# Deliverable III

## A. User manual

On any Computer or mobile device, access any web browser and type in the URL box “localhost:8080”.

## B. Instructions to Compile/Run

### How to Compile/Run Application

1. Pre-requirements (NOTE: Architectures must match)
    a. Java JRE
    b. Eclipse
2. Clone repository
    a. From the command line: “git clone https://github.com/jacobhanson1010/csce4444”

*****IMPORTANT*****

- **Steps 3 - 7 are for converting all xlsx sheets to csv format.**
- **We have already provided All_training_data.csv in the repository.**
- **Skip to step 8 when attempting to execute the program.**

3. Change directory into csce4444/data/
4. Open All_training_data.xlsm
    a. NOTE: In order for the following VBA macro to work, you must:
        i. Be using Microsoft Windows
        ii. Open the workbook using Microsoft Excel
    b. There may be a popup about macros, MAKE SURE YOU PRESS ENABLE. Otherwise, be sure Macros are enabled in your Microsoft Excel settings.
5. Under the “Views” tab of the ribbon, click the “Macros” icon
6. Select “main” from the list of macros, then press “Run”
    a. This will execute the VBA macro that does the following:
        i. Formats all of the input files
        ii. Appends said files to All_training_data.xlsm
        iii. Converts All_training_data.xlsm to a .csv
7. Close out of Excel
    a. Press “Don’t Save” in the dialog
    b. The macro already saved the .csv file
8. Open project in Eclipse
    a. File→import →General→ Existing Projects into Workspace → Select root directory → Finish
    b. You may have to point your project to your JRE:
        i. Right-click package→Properties→Java Build Path→JRE System Library→Check box
9. Run project
    a. Right-click package → Run As → Java Application
    b. Select “SwolePatrol - csce4444” when prompted to Select Java Application

10. Confirm project has been started
    a. Visit “localhost:8080” within a web browser

### How to Compile/Run Test Suite

1. Open project in Eclipse
2. Run Test Suite
    a. Right-click package → Run As → JUnit Test
3. Confirm status of all test cases on JUnit tab, next to Package Explorer

## C. Unit Testing

Our unit tests are composed of one test suite, and two test class files. The purpose of the test suite is to execute the explicitly written tests defined in the test class files, which correlate to each of our two implementation classes. The bigger of the two test files -- the one for the EntryLookupEngine.java -- contains multiple test methods to supply a fixed input to test with. By supplying a test ‘Date’, we can check the output of the engine to ensure the desired calculation results. We used assertEquals to compare the generated output values to the pre-generated expected values. In addition to testing the main functionality of the application, we also tested some incorrect values, to see if they would throw the correct exceptions.

Finally, because our application uses many small helper methods, we wrote test cases to ensure those helper functions are performing their tasks appropriately.

To run the test suite, you must have Apache maven installed on your computer. Run the command “mvn clean install” from the command line from within the project directory. This will compile the program into an executable jar file, and run the test suite.

## D. Functionality Implementations and Future works

In this program we were able to implement most of our functionalities that we had initially intended to, that is we were able to approximate the entrances to the Pohl Recreation center based on the data UNT provided us. The data we received was not in real time though, UNT sent us reports from the previous year that documented the gym’s entrances per day per 15 minutes. The data was received in the form of Excel workbooks, so we wrote a script for excel to reorder and reformat the sheets in order to extract the important information from their reports, we initially implemented some of the Apache functionalities to read the excel sheets into our java code and store the averages per day per 15 min in lookup tables. We then used the original data to make our predictions by implementing the Weka library in java. The Weka library is an open source library that contains multiple machine-learning algorithms used for data mining making it a great choice for this project.

One of the functionalities we had wanted to include was allowing the user to choose a specific day and time to check the availability of the gym rather than only showing the current entrances. Upgrading our UI would also be something like we’d like to do, because right now it doesn’t look very clean and organized. Having something like a live background that shifts through different photos of UNT and the gym would make it feel more personal for the students who use it. Another feature to be implemented was a visual representation of the entrances (i.e. diagram of gym max capacity and our estimated number of entrances), similarly to this, showing the population of the gym rather than just the entrances. Since the UNT gym only monitors entrances there is no data regarding how long people stay or how many people exit at any given time so being able to calculate this figure would make our software a lot more useful since

it’ll give the users a more realistic figure of how many people are actually at the gym and give a breakdown of how the population is distributed through the facility (i.e. 30 people are at the weight room, 20 people at the basketball courts) rather than just how many have recently arrived at the gym.

From a functionality perspective we’d like to be able to get live data from UNT rather than getting Excel sheets that require more work to organize and read in. With the success of our webpage, the next step would be to communicate with UNT officials and see if they would endorse our software by integrating this feature into their systems allowing all of UNTs students to use this product, furthermore the quality of predictions would drastically increase as well as the usability of our software.

# E. Meeting Minutes

| Date | Time | Team | Progress Description |
|---|---|---|---|
| 10/24/2017 | 3PM-3:40PM | John<br>Anthony<br>Jacob<br>Ibrahim | Prototype different style layouts for the website while consider what is possible code wise.<br>Worked on the server framework. |
| 10/26/2017 | 3:20PM-3:50PM | John<br>Anthony<br>Jacob<br>Ibrahim | Planned out we will simplify and use the data. |
| 10/28/2017 | 5PM-10PM | John<br>Anthony<br>Jacob<br>Ibrahim | Worked on scripts that condensed years worth of excel data sheets. |
| 11/4/2017 | 3PM-11PM | John<br>Anthony<br>Jacob<br>Ibrahim | Finished script algorithm prototype. |
| 11/9/2017 | 3:10PM-3:50PM | John<br>Anthony<br>Jacob<br>Ibrahim | Enhanced our algorithm to do machine learning to output a more precise number. |
| 11/19/2017 | 7PM-11PM | John<br>Anthony<br>Jacob<br>Ibrahim | Finished programing HTML and CSS code for UX.<br>Maded changed based off other team member feedback for our back end. |
| 11/25/2017 | 3PM-7PM | John<br>Anthony<br>Jacob<br>Ibrahim | Distribute the workload for Deliverable III and finished it. |
| 11/27/2017 | 5PM-11PM | John<br>Anthony<br>Jacob<br>Ibrahim | Prepared ourself for the presentation on the 28.<br>Optimized our code for the presentation. |

NOTE: Previous time meetings are in the past deliverable.

# F. Member Contribution

| Member name | Contribution description | Overall Contribution (%) | Note (if applicable) |
|---|---|---|---|
| Jacob Hanson | • Created repository and setup the Spring framework for java<br>• Managed the repository and branches<br>• Wrote JUnit test cases | 25% | |
| Anthony Hicks | • Integrated Weka library into Java source code<br>• Automated conversion and concatenation of Excel workbooks<br>• Wrote clear instructions on how to compile/run the application | 25% | “Kind and, a hard worker, and smart” -John |
| Ibrahim El-Rayes | • Implemented Apache functionalities to allow the program to read from the excel workbooks.<br>• Helped obtain entrance data (with John) | 25% | |
| John Nguyen | • Worked on HTML and CSS code.<br>• Kept Track of group effort<br>• Obtained source data for the back end. | 25% | “Great worker” -John |