

SESSION: 2017/18 DIET 1

**Module Title: Programming 2 (Marking Sheet)**

**Module Code:** M2I324178-17-B

**Level:** 2

**Module Leader:** Dr. Richard Holden

Summary of marks per requirement:

|  |  |
| --- | --- |
| **Requirement** | **Marks** |
| 1. Narrative document | /6 |
| 1. Swipe class | /5 |
| 1. VisitorSwipe class | /3 |
| 1. Repository class | /4 |
| 1. DAOImpl class | /5 |
| 1. AttenanceController class (Increment1) | /7 |
| 1. AttenanceController class (Increment2) | /5 |
| 1. AttenanceController class (Increment3) | /5 |
| 1. AttenanceController class (Increment4) | /5 |
| 1. Use of nested classes (inner/local/anonymous) | /5 |
| Use of abstract & generic classes |
| Use of Enums |
| Use of lambda expressions/aggregate operations |
| **Total** | /50 |

|  |  |  |
| --- | --- | --- |
| Student Name | Student ID | Student E-mail |
| Mary Ann Achieng | S1719027 | machie200@caledonian.ac.uk |

Student Declaration: This piece of work is not plagiarized. It is my own original work and has not been submitted elsewhere in fulfilment of the requirements of this or any other award.

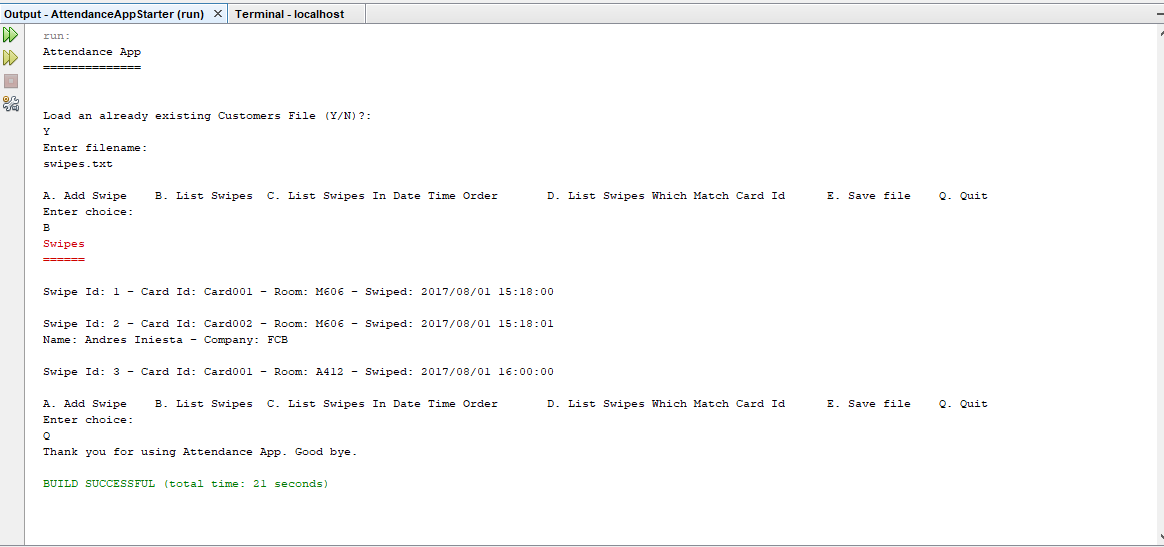
Signature: Mary Ann Achieng Date: April 13, 2018

Attendance App Narrative

Increment 1: List swipes in ID order

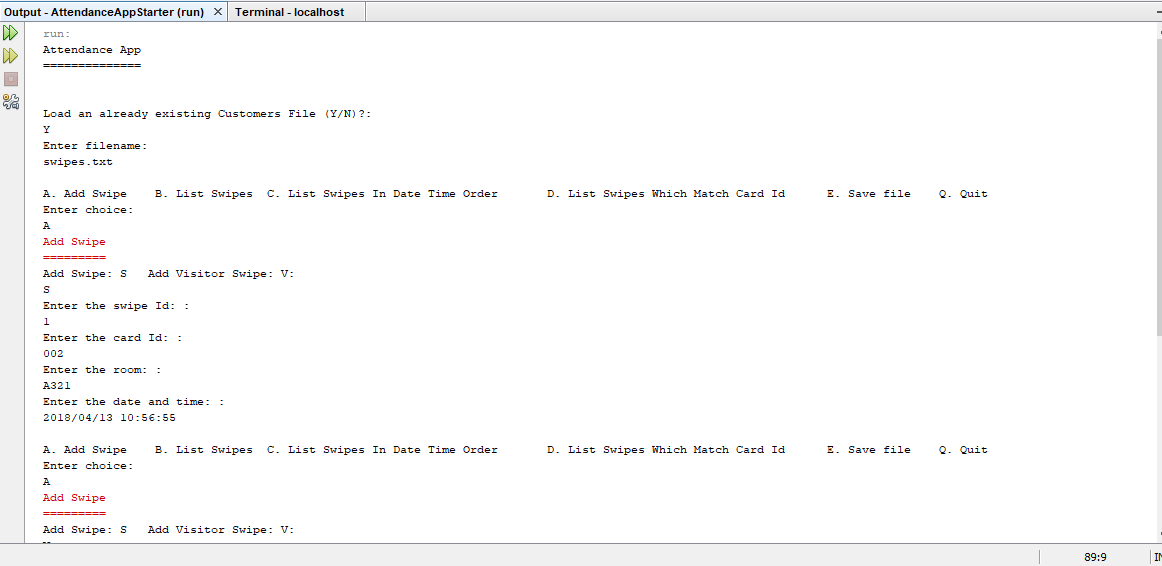
The first increment required that I complete the model classes: Swipe and VisitorSwipe. I decided to implement the collection type List as it is easier to implement. ArrayLists are also flexible and are easy to implement: they do not have a fixed length which gives me the freedom to input as many elements into it as I would like to.

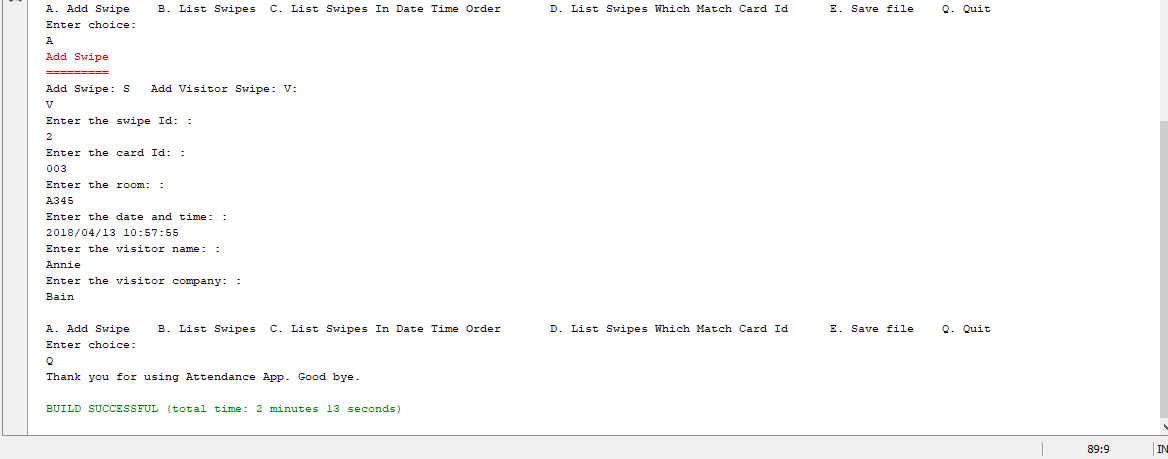
I used an iterator to go through the repository and get the swipes then list them. After this I printed the swipes in the *listSwipes()* method.



Increment 2: Add swipe

For the *addSwipes()* method, I prompted the user to select whether they want to enter a swipe or visitor swipe. I then implemented the InputHelper to collect swipe or visitor swipe data from the user and created new swipe and visitor swipe containers to store this. Finally, I added each of the created swipes or visitor swipes into the repository.

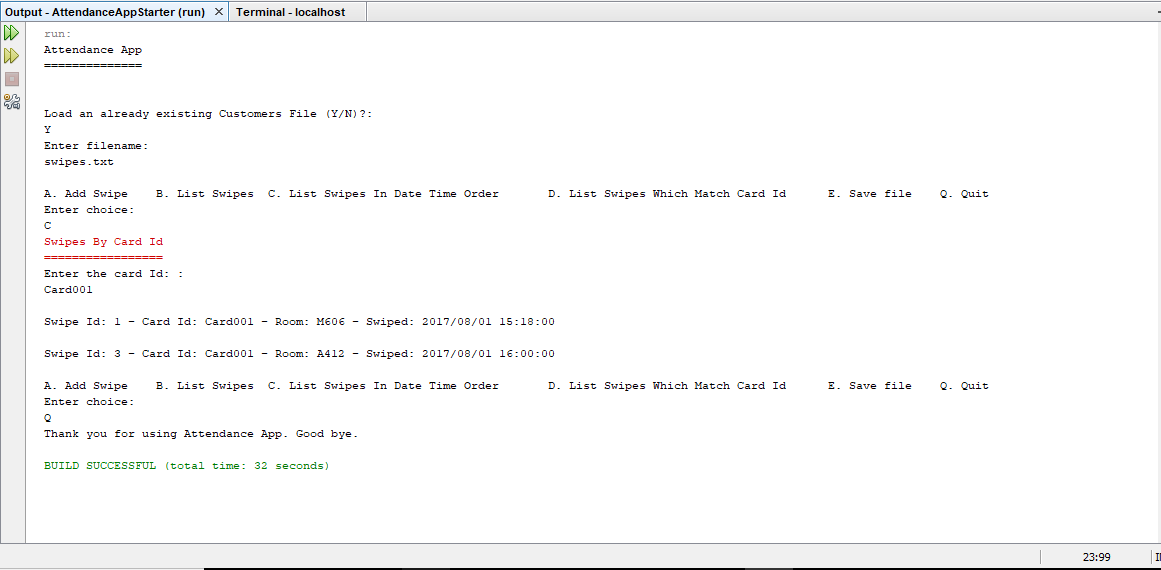




Increment 3: List swipes for a specified card id in order of swipe date-time

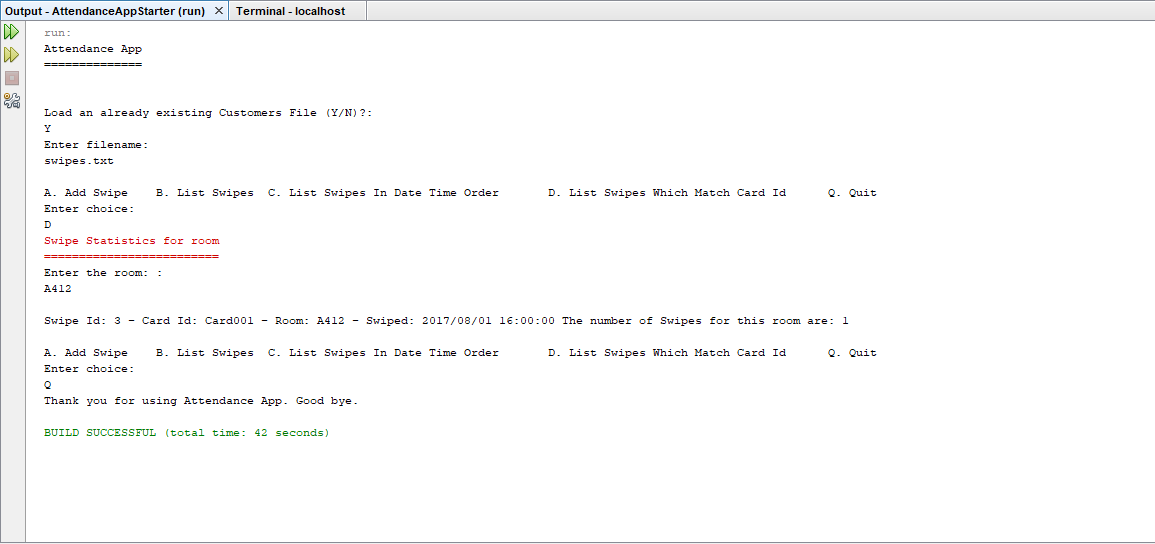
First, I implemented the comparator method in the swipe class to compare swipes using the swipe date and time.

To list the swipes according to a specified card id, I used the Input Helper to collect the card id from the user. I then used a list iterator to go through the items/swipes from the repository. I would then check through the swipes to find the ones with a similar card id. The code would then put the swipes for the card id similar to the one entered into an ArrayList. The code then prints out the contents of the ArrayList as a String(s). The code automatically arranges the swipes according to the date and time which the swipe was done.



Increment 4: List swipe statistics for a specified room

To list the swipes from a specific room, I used the Input Helper to get the room from the user. I then went through the swipes from the repository using the Iterator. The code then compares the entered swipe room with the rooms currently in the swipe repository. If the repository contains a swipe with the same room as the entered room, the swipe is input into an ArrayList. The code then goes through the ArrayList and prints out the last swipe listed on the list. It also prints out the number of swipes contained in that ArrayList.



Store method

I implemented the store() method in the DAOTextImpl to store any new file that the user may decide to create. The method takes the filename input by the user and a repository object as parameters. It then creates a new file if called by the user. The store method can be called in the Attendance Controller class if the user selects the *saveFile()* method(if the user selects option E).

The *saveFile()* method in the Attendance Controller class prompt the user to conform if they want to save the file then asks for a new filename.

