Web Platform Development 2
Group Report

Module Code: M3I324182-18-B
Module Leader: Katrin Hartmann

Declaration

We declare that all work submitted for this coursework is the work of Angela Lum Neh, Ariyike Adetimehin and Mary Ann Achieng alone unless stated otherwise.

Group 6

Group members	Student ID	Email
Adetimehin Anjolaoluwa Ariyike	S1719003	aanjol200@caledonian.ac.uk
Lum-neh Angela	S1719007	alumne200@caledonian.ac.uk
Mary Ann Achieng	S1719027	machie200@caledonian.ac.uk

Table of Content

1 Introduction

As part of the Web Platform Development 2 module, a coursework is required to display understanding of content taught through the course of the term. In the sections that follow, the report highlights some important aspects of the artifact submitted. The main points discussed include: a discussion of the link design within the application and how these are linked to the functionality of the web application as well as the logic behind the way they have been built, a description of the database schema used within the project and the mechanisms running it, a documentation of the functionality of the web application alongside all tests run with their results, and an appraisal of the application security.

2 Application Link Design

2.1 Servlets, Pages and Sitemap

For the purpose of building the web application, a local Tomcat server was used to build and run tests. All the links that the application is able to handle are included in the web xml file that accompanies the project and are linked to java servlet files which run the logic of the interaction between the components of the webpage.

As part of the design of the application, a story board was created for the assignment. This story board, shown below in *figure 1* and *figure 2*, give a brief overview of how the pages flow from one to the other and the sitemap in *figure 3* gives a clearer picture of this. In the sections that follow, the each of the pages and the servlets that power them are described. Table 1 shows the mapping of the servlet links to the pages and the corresponding functionality.

Link	**Functionality**	**Pages**
/dashboards	Loads up the dashboard, a page with all the projects attached to the current user	Sends to: dashboard.jsp
/AddProjectServlet	Adds a new project to the user and sends a response to the dashboard	Sends to : dashboard.jsp Receives from: add_project.jsp
/RemoveProjectServlet	Removes a list of project(s) from the user and sends a response to the dashboard	Sends to: dashboard.jsp Receives from: remove_project.jsp
/MilestoneMenuServlet	Loads up the milestone menu, a page with possible actions for the milestones	Sends to: project.jsp Receives from: dashboard.jsp

/AddMilestoneServlet	Adds a new milestone to the current project and sends a response to the page with all milestones	Sends to: all_milestones.jsp Receives from: add_millestone
/RemovalServlet	Loads up the (form) checklist for the removal of projects from the current user	Sends to: remove_project.jsp Receives from: dashboard.jsp
/RemoveMilestoneServlet	Removes a list of milestone(s) from the current project and sends a response to the page with all milestones	Sends to: all_milestones.jsp Receives from: remove_milestone.jsp
/EditMilestoneServlet	Edits a given milestone with a completion date and/or changes its status to complete	Sends to: all_milestones.jsp Receives from: edit_milestone.jsp
/LoginServlet	Logs a user into the system if they exist in the database	Sends to: dashboard.jsp Receives from: login.jsp
/LogoutServlet	Logs the user out of the system and ends the current session	Sends to: index.jsp Receives from: all pages
/RegisterationServlet	Registers a new user in the system by creating them in the database	Sends to: login.jsp Receives from: index.jsp

Table 1: *Mapping of servlet links to pages and the corresponding functionality*

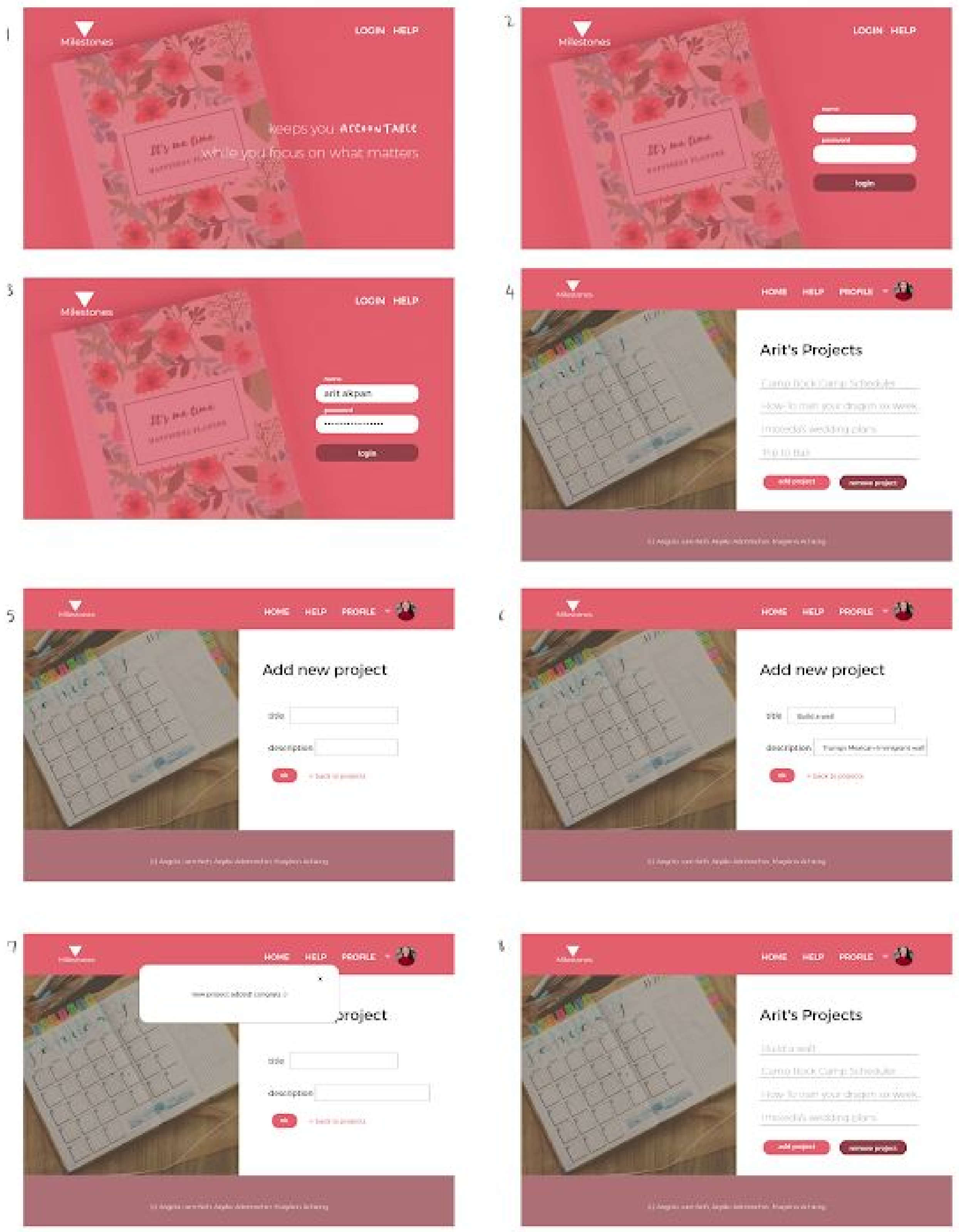


Figure 1: Screens 1-8 of the design storyboard

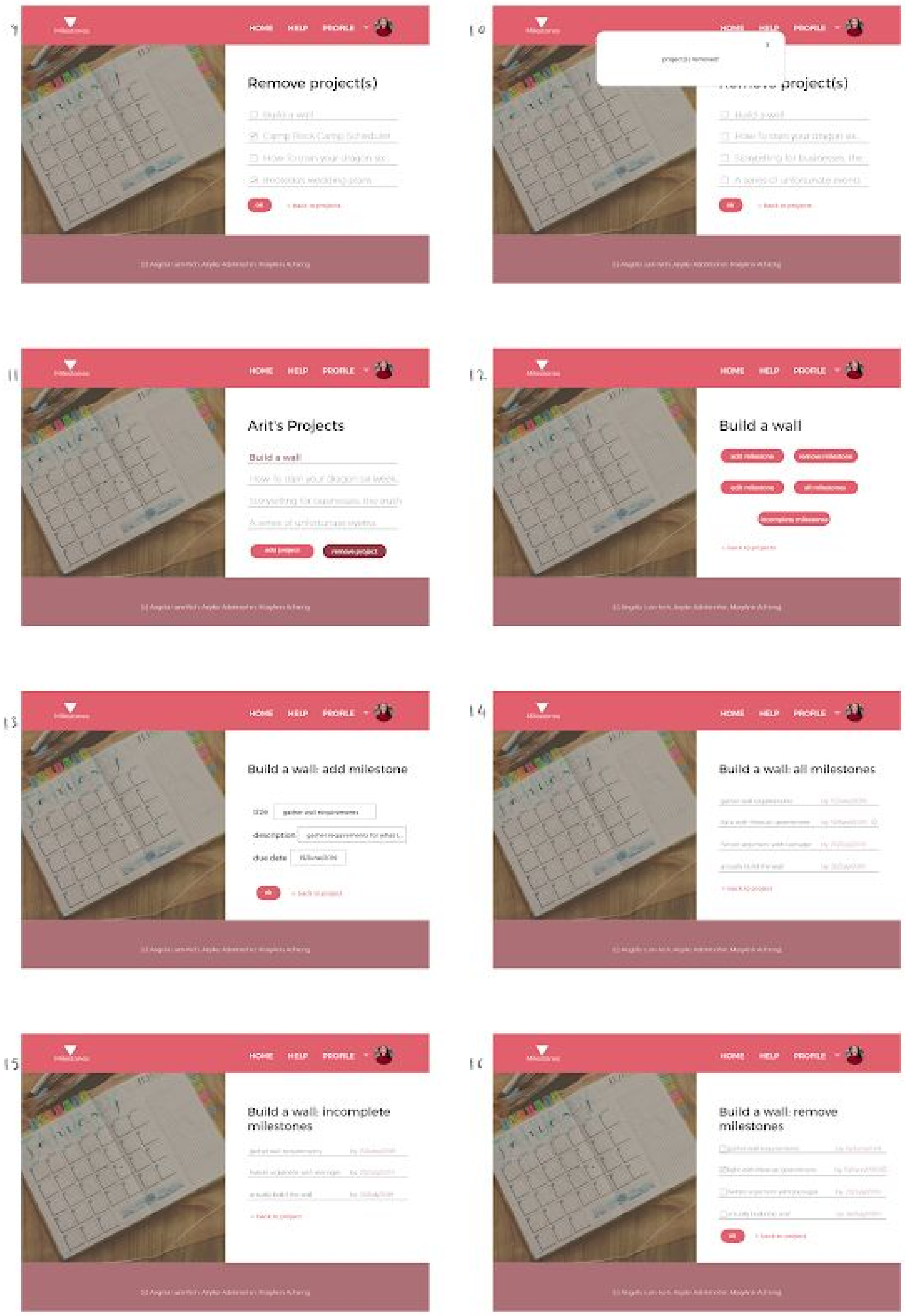


Figure 2: *Screens 9-16 of the design storyboard*

2.2 Logic of the link schema

The reasoning behind the links that have been used for the website is simply to use the base site url (retrieved by using getContextPath() method of the HTTPServlet class), and the applicable servlet name in full. This is done in consideration of the way design of the classes and Java Server Pages (JSP) have been done which is that there is a corresponding servlet handling each major operation within the web application. In many of the scriptlet methods and the servlet logic, the parameters for operations are retrieved from the url of the pages themselves. Figure 3 provides a graphical view of how the pages have been laid out.

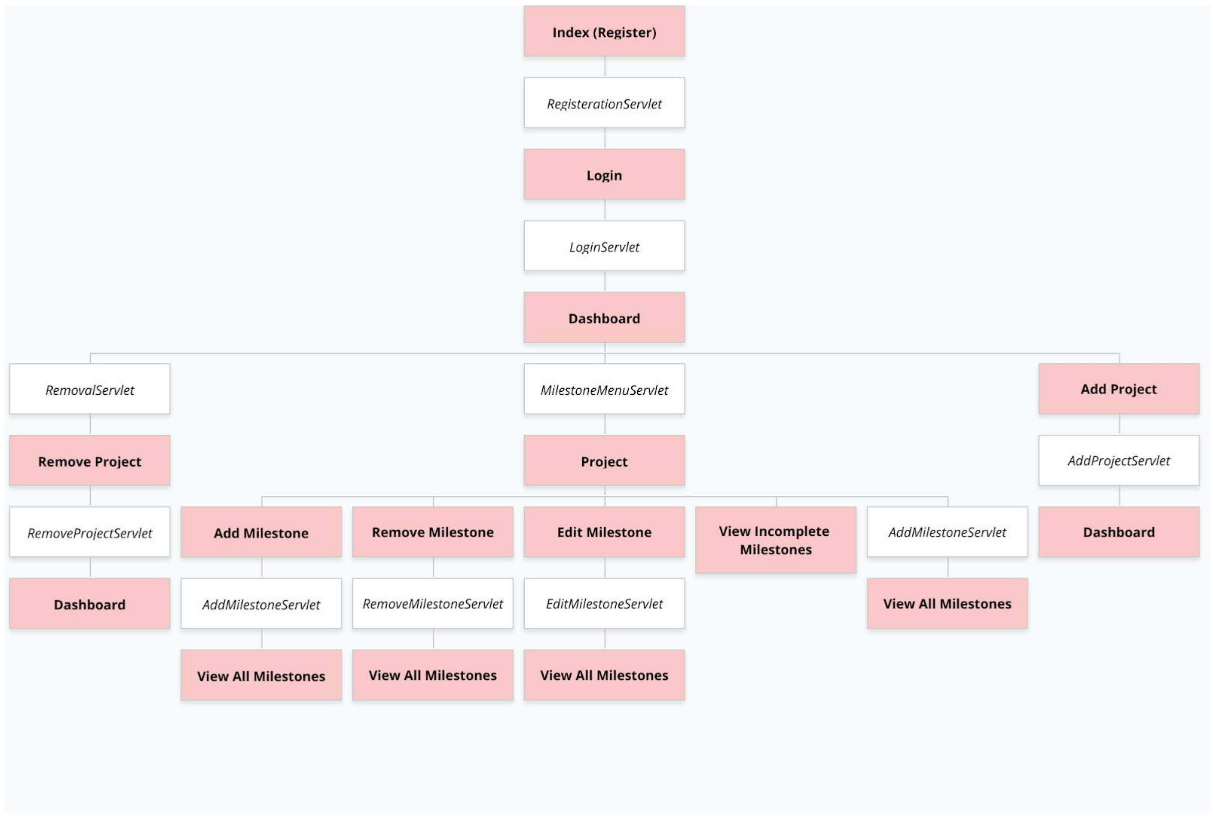
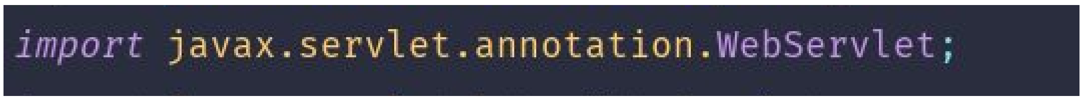


***Figure 3:** Sitemap showing all pages and related servlets*

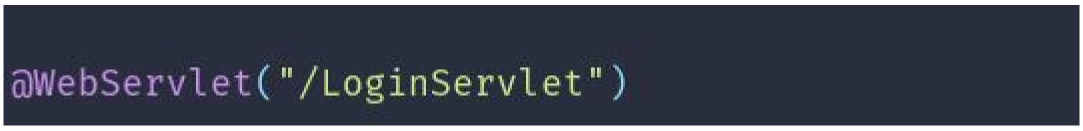
For this reason, it was necessary to keep the names as they exist within the project structure for the best possible access to parameters and variables as they are needed. Using this structure for the links ensured the fastest possible access to the pages and servlets whenever they are needed to be run on as part of the web application.

Servlet referencing was handled in the servlets themselves. This was done using the "@WebServlet" annotation before the declaration of each of the servlets, see *figures 4 and 5* for screenshots. The names configured are the names of the servlets themselves for reasons discussed earlier. The ideal linking method would be to use the web.xml file that accompanies the project folder as is shown in documentation which can be found here.



```
import javax.servlet.annotation.WebServlet;
```

Figure 4: *Import statement for the WebServlet*



```
@WebServlet("/LoginServlet")
```

Figure 5: *Annotation for the servlet*

3 Application Data Access and Persistence Mechanism

3.1 Database Schema

In this application, the H2 database was used in embedded mode to store user information, project(s) and milestone(s) information. The database ensures the data is persistent throughout and after a session ends and allows for more than one user to use the application.
There are 3 classes, H2User, H2Project, and H2Milestone which enable this functionality. In each of these classes, a table is created that holds the information acquired from the forms that collect information on the user(s), project(s) and milestone(s) respectively in the application. The database was designed to ensure that every user has a milestone board which holds the projects for that user. In tend, each project has a list of milestones that belong to that project. To achieve this, the user's id is used as a foreign key in the projects table while the project id is used as a foreign key in the milestone table as shown on the ER diagram below.

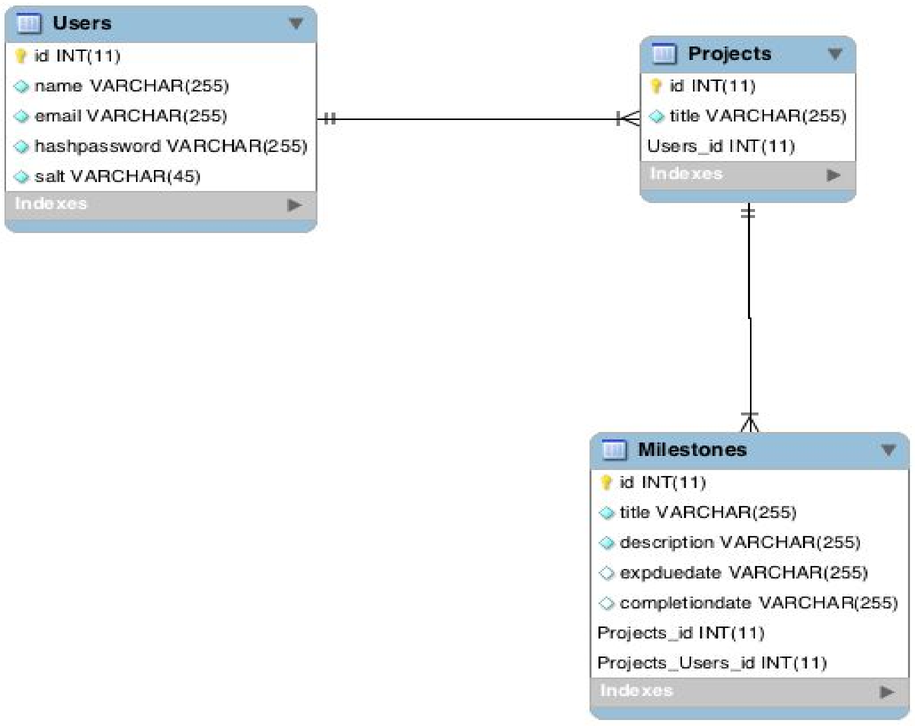
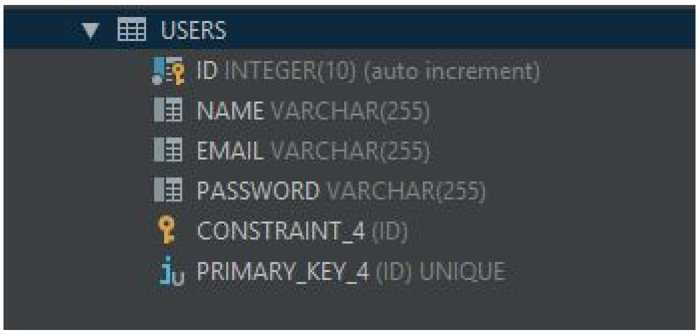


Figure 6: *ER diagram of Database Schema*

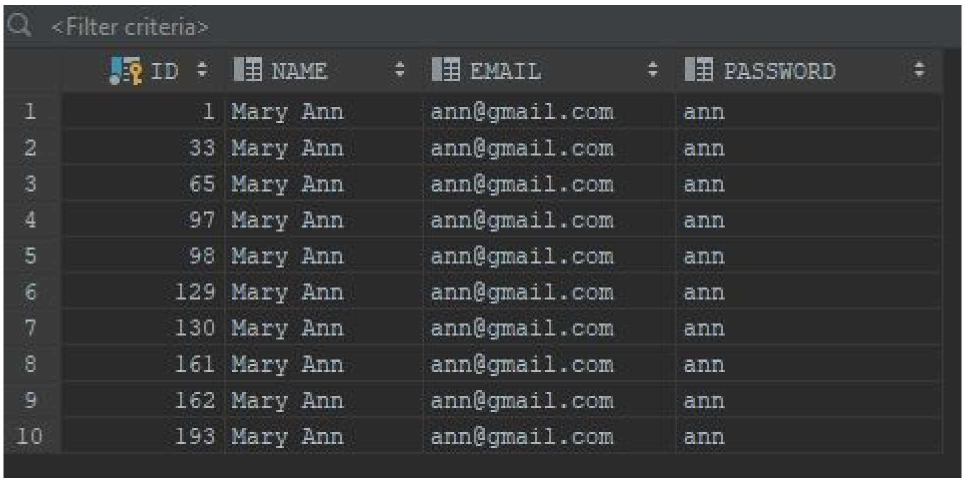
The different tables as seen on the H2 are described below in more details. The table in H2User holds information as shown in *figure 7* below.



```
▼ USERS
    ID INTEGER(10) (auto increment)
    NAME VARCHAR(255)
    EMAIL VARCHAR(255)
    PASSWORD VARCHAR(255)
    CONSTRAINT_4 (ID)
    PRIMARY_KEY_4 (ID) UNIQUE
```

***Figure 7:** The Users Database Table Variables*

The information is collected during registration when the user inputs their details and stored in columns and rows as shown in *figure 8* below. An instance of when this data is used is when the user is directed to the login page after registration. When they enter their name and password, the password they enter is checked against the password they registered with to determine whether it is a correct or incorrect password

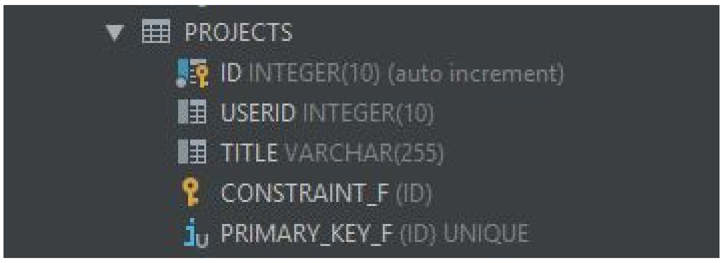


<Filter criteria>

	ID	NAME	EMAIL	PASSWORD
1	1	Mary Ann	ann@gmail.com	ann
2	33	Mary Ann	ann@gmail.com	ann
3	65	Mary Ann	ann@gmail.com	ann
4	97	Mary Ann	ann@gmail.com	ann
5	98	Mary Ann	ann@gmail.com	ann
6	129	Mary Ann	ann@gmail.com	ann
7	130	Mary Ann	ann@gmail.com	ann
8	161	Mary Ann	ann@gmail.com	ann
9	162	Mary Ann	ann@gmail.com	ann
10	193	Mary Ann	ann@gmail.com	ann

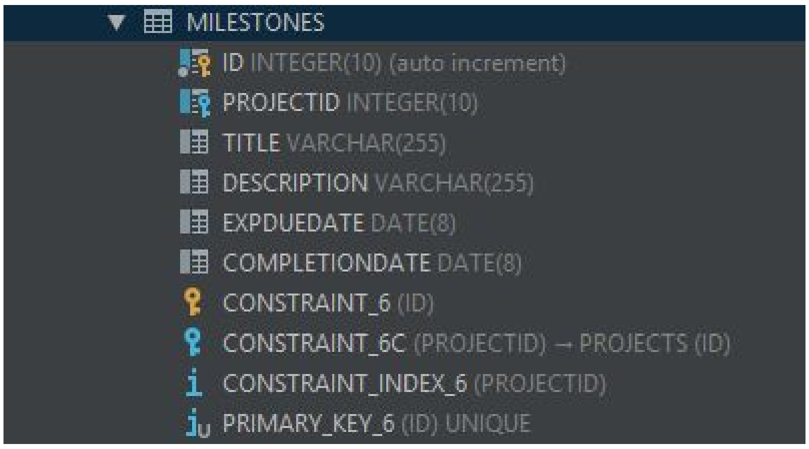
***Figure 8:** Data stored in the user database in columns and rows*

The project(s) and milestone(s) are stored in the same format as the user(s), in columns and rows, database with the only difference being variables. The *figures 9 and 10* below show the information held by the project(s) and milestone(s) databases.



```
▼ PROJECTS
    ID INTEGER(10) (auto increment)
    USERID INTEGER(10)
    TITLE VARCHAR(255)
    CONSTRAINT_F (ID)
    PRIMARY_KEY_F (ID) UNIQUE
```

***Figure 9:** The Projects Database Table Variables*



```
▼ MILESTONES
    ID INTEGER(10) (auto increment)
    PROJECTID INTEGER(10)
    TITLE VARCHAR(255)
    DESCRIPTION VARCHAR(255)
    EXPDUEDATE DATE(8)
    COMPLETIONDATE DATE(8)
    CONSTRAINT_6 (ID)
    CONSTRAINT_6C (PROJECTID) → PROJECTS (ID)
    CONSTRAINT_INDEX_6 (PROJECTID)
    PRIMARY_KEY_6 (ID) UNIQUE
```

Figure 10: *The Milestones Database Table Variables*

4 Functionality of the application

4.1 Functionality of the application

The entire milestone application is designed to enable a user to create and store milestones. At the very base of the application, it should store the user's milestones and enable the user to add, remove and edit them to their liking. This application has not only the basic functionality, but is comprehensive to include functionality that supports the basic functionality. The flow of the application include the following functionality:

4.1.1 Registration

When a user loads the application, their name, email and password are required to registered. These variables and a salt are read and stored in the database using the addUser method. The user's password is hashed before storing for security measures. The user is directed to the login page.

4.1.2 Log in

A registered user can log in to the application with a username and a password. The user's salt from the database is used to hash the password. The hashed password is then compared with that stored in the database. If the same, the user is directed to their dashboard. In the case that a wrong password is entered, a password that does not match the one stored in the database, the user is returned to the login page and required to input the username.

4.1.3 Add project

The user dashboard contains a list of projects that the user may have previously added to the application. The dashboard has a "add project" button. When the user selects this button, they are requested to add the project name and when they select the "ok" button, the project is added onto their current list of projects and the projects table in the database is updated.

4.1.4 Remove project

In the dashboard, there is a “remove project(s)” button. When the user selects this button, they are directed to a page where they can select one or more projects and remove them. This deletes the projects from the project table. When the user goes back to the dashboard, they will notice that the project(s) that they removed has been removed from the dashboard.

4.1.5 List project milestones

While in the dashboard, the user can select an individual project and be able to view the milestones under that project. This is implemented using the findProjects method from the H2Projects class. The user will have access to the list of milestones submitted for that particular project. They can view the project’s full list of milestones and the project’s incomplete milestones.

4.1.6 Add milestone

While in the view of an individual project’s milestones, the user can select the “add milestone” button to add a milestone to the project. The user will be prompted to enter the milestone title, it’s description, its expected due date and then to select “ok” to add the milestone and the milestone table in the database is updated. When the user goes back to their project page, they shall view the added milestone with the previously added milestones. It is important to note, that when a new milestone is added, its completion date is set to the same date as the due date and can be edited once actually completed. This edition will change the status of completion as necessary.

4.1.7 Edit milestone

The user can select a single milestone and edit it. This calls the updateMilestone method from the H2Milestone class. The milestone is then updated on the list of milestones. How this works is that a user selects the milestone to be edited from a list of radio options, then clicks ok. Once sent to the EditMilestone servlet, the new completion date selected will determine whether the milestone status is changed from incomplete to complete automatically by logic written within the milestones jsp page.

4.1.8 Remove milestone

While in the view of an individual project’s milestones, the user can select the “remove milestone” button to remove a milestone(s) from the project. When the user selects this button, they are directed to a page where they can select one or more milestones and remove them using the removeMilestone method from the H2Milestone class. When the user goes back to the milestone list, they will notice that the milestone(s) that they removed has been removed from the milestone list.

4.1.9 Log out

In each page, there is a “logout” option which the user can select. When the user selects this option, they are directed out and back to the login page and may choose whether or not to log in again.

4.2 Unachieved Functionality

4.2.1 The “Share” Feature

At the beginning of the development of the milestone application, it was expected that there would be a share feature. This would enable the user to share their milestones with other users. This functionality has not been implemented in this version of the application due to time constraints. In the quest to achieve the best functionality for the other features, there was no time left to fully implement the share feature in this version of the application. Given more time, however, the development team would look into implementing the feature.

4.3 Visual Design Considerations

The design and layout of the application is intentional towards creating a smooth experience for the user. The navigation bar is present in all pages to give the user ease of moving through the application. The “back” button is also present in all pages to ensure ease of moving back and forth within the application.
The bootstrap grid system has been used to determine the general layout of the pages of the application. It has been used to divide each page into reasonable divs. This ensures the pages are well-planned and the content well-positioned on the pages.
The colour scheme used throughout the application is calm and yet not dull. This is intended to keep the usee engaged and in the ‘right space of mind’ to plan their goals/milestones for their projects. The colour scheme is emphasized in the little details such as the buttons and links which provides a continuous feel to the flow of the application from start to finish.

5 Testing and test reports

Testing for the application was done for every required functionality. Throughout development, after the creation of a new feature, testing would be done and an iteration would follow if the desired results were not achieved. If the desired results were achieved, the development of the next feature would proceed. After the full development of the application, all the features were tested again to ensure they were all well-connected and could work seamlessly in a flow.
In the *table 2* below, the tests carried out are listed together with the expected results and the actual achieved results. Screenshots of the test results can be seen in Appendix A.

Tested Feature	**Expected Result**	**Achieved Result**
Registration	The application would display a registration form in which a user can input their name, email and password. When the user selects “register” they would be directed to the login	The application displays a registration form in which a user can input their name, email and password. When the user selects “register” they are directed to the login page.

	page.	
Log in	The application would display a login form in which a user can input their name and password. **Scenario 1: Correct password** When the user inputs a correct password, they would be directed to the dashboard. **Scenario 2: Incorrect password** If the user inputs an incorrect password, they would be taken back to the login page and prompted to enter the password again.	The application displays a login form in which a user can input their name and password. **Scenario 1: Correct password** When the user inputs a correct password, they are directed to the dashboard. **Scenario 2: Incorrect password** If the user inputs an incorrect password, they are taken back to the login page and prompted to enter the password again.
Add project	While on the dashboard, when the user selects the “add project” button, they would be directed to a page where they would be prompted to enter the project title and description. When they select the “ok” button, the project would be added to the list of projects on their dashboard.	While on the dashboard, when the user selects the “add project” button, they are directed to a page where they are prompted to enter the project title and description. When they select the “ok” button, the project is added to the list of projects on their dashboard.
Remove project(s)	While on the dashboard, when the user selects the “remove project” button, they would be directed to a page where they would be prompted to select the project(s) they would like to remove. When they select the “ok” button, the project(s) would be deleted from the list of projects on their dashboard.	While on the dashboard, when the user selects the “remove project” button, they are directed to a page where they are prompted to select the project(s) they would like to remove. When they select the “ok” button, the project(s) is deleted from the list of projects on their dashboard.
List project milestones	When the user selects one project, they would be directed to a page with different choices on what to do with the project. When the user selects “All milestones” they would view a list of the currently added milestones for that particular project.	When the user selects one project, they are directed to a page with different choices on what to do with the project. When the user selects “All milestones” they view a list of the currently added milestones for that particular project.
Add milestone	When the user selects one	When the user selects one

	project, they would be directed to a page with different choices on what to do with the project. When the user selects “Add milestone” they would be directed to a page where they would be prompted to enter the milestone title, its description, and its expected due date. When the user selects “ok”, this milestone would be added to the list of milestones of the current project.	project, they are directed to a page with different choices on what to do with the project. When the user selects “Add milestone” they are directed to a page where they are prompted to enter the milestone title, its description, and its expected due date. When the user selects “ok”, this milestone is added to the list of milestones of the current project.
Edit milestone	When the user selects one project, they would be directed to a page with different choices on what to do with the project. When the user selects “Edit milestone” they would be directed to a page where they would be prompted to select and edit a milestone’s date of completion. When the user selects “ok”, the edited milestone would be added back to the list of milestones of the current project.	When the user selects one project, they are directed to a page with different choices on what to do with the project. When the user selects “Edit milestone” they are directed to a page where they are prompted to select and edit a milestone’s date of completion. When the user selects “ok”, the edited milestone is added back to the list of milestones of the current project.
Remove milestone	When the user selects one project, they would be directed to a page with different choices on what to do with the project. When the user selects “Remove milestone” they would be directed to a page where they would be prompted to select the milestone(s) they would want to remove. When the user selects “ok”, the milestone(s) would be deleted from the list of milestones of the current project.	When the user selects one project, they are directed to a page with different choices on what to do with the project. When the user selects “Remove milestone” they are directed to a page where they are prompted to select the milestone(s) they would want to remove. When the user selects “ok”, the milestone(s) is deleted from the list of milestones of the current project.
Log out	When the user selects the “logout” button on top of a page, they would be directed out and back to the login page.	When the user selects the “logout” button on top of a page, they are directed out and back to the login page.

Table 2: *Test reports for each of the functionalities of the application*

6 Application Security

Password Encryption: Hashing and Salting

Given that the implementation of the project requires storing users' information, protecting this information most especially the password is important in case of a database hack. Attacks on the database could range from brute force to dictionary attacks, or lookup tables, reverse lookup tables to rainbow tables. To prevent these attacks on our milestone app, a SHA-512 (Secure Hashing Algorithm) was used with a salt. During user registration, the user's password is hashed with a random salt and both the hash and salt for this user are stored in the database. During login, the password entered by the user is then hashed using the salt stored for this user and the this hashed password compared with that of the user. If both are the same, the user is then logged in. The database therefore contacts the hashed password which can not be decrypted with the salt hence, on the hashed password will be available to a hacker.

SHA-512 was chosen over Message Digest 5 (MD5), SHA-256 and SHA-384 since it is longer and more difficult to break. Nonetheless, there exists more advanced algorithms like PBKDF2, Argon2, Bcrypt or Scrypt which could provide better security since they are well-designed key stretching algorithms.

Appendices

Appendix A

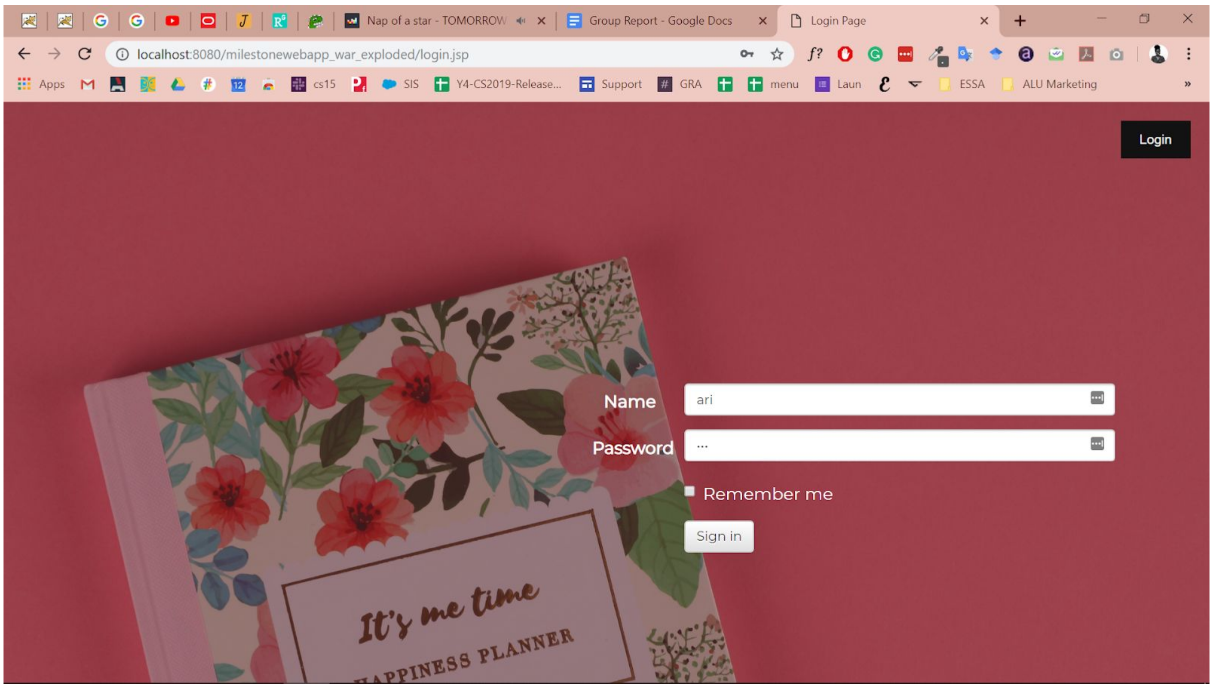
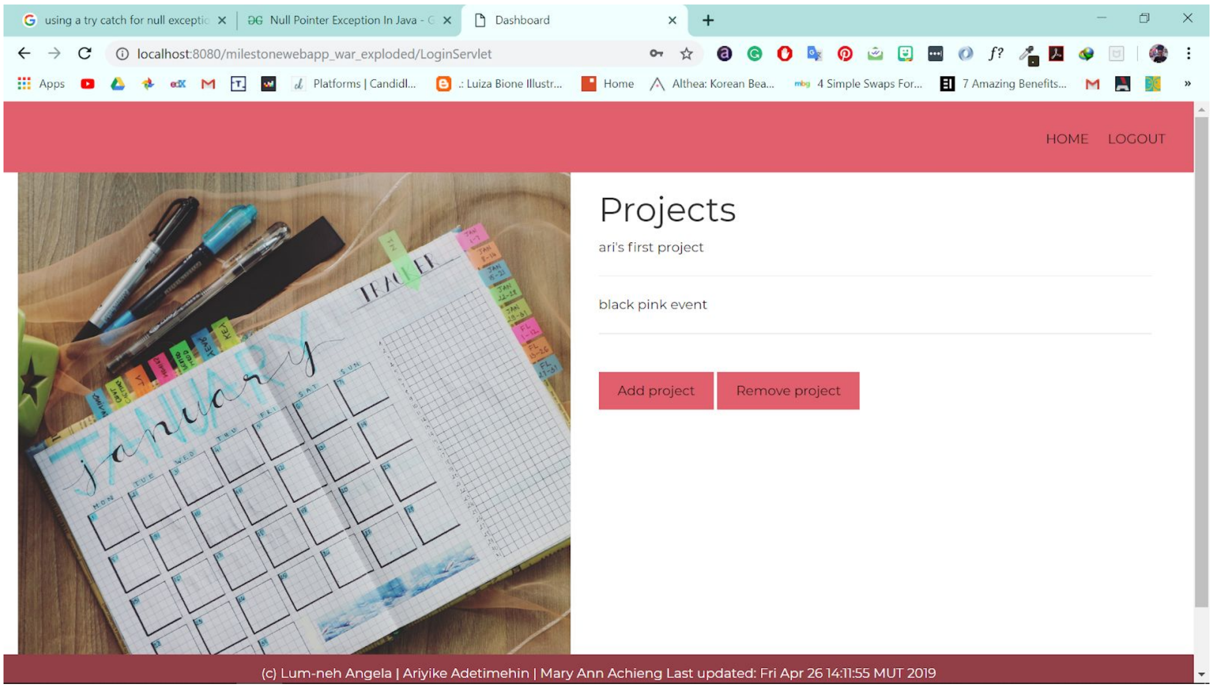


***Figure A:** login screen to the site showing user input*



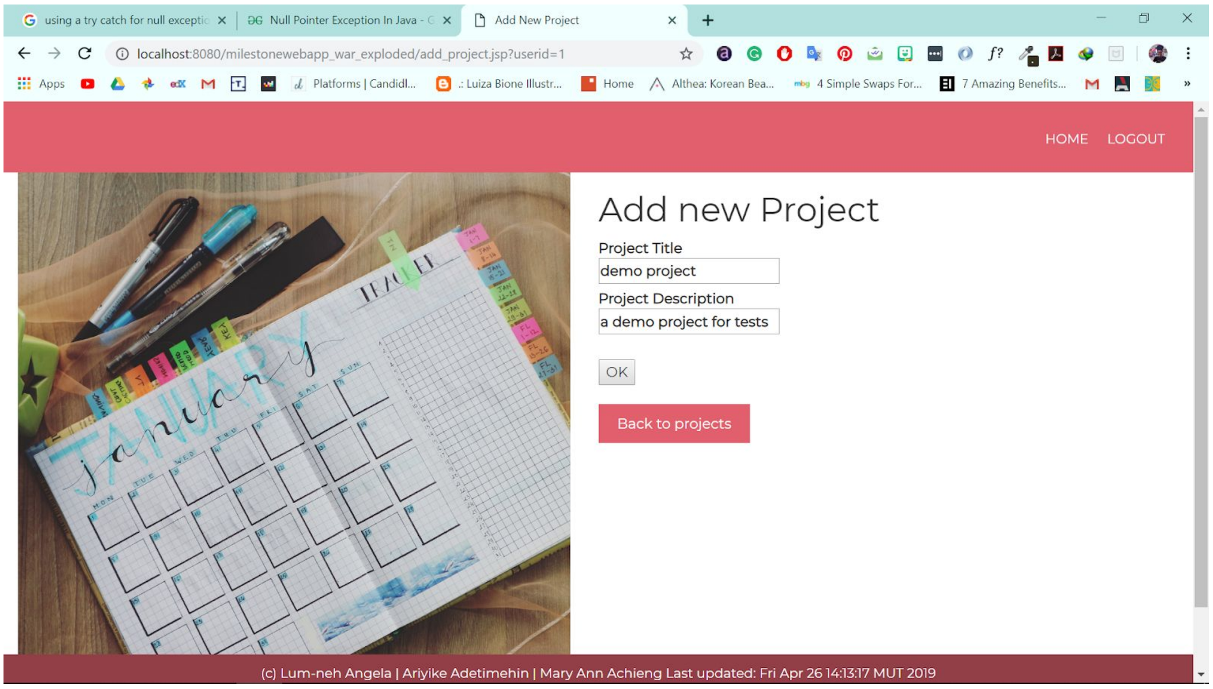
Projects

ari's first project

black pink event

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:11:55 MUT 2019

***Figure B:** successful login showing user dashboard*


using a try catch for null exceptic
Null Pointer Exception In Java
Add New Project
localhost:8080/milestonewebapp_war_exploded/add_project.jsp?userid=1
HOME
LOGOUT

Add new Project

Project Title
demo project

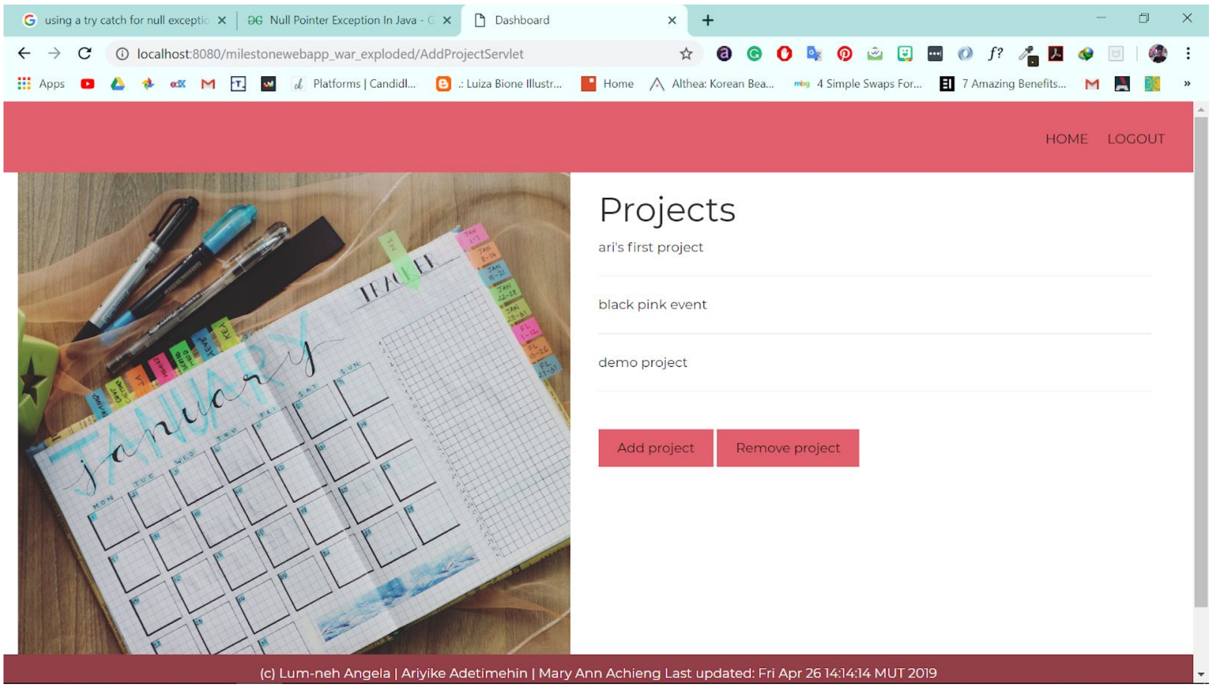
Project Description
a demo project for tests

OK

Back to projects

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:13:17 MUT 2019

Figure C: *Add new project screen with user input*


using a try catch for null exceptic
Null Pointer Exception In Java
Dashboard
localhost:8080/milestonewebapp_war_exploded/AddProjectServlet
HOME
LOGOUT

Projects

ari's first project

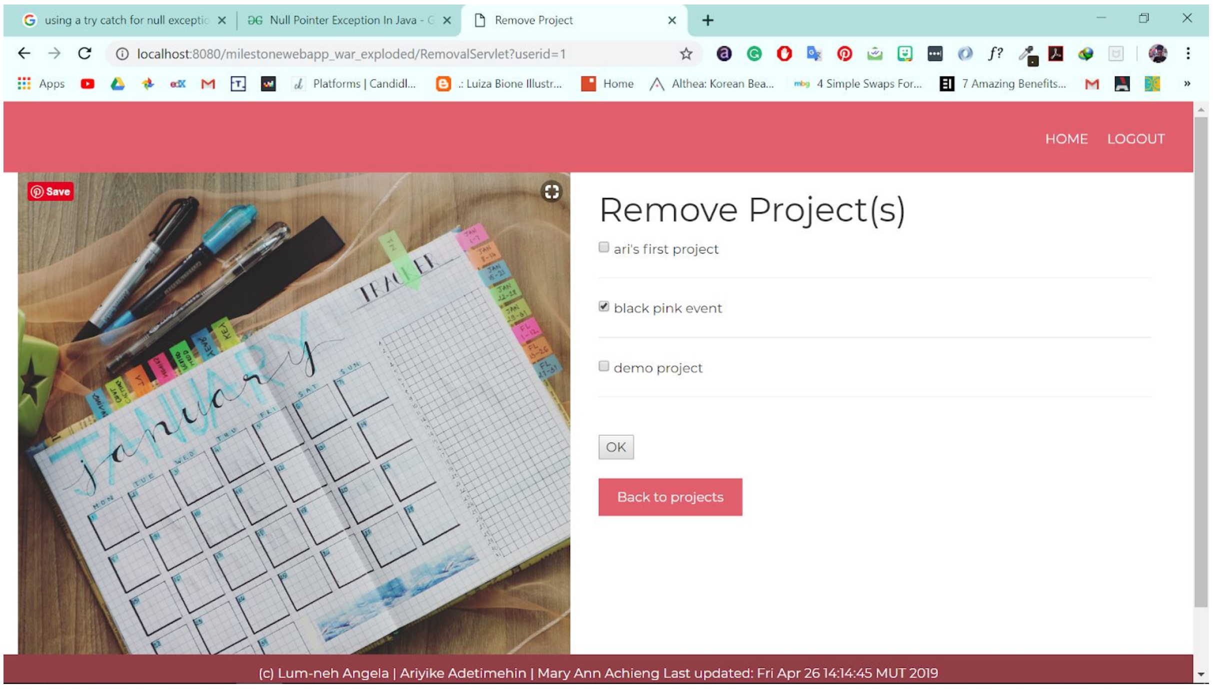
black pink event

demo project

Add project
Remove project

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:14:14 MUT 2019

Figure D: *Project added successfully*



HOME LOGOUT

Remove Project(s)

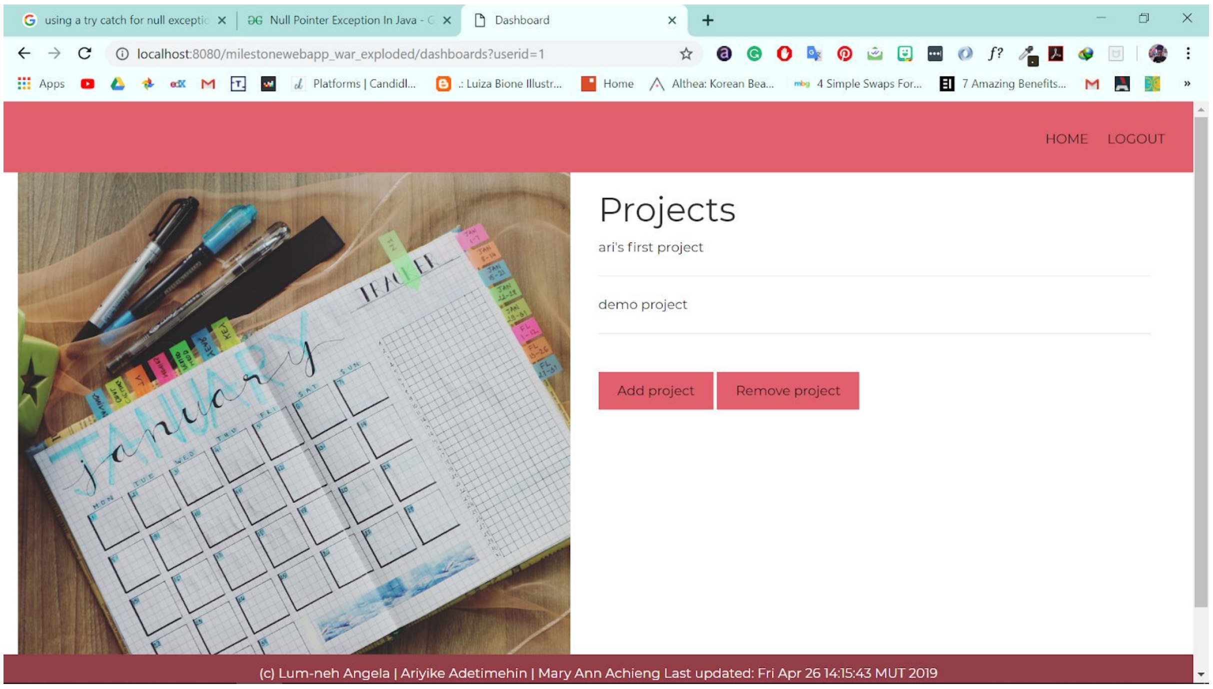
- [ ] ari's first project
- [✓] black pink event
- [ ] demo project

OK

Back to projects

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:14:45 MUT 2019

Figure E: *select a project to remove*



HOME LOGOUT

Projects

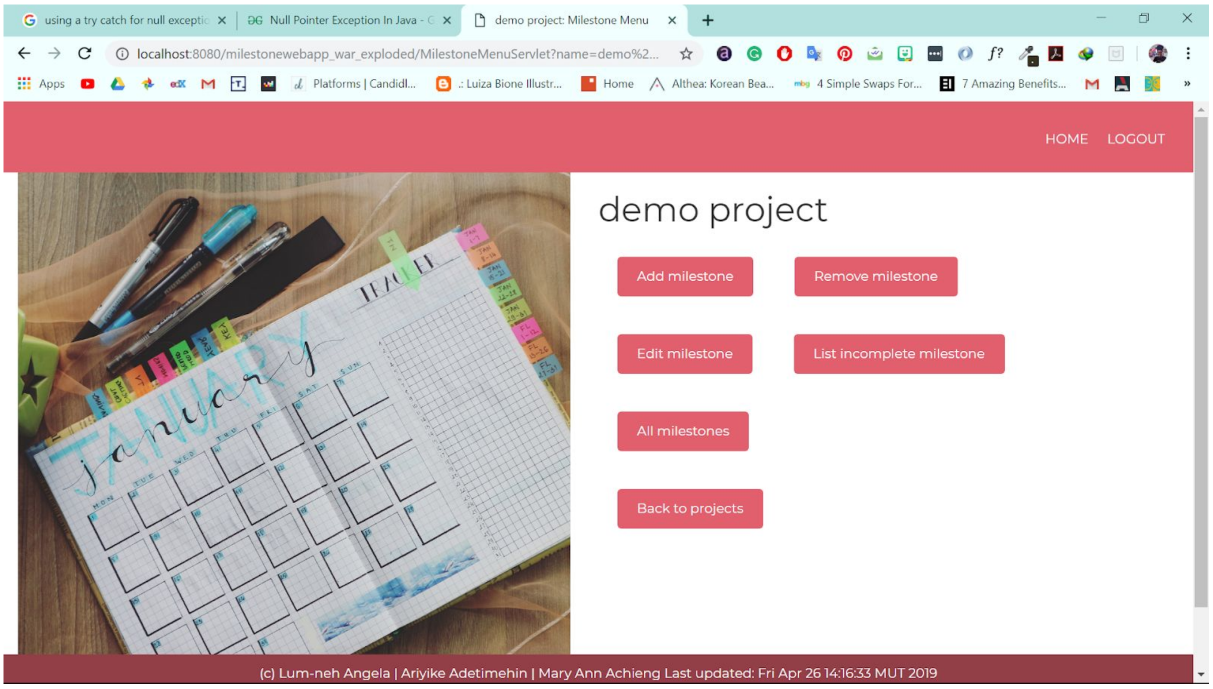
ari's first project

demo project

Add project Remove project

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:15:43 MUT 2019

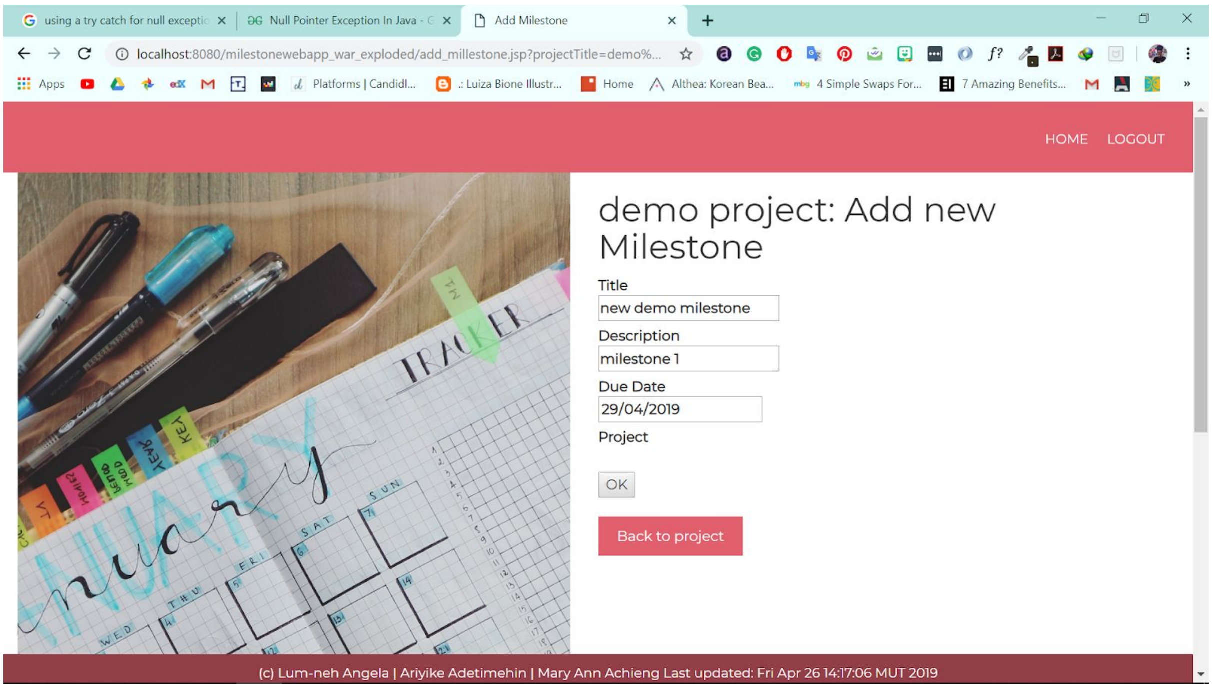
Figure F: *Updated user dashboard without the removed project*


using a try catch for null exceptic
Null Pointer Exception In Java - G
demo project: Milestone Menu
localhost:8080/milestonewebapp_war_exploded/MilestoneMenuServlet?name=demo%2...
HOME
LOGOUT
Add milestone
Remove milestone
Edit milestone
List incomplete milestone
All milestones
Back to projects

demo project

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:16:33 MUT 2019

Figure G: *Milestone menu*


using a try catch for null exceptic
Null Pointer Exception In Java - G
Add Milestone
localhost:8080/milestonewebapp_war_exploded/add_milestone.jsp?projectTitle=demo%...
HOME
LOGOUT
OK
Back to project

demo project: Add new Milestone

Title
new demo milestone

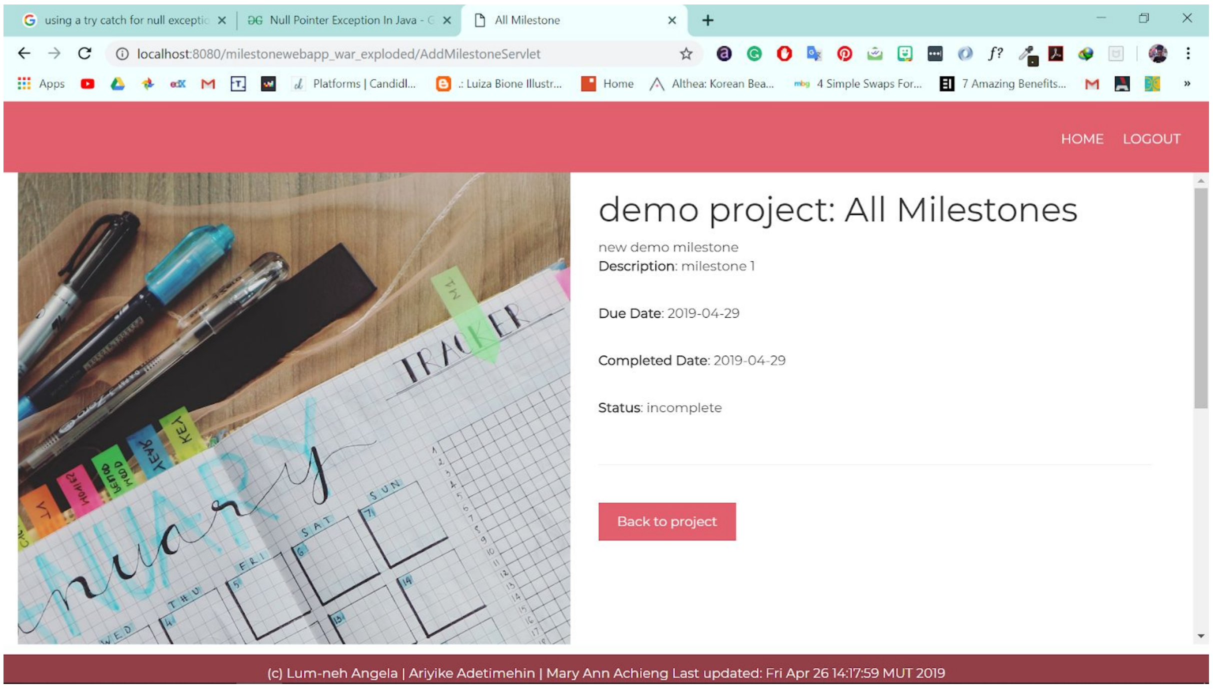
Description
milestone 1

Due Date
29/04/2019

Project

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:17:06 MUT 2019

Figure H: *Adding a new milestone to the project*



localhost:8080/milestonewebapp_war_exploded/AddMilestoneServlet

HOME LOGOUT

demo project: All Milestones

new demo milestone
Description: milestone 1

Due Date: 2019-04-29

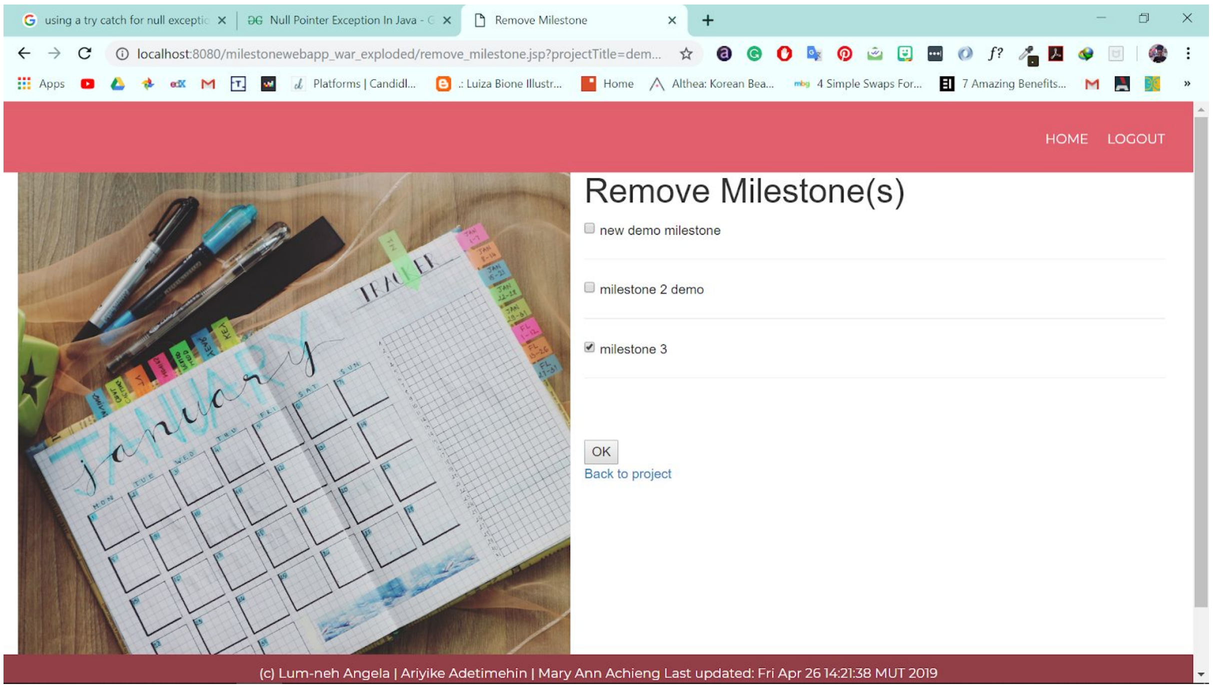
Completed Date: 2019-04-29

Status: incomplete

Back to project

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:17:59 MUT 2019

Figure I: *Milestone added successfully*



localhost:8080/milestonewebapp_war_exploded/remove_milestone.jsp?projectTitle=dem...

HOME LOGOUT

Remove Milestone(s)

☐ new demo milestone

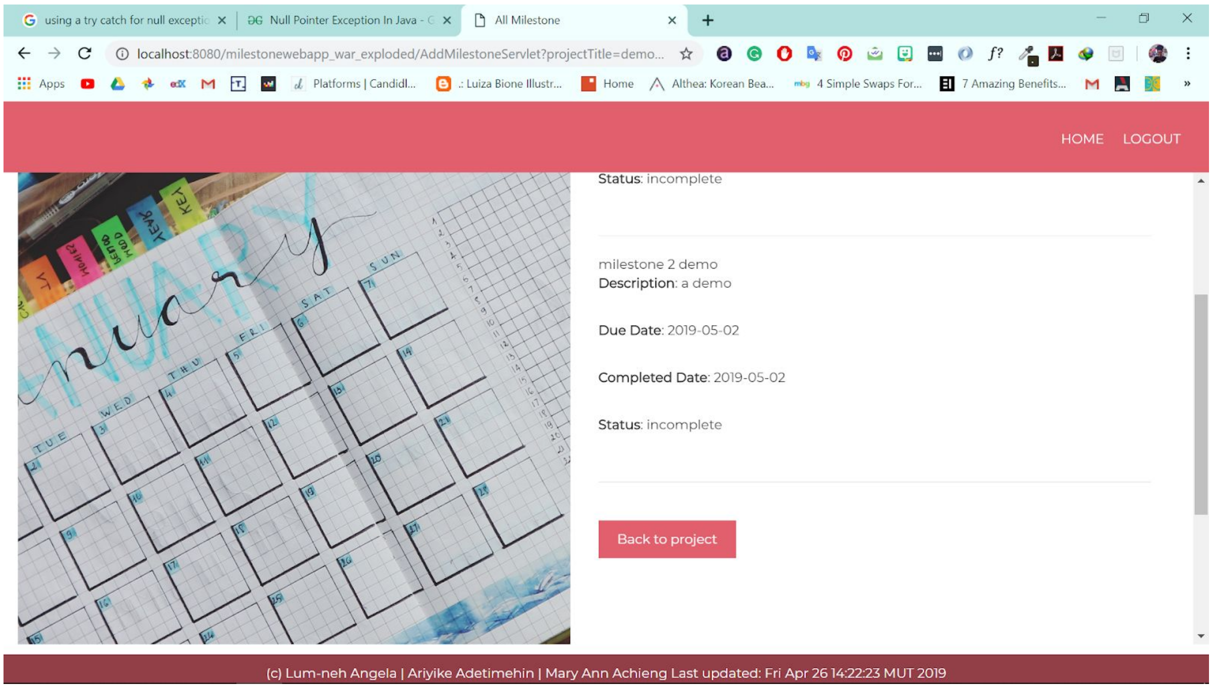
☐ milestone 2 demo

☑ milestone 3

OK
Back to project

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:21:38 MUT 2019

Figure J: *removing milestone 3 from the project*


using a try catch for null exceptio | Null Pointer Exception In Java - G | All Milestone

localhost:8080/milestonewebapp_war_exploded/AddMilestoneServlet?projectTitle=demo...

HOME LOGOUT

Status: incomplete

milestone 2 demo
Description: a demo

Due Date: 2019-05-02

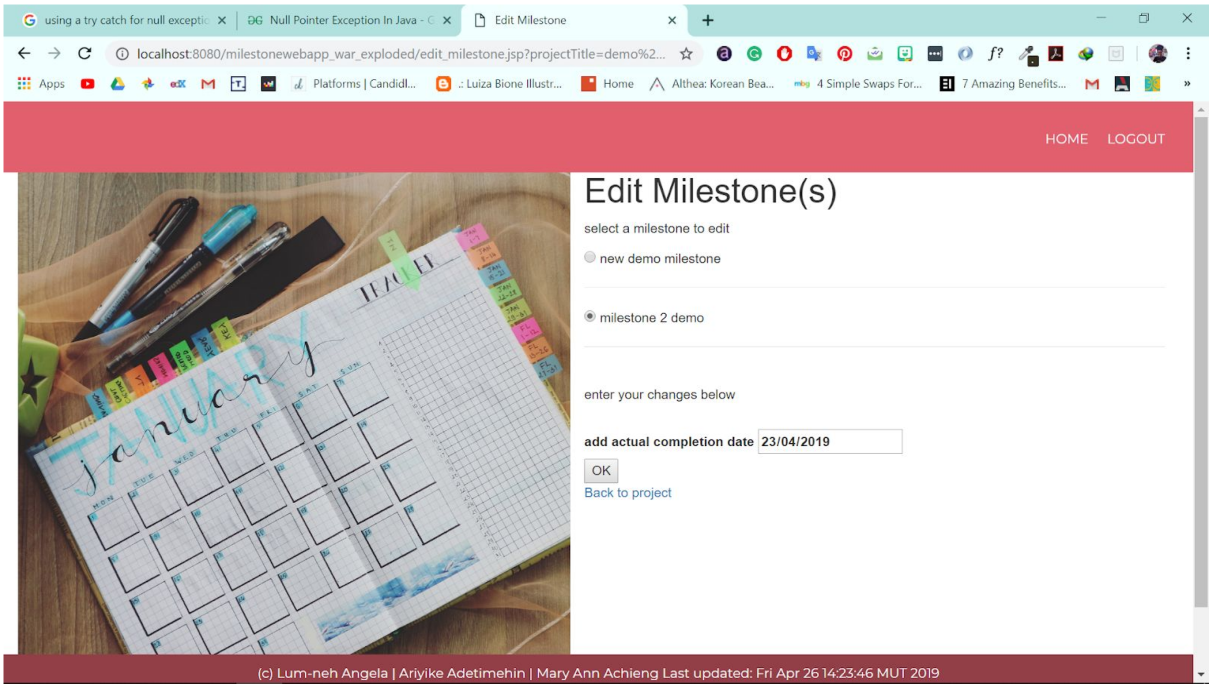
Completed Date: 2019-05-02

Status: incomplete

Back to project

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:22:23 MUT 2019

Figure K: *Milestone 3 removed successfully*


using a try catch for null exceptio | Null Pointer Exception In Java - G | Edit Milestone

localhost:8080/milestonewebapp_war_exploded/edit_milestone.jsp?projectTitle=demo%2...

HOME LOGOUT

Edit Milestone(s)

select a milestone to edit

( ) new demo milestone

(●) milestone 2 demo

enter your changes below

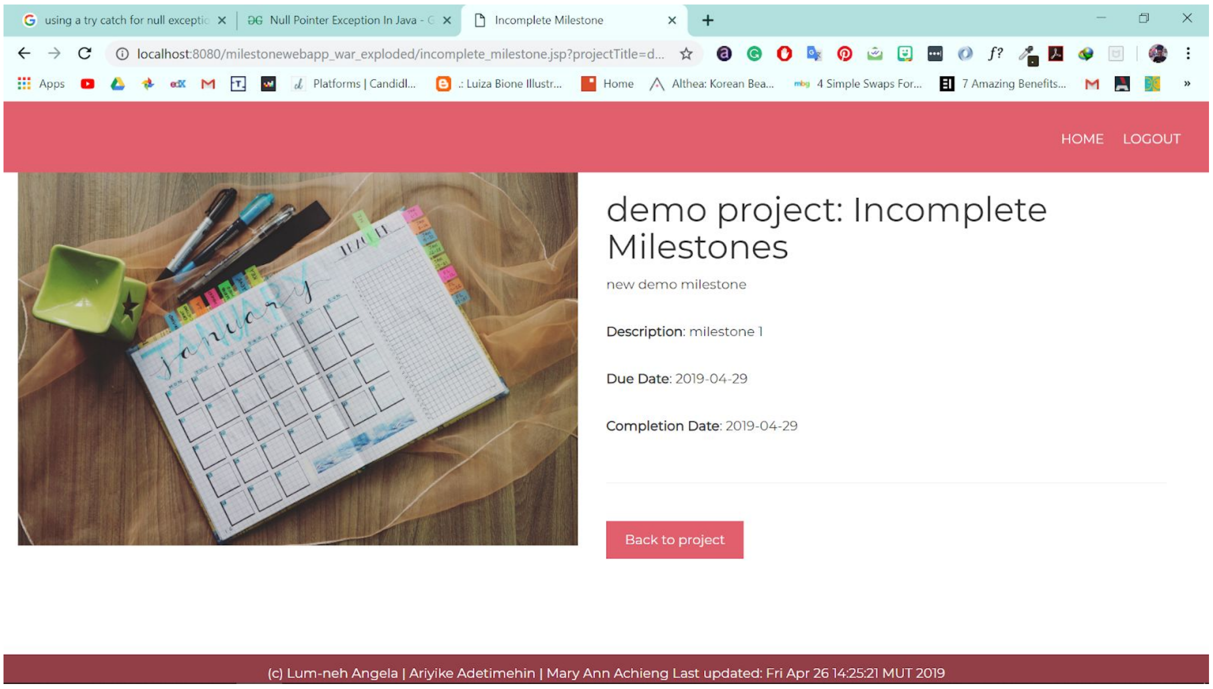
add actual completion date 23/04/2019

OK

Back to project

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:23:46 MUT 2019

Figure L: *Editing a new milestone with its completion date*


using a try catch for null exceptio
Null Pointer Exception In Java
Incomplete Milestone
localhost:8080/milestonewebapp_war_exploded/incomplete_milestone.jsp?projectTitle=d...
Apps
Platforms | Candidl...
.: Luiza Bione Illustr...
Home
Althea: Korean Bea...
4 Simple Swaps For...
7 Amazing Benefits...
HOME
LOGOUT
Back to project

demo project: Incomplete Milestones

new demo milestone

Description: milestone 1

Due Date: 2019-04-29

Completion Date: 2019-04-29

(c) Lum-neh Angela | Ariyike Adetimehin | Mary Ann Achieng Last updated: Fri Apr 26 14:25:21 MUT 2019

Figure M: *Listing incomplete milestones*