Web Developer 2 Coursework 2

Group Report

|  |  |
| --- | --- |
| Group Letter: | F |
| Project Title: | Coursework 2 |
| Date: | 29/4/2019 |
| Group Members: | Basharat Afzal |
|  | Zarko Ivanov |
|  | Merin Haslacher |
|  | Mark Cottrell |
| Repo: | <https://github.com/markonline37/webdev2> |

I declare that all work submitted for this coursework is the work of Basharat Afzal, Zarko Ivanov, Merin Haslacher and Mark Cottrell alone unless stated otherwise.

# Link design within the application

As a tool for building and managing Java-based projects, Maven’s main objectives are to provide a uniform build system with quality project information and guidelines for best practices development for the easy build process while allowing transparent migration to new features (<https://maven.apache.org/what-is-maven.html>). Due to its functionality that provides, Group ‘F’ have decided to use Maven for their ‘Milestoner’ project given in the Web Platform Development 2 course at Glasgow Caledonian University (GCU). This ‘Milestoner’ project main features were:

* A description of the milestone.
* An intended due date.
* The actual completion date.

The features of the website should have provided the following functionality:

* Milestones can be removed from the list.
* Milestones can be edited.
* A listing of al incomplete milestones.
* A milestone (list) can be shared with friends using a link.

For the structure of the project group ‘F’ have created a project in which the Maven dependencies are set, the properties are built, and a file which is used to configure the servlets as a Java classes. The main ideology in the code is that the main method is linking the servlets together and afterwards, the main class operates all directing of directories to servlets.

As the servlets are similarly built, the process of creating a ‘New Project’ will be investigated. The idea behind the servlet is as it follows:

There is a text field in which a user will have to type the name of the new project. When the user has typed the name and pressed submit, the information is sent to ‘/newProject’. As the main method is responsible for handling all the links as stated above, it passes it to the servlet responsible for handling ‘/newProject’ which is automatically loaded into the Do Post method. If a user is not logged in, it will be redirect. The code checks if the text input is blank as it was is a non-null field in the MySQL database which may throw up errors if the server attempts to enter the empty text. The functionality across all servlets is similar where forms are submitted to the servlets, main class handles directing to the appropriate servlets and data is processed and inserted/deleted/updated etc as appropriate from the database.

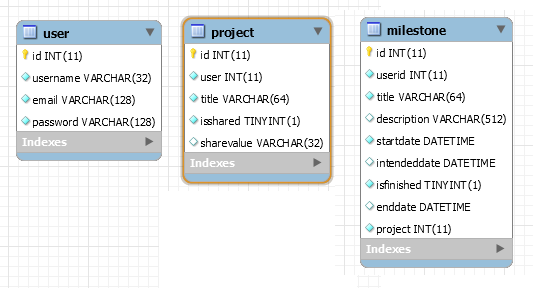
# Persistence

There are a few options when it comes to persistence for a web-application. Theoretically, the server could store all of its data in a text file, but that would be hard to organize, and it would require a persistent structure. An often more desirable thing to do is setting up a database, which is a rather vague term for an organized set of data.

There are two main kinds of databases: relational and non-relational. Relational databases are structured like tables, and they have keys that indicate their unique values and relations to other values in other tables. Non-relational databases aren’t as popular and all databases that don’t adhere to the relational module are typically defined as such. They are also known as NoSQL databases. We chose to use a relational database.

There are many different relational database management systems, but we chose MySQL, a very popular management system that has even spawned popular forks, such as MariaDB. Another option would have been H2, a database management system written in java. Due to our familiarity and experience with MySQL we didn’t chose h2, although it would have been fitting for a java-written web-application.

Using the java.sql package, we could send queries to the database that added, removed, modified, and retrieved entries in the different tables. This is used in parts where a user’s data is to be altered or displayed, such as a removal of milestones or a list of projects. There are also different tools to set up databases, one of which is phpMyAdmin. This is a MySQL database administration tools, which means it allows an administrator to easily view and edit the databases. This was used to set up the tables and the important keys in a manner which made sense. The database tables can be viewed below:



The data access layer gives simplified access to this database. In this case, it consists of a java class called DB in the milestone.util package. This class uses the java.sql package mentioned before to ease interactions with the database. A connection method is defined to first establish a connection. This does not require parameters, as the host, port, and name are defined as static variables separately. Naturally, a disconnect method accompanies this. A query method is defined which allows the programmer to put in any SQL statement that will be processed by the database. This does not contain a return statement, as it is only for altering the database, so it would be used for inserting, delete, and update queries. There is also a method for updating the database and different methods for each type of data the database can return. In other words, a query that requires a single string would call a different function than one that requires a set of data. This improves the stability of the program, as it decreases the chance of side-effects and unexpected return types. The methods with return types return string, array, Boolean, and ResultSet. The array returns multiple single values, while the ResultSet represents multiple rows of the database.

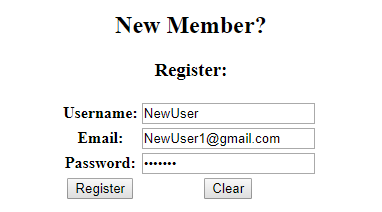
# Functionality and Testing

## Register

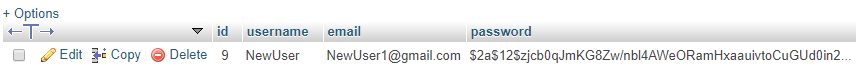
Users can register on the system;

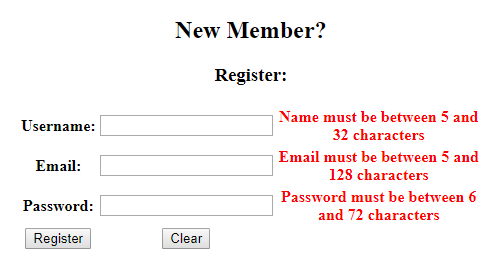
Emails must be unique, and all fields must be certain (different) lengths.

Emails are checked to ensure it is unique and passwords, using BCrypt encryption, are hashed when stored in the database.

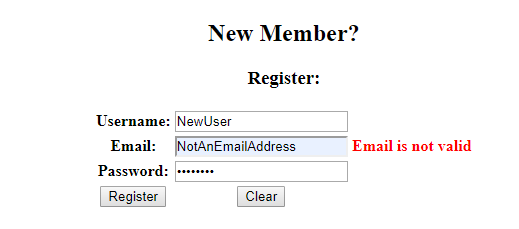


New User viewed in PhpMyAdmin:



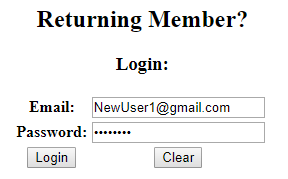


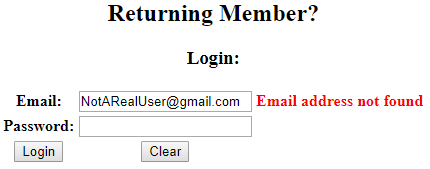
Additionally, emails are checked to make sure they are in appropriate format

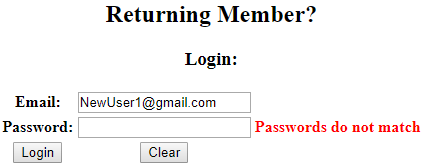


## Login

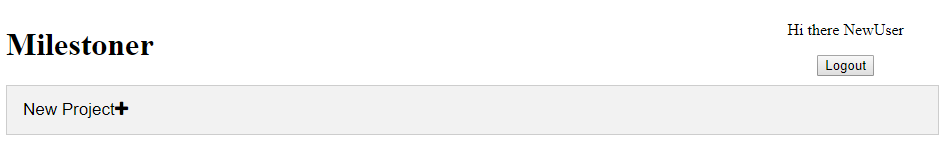
A user can login using email and password, email is looked up and password retrieved; passwords are compared to stored password to login.







Users are redirected to the project view once logged in (or when they have registered).



Logout

User’s can logout of the application at any time (when logged in) which will automatically redirect them to the login page which functions as the website’s main page.

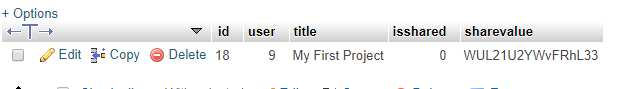
## New Project

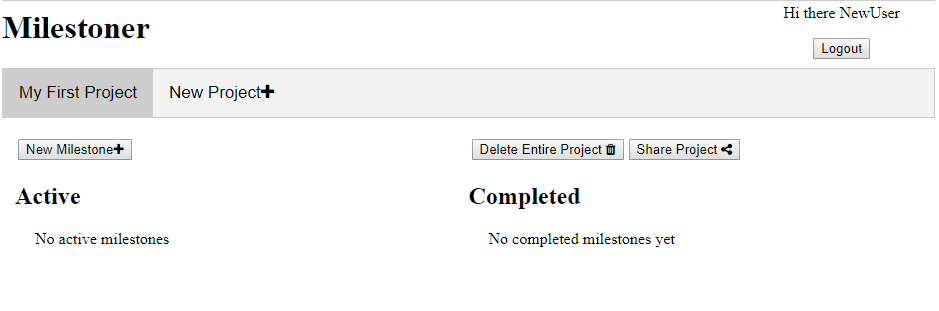
Users can create project’s using the new project button:



Which displays a form they can enter a project name into and a button to submit. JavaScript is used to return validation before submission to ensure the project title is of a certain length.



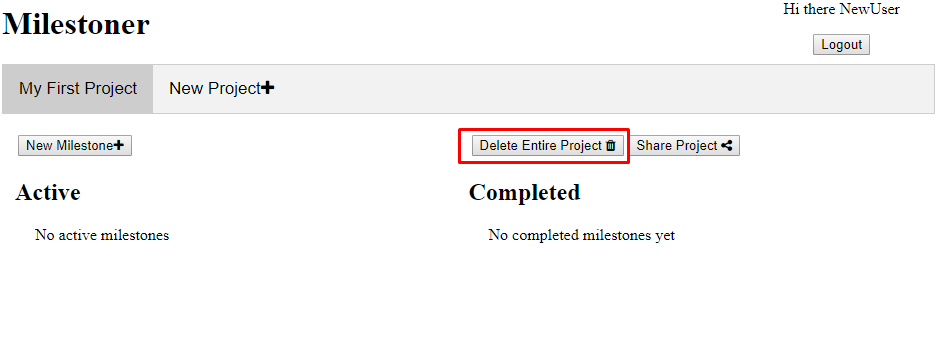


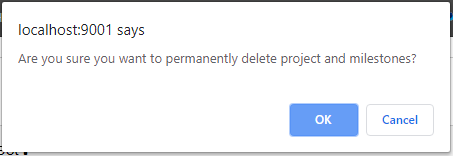


The new project is now created, and users can begin to add milestones, share or delete the project (and associated milestones)

## Delete Project

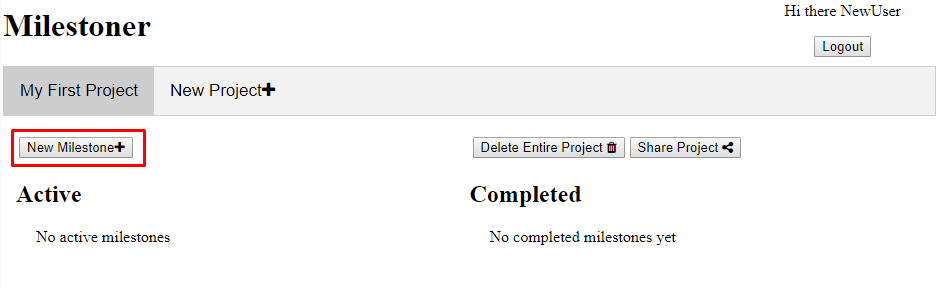
Any project can be deleted by pressing the delete project button, since this is a major action which will also delete the milestones and cannot be reversed, users are prompted to confirm deletion.

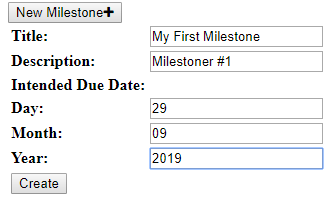




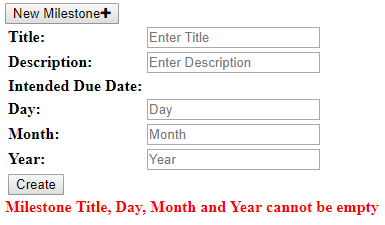
# New Milestone

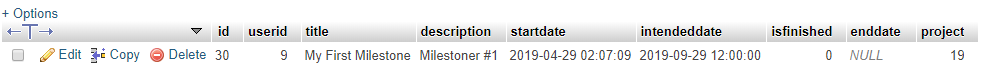
Users can create new milestones by pressing the new milestone button

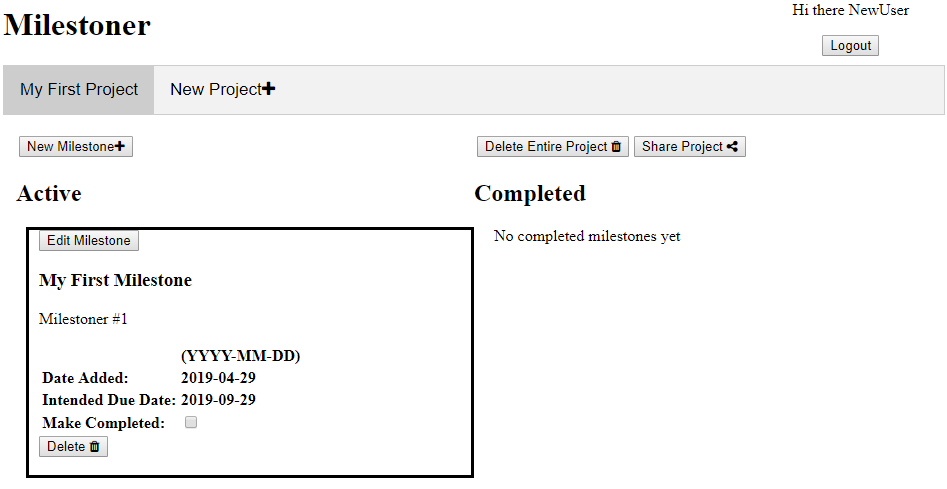




Since all values are required, except description, a JavaScript function is used to ensure the form is valid before submission.

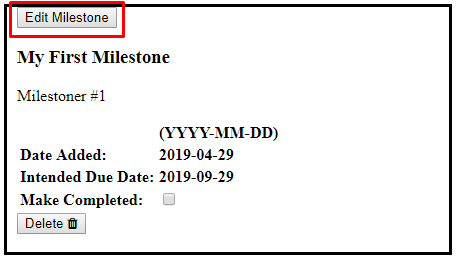




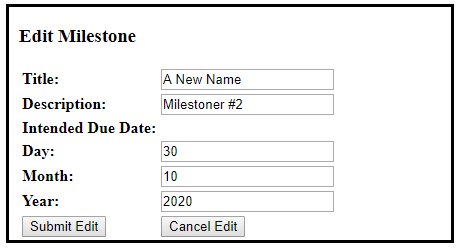


## Edit Milestone

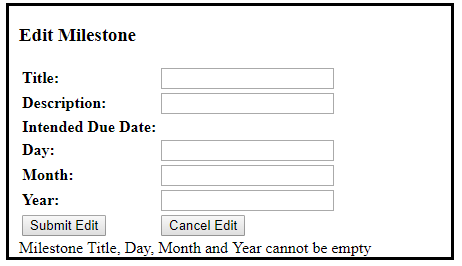
Users can edit a milestone to change the title, description or due data.



Edit form:

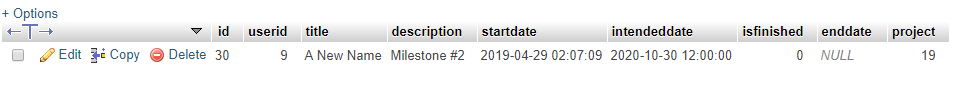


Error with submission:

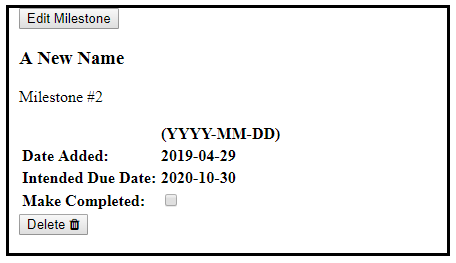


Which are again checked to make sure that the fields are correct before submitting to the database.

Updated Database:

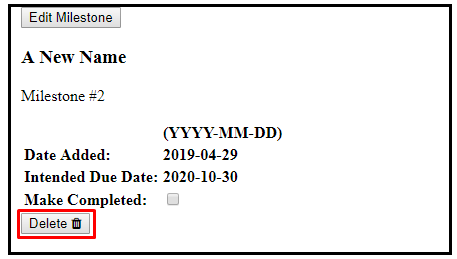


Updated milestone:

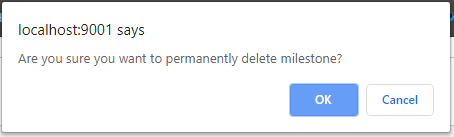


## Delete Milestone

Users can delete milestones which removes them from the database.

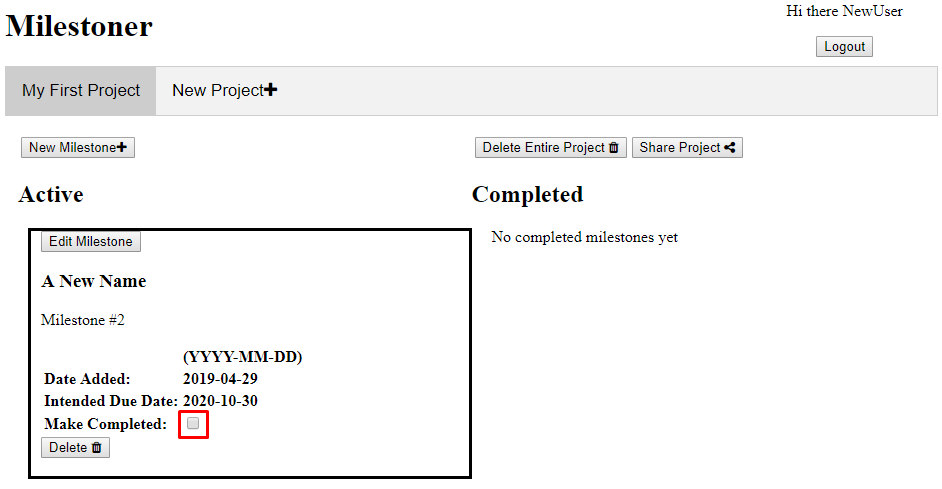


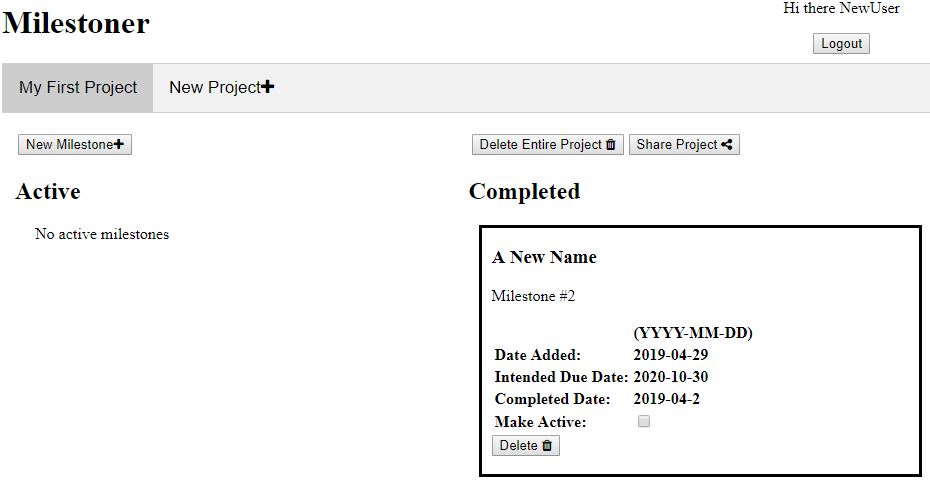
This requires confirmation:



## Mark Milestone as completed

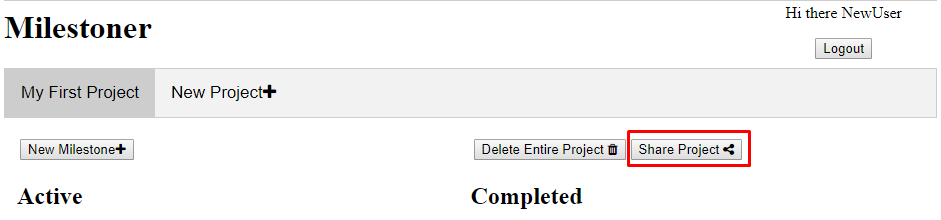
User can mark milestones as completed.



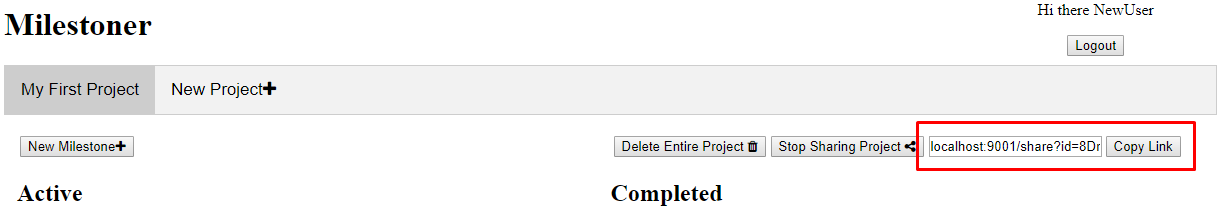


## Share Project

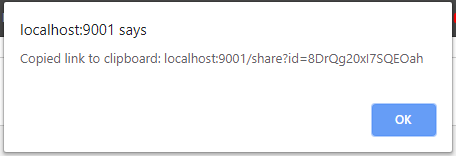
Users can enable sharing for a project, this enables sharing in the database and provides a link that can be copied manually or press the copy to clipboard button which copies the link to clipboard.



Once sharing is enabled:



When a user copies the link, an alert pop’s up to let them know it has copied successfully:

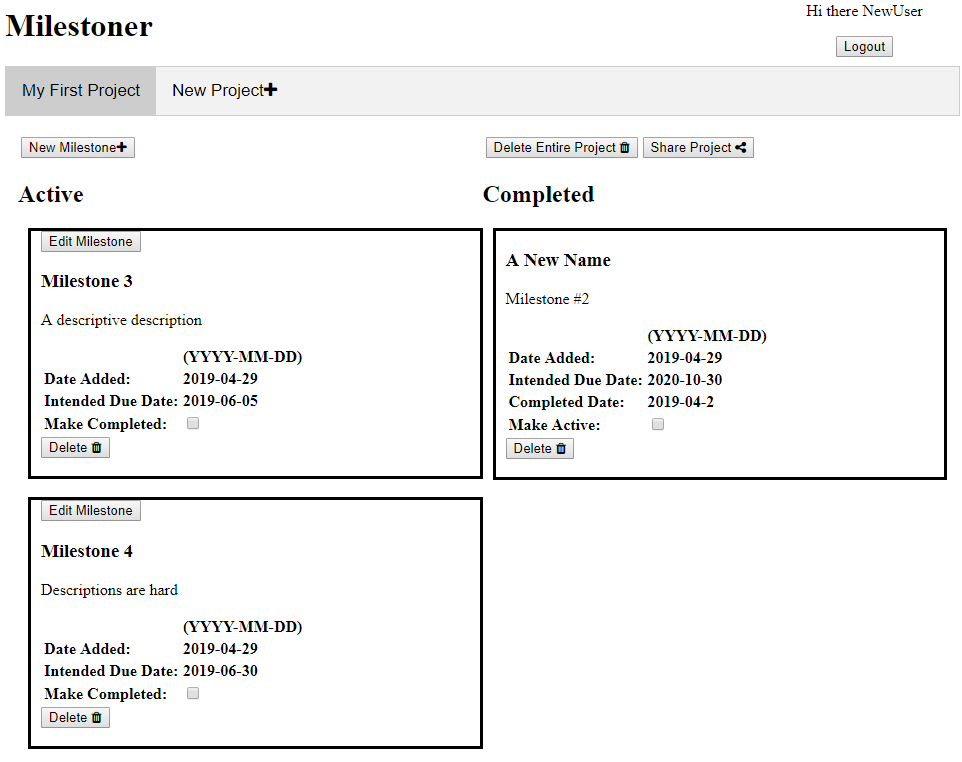


## View Shared Project

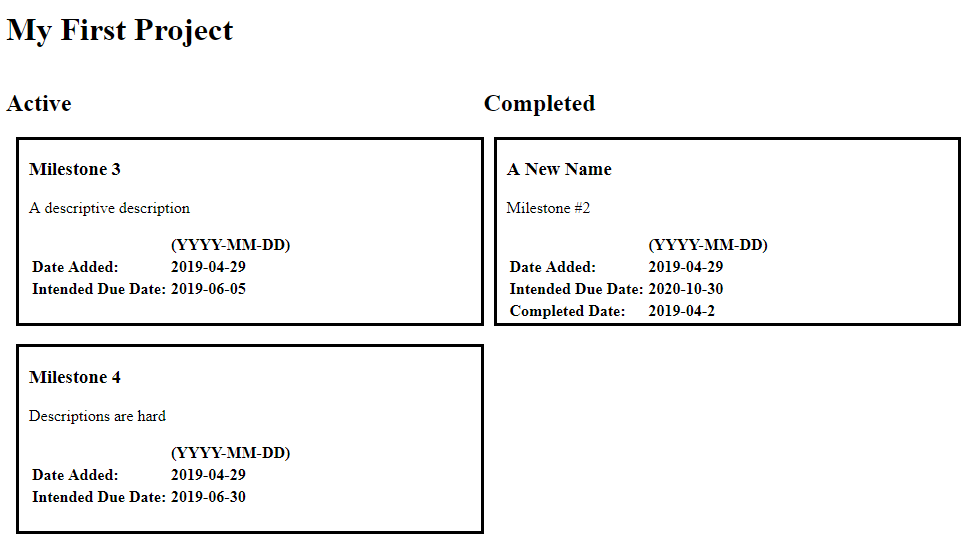
Anyone can view a shared project with the appropriate link.

The view is different to a logged in user as a user viewing the shared link can only see the milestones and project.

A user’s project view:



View of a shared project:



## User Authorisation Protection

All pages apart from the view shared page automatically redirect a user to the login page (which is also the homepage). Additionally, all SQL commands where appropriate are bound to the user set in the session to prevent user 1 from being able to modify user 2’s project/milestones etc.

# Application Security

Overall our site features some very simple security measures which in a perfect world without any malicious attackers would be enough. However, there are many kinds of malicious attacks and general oversites that can happen to our site such that it isn't fully protected.

Our site has a feature where if you have accessed it somehow without logging in it sends you back to the login page in order to make sure than people without an account can't access the site which is incredibly important for security as it’s the first line of defence against any unauthorised entries.

Another feature our site has is that certain SQL statements are bound to the user this means that every user can only update their own milestones they can't actually interact with milestones that they don’t own which helps keep each users data a little safer.

When users register with the system their password is both hashed and salted to provide extra security when a user tries to log back in there password entry is hashed and compared to the passwords in the data base if a match is found the user will be granted access if no the user will be denied access and get another chance to input the right password.

While our site does feature some security measures below, we have listed some more security measures that we were unfortunately unable to implement owing to time constraints.

Having some form of anti-spam system that limits the amount of sessions that connect to the website will help against a denial of service attack (DDOS). Another possible way to help prevent DDOS attacks is to increase the amount of bandwidth on the site in order to handle the spikes in internet traffic. Another way to prevent DDOS attacks would be to use a DDOS protection service such as Cloudflare which routes your websites internet traffic into their network which enables them to block any kind of attacks or threats.

Another threat to the website is a Cross Site Scripting Attack (XXS) in which the user is sent a link by the attacker which takes them to a web page that includes malicious code when the user enters that page they are sent malicious scripts through the page that the attacker designs to do various malicious acts. XSS attacks can also be used to damage a website by using injected scripts they can be used to change the content on the site or even perhaps redirect the user towards a site filled with malicious code. One of the ways that our site could prevent an XSS attack is by asking for the user password or username before they access certain features in our websites case, we could have had a password requirement for users that are using a share link to view a user's milestone list this would make sure that only the intended users will be able to access the website in case the link was accidently mishandled.

In order to prevent session hijacking the session key needs to both encrypted and randomised and session keys need to be dropped on logout to make sure that the user doesn’t use the same key repeatedly this can also help in preventing XSS attacks.

Using prepared statements to stop SQL injection attacks this is done because the query and the data are split and sent to the server separately SQL injection only works when it can mix the code and the data.

Some more of the basic ways of keeping security up to date are keeping all software up to date, keeping the server the website is running on up to date and making sure to test pre-existing security options thoroughly.