# CS415 ASSIGNMENT 1

## JUNIT TESTING AND GITHUB


**GROUP Customer.java members:**

Prasant Prasad (S11075110)

Alvindra Dutt (S11074661)

## Junit Testing (Customer.java)

Junit testing involves testing individual units of code with associative control data, usage procedures and operating procedures. We had chosen to unit test Customer.java and the modules to be tested were Add, Update, Find and Delete. The following unit tests were created to test the above modules;

| Unit | Functionality | Parameters | Results | | Unit Test | Reasons |
|---|---|---|---|---|---|---|
| | | | **Positive** | **Negative** | | |
| **Add()** | This function executes the "INSERT" query and inserts data into Person and Customer tables. | Connection con,<br>String LNAME,<br>String FNAME,<br>Int Age,<br>Int ID,<br>String Address,<br>String Phone | 1. The Customer Has Been Added | 2. Error java.lang.NullPointerException<br>3. Error: java.sql.SQLException: [Microsoft][ODBC Microsoft Access Driver] Query is too complex. | testAdd() | This unit tests if the required inputs to the Add() function does yield the expected positive output. |
| | | | | | testAdd_2() | This unit tests if the required inputs to the Add() function does yield the expected negative output, if there was no connection to the database. |
| | | | | | testAdd_3() | This unit test tests the size of the input Strings used in the parameters. The unit creates Strings of 10000000 characters and produces error 3. The database accepts characters less than or equal to 1000000. |

| | | | | | | |
|---|---|---|---|---|---|---|
| **Update()** | This function executes "UPDATE" query to update Person and Customer tables at a given ID. | Connection con,<br>String LNAME,<br>String FNAME,<br>Int Age,<br>Int ID,<br>String Address,<br>String Phone | 1. The Record Was Updated | 2. Error java.lang.NullPointerException<br>3. Error: java.sql.SQLException: [Microsoft][ODBC Microsoft Access Driver] Query is too complex. | testUpdate() | This unit tests if the required inputs to the Update() function does yield the expected positive output. |
| | | | | | testUpdate_2() | This unit tests if the required inputs to the Update() function does yield the expected negative output, if there was no connection to the database. |
| | | | | | testUpdate_3() | This unit test tests the size of the input Strings used in the parameters. The unit creates Strings of 10000000 characters and produces error 3. The database accepts characters less than or equal to 1000000. |
| **Find()** | This function locates the required data from the database using the "SELECT" query. | Connection con,<br>Int ID | 1. The Above Record Was Found | 2. Error: java.lang.NullPointerException | testFind() | This unit tests if the required inputs to the Find() function does yield the expected positive output. |
| | | | | | testFind_2() | This unit tests if the required inputs to the Find() function does yield the expected negative output, if there was no connection to the database. |

| | | | | | | |
|---|---|---|---|---|---|---|
| **Delete()** | This function deletes a record at a given ID, using the "DELETE" query. | Connection con,<br>Int ID | 1. The Above Record Was Found | 2. Error: java.lang.NullPointerException | testDelete() | This unit tests if the required inputs to the Delete() function does yield the expected positive output. |
| | | | | | testDelete_2() | This unit tests if the required inputs to the Find() function does yield the expected negative output, if there was no connection to the database. |

**Note:**

The variable **"Connection con"** is the only parameter passed to the above functions, the other variables are global variables.

## Github

Github was great tool in version control in this assignment. Group members were able to make changes on their local repository and push changes to the remote repository. These changes can be pulled from the remote repository to their local repository. Github had the following advantages;

- Changes made was easily distributed amongst team members.
- A single version was worked on by all members.
- We were able easily role back to the previous version.
- Github kept a history of all changes made by the group members.

The Netbeans IDE and the Github desktop application was used in this assignment. The group was able to interact with the repository using the following steps;

1. Github account was created on https://github.com/ and a desktop application was downloaded from the site.
2. A clone was made from the copy on the remote repository to the local repository using the desktop application.
3. Netbeans IDE was used to make changes on the local copy of the Book Management System.
   *(Note: Since the project is already in the Git repository, we did not have to initialize the repository on Netbeans)*
4. The snap shot of the changes were stored in the local repository using the commit command with a commit message.
   *(Go to Team -> Commit -> Select the file to commit -> Add a message -> Click commit)*
5. The changes were then pushed from the local repository to the remote repository using push command.
   *(Go to Team -> Remote -> Push -> Select Specify Git Repository Location -> Enter username and password -> Select Next -> Select local branch -> Select Next -> Select remote branch and click finish)*
6. Changes were pulled from the remote repository to the local repository using the pull command.
   *(Go to Team -> Remote -> Push -> Select Specify Git Repository Location -> Enter username and password -> Select Next -> Select remote branch and click finish)*

The interaction between the remote repositories can be found at https://github.com/Prasant555/2014s2

The CustomerTest.java can be found at

https://github.com/Prasant555/2014s2/blob/master/test/bookcompany/CustomerTest.java

The list of commits can be found at https://github.com/Prasant555/2014s2/commits/master