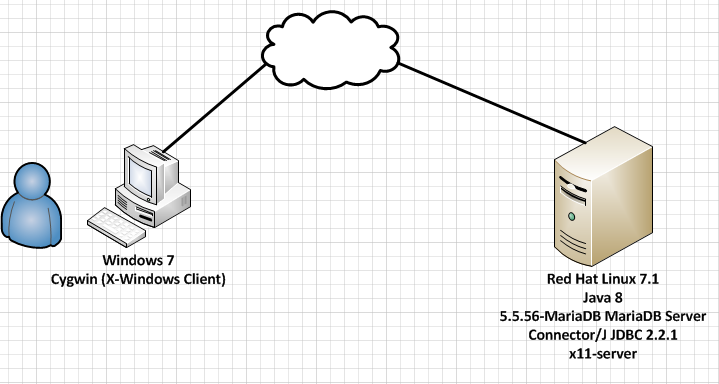
**Library System User Guide** – work in progress

last update – 2/21/2018

Doug Skayer

Bhavani Putrevu



**Table of Contents**

1. Requirements
2. Database Tables
3. Use Cases
4. Class Diagrams
5. MariaDB Installation and Configuration
6. X11 Server-side installation
7. Java Installation
8. ConnectorJ Installation and Testing
9. X11 Client-side installation
10. Application startup
11. Testing
12. **Requirements:**
13. Adding a new user to the User List in the Library
14. Adding a new book to the Book List in the Library
15. User checks-in a book from the Library
16. User returns a book to the Library
17. User holds a book
18. Searching for a book
19. Deleting a book from the Book List
20. Removing a hold on a book
21. Renewing a book
22. Printing user’s transactions
23. **Database Tables**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  | **Table Name** | **Book** | |  |
|  | Column | DataType | Null |  |
|  | Id | String | Not Null |  |
|  | Title | String | Not Null |  |
|  | Author | String | Not Null |  |
|  | Library\_id | String | Not Null |  |
|  | Borrowd\_User\_Id | String | Null |  |
|  | Due\_date | Date | Null |  |
|  |  |  |  |  |
|  | **Table Name** | **User** | |  |
|  | Column | DataType | Null |  |
|  | Id | String | Not Null |  |
|  | Name | String | Not Null |  |
|  | Address | String | Not Null |  |
|  | Phone | String | Null |  |
|  |  |  |  |  |
|  | **Table Name** | **Hold** | |  |
|  | Column | DataType | Null |  |
|  | Book\_Id | String | Not Null |  |
|  | User\_Id | String | Not Null |  |
|  | Date | Date | Not Null |  |
|  |  |  |  |  |
|  | **Table Name** | **Library** | |  |
|  | Column | DataType | Null |  |
|  | Id | String | Not Null |  |
|  | Name | String | Not Null |  |
|  | Address | String | Not Null |  |
|  | Phone | String | Null |  |
|  |  |  |  |  |

1. **Use Cases: (Incomplete)**

|  |  |
| --- | --- |
| Use Case Name | **Checkout book** |
| Scope | Library System |
| Level | User Goal |
| Primary Actors | User |
| Stakeholders and Interests | Use case applies to checkout station or website. |
| Preconditions | User or librarian is at book checkout station, the checkout station is fully operational and displays the authentication screen and user has one or more books to checkout. |
| Success Guarantee | Books are successfully removed from inventory and their check-out status is updated. |
| Main Success Scenario | 1. User authenticates (scan library card or enters library card number). 2. System returns overdue book and charge information (if true). 3. System allows checkout if dues are less than $25, otherwise instructs user to see library personnel. 4. User scans or enters book ISBN number. 5. System updates book checkout status record. 6. System indicates successful checkout. 7. Repeat 4-6 until all books are checked out. 8. System prints list of checked out books and their due dates. 9. System returns to authentication screen. |
| Special Requirements | None |
| Technology and Data Variations List | Checkout station has a barcode scanner, touch-screen and printer. |
| Frequency of Occurrence | Medium frequency. |
| Miscellaneous | None |
| Last update | Doug Skayer - 1/18/2018 |

|  |  |
| --- | --- |
| Use Case Name | **Access Account Information** |
| Scope | Library System |
| Level | User Goal |
| Primary Actors | User |
| Stakeholders and Interests | Use case applies to web access. |
| Preconditions | User or librarian is accessing system over the internet using a browser. |
| Success Guarantee | Book information is successfully displayed. |
| Main Success Scenario | 1. User authenticates with system. 2. System returns list of charges (if any). 3. System returns list of books reserved (if any). 4. System gives option to modify account information (password). 5. System gives option to cancel book reserve. |
| Special Requirements | None |
| Technology and Data Variations List | None. |
| Frequency of Occurrence | Low frequency |
| Miscellaneous | None |
| Last update | Doug Skayer - 1/20/2018 |

1. **Class Diagrams (Incomplete)**

|  |
| --- |
| **Book** |
| * id: String * title: String * author: String * dueDate: Date * borrowdBy: User * holds : List |
| + book ( id: String, title: String, author: String) : Book  + issue ( user: User) : Boolean  + returnBook ( ) : User  + renewBook ( user: User) : boolean  + placeHold ( hold: Hold) : void  + removeHold ( userId: String) : Boolean  + hasHold ( ) : Boolean  + getBorrower ( ) : User  + getId ( ) : String  + getTitle ( ) : String  + getAuthor ( ) : String  + getDueDate ( ) : Calendar |

|  |
| --- |
| **BookList** |
| - books : List |
| + search (bookId: String) : Book  + getBooks ( ) : Iterator  + insertBook ( book: Book) : boolean  + removeBook ( bookId: String) : boolean |

|  |
| --- |
| **Library** |
| - users: UserList  - books: BookList |
| + getBooks ( userId: String) : Iterator  + addBook (title: String, author: String, id: String) : Book  + searchUser ( userId: String) : User  + addUser (name: String, address: String, phone: String) : User  + issueBook (bookId: String, userId: String) : Book  + renewBook (bookId: String, userId: String) : Book  + returnBook (bookId: String) : int  + removeBook ( bookId: String) : int  + placeHold (bookId: String, userId: String, duration: int) : int  + removeHold ( bookId: String, userId: String) : int  + processHold ( bookId: String) : User  + getTransaction ( userId: String, date: Calendar) : Iterator |

|  |
| --- |
| **User** |
| - name: String  - address : String  - phone: String  - booksOnHold: List |
| + user ( name: String, address: String, phone: String) : User  + getName ( ) : String  + setName ( name: String) : void  + getId ( ) : String  + setId ( id: String) : void  + getAddress ( ) : String  + setAddress (address: String) : void  + getPhone ( ) : String  + setPhone ( phone: String) : void  + issue ( book: Book) : boolean  + returnBook (book: Book) : Boolean  + renew ( book: Book) : Boolean  + placeHold ( hold: Hold) : void  + removeHold ( bookId: String) : Boolean |

|  |
| --- |
| **UserList** |
| - members : Live |
| + search ( userId: String) : User  + getUsers ( ) : Iterator  + insertUser ( user: User) : boolean |

1. **MariaDB installation and Configuration (Incomplete)**

**MariaDB** (MySQL) - Server version: 5.5.56-MariaDB MariaDB Server

Documentation: <https://mariadb.com/kb/en/library/documentation/>

**Procedure to install**

yum install mariadb-server mariadb

systemctl enable mariadb

systemctl start mariadb

**Login to MariaDB as root user**

mysql –u root -p

help

status

**Create user01 for database application operations**

CREATE USER 'user01'@'localhost' IDENTIFIED BY 'password';

grant select, insert, delete, update on testdb.\* to 'user01' identified by 'password';

FLUSH PRIVILEGES;

**Create testdb – database for POC**

create database testdb; # For POC

use testdb;

**Create BOOKS TABLE for POC**

create table BOOKS

(grade int(3) NOT NULL,

title varchar(100) NOT NULL,

author varchar(50) NOT NULL,

isbn varchar(15) NOT NULL);

MariaDB [(none)]> use testdb;

Database changed

MariaDB [testdb]> create table BOOKS

-> (grade int(3) NOT NULL,

-> title varchar(100) NOT NULL,

-> author varchar(50) NOT NULL,

-> isbn varchar(15) NOT NULL);

Query OK, 0 rows affected (0.00 sec)

MariaDB [testdb]> show tables;

+------------------+

| Tables\_in\_testdb |

+------------------+

| BOOKS |

+------------------+

1 row in set (0.00 sec)

**Add some data into BOOKS table for POC:**

insert into BOOKS(grade, title, author,isbn) values ('3', ' *My Librarian Is a Camel: How Books Are Brought to Children around the World*', 'Margriet Ruurs', ' 1590780930');

insert into BOOKS(grade, title, author,isbn) values ('3', ' *Rain School* ', ' James Rumford', ' 0547243073');

insert into BOOKS(grade, title, author,isbn) values ('3', ' *The Librarian of Basra* ', ' Jeanette Winter', ' 0152054456');

insert into BOOKS(grade, title, author,isbn) values ('3', ' *The Librarian of Basra* ', ' Patricia Polacco ', ' 0399237321');

insert into BOOKS(grade, title, author,isbn) values ('3', ' *The Incredible Book Eating Boy* ', ' Oliver Jeffers ', ' 0399247491');

insert into BOOKS(grade, title, author,isbn) values ('3', ' *The Boy Who Loved Words* ', ' Roni Schotter ', ' 0375836012');

MariaDB [testdb]> DESCRIBE BOOKS;

+--------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+--------+--------------+------+-----+---------+-------+

| grade | int(3) | NO | | NULL | |

| title | varchar(100) | NO | | NULL | |

| author | varchar(50) | NO | | NULL | |

| isbn | varchar(15) | NO | | NULL | |

+--------+--------------+------+-----+---------+-------+

MariaDB [testdb]> SELECT \* FROM BOOKS;

+-------+------------------------------------------------------------------------------+--------------------+-------------+

| grade | title | author | isbn |

+-------+------------------------------------------------------------------------------+--------------------+-------------+

| 3 | My Librarian Is a Camel: How Books Are Brought to Children around the World | Margriet Ruurs | 1590780930 |

| 3 | Rain School | James Rumford | 0547243073 |

| 3 | The Librarian of Basra | Jeanette Winter | 0152054456 |

| 3 | The Librarian of Basra | Patricia Polacco | 0399237321 |

| 3 | The Incredible Book Eating Boy | Oliver Jeffers | 0399247491 |

| 3 | The Boy Who Loved Words | Roni Schotter | 0375836012 |

+-------+------------------------------------------------------------------------------+--------------------+-------------+

6 rows in set (0.00 sec)

Template: insert into BOOKS(grade, title, author,isbn) values ('3', ' ', ' ', ' ');

/home/dskayer1/PROJECT/TEST/mariadb

MariaDB [(none)]> status

--------------

mysql Ver 15.1 Distrib 5.5.56-MariaDB, for Linux (x86\_64) using readline 5.1

Connection id: 2

Current database:

Current user: root@localhost

SSL: Not in use

Current pager: stdout

Using outfile: ''

Using delimiter: ;

Server: MariaDB

Server version: 5.5.56-MariaDB MariaDB Server

Protocol version: 10

Connection: Localhost via UNIX socket

Server characterset: latin1

Db characterset: latin1

Client characterset: utf8

Conn. characterset: utf8

UNIX socket: /var/lib/mysql/mysql.sock

Uptime: 4 min 4 sec

Threads: 1 Questions: 5 Slow queries: 0 Opens: 0 Flush tables: 2 Open tables: 26 Queries per second avg: 0.020

--------------

1. **X11 server-side installation**

yum install x11-server

yum install x11-server-common

yum install xauth

yum install xorg-x11-xauth

yum update xorg-x11-xauth

yum update xauth

1. **Java installation components (Incomplete)**

yum install java-1.8.0-openjdk-devel

1. **Java Connector/J (jdbc driver) for MariaDB installation (Incomplete)**

**Java connection to database :** MariaDB Connector/J is used to connect applications developed in Java to MariaDB and MySQL databases using the standard JDBC API. The client library is LGPL licensed. See: <https://mariadb.com/kb/en/library/about-mariadb-connector-j/>

* Download mariadb-java-client-2.2.1.jar into /home/dskayer1/PROJECT/TEST/java/ on mrtsdoug-vm. Note: 2.2 for java 8 and java 9 (JDBC 1.4.2 compliant).
* Copy to /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7\_4.x86\_64/jre/lib/ext/

cp mariadb-java-client-2.2.1.jar /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7\_4.x86\_64/jre/lib/ext/

Note: jar tvf mariadb-java-client-2.2.1.jar lists jar file contents. I needed to change code to point from com.mysql.jdbc.Driver to org.mariadb.jdbc.Driver and recompile. Then the jdbc code started working.

**Example:**

**[dskayer1@mrtsdoug-vm java]$ java MySqlJdbcTest**

**Grade: 3, Title: My Librarian Is a Camel: How Books Are Brought to Children around the World, Author: Margriet Ruurs, ISBN: 1590780930**

**Grade: 3, Title: Rain School , Author: James Rumford, ISBN: 0547243073**

**Grade: 3, Title: The Librarian of Basra , Author: Jeanette Winter, ISBN: 0152054456**

**Grade: 3, Title: The Librarian of Basra , Author: Patricia Polacco , ISBN: 0399237321**

**Grade: 3, Title: The Incredible Book Eating Boy , Author: Oliver Jeffers , ISBN: 0399247491**

**Grade: 3, Title: The Boy Who Loved Words , Author: Roni Schotter , ISBN: 0375836012**

1. **Application startup - client side**
2. Insert sysadmin USB drive with Cygwin terminal
3. Open My Computer icon on desktop
4. Sign into VPN
5. Select CYGWYN folder
6. Right-click **Cygwin** batch program and “Run as administrator”. Wait for C:\ CMD window to open.
7. Run: **startxwin &** #If problems run ps to verify xinit & xwin are running
8. Run: **export DISPLAY=:0.0** #Set this computer to open X Windows display
9. Run: **xterm** wait for X terminal session to open.
10. **ssh -Yv mrtsdoug-vm** # Use ssh to connect to server running Java & MariaDB.

# Y is for debug output; v is for X11 forwarding

1. **xhost +** # Allows X11 protocol access.
2. export CLASSPATH=/home/dskayer1/PROJECT/TEST/java/
3. **cd /home/dskayer1/PROJECT/TEST/java** # Application location
4. systemctl start mariadb (if not running) # Start MariaDB database
5. **java FahrenheitToCelcius** # Test JavaSwing/X Windows functionality
6. **Application startup/shutdown - server side**

systemctl start mariadb

systemctl stop mariadb