The documents summarized the procedures and technology used in building a Bug Tracker Application. Note that in Login ID is admin and password is admin.

### A Study User Requirements

Followings are the user requirements:

Your goal is to implement a simple bug tracker. The features we would like, in no particular order:

● It should be possible to view the list of open bugs, including their titles

● It should be possible to view the detail of individual bugs, including the title the full

description, and when it was opened

● It should be possible to create bugs

● It should be possible to close bugs

● It should be possible to assign bugs to people

● It should be possible to add people to the system

● It should be possible to change people’s names

● The web application should look nice

● The web application should expose some sort of API

● The data should be stored in some sort of database

According to the user requirements, I understood that it should use a technology that could connect to database to store information, and present them in nice, reliable, and readable way.

### B Design Application

Because it has concrete requirements and a defined schedule for this Bug Tracker Application, I decided to use a simple Software Development cycle – Water Fall model to make this application happen. But if the requirements and schedule are not defined clearly, I can also agile methodology to proceed the project

According to the requirements, I designed the Bug Tracker Application with :

* Use Microsoft C# MVC framework
* Use MS SQL server to store the information
* Have a simple authorization to check user login and password
* Have a nice user-friendly interface for users to input bugs/issues and related information
* Communication with database and server are using Web API which is default in MVC framework

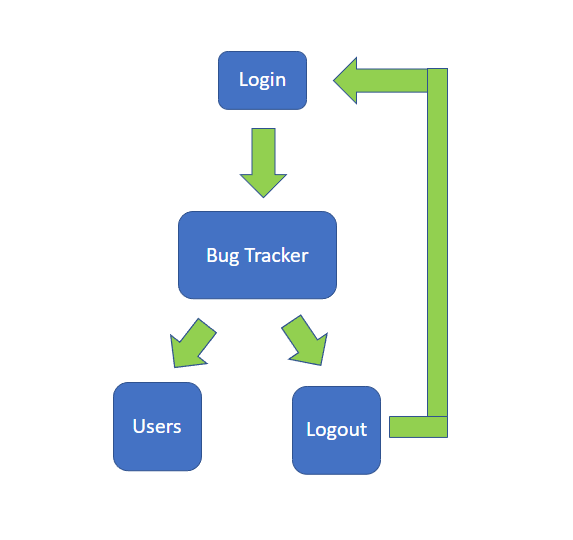
### C Schedule Tasks

The schedule and resource of the project is defined as

|  |  |  |
| --- | --- | --- |
| **Items** | **Schedule (days)** | **Responsible** |
| Requirements studying | 0.3 | Kevin |
| Design | 0.7 | Kevin |
| Development | 1 | Kevin |
| Testing | 0.5 | Kevin |
| Documentation | 0.5 | Kevin |

### D Functional design

**a Flow of Functions and Layout design**

****

|  |  |
| --- | --- |
| **Login** | **Graphical user interface, application  Description automatically generated** |
| **Bug Tracker** |  |
| **Users** |  |

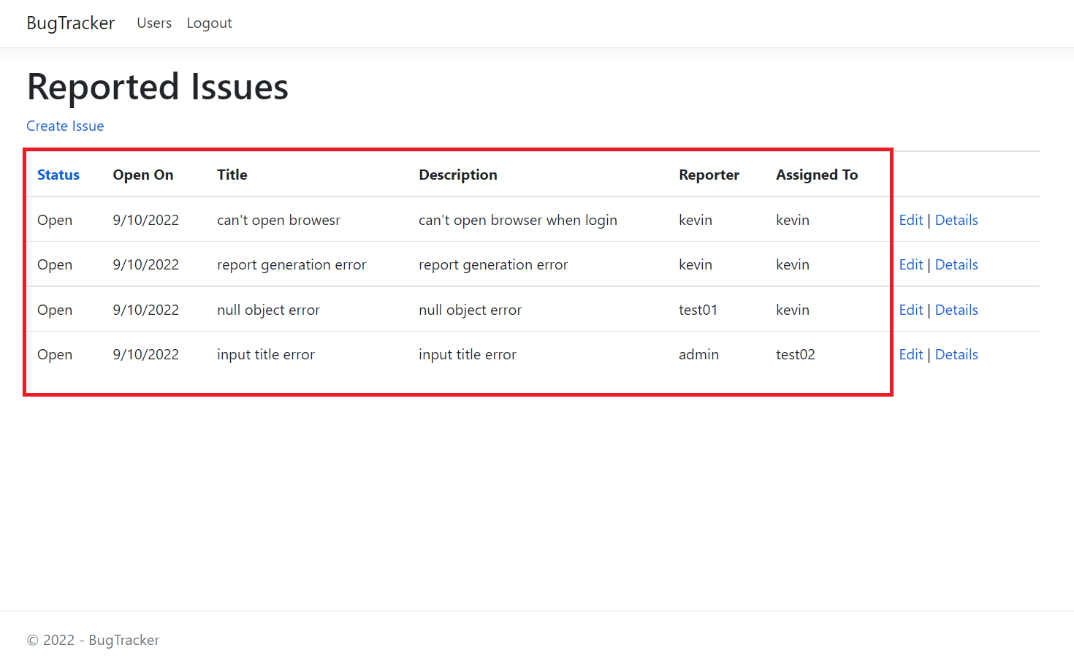
**b Bug Tracker Functions**

Basic information of a Bug Tracker record

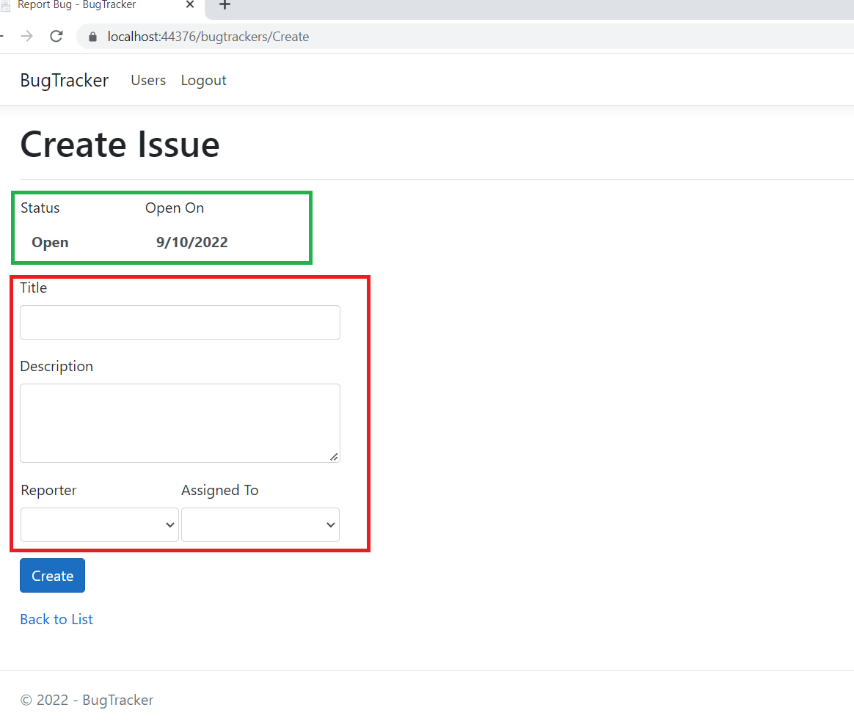
* Status – Open, Close
* Open Date – date of open of Bug/Issue that is not editable
* Title – Title of Bug/Issue with length less than 255 characters
* Description – Description of Bug/Issue with no limit in length
* Reporter – Person who reports the Bug/Issue with length less than 30 characters
* Assigned To – Person who will follow the Bug/Issue with length less than 30 characters

Basic Action

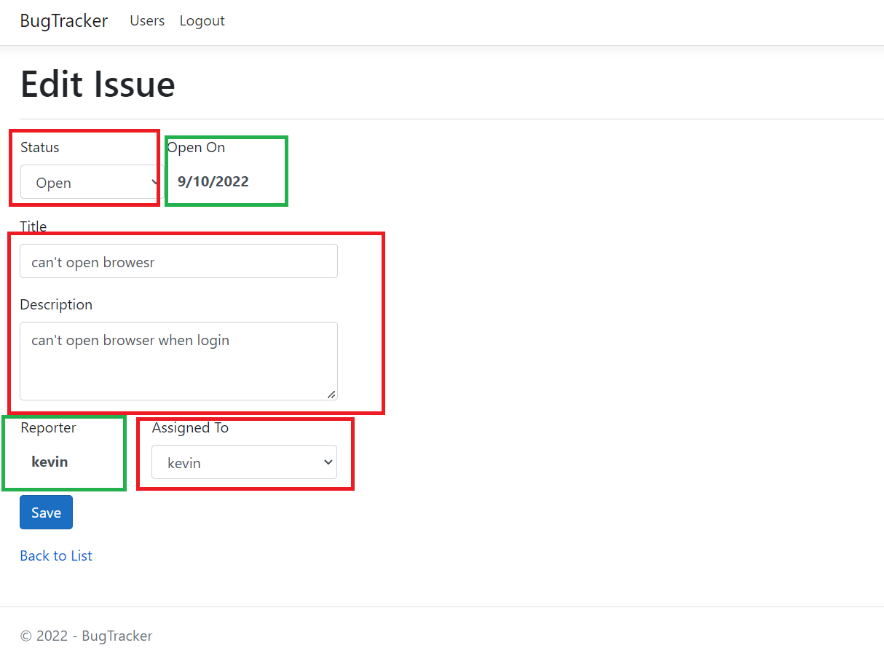
* List of records – shows Status, Open date, Title, Description, Reporter and Assigned To



* Creation – allows input Status, Title, Description, Reporter and Assigned To, but not Open Date



* Edition – allows edit Status, Title, Description and Assigned To, but not Open Date and Reporter



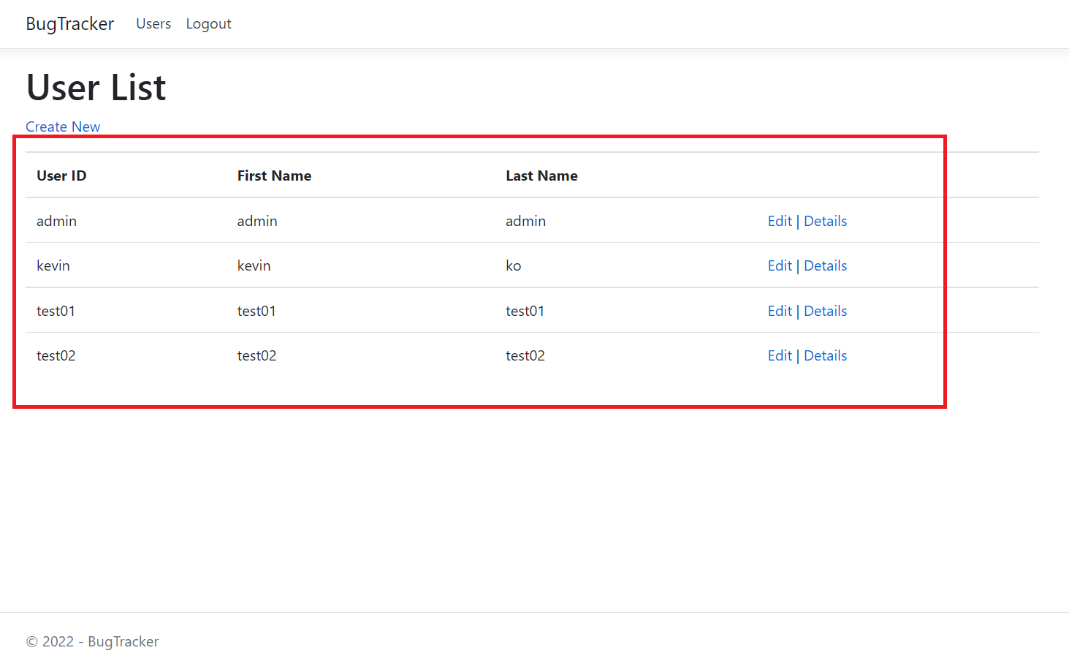
**c User Functions**

Basic information of a User record

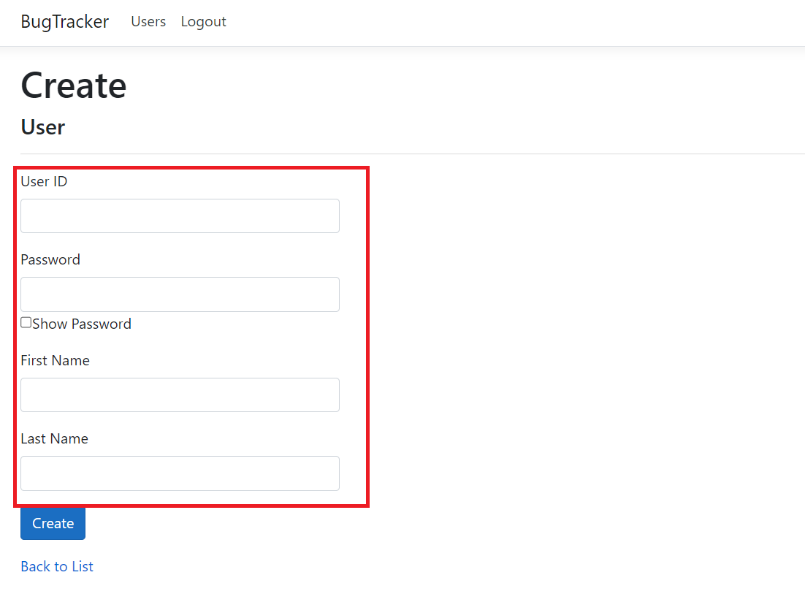
* User ID – for user to login with length between 5 and 30
* Password – Password of user with length between 5 and 30
* First Name – First Name of User with length less than 50
* Last Name – Last Name of User with length less than 50

Basic Action

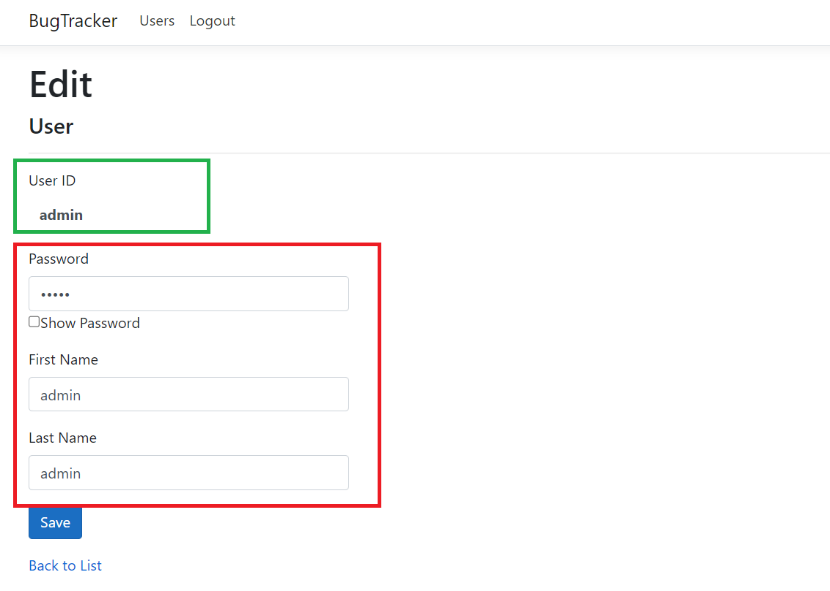
* List of records – shows User ID, First Name and Last Name but not Password



* Creation – allows input User ID, Password, First Name and Last Name



* Edition – allows edit Password, First Name and Last Name, but not User ID



### E Setup Environment for development

For setup Database

1. Open MS SQL server, create database called BugTracker
2. Run script file – “setup database.sql”

For setup environment to develop application

* Microsoft Visual Studio 2019 or higher
* .net Core 2.1 or higher
* Microsoft SQL server Express

### F Functionalities and Performance testing

For Functionalities, there are 3 main testing – unit test and integration testing, UX/UI testing

However, for unit test, it was not success to start the unit for some reasons.

For integration test and UX/UI, I do it manual. Testing included validation of values and Creation/Edition of Bugs and Users

For performance testing, it is very important because when data of application becomes bigger and bigger, performance will be degraded, and it needs some tuning to database and also the program logic. Usually, programmer ignores performance testing because it needs tons of data and takes hours of time to do. However, without it, the application run slower and slower, would finally become a great problem.

### G Good Practice

I would suggest some guidelines:

* Define variables in descriptive manner
* Comment codes at appropriate lines are necessary for programmers or supporters to better understanding and readable.
* Refactoring makes codes clean and reusable which is good for maintenance and speeding up development for future.
* Code Review makes sure codes follow standard
* Uses code analysis software

### H Improvement

Some improvements can be done

* Authorization
* Security
* Data validation
* Layout
* Mobile friendly

### I Conclusion

This project implemented successfully in terms of functionalities and Usability. Users are happy to use the application to report issues or bugs that are found in the applications or software they are using. Reporting is seamless and user-friendly due to the technology. Supporters can review the reported issues and prioritize the resolution. At the end, the project can help to improve the productivity of software development and maintenance.