Milestone 3

CST - 247

February 11th, 2018

Connor Low && Mick Torres

**.NET Application Programming**

**Project Status and Design Report**

|  |  |  |
| --- | --- | --- |
| **Topic:** | Milestone 3 | |
| **Date:** | February 11th, 2018 | |
| **Revision:** | 3.3 | |
| **Team:** | Connor Low | |
| Mick Torres | |
|  | |
|  | |
| **Weekly Team Status Summary:** | |  |  |  |  | | --- | --- | --- | --- | | **User Story** | **Team Member** | **Hours Worked** | **Hours Remaining** | |  |  |  |  | | Updated Burndown, Product Log, and Backlog | mick | *2* |  | | Pseudo code, Burndown, wireframes, flowchart | Connor | 2.5 |  | | Updated views, (new layout to match wireframe, pages) | mick | *5* |  | | Merge game engine / existing code | *both* | *3* |  | | Class diagrams for game engine | Connor | 1 |  | | Fixed bugs | Connor | 2 |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | | |
| **GIT URL:** | To access this resource, go to Student Success Center site and search for “GIT.” LINK: https://github.com/ConnorAndMickDoingCode/Charp | |
| **Loom video:** | https://www.useloom.com/share/76675413e01b4086b401301d0527701f | |
| **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://github.com/ConnorAndMickDoingCode/Charp/tree/master/SCRUM

**Agile Scrum Sprint Backlog:**

https://github.com/ConnorAndMickDoingCode/Charp/tree/master/SCRUM

**Agile Scrum Burn Down Chart:**

https://github.com/ConnorAndMickDoingCode/Charp/tree/master/SCRUM

**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** |
| * Connection to DB was quick / easy since done in class |
| * Creation of models, views, and controllers |
|  |

*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** |
| * Creation of functional pages / incorporation of Razor | * Brush up on Razor tutorials, read template guides |  |
| * **Pushing projects back and forth over the repository causes errors when the other party tries to load the solution** | * **Unknown at this time** * **Will ask Reha in class regarding this** |  |
|  |  |  |

**Design Documentation**

**Install Instructions:**

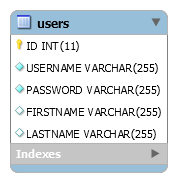
*Step by step instructions for setting up your database, configuring, and deploying/installing your application. This section should also include detailed instructions for what configuration files are required by your application, what configuration settings need to be adjusted for various runtime (development 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:**

**Key Technical Design Decisions:**

*Any final technical design decisions, such as framework decisions, etc., should be documented here. This should list the technology/framework, its purpose in the design, and why it was chosen.*

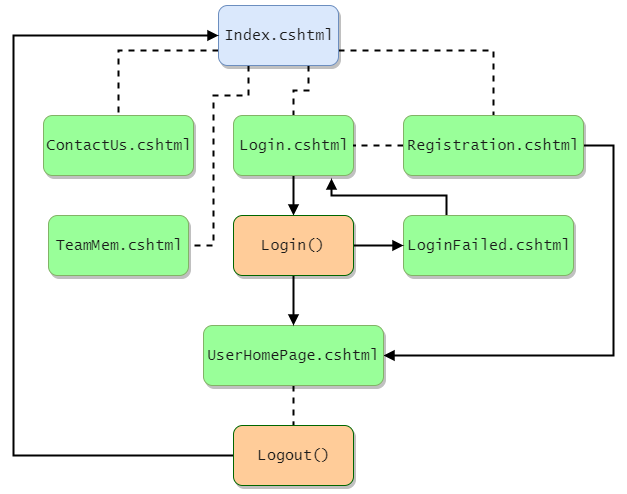
**ER Diagram:**

**

**DDL Scripts:**

https://github.com/ConnorAndMickDoingCode/Charp/tree/master/Documentation/DB%20Design

**Sitemap Diagram:**

**

**Security Design:**

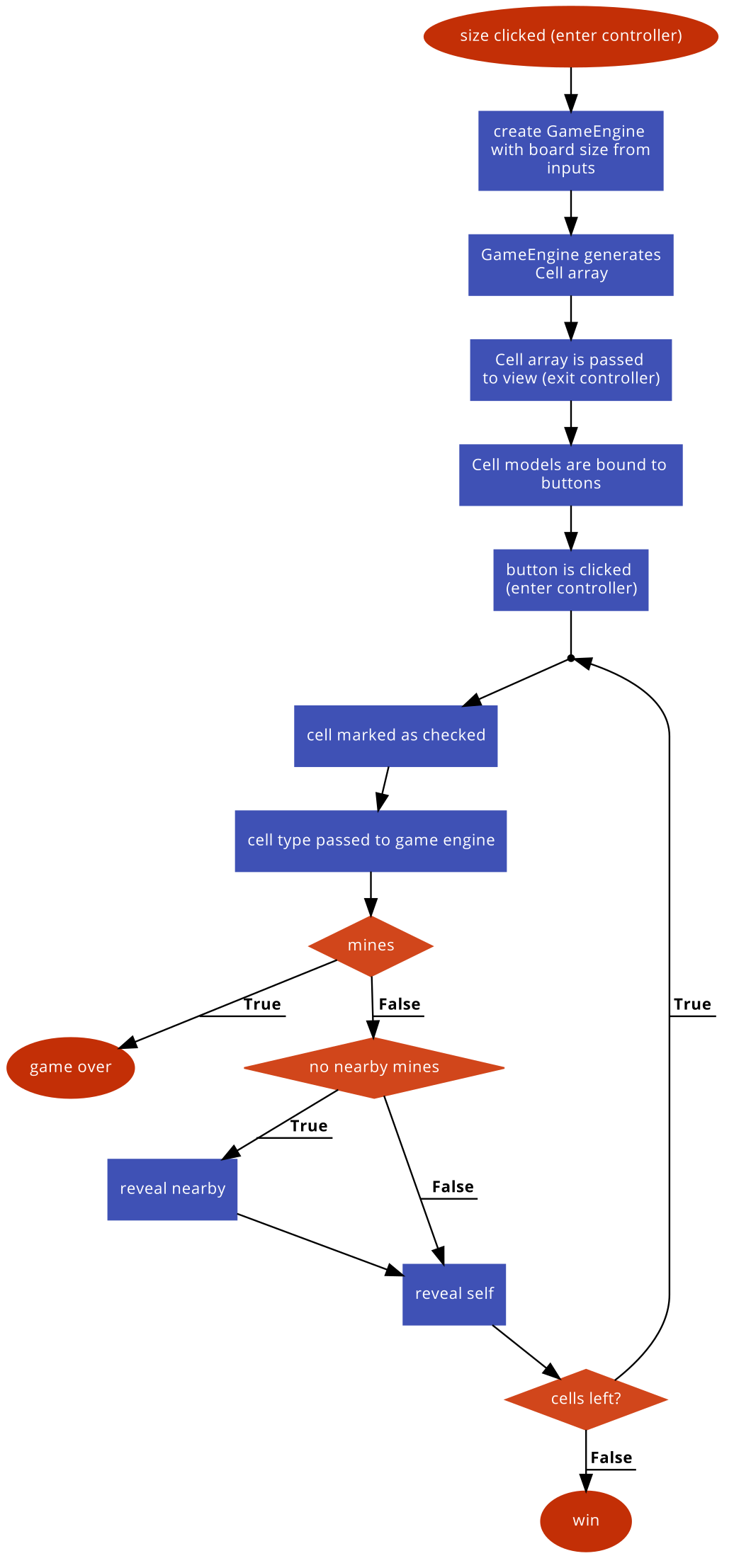
*This section should outline the design for how authentication and authorization was supported. This section should also contain all of the roles and privileges that are supported by the design.*

**Third Party Interface Design:**

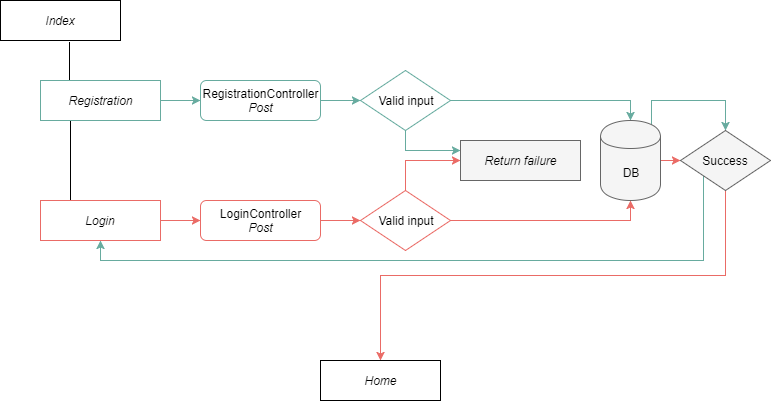
*This section should fully document any Third Party Service Interface API’s, how to access the service, what parameters are required by the API, and the detailed JSON data format specification that could be used by a third party developer to integrate with the service and API.*

**Flow Charts:**

Game-button pressed

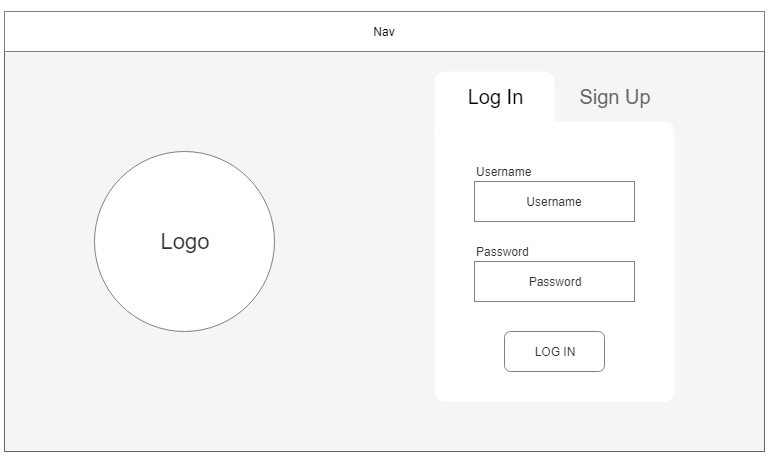


Login

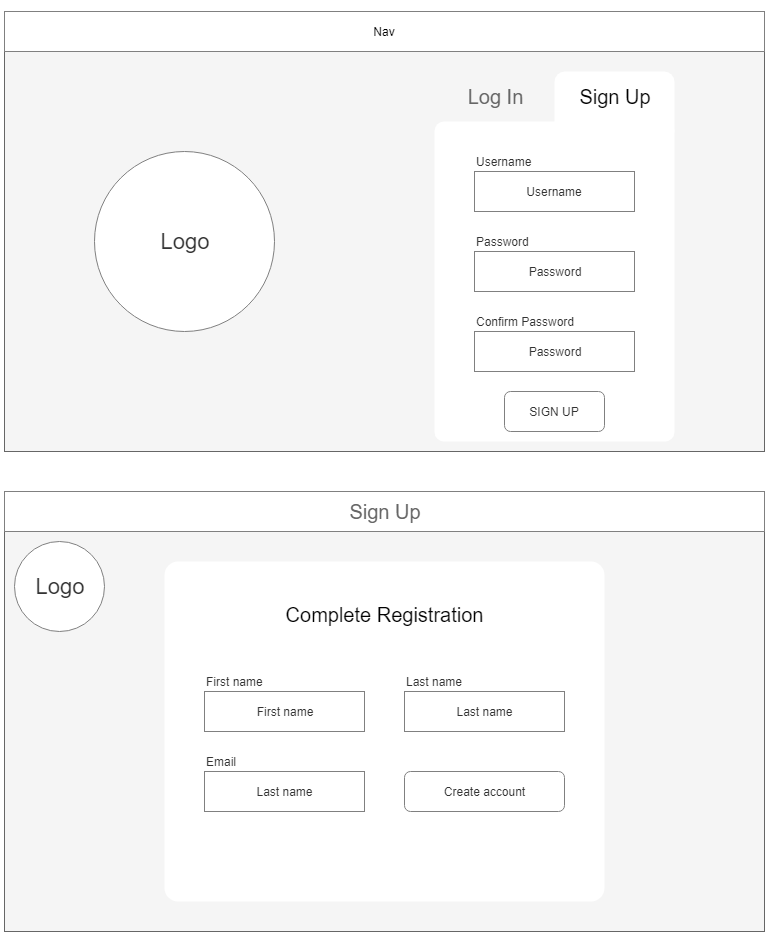
**

**User Interface Diagrams:**

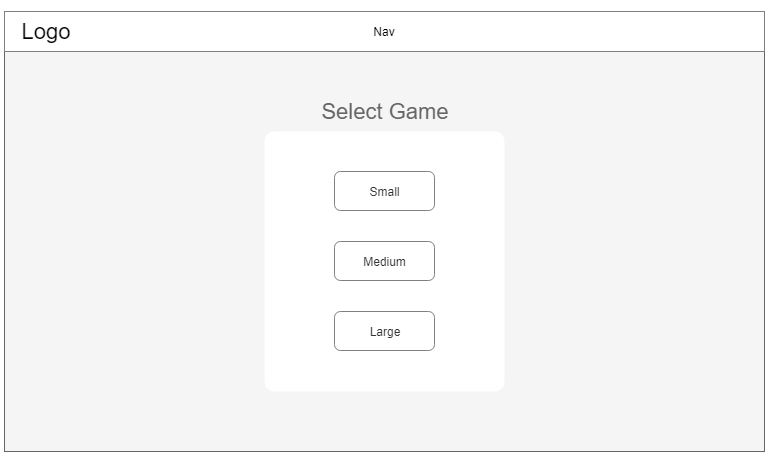
Login



Registration



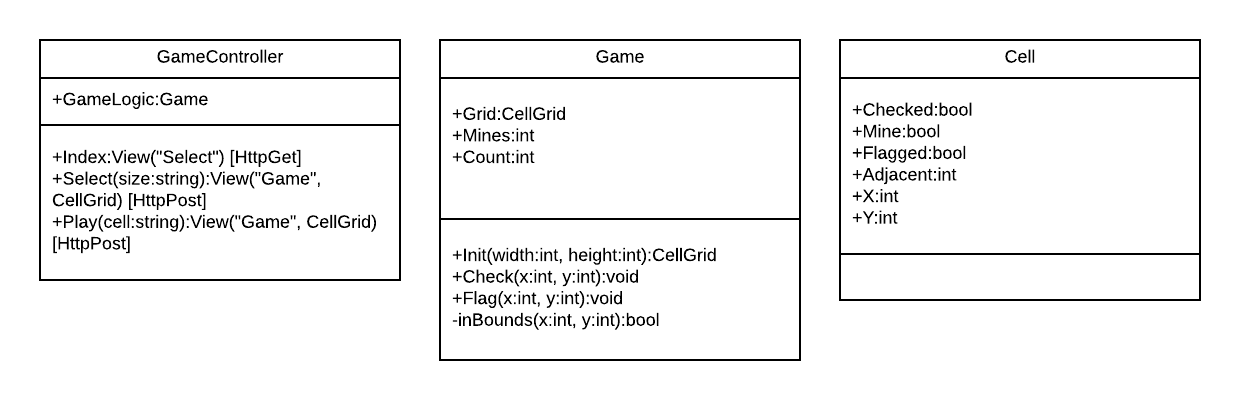
Select Game size

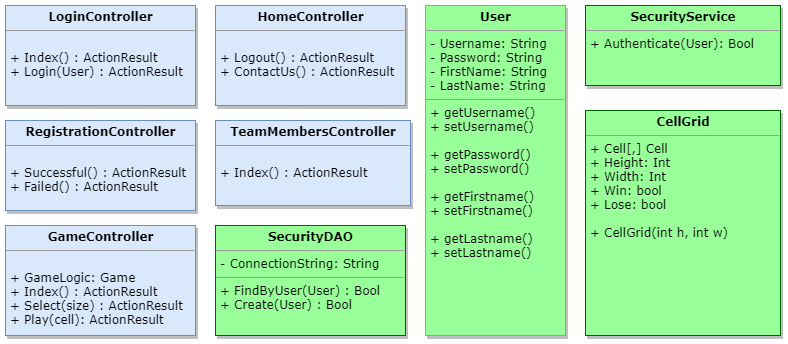


Game play



**Class Diagrams:**



**

**Pseudo Code:**

**Board generation**

* User selects game board size.
* Controller receives input and instantiates a Game Logic object with height and width.
* A 2D array of Cell objects is created.
* The Cells are updated with the correct information.
* The Game Logic and Cells 2D array are passed into the view.
* In a <table> loop through each row:
  + In a <tr> Loop through each column:
    - Create <td>button</td> for each cell
  + Close row
* Close table.

**Update board**

* Cell is clicked.
* Pass array into controller.
* Update checked property.
* Check for bomb.
* Update nearby cells if necessary.
* Pass into view.
* Build table.
  + For checked cells, button replaced with text

**Other Documentation:**

**Issues**

Unsure of how to set up controller/view relationships. RegistrationController cannot access Views from Login directory without an ActionRedirect. Forced to go through the Login page after registration.

Unable to use Viewbag to give notification of successful user-creation.