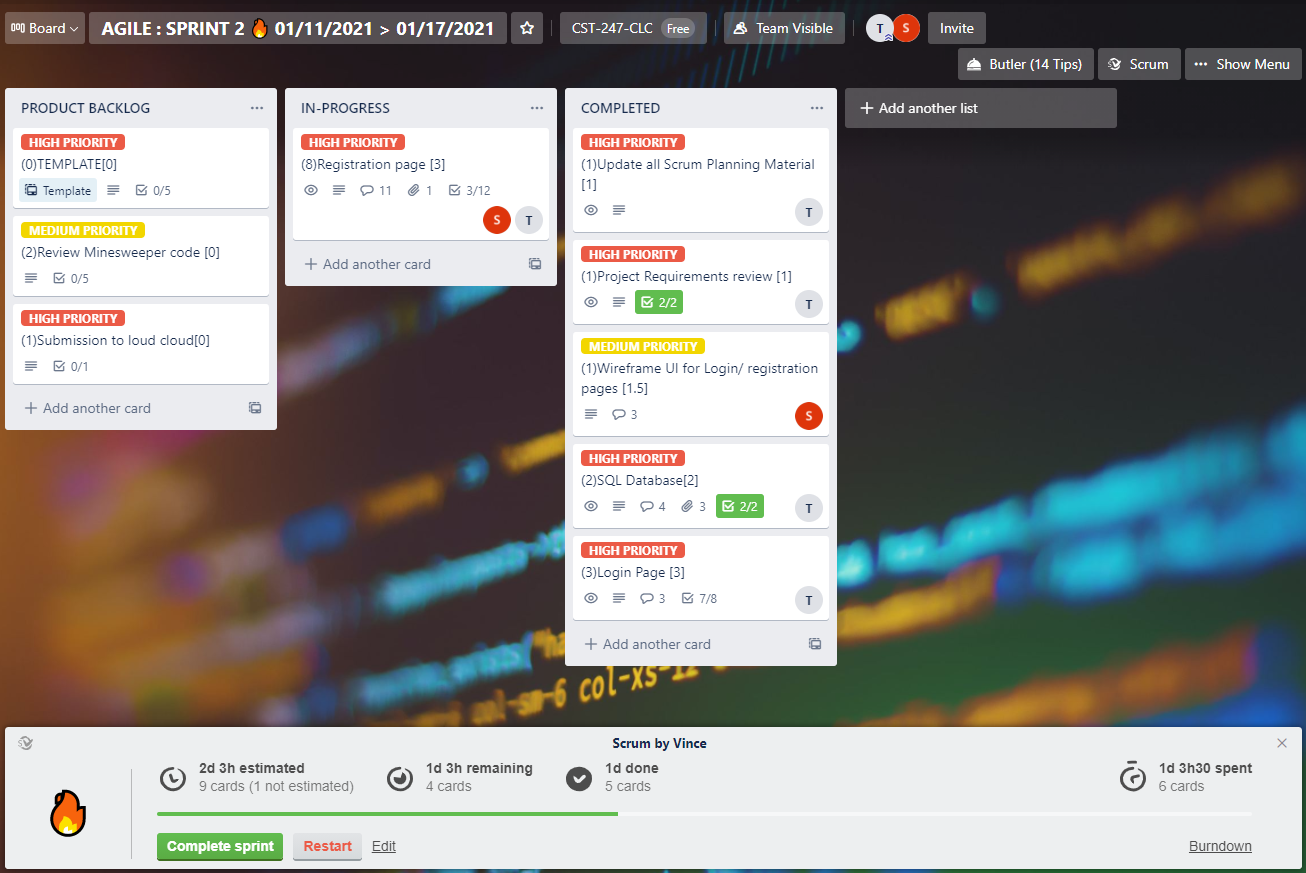
**.NET Application Programming**

**Project Status and Design Report**

|  |  |  |
| --- | --- | --- |
| **Topic:** | *Topic 2: Login and Registration Modules* | |
| **Date:** | *1/16/2020* | |
| **Revision:** | *1.0* | |
| **Team:** | 1. *Tim James* | |
| 1. Stephan Moncavage | |
|  | |
|  | |
| **Weekly Team Status Summary:** | |  |  |  |  | | --- | --- | --- | --- | | **User Story** | **Team**  **Member** | **Hours**  **Worked** | **Hours Remaining** | | *As a dev I would like to have an overview of the entire project and the tasks that it will take to bring the application to completion.* | *Tim James* | *1* | *0* | | *As a Dev I would like to have a form that takes a Username and Password. Upon Login submission the SQL database will be scanned for authentication of credentials. A success or error page will be displayed accordingly. So that users can login to their accounts and so that others cannot access secure data.* | *Tim James/ Stephan Moncavage* | *5* | *0* | | *As a Dev I would like to store users credentials in an SQL database. So that users can be remembered each time they login or change account information.* | *Tim James/ Stephan Moncavage* | *2* | *0* | | *As a dev group we would like to have an initial design idea for the Registration and login page.* | *Stephan Moncavage* | *1.5* | *0* | | *As a Dev I would like to have input fields for First Name, Last Name, Sex, Age, State, Email Address, Username, and password. I would also like there to be form validation on the server side. Upon submit I would like an HTTP POST to route to a controller and save the data submitted to an SQL table. Upon a successful or non-successful registration an error or success page displays. So that users can register accounts on the website securely.* | *Stephan Moncavage* | *14* | *0* | | *As a dev I would like to have UML diagrams, and ER diagrams* | *Tim James* | *1* | *0* | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | | |
| **GIT URL:** | https://github.com/smoncavage/GCU\_CST247\_CLC\_Project.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**



We chose to use Trello for our Scrum planning process. Below you will find the links to get to the trello board.

**Agile Scrum Product Backlog:**

*This needs to contain a URL to BitBucket Scrum Product Backlog Artifact*

[***CST-247-RS-SprintProductLog***](https://trello.com/invite/b/CZPOLBzu/cc1de420b5b1f7d8e2b72f23e5b2321e/agile-sprint-2-%F0%9F%94%A5-01-11-2021-01-17-2021)

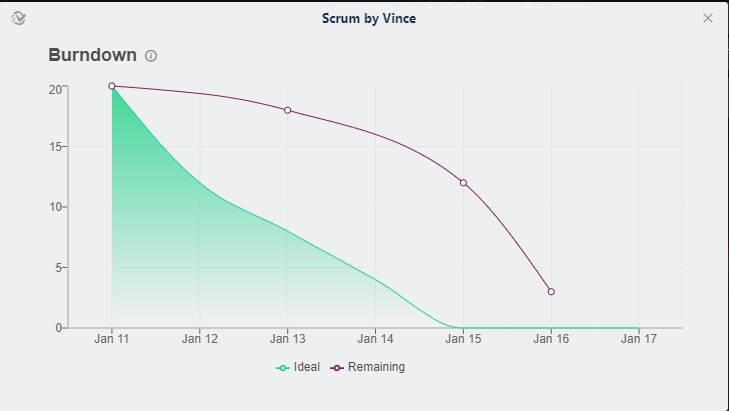
**Agile Scrum Sprint Backlog:**

*This needs to contain a URL to BitBucket Scrum Sprint Backlog Artifact. This current week’s progress should be reflected in the above section of this Design Report.*

[**CST-247-RS-SprintBackLog**](https://trello.com/invite/b/CZPOLBzu/cc1de420b5b1f7d8e2b72f23e5b2321e/agile-sprint-2-%F0%9F%94%A5-01-11-2021-01-17-2021)

**Agile Scrum Burn Down Chart:**

[***CST-247 Scrum Burndown chart***](https://trello.com/invite/b/CZPOLBzu/cc1de420b5b1f7d8e2b72f23e5b2321e/agile-sprint-2-%F0%9F%94%A5-01-11-2021-01-17-2021)

****

**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** |
| The team communicated early and was able to get up and running quickly with the tasks for the milestone. |
|  |
|  |

*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** |
|  |  |  |
|  |  |  |
|  |  |  |

**Design Documentation**

**Install Instructions:**

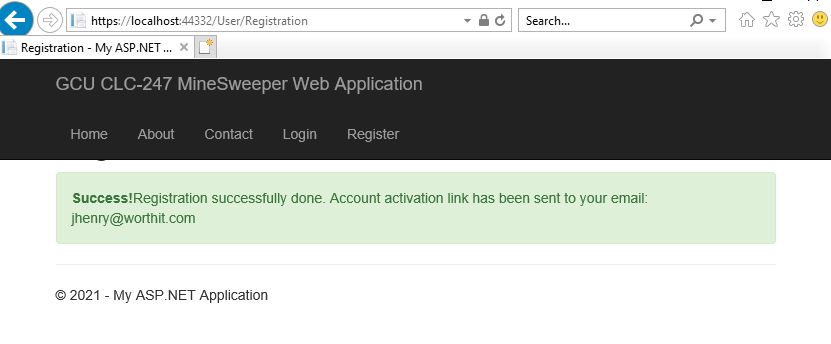
*Step by step instructions for setting up your database, configuring, and deploying/installing your application. This section should also include detailed instructions for what configuration files are required by your application, what configuration settings need to be adjusted for various runtime (development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

No Install instructions at the moment.

**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.*

**Successful Registration**

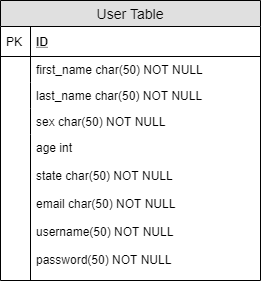
****

**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:**

*Image file of your ER database diagram.*

**

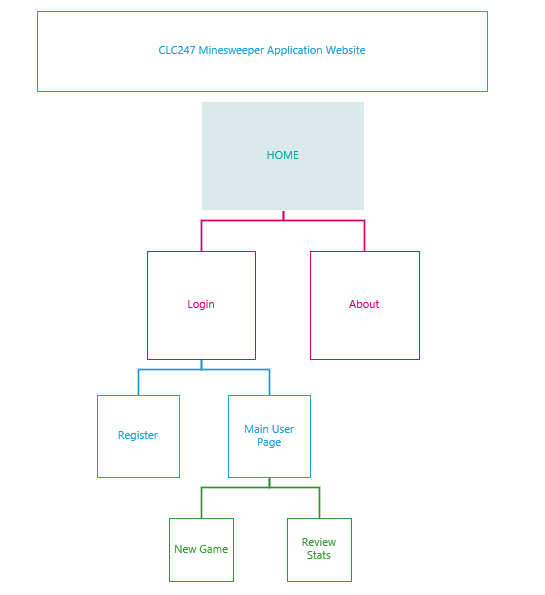
**DDL Scripts:**

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

[*DDL SCRIPT*](https://github.com/smoncavage/GCU_CST247_CLC_Project/blob/main/CLC_Milestone_2/Planning%20and%20Design/UserTable%20DDL%20Script.txt)

**Sitemap Diagram:**

*Image file of your 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 Part Interface Design:**

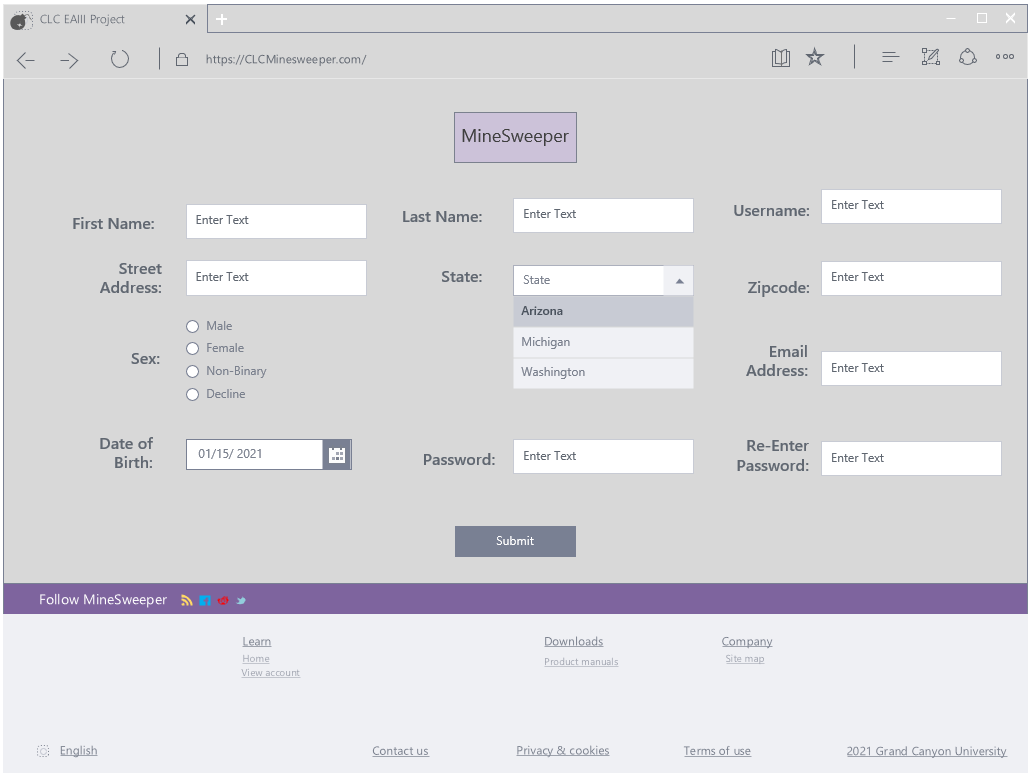
*This section should fully document any Third Party Service Interface API’s, how to access the service, what parameters are required by the API, and the detailed JSON data format specification that could be used by a third party developer to integrate with the service and API.*

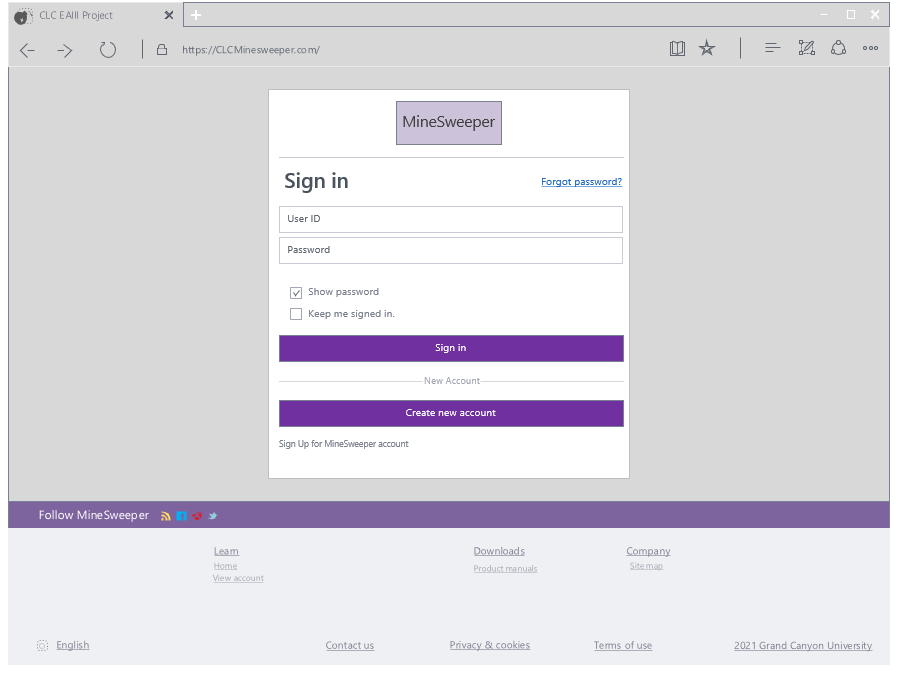
**Flow Charts:**

*You should insert any flow charts here. Flow charts should document algorithms or workflow that will be implemented in your program. At a minimum this should contain a flow chart of the Minesweeper game logic.*

**User Interface Diagrams:**

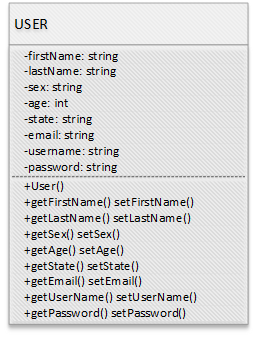
*You should insert any wireframe drawings or white board concepts that were developed to support your application. If you have no supporting documentation please explain the rational why you are able to leave this section as N/A.*

**

**

**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.*