**Group Journal –** last update 2/17/2018 dls

**Meeting Date : 01-16-2018**

Members Present: Bhavani Putrevu and Doug Skayer

**Doug’s GitHub link:** [**https://github.com/as0371zk/Doug\_and\_Bhavani/tree/master/docs**](https://github.com/as0371zk/Doug_and_Bhavani/tree/master/docs)

**Bhavani’s GitHub link:**

<https://github.com/bputrevu/ICS499_Project_Library/blob/master/Group%20Journal.docx>

**Current Requirements:**

* We are planning to build a Library Book Management System.
* We will be building a 3-tier system, but haven’t determined the platform (Bhavani’s laptop, Amazon AWS or C.S. server) or framework yet (HTML/PHP/MySQL or Java?).
* The following requirements have been identified:
  + User can login to system via Internet browser.
  + User can search for a book
  + User can check-out a book (any location).
  + User can check-in a book (any location).
  + User can reserve a book (any location).
  + User can check the status of a book
  + User can cancel book-reservation.
  + User can check for overdue charges.
  + User can manage their account.

**Meeting date #1:** 1/9/2018

**Members:** Doug & Bhavani

**Major discussions**: Basic design of Library system, some requirements, and some db tables.

**Major decisions**:

* The 3-tier development project we decided to do is a library checkout system.
* Bhavani and I outlined and submitted on 1/9/2018 to Ryan the entities we thought were relevant to our implementation.
* **Book** (title, author, ISBN, type, description), **location** (name, address, hours), **user** (name, address, library id), **book\_status** (hold, location, check-in/out).
* Other functions/entities/objects/attributes include: book\_search, due\_date, user\_login, account\_management, cancel\_hold, book\_inventory.
* We decided to exclude the library-employee aspect of the project, focusing more on book management.
* The platform suggested by Ryan was Amazon AWS. I’m not familiar with it, but if that is likely what others would use, I will look into that. It sounds like use of the CS server at metro may also be an option.
* While somewhat familiar with Java, I’m not sure how the 3-tier framework would work. The only practical way I can think of deploying this project is by using html/javascript, php and MySQL. We hadn’t talked about it, but perhaps Bhavani has better ideas for the implementation.

**Meeting date #2:** 1/16/2018

**Members:** Doug & Bhavani

**Major discussions**: How to divide work requirements for next week’s class.

**Major decisions**:

* I would setup GitHub repository and add Bhavani as a contributor.
* We would split the use cases.
* We would split the DB tables.
* We would split the class diagrams with methods.
* We would both contribute requirements.
* I would do a high-level over view of what the project is in the first 5 minutes of the presentation.

**Meeting date #3:** 1/23/2018 **– missed due to snow storm.**

**Notes:**

**1/16/2016** – setup Github, repository and docs directory. Location: <https://github.com/as0371zk/Doug_and_Bhavani>

**1/18/2018** – Uploaded Excel doc with 4 DB-table examples

1/18/2018 – Uploaded start of User Guide (Word doc) including requirements and 1 use case (Checkout Book).

**1/22/2018** – Test pulling and pushing content. This is clunky, but seems to work ok. I added another use case as a test.

**1/23/2018** – Upload journal, resave tables and user guide as PDF’s.

**Issues**:

* We are not able to collaborate on GitHub. I setup a GitHub repository, added tables and a user guide with a couple of use cases to the repositories document folder, but Bhavani was not able to access it. Bhavani setup a separate GitHub repository. As a result, we have both setup GitHub repositories.
* We still don’t have a website platform to develop on.

**1/27-28/2018 – dls** – Work with GitHub command line interface. Practiced download/upload project, along with other GitHub commands.

**Meeting #4:** 1/30/2018 –

* Decided that I need to make the DOCS folder
* Still don’t have a platform to work on.
* Decided that we would use Java-Swing for the web tier, Java for the application and MySQL for the database.

**1/31/2018 – dls**

* I setup a Linux VM and installed Java8 & MariaDB to use as an application development platform (2 hours).
* Research java code needed to access a database and web front end (2 hours).

**2/2/2018 – dls**

* 7 hours trying to get Java Swing to work. Tried Ming64 and MAC. Installed Cygwin and was able to test Swing from laptop.

**2/3/2018 – dls**

* Tried to get Java Swing class to work with MariaDB.
* Configure test database table and add some data.

**2/8/2018 – dls**

* Try to get Java jdbc to work with MariaDB. Tried various classpath options. Researched possible solutions in books and on internet. Solution seems straight forward with MySQL, but I am unable to get it to work.

**2/9/2018 – dls**

* Finally got the jdbc to work to have a java class print a text output of a MariaDB table. Code examples reference MySQL database path (com.mysql.driver), but MariaDB, which is supposed to be an exact replacement for MySQL, uses a different path(org.mariadb.driver). I figured this out by looking at the JAR file contents (jar tvf maria.jar).

**2/10/2018 – dls**

* Try to get Java-Swing to list database table contents. No luck.

**Meeting #5** : 2/13/2018

Doug will do the following:

* Go to Java tutor to determine how to read data from MariaDB using a Java-Swing class.
* Will update the user guide
* Will update the status report
* Will update this journal.

**2/17/2018 – dls**

* Update this journal.
* Create DOCS folder and copy this journal to it.
* Push changes to repository