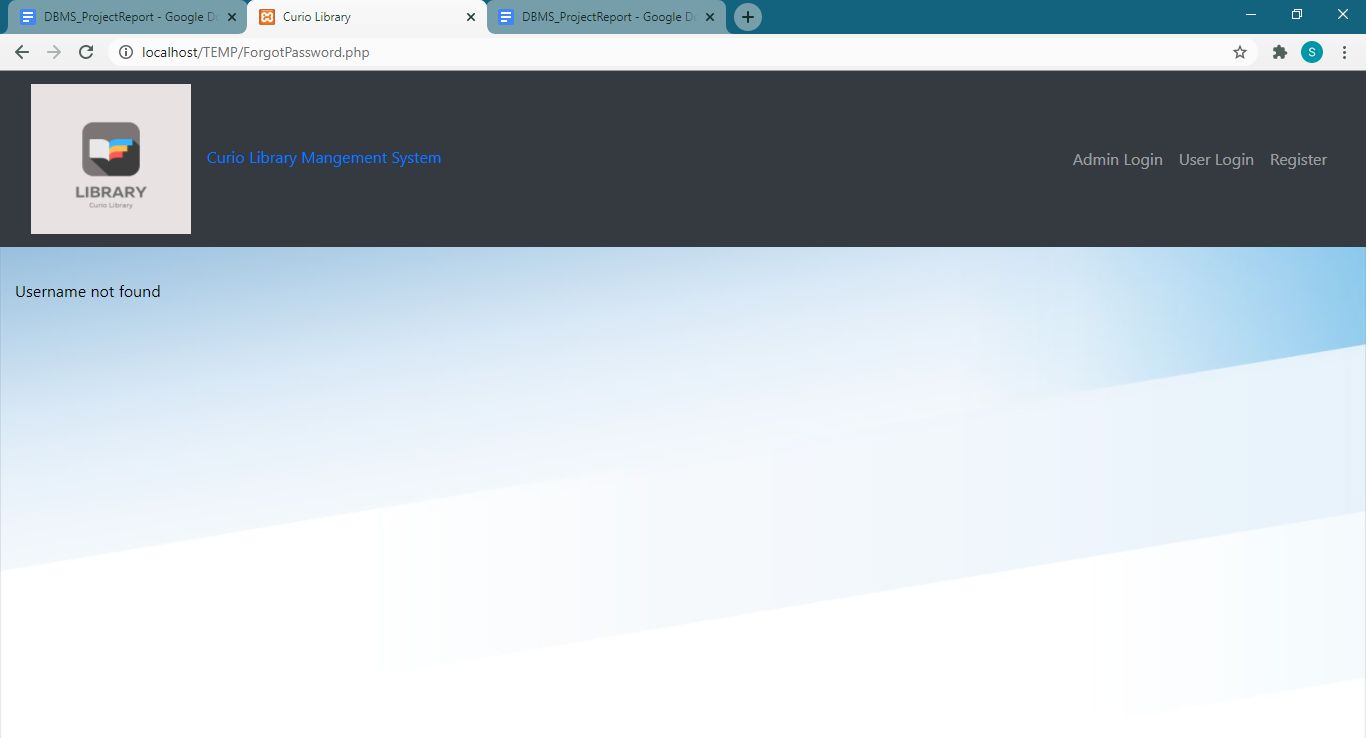
****

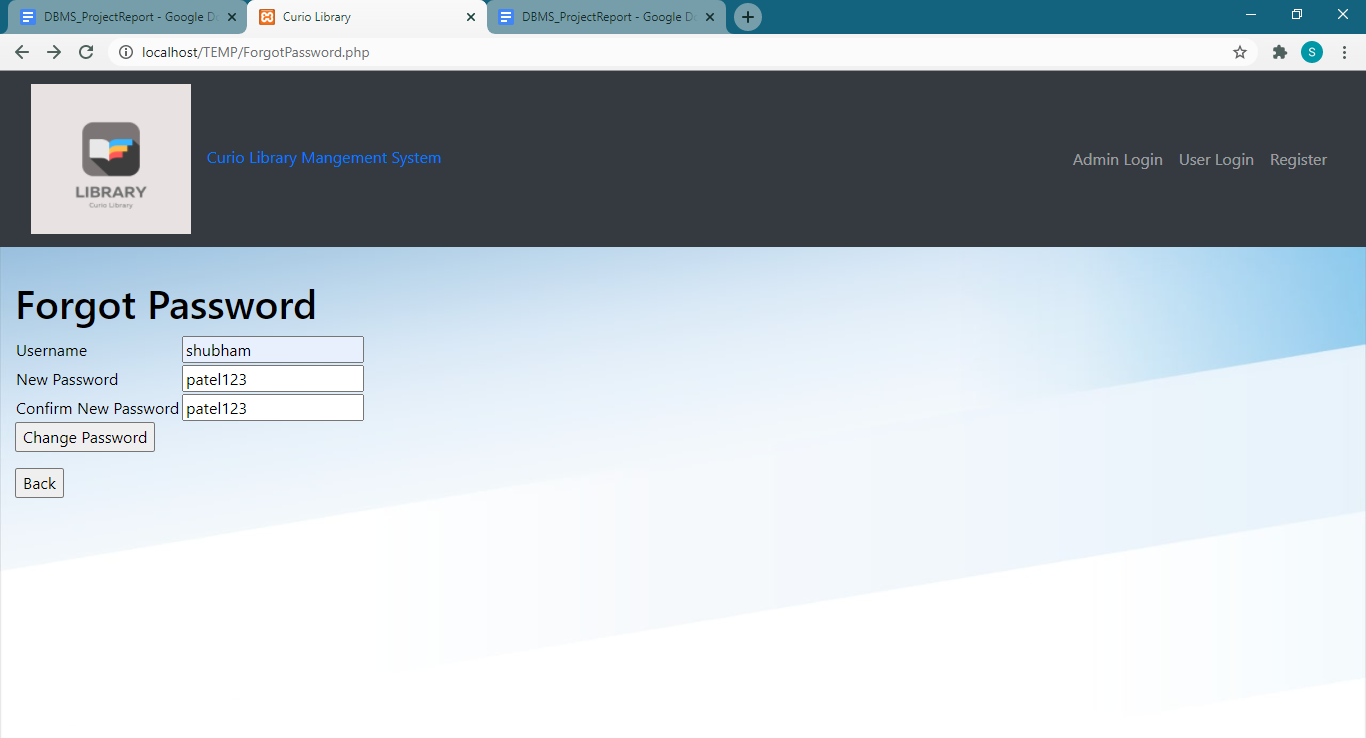


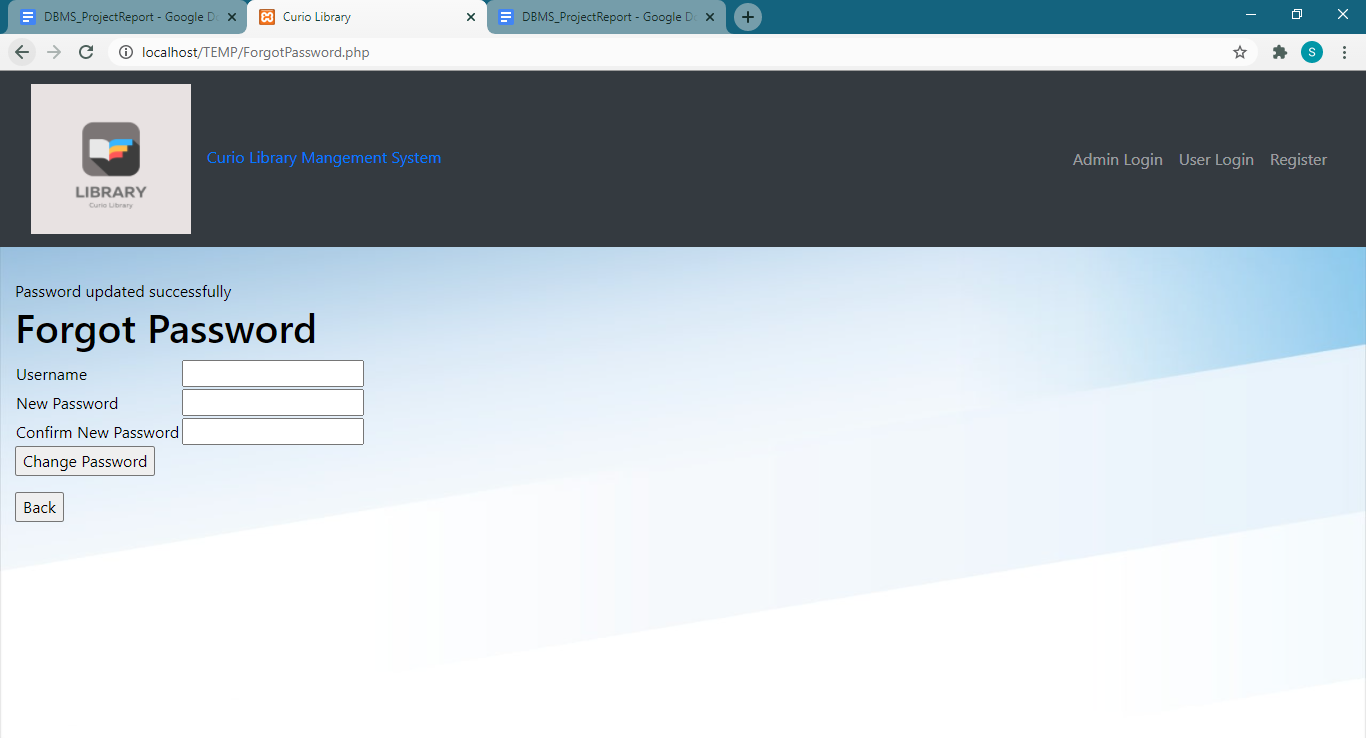




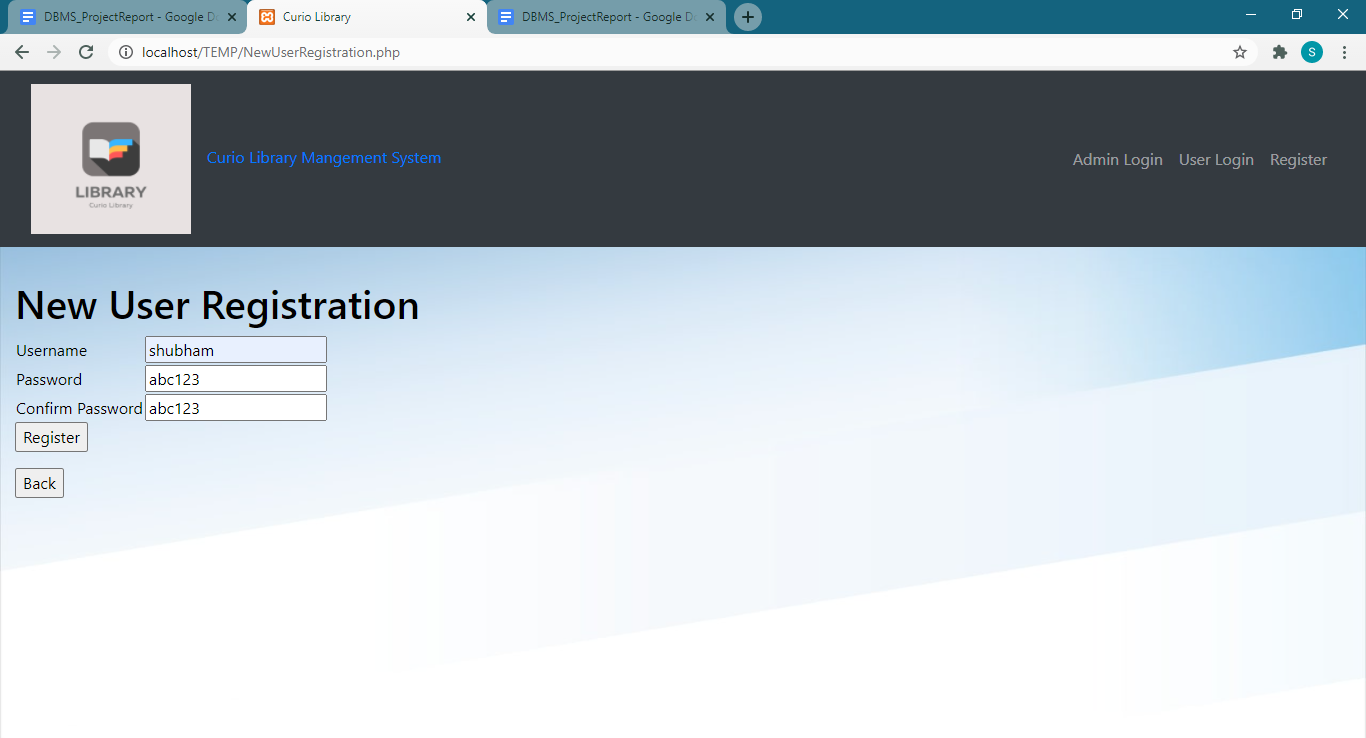






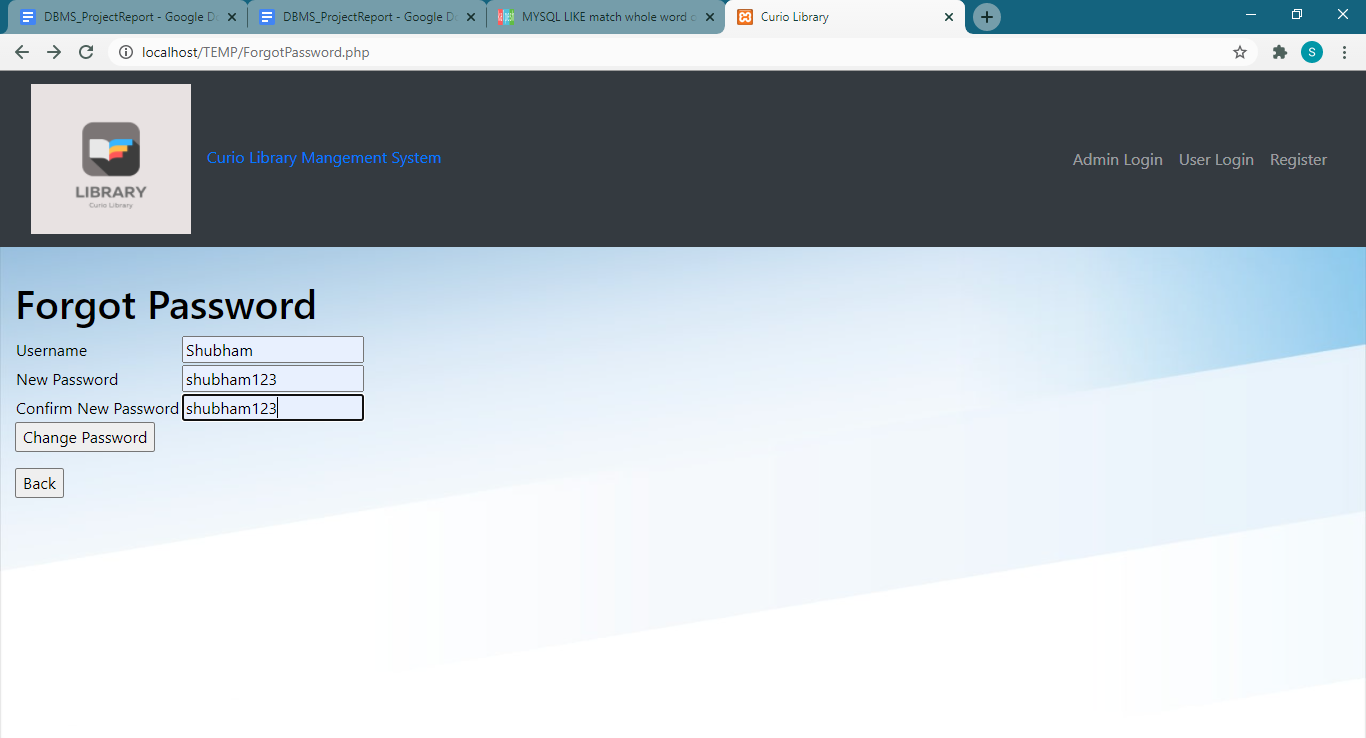


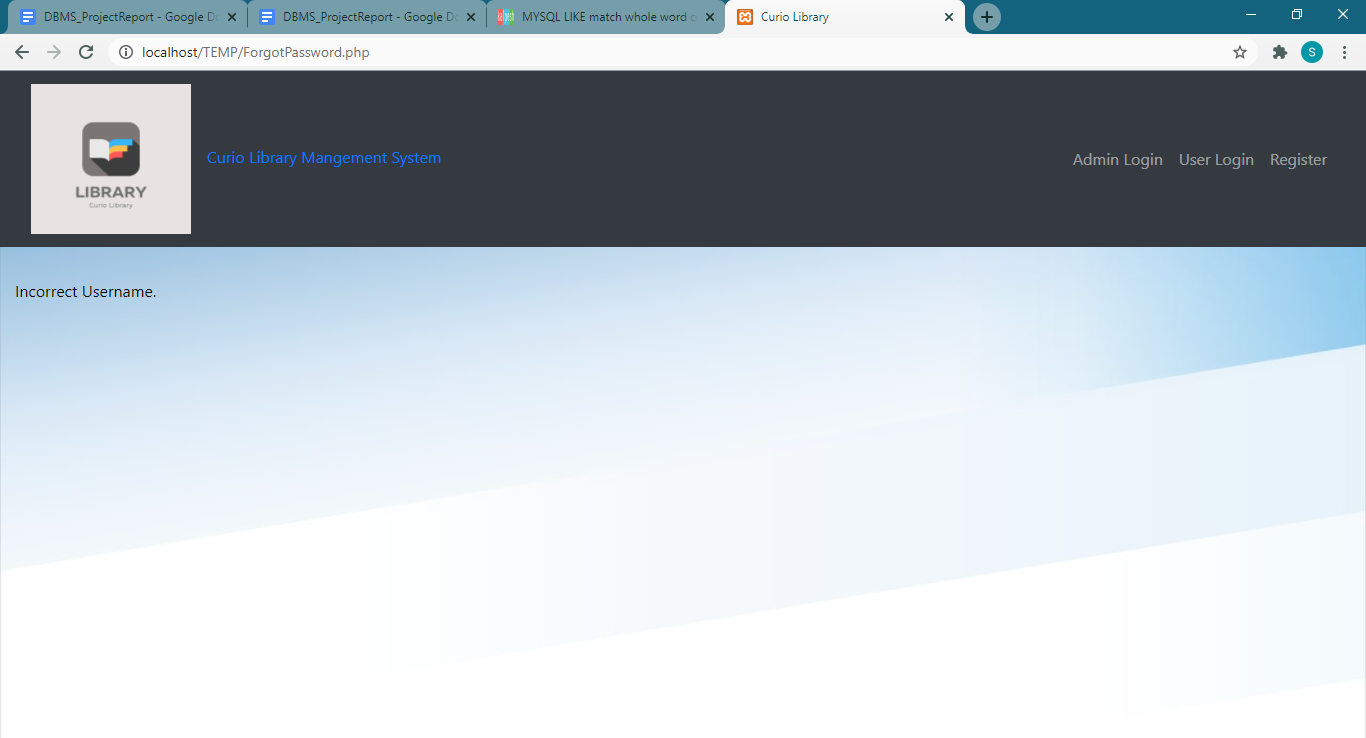
2. **userregistration**



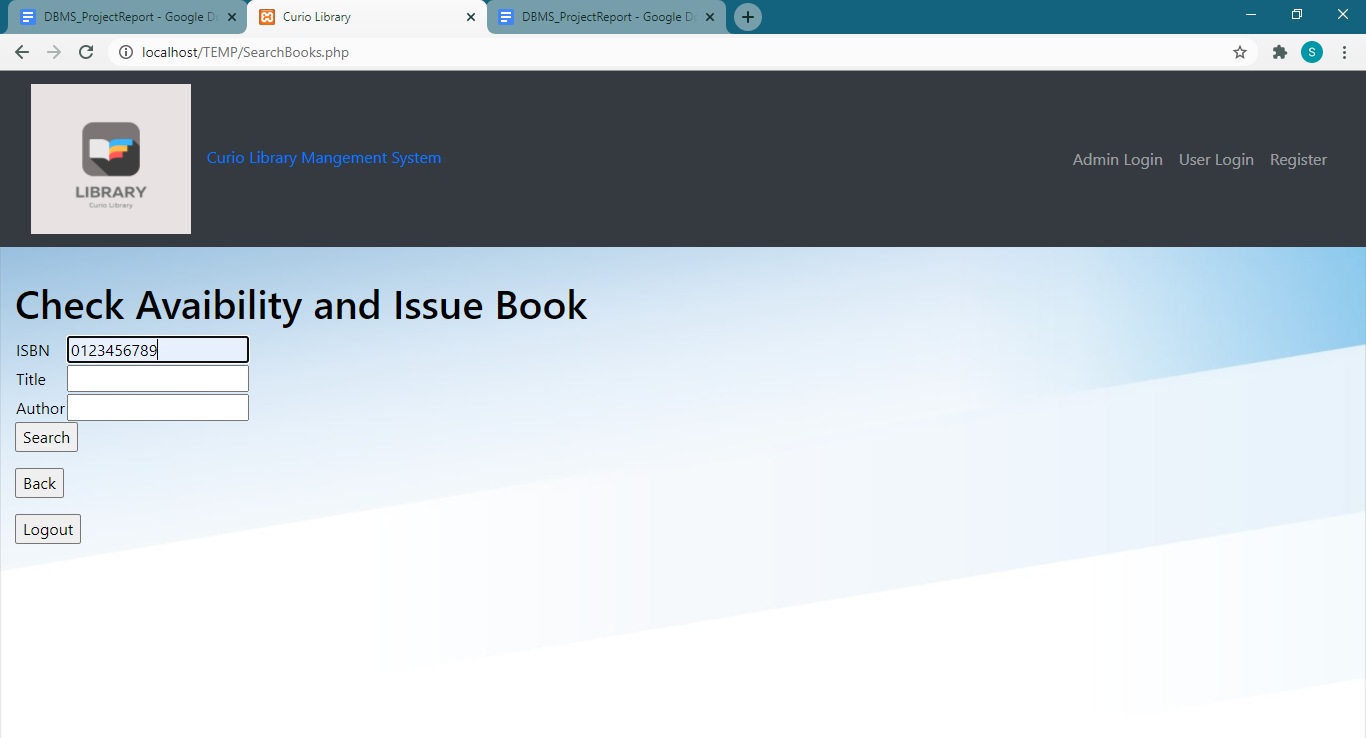


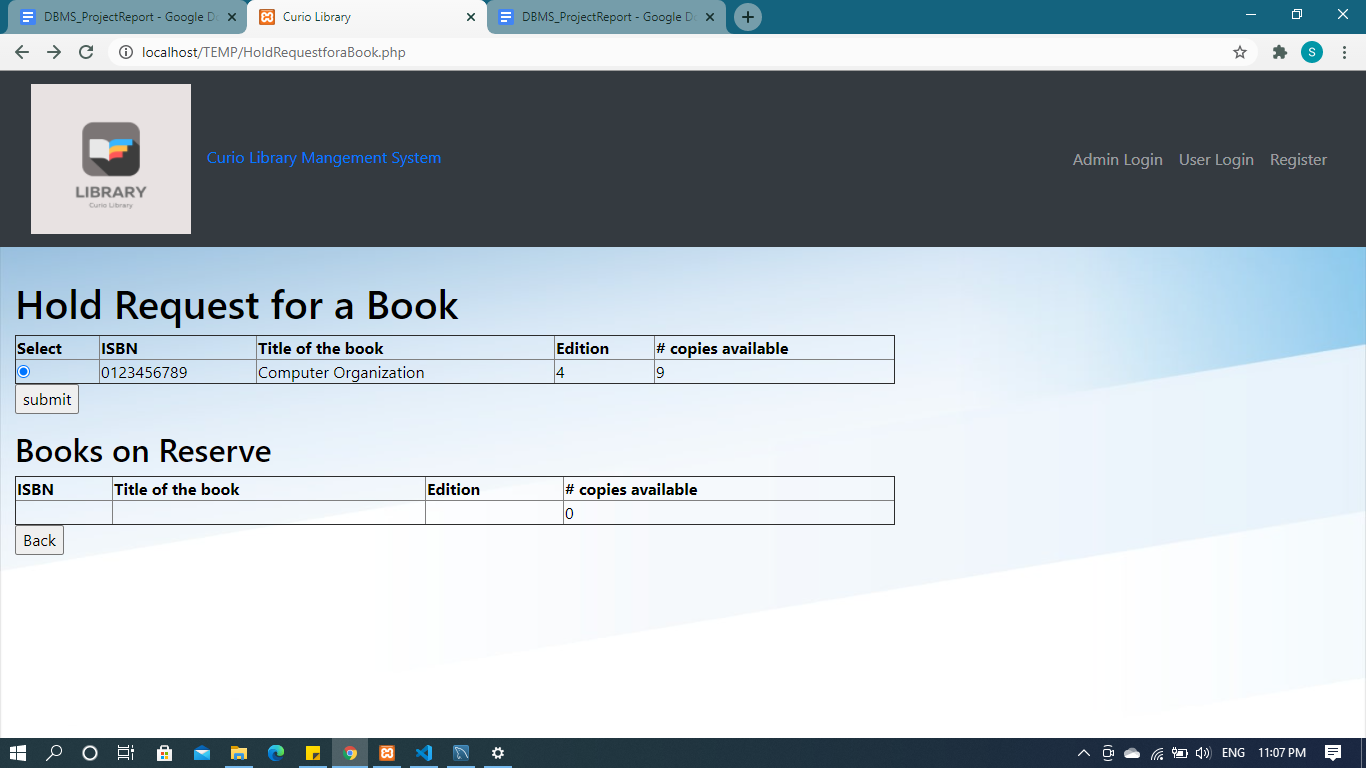
**3. verifyuser**

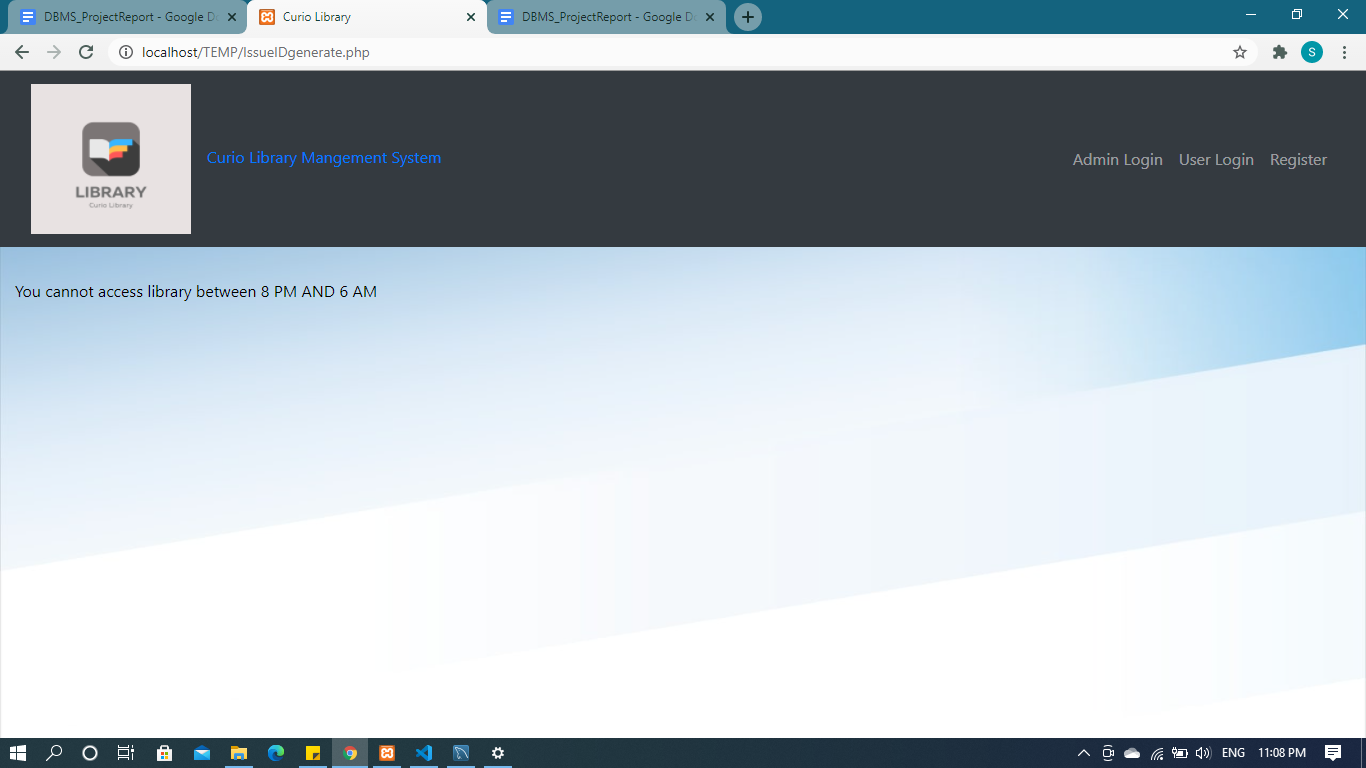
****

****

4. **chktimeinsertissue**







**Similarly the triggers:**

**chktimeupdateissue, chktimedeleteissue, chktimeinsertauthor, chktimeupdateauthor, chktimedeleteauthor, chktimeinsertbook, chktimeupdatebook, chktimedeletebook, chktimeinsertbookcopy, chktimeupdatebookcopy, chktimedeletebookcopy**

**generates the above errors on accessing the library on Saturday and Sunday or accessing the library from 10:00 PM to 6:00 AM or regular days as well as on Saturdays and Sundays.**