

Web platform development 2

Group X Report

*“*I declare that all work submitted for this coursework is the work of Aaron Amjid, Andrew Mathieson, David Fagan and Matthew Alston alone unless stated otherwise.*”*

# Introduction

The purpose of this report is to give a full and detailed report on the process Group X took when developing the application necessary for satisfying the specifications for the Web Platform Development 2 module (M3I322955-18-B), second semester of the third year at Glasgow Caledonian University 2018/2019.

After forming a group, the members took to planning, designing and implementing a milestone planner application. In order to satisfy the requirements, the team chose to start a GitLab project where each member could pull, push from their own separate branch. By doing this each team member could work on their own individual section at the same time allowing the project to conclude successfully and equally as importantly, on time. Once work on an individual team members branch was ready, it would be merged with the main branch eventually leading to a finished milestone planner application.

The milestone planner application must allow the user to define their own milestones on the application but remain only viewable by them. The application will implement a log in system and session ids in order to achieve this. The milestone itself will include a description of the milestone, its intended due date and finally the milestones actual completion date. Each milestone must be able to be removed from the list and be able to be edited. The milestone must be listed if incomplete and have the capability of being shared with others using a link.

In order to satisfy the requirements each team member should pull when deemed necessary in accordance to the division of work on the project. GIT functionality had to be utilized and in order to do that the group would make sure there was regular commits, useful comments and branches.

Group X decided to use GitLab for this project, partly due to familiarity by the members but also due to its useful and unique features (additionally, it’s free and open source). It allows for up to 10gb of storage for free which was deemed more than enough for this project. It also allows for editing of files directly in the web interface that was important when small changes had to be made. It has an interface capable of moving through the repository at a fast pace, this was imperative due to deadlines not only for this module, but others too. In comparison to GitHub, Gitlab offers project teams more control over their repositories.

When choosing a development environment, the team selected to use IntelliJ IDEA. IntelliJ comes with features comes with a vast number of features and plugins, supporting both Maven and Mustache that were both used heavily in the development of this project. The integrated tools help to develop a precise and streamlined finished product that helped lead this project successfully. When comparing to past development environments the team used (Eclipse, Brackets, Atom, Notepad++) it was deemed the most efficient and practical.

Finally, when deciding on the design of the application the group decided it must have a simple colour scheme and layout. After the creation of wireframes and a brief consideration of HCI we decided in order to best satisfy the requirements the application will remain simple allowing every user, experienced or new to milestone planning, a streamlined and easy experience when utilising the applications functionality. The best way to achieve this was to use Bootstrap. Bootstrap is a framework that allows the design of modern websites and web apps. It comes with numerous HTML and CSS templates that gears the User Interface towards being user friendly. Again, as previously stated the application had to be user friendly in order to satisfy the requirements specified when first taking on this project.

# Link design within the application

Within the application, we designed our links to each link to some form of functionality within the application. Within the web app, the standard design layout we decided to implement utilises a navigation bar to provide the necessary links to functions within the application, providing a standardised method of linking pages to functions. From the index page, the Log In page is also linked in a clear fashion, to avoid confusion from users upon viewing. The URLs within the website have mostly been shortened in some form, such as milestones being shortened to “ms” to maintain a short form of URL but also keep the URLs being named meaningfully. From the Milestones page, a user can click on any milestone to be taken to the milestones update page. On top of this, the user can also navigate to the update milestones page by using the navigation bar at the top of the application. The add milestone page is also linked within the rest of the application through the navigation bar.

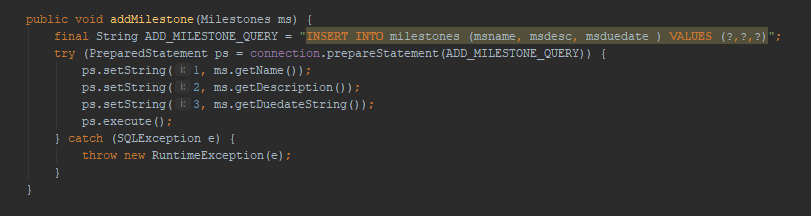
# Persistence Mechanism

Within context of persistence regarding the application, there is a degree of persistence present within the application. There are two main aspects of the application which are persisted between sessions, the list of users and the list of milestones. For both of these lists, the application utilises an H2 database to handle persistence. H2 is a java database which functions through imbedding databases into applications. A general benefit of H2 databases is that they are lightweight and relatively fast. Within the context of the application, the benefits of utilising an H2 database are that it allows for the application to maintain some form of persistence through the embedded database.

Within the application’s database, three tables are stored, Persons, Users and Milestones. Persons is the main table which stores the user’s details, their first name, last name and email. The Users table is part of a partially unimplemented function of the system. There was an attempt to implement a proper login system which used usernames and passwords which would allow for milestones to be linked to specific users and allow for a more solid implementation of sessions. This table holds an ID number for each user, a username and password. We recognised when designing the functionality of this table that password should be stored as hashed to implement a degree of security into the application.

The milestones table contains the data relating to any milestones created by a user. The table contains an id for each milestone, this allows for milestones to be searched for and identified. Each milestone has a name, a textual description of the milestone’s details, a due date, a completion date and a userID field. The userID field is another unimplemented aspect of the application’s functionality. The userID field works as a foreign key which would link to the perons database’s ID column, which would allow for milestones to be linked to particular users. As it stands, however, within the final implementation, all milestones are stored without user IDs.

All database functions are handled by a userDB class which acts as a data access object for the application. Methods from this class are called within the necessary servlets, within the doPost or doGet method for each. Within each method in the userDB class, an SQL statement is generated using parameters which are passed into the method, which is then executed upon the database and, where applicable, the results of the statement are returned by the method. An example of one of these methods can be seen below, the Add Milestone method.



As shown, the method stores the appropriate SQL statement as a string. After the parameters are filled using a prepared statement which allows for an extra layer of security against malicious attempts to access the database, the statement is executed. A try catch is used to catch SQL errors.

# Functionality

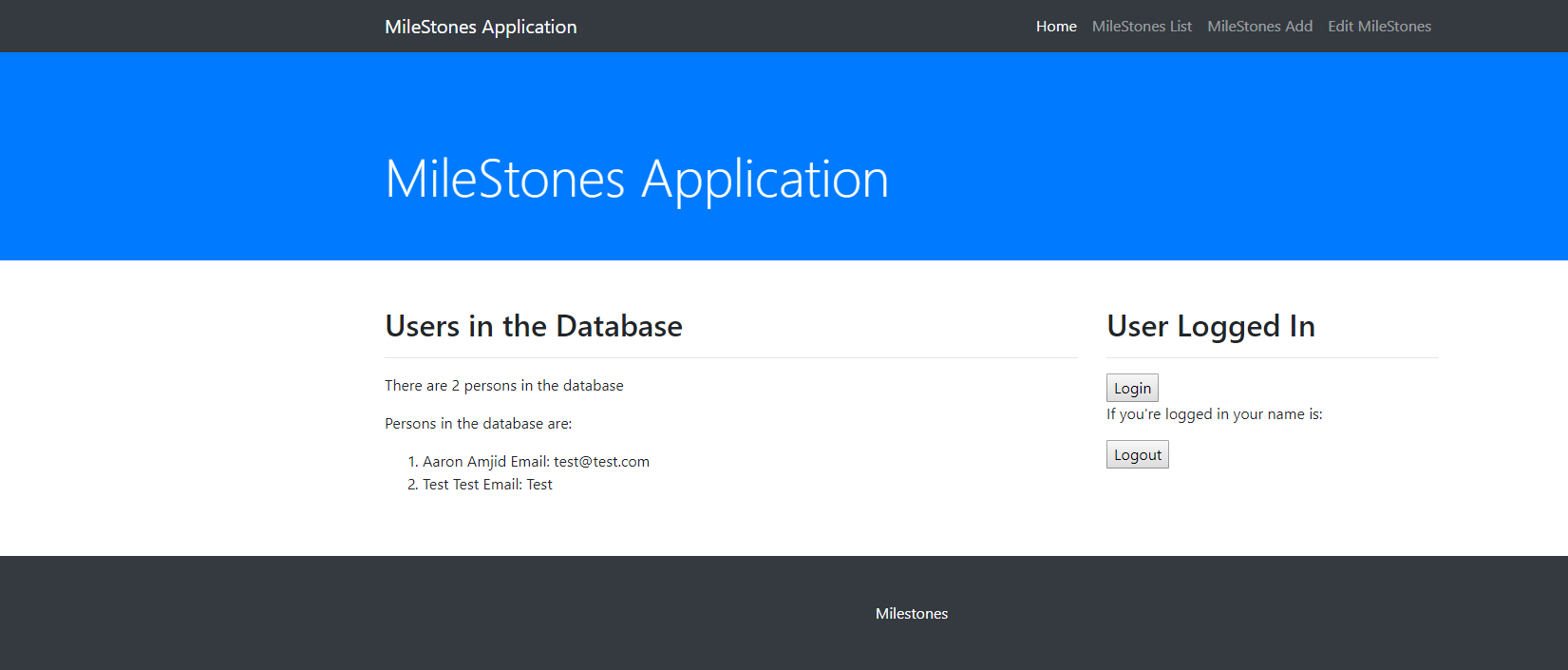
The final application which was produced by group covers a few of the specified requirements detailed within the coursework specification, however, due to a number of circumstances such as distractions to the project from other module deadlines and issues with some errors, the final product fails to meet some specified functionality, however, overall the application keeps within the spirit of the specification and a fair attempt was made at implementing all of the functionality.

The milestones application is comprised of 5 main pages: The index, Login page, Milestones page, Add Milestones and Edit Milestones page.

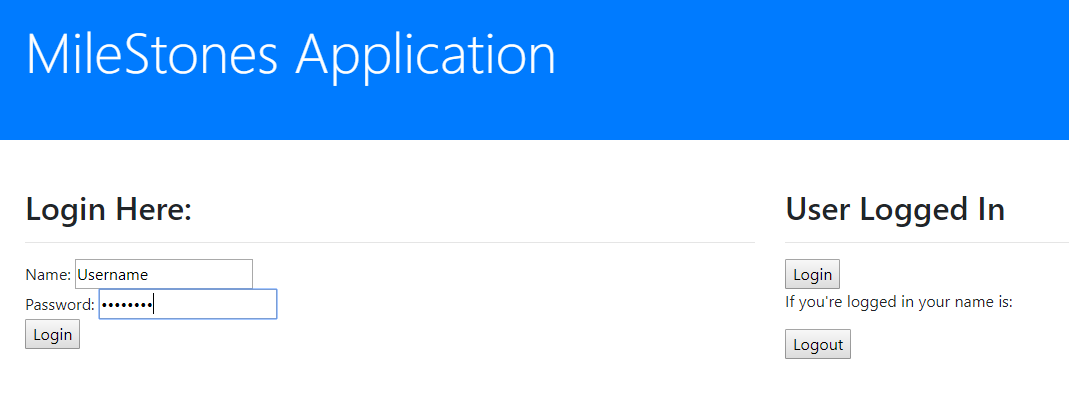
**Index Page**

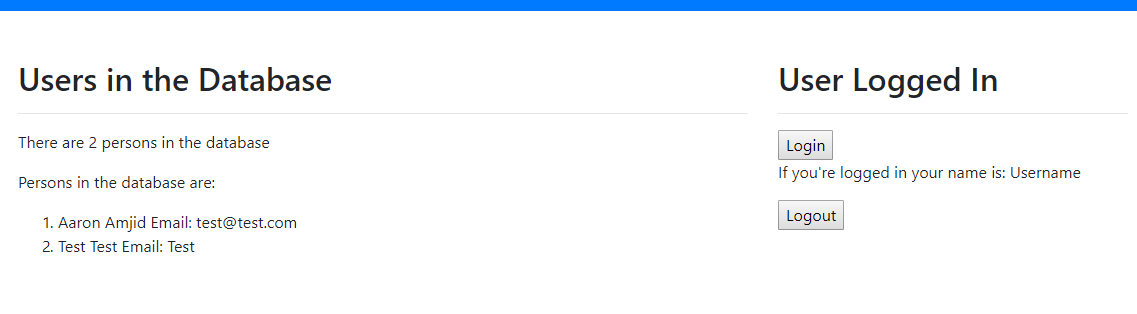
The main purpose of the index page is to act as a landing page for the website and provides some details, such as the contents of the Persons database. This is provided using a doGet method in the servlet which pulls from the database using a data access object. The page also contains a link to the Login page and also provides a Log Out button. Both the actions of logging in and logging out are functional, however the act of doing so is mostly inconsequential to the data displayed by the application.

Index page without logging in



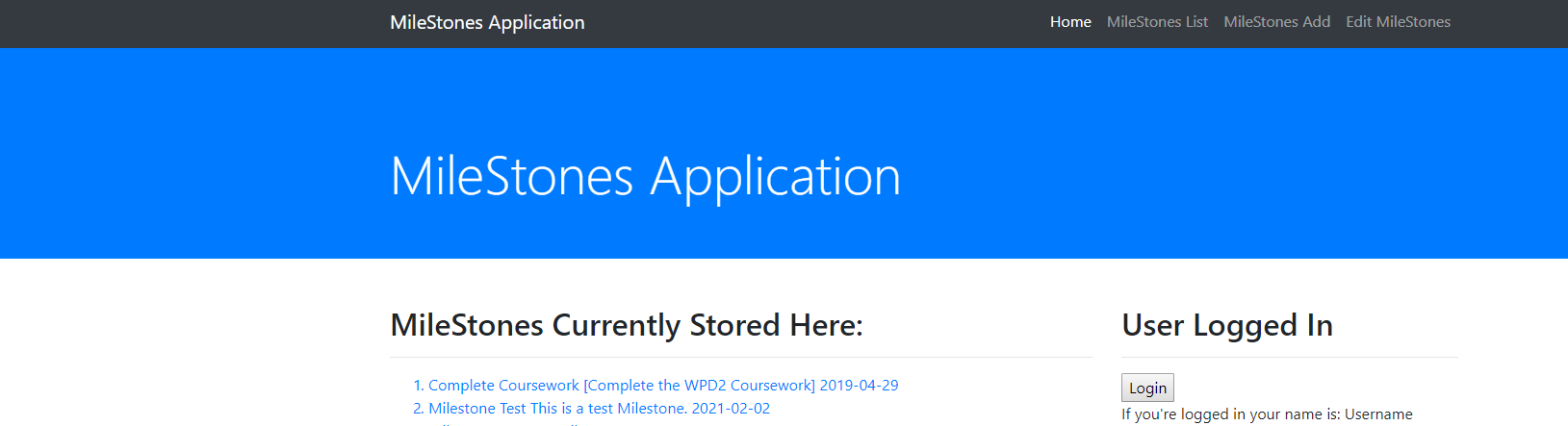
Index page when logged in





**Milestones Page**

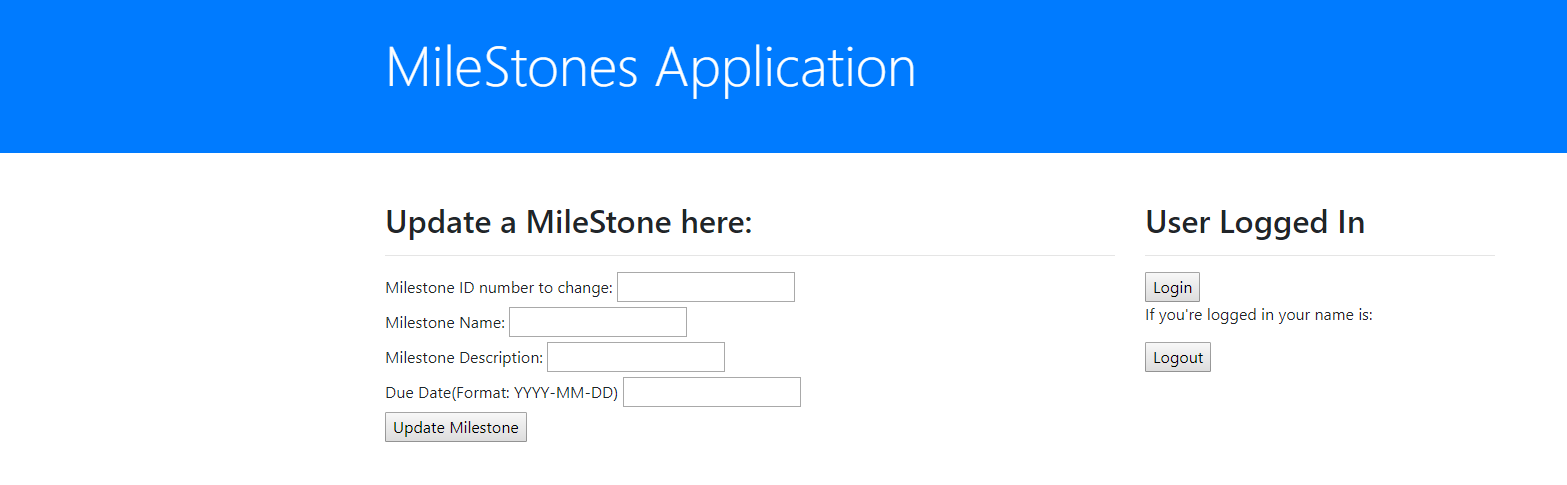
The Milestones page contains the information for all current milestones stored within the application’s database. This information is retrieved, similar to the index page, by using the doGet method within the milestone page servlet to retrieve the data from the milestones database using a data access object. For each milestone, a name, description and due date for each milestone is listed.



From the milestones page, a user can click on any milestone and they will be taken to the edit milestone page. Alteratively, if there are no milestones, a user can use the navigation bar to go to the Add Milestones page.

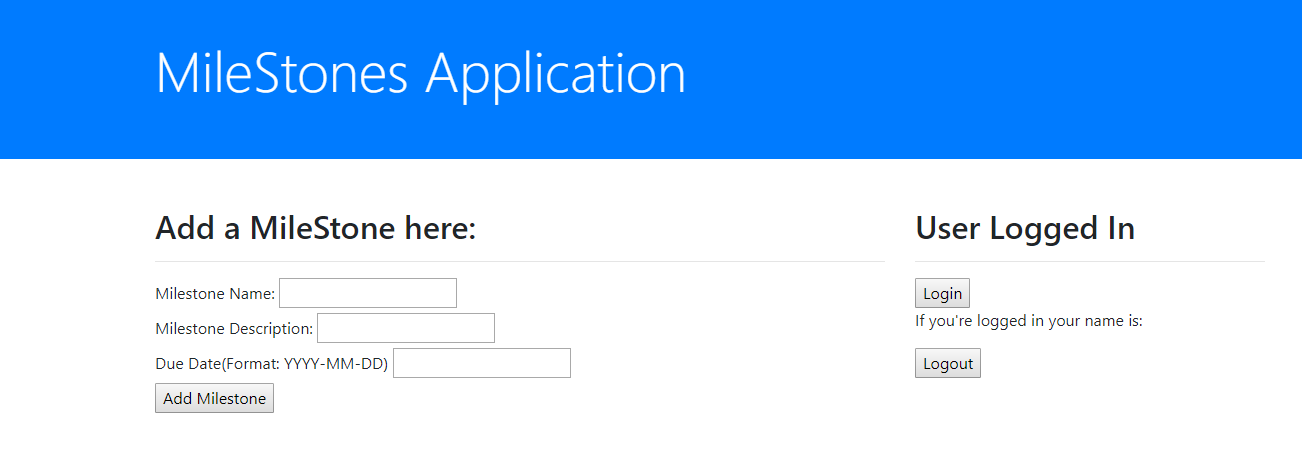
**Update Milestones Page**

The Update Milestones page contains a form which is used to update the milestone specified. In this version of the application, an ID number is requested to change the milestone, however, the group’s intention was to implement a method of editing a specified milestone by clicking on it in the milestone page, rather than add another layer of user input. The application works by using the update milestone servlet’s doPost method which is called by the submit button and runs an appropriate SQL statement on the database. Once all information is filled out, the user is redirected to the milestones page and the specified milestone is updated.



**Add Milestone Page**

The Add Milestone page allows a user to fill out a form which creates a new Milestone and adds it to the Milestones list. The page specifies a name, description and due date in a specified date format. The code behind the add milestones page works similarly to the update milestones page, using a doPost method to post the parameters needed for the data access object to perform the appropriate action within the milestones database.



A minor additional function which was implemented was that the application is responsive, due to the webpages utilising Bootstrap. While not a major function, or very recognisable, we felt that it should be recorded within the functionality section of the report for posterity.

## Testing

For the purpose of testing the application, various tests were done throughout the development process. This included, testing of application navigation, testing of user database, testing of milestones database, testing of login system and the testing of milestones management.

To begin with the website navigation was very bare, only including links to each page with no further hints of how to navigate the app, through the use of testing it was determined that a navigation bar and changes to the links to pages would be required and so this was added through the use of Bootstrap, Allowing for a consistent presence of links throughout the app.

For the testing of the user database users were added to the database to ensure that the entries went in correctly, numerous variations were attempted with some being attempts at SQL injection to ensure that the database was safe and secure.

The testing of the milestones database was done in a similar way to the user’s database, milestones were added to the database, some inside and some outside of the parameters defined on the apps input boxes, this is to guide the user away from typing in information which is incorrect. Due to time constraints, there was not enough time to implement a way of validating the information the user types in, and so incorrect date formats are accepted into the system.

The testing of the login system was done by typing in a username and password into the system, through testing it was found that passwords had no link to the username and so anything could be typed in, this however did show that the logging in and out functionality did work, it however was just not able to validate entries, as the login and logout buttons did function correctly.

To test the milestones management aspect of the application, tests were carried out by creating milestones that both were inside and outside of the parameters defined in the text boxes (wrong date format), these milestones were then edited to be out of or inside the boundaries, whichever they were not originally, this caused no issues.

# Application Security

Within the application, security was one factor which was considered during the initial design and this was reflected in a few of the decisions made during implementation. One of the main security measures which was implemented into the application was within the methods which handled the database operations. As stated within the section discussing persistence, the methods implemented made use of Prepared Statements. The main security benefit of Prepared Statements is that they reduce the chance of SQL injection attacks. This helps to prevent the possibility that malicious code can be inserted into the application to affect the database. Within the database, we also recognised that there should be some security consideration regarding the storage of usernames and passwords. When designing the implementation of the username and password table, we recognised that passwords which are stored should be hashed, to uphold security, therefore if passwords are stolen from the database, security is upheld.

Another security aspect which we took into consideration within the design and implementation of the program is the usage of sessions and security measures associated with sessions. Within the application, we attempted to implement sessions and partially succeeded, however, some functionality within the application does not function with sessions. Originally, the intent was to have milestones be associated with particular users based off of using sessions and retrieve milestones based off of the current session user’s login, however, this function did not pan out. Another security aspect associated with sessions which could be implemented is session tokens. Session tokens are used to ensure that the current session is valid. Instead of sending the entirety of the user’s information for each request, a token is generated which is sent along with the request instead to ensure that the session is valid.

Another aspect associated with web application security which should be considered is cross origin resource sharing. This refers to the concept of sharing a website’s resources through a request which is made external to the website’s origin. Usually this is blocked by a CORS error which states that cross-origin requests cannot be made. The problem with CORS is that it is handled usually by a user’s browser. Browsers handle all requests in a similar manner and often do not distinguish between the circumstances from which the request is being sent, therefore a malicious site may use its code to use a user’s browser to access the website using the user’s details.

One security issue which was noted regarding the web application is the displaying of the contents of the user detail database on the index page. Obviously, this is a big security issue, as this information should be kept confidential. The purpose of displaying the database content was as a method of testing the ability to access the database and display contents and as a way of showing that this is possible. If the application were for public use, this aspect of the website would be removed to maintain user confidentiality.

# Conclusion

After deliberation, the group believe that together they have produced a working application that satisfies some of the major requirements decided upon before we set out development. Throughout the production the team encountered problems that hindered the development process and the timescale. With other deadlines approaching from other modules the team felt that they were not able to give the appropriate time and resources to the project. This had a direct effect on the functionality of the final product. In summary we as a group feel we have produced an application that capable of managing milestones for users to organise and edit. With more time and more educational resources we believe the finished product could have been of a higher quality and calibre.