**.NET Application Programming**

**Project Status and Design Report**

|  |  |  |
| --- | --- | --- |
| **Topic:** | *Login and Registration Pages* | |
| **Date:** | *1/21/18* | |
| **Revision:** | *1.0.0* | |
| **Team:** | 1. *Ali Cooper* | |
| 1. *Caleb Ljunggren* | |
|  | |
|  | |
| **Weekly Team Status Summary:** | |  |  |  |  | | --- | --- | --- | --- | | **User Story** | **Team**  **Member** | **Hours**  **Worked** | **Hours Remaining** | | *As a user, I need to register so that I can have an account* | *Ali Cooper* | *0* | *2* | | *As a user, I need to be able to login so that I can play minesweeper* | *Ali Cooper* | *0* | *2* | | As a dev, I need design documentation to properly implement project code | *Ali Cooper* | *4* | *4* | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | | |
| **GIT URL:** | https://drowsyWarble@bitbucket.org/drowsyWarble/cst247.git | |
| **Peer Review:** | *Y* | We acknowledge that our team has reviewed this Report and we agree to the approach we are all taking. |

**Planning Documentation**

**Agile Scrum Product Backlog:**

https://bitbucket.org/drowsyWarble/cst247/raw/dc3c7d23605eb84fd2360341b16aed81f09fa4b5/Documentation/Scrum/ProductLog.xlsx

**Agile Scrum Sprint Backlog:**

https://bitbucket.org/drowsyWarble/cst247/raw/dc3c7d23605eb84fd2360341b16aed81f09fa4b5/Documentation/Scrum/Sprint1Backlog.xlsx

**Agile Scrum Burn Down Chart:**

https://bitbucket.org/drowsyWarble/cst247/raw/dc3c7d23605eb84fd2360341b16aed81f09fa4b5/Documentation/Scrum/Sprint2Backlog.xlsx

**Agile Retrospective Results:**

*The following table should be completed after each Retrospective on Things That Went Well (Keep Doing). An alternative to the following table is to use a Mind Mapping tool such as Coggle. If you use a Mind Mapping tool you must include a URL or Image File.*

|  |
| --- |
| **What Went Well** |
| No problems with git |
| Communication |
| Adding Login and Registration functionality |
| Adding DB functionality |

*The following table should be completed after each Retrospective on Things That Didn’t Go Well (Stop Doing) and What Would Be Done Differently Next Time with an Action Plan to Improve (Try Doing and Continuous Improvement). An alternative to the following table is to use a Mind Mapping tool such as Coggle. If you use a Mind Mapping tool you must include a URL or Image File.*

|  |  |  |
| --- | --- | --- |
| **What Did Not Go Well** | **Action Plan** | **Due Date** |
| Would sometimes start at the wrong location | Read VS Documentation | 1/29/18 |
| When adding a Model to a view, it defaulted the password property to the text input field | Use PasswordFor, for input | **1/28/18** |
|  |  |  |

**Design Documentation**

**Install Instructions:**

*Step by step instructions for setting up your database, configuring, and deploying/installing your application. This section should also include detailed instructions for what configuration files are required by your application, what configuration settings need to be adjusted for various runtime (development or production) environments, and where the files need to be deployed to. This section should also contain detailed instructions for how to clone your application source code from BitBucket and deploy the application to an externally hosted site.*

**General Technical Approach:**

*You should, in words, describe your approach and design here. You should also summarize any meeting notes, brainstorming sessions, etc. that you want to retain thru the design of your project.*

**Key Technical Design Decisions:**

Board State will be serialized and stored into the database

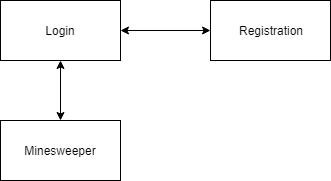
Will use recursive DFS algorithm for 0 tile discovery.

**ER Diagram:**

**DDL Scripts:**

*This should contain a link to BitBucket where the DDL script can be downloaded from.*

**Sitemap Diagram:**



**Security Design:**

*This section should outline the design for how authentication and authorization was supported. This section should also contain all of the roles and privileges that are supported by the design.*

With the login and registration pages, there was a need to make the pages secure enough to prevent users from taking it apart or breaking it. The first part we did to prevent unwanted viewings of data was to use the POST submission type. This prevents users from viewing the posted material (which is why GET was not used for handling the registration and login information). Authentication for the registration process was done in two steps. The first was to verify the credentials given by the user, so that they do not contain invalid fields. Invalid fields for example could be when a user does not enter a .com or similar URL ending to the email, which would in turn render the email they entered useless to the program. Based off the specific field, the program will check the input to see if it passes the specified conditions (length being the most common parameter check). If the input for the given field passes every authentication check, the program moves on to the next field until all fields have been authenticated. Next is the second step, which takes the username given by the user and checks the entire database for any matches. This prevents multiple users from registering under the same username, which would cause many problems further down the road if this was allowed to happen.

**Third Part Interface Design:**

*This section should fully document any Third Party Service Interface API’s, how to access the service, what parameters are required by the API, and the detailed JSON data format specification that could be used by a third party developer to integrate with the service and API.*

**Flow Charts:**

https://bitbucket.org/drowsyWarble/cst247/raw/53cd0e234da335b043aeb0381f2e6f717e2b26e5/Documentation/Flow%20Charts/MinesweeperDFS.pdf

**User Interface Diagrams:**

<https://bitbucket.org/drowsyWarble/cst247/raw/dc3c7d23605eb84fd2360341b16aed81f09fa4b5/Documentation/StoryBoard/StoryBoard.pdf>

**Class Diagrams:**

*You should insert any class diagrams here. Your class diagrams should be drawn correctly with the three appropriate class compartments, + and – minus to indicate accessibility, and the data types for the state/properties as well as method arguments and return types. If you have no supporting documentation please explain the rational why you are able to leave this section as N/A.*

**Pseudo Code:**

*You should provide BitBucket URL references to any code stubs & pseudo code. If you have no supporting documentation please explain the rational why you are able to leave this section as N/A.*

**Other Documentation:**

*You should insert any additional drawings, storyboards, white board pictures, project schedules, tasks lists, etc. that support your approach, design, and project. If you have no supporting documentation please explain the rational why you are able to leave this section as N/A.*