**.NET Application Programming**

**Project Status and Design Report**

|  |  |  |
| --- | --- | --- |
| **Topic:** | *FR-2, FR-3* | |
| **Date:** | *09-25-2020* | |
| **Revision:** | *2.0* | |
| **Team:** | 1. *Matt Sievers* | |
| 1. Crystal Sciarrino | |
|  | |
|  | |
| **Weekly Team Status Summary:** | |  |  |  |  | | --- | --- | --- | --- | | **User Story** | **Team**  **Member** | **Hours**  **Worked** | **Hours Remaining** | | *As a developer I need to create a registration page that has the following fields. Firstname, LastName, Age, Gender, State, emailAddress, Username, Password* | *Crystal Sciarrino* | *0* | *0* | | *As a developer I need to create the player model to help facilitate the creation of the registration page* | *Matt Sievers* | *1* | *0* | | *Persist registration info into localDb* | *Matt Sievers* | *3* | *0* | | As a developer, I need to create a success page for both login and registration | *Crystal Sciarrino* | *0* | *0* | | As a developer, I need to create a login page that will ask the user for user name and password | *Crystal Sciarrino* | *0* | *0* | | As a developer, I need to create a SecurityDAO for login validation (mimic w1 assignment) | *Matt Sievers* | *2* | *0* | | As a developer, I need to create a UserService to use for creating new users on the registration page | *Matt Sievers* | *2* | *0* | | As a developer, I need to create the user login model to help create a teamplate for the login page | *Matt Sievers* | *1* | *0* | | As a team, the weekly team paperwork deliverables need to be completed | *Matt Sievers Crystal Sciarrino* | *2* | *1* | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | | |
| **GIT URL:** | https://github.com/Malleas/CLCMinesweeperApp | |
| **Peer Review:** | *Y/N* | We acknowledge that our team has reviewed this Report and we agree to the approach we are all taking. |

**Planning Documentation**

**Agile Scrum Board:**

[**https://www.meistertask.com/app/project/sSO4jU4u/cst-247**](https://www.meistertask.com/app/project/sSO4jU4u/cst-247)

**Agile Scrum Product Backlog:**

<https://github.com/Malleas/CLCMinesweeperApp/tree/master/CLCMinesweeperApp/Planning%20and%20Design/Week%202>

**Agile Scrum Sprint Backlog:**

[*https://github.com/Malleas/CLCMinesweeperApp/tree/master/CLCMinesweeperApp/Planning%20and%20Design/Week%202*](https://github.com/Malleas/CLCMinesweeperApp/tree/master/CLCMinesweeperApp/Planning%20and%20Design/Week%202)

**Agile Scrum Burn Down Chart:**

[*https://github.com/Malleas/CLCMinesweeperApp/tree/master/CLCMinesweeperApp/Planning%20and%20Design/Week%202*](https://github.com/Malleas/CLCMinesweeperApp/tree/master/CLCMinesweeperApp/Planning%20and%20Design/Week%202)

**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 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** |
| Initial Communication | We came up with a time window and convenient communication method that would work best for us going forward. |  |
|  |  |  |
|  |  |  |

**Design Documentation**

**Install Instructions:**

*Connect and pull from master from repo. Refer to DDL scripts before running as a local DB will need to be created and table added. Click debug menu option and select Run without debug*

**General Technical Approach:**

*We as a team wanted to make sure that the login and registration pages were simple to use and met all requirements listed. We also wanted to add some basic validation in on the login and createUser to ensure the userName and password for login matched as well as the createUser was able to insert the data into the localDB.*

**Key Technical Design Decisions:**

*Reuse the SecurityService and SecurtiyDAO from assignment 1 for the login page as well as create a new UserService to house all user related queries such as createUser or possibly later updateUser.*

**ER Diagram:**

*n/a for this requirement*

**DDL Scripts:**

*CREATE TABLE [dbo].[Player]*

*(*

*[USERID] INT NOT NULL PRIMARY KEY IDENTITY(1,1),*

*[FIRSTNAME] VARCHAR(50) NOT NULL,*

*[LASTNAME] VARCHAR(50) NOT NULL,*

*[GENDER] VARCHAR(50) NOT NULL,*

*[AGE] INT NOT NULL,*

*[STATE] VARCHAR(2) NOT NULL,*

*[EMAILADDRESS] VARCHAR(100) NOT NULL,*

*[USERNAME] VARCHAR(50) NOT NULL,*

*[PASSWORD] VARCHAR(50) NOT NULL*

*)*

**Sitemap Diagram:**

*N/A not required for this FR*

**Security Design:**

*Not needed till FR-4 but as stated above for key technical decisions, we should work on SOP for passwords.*

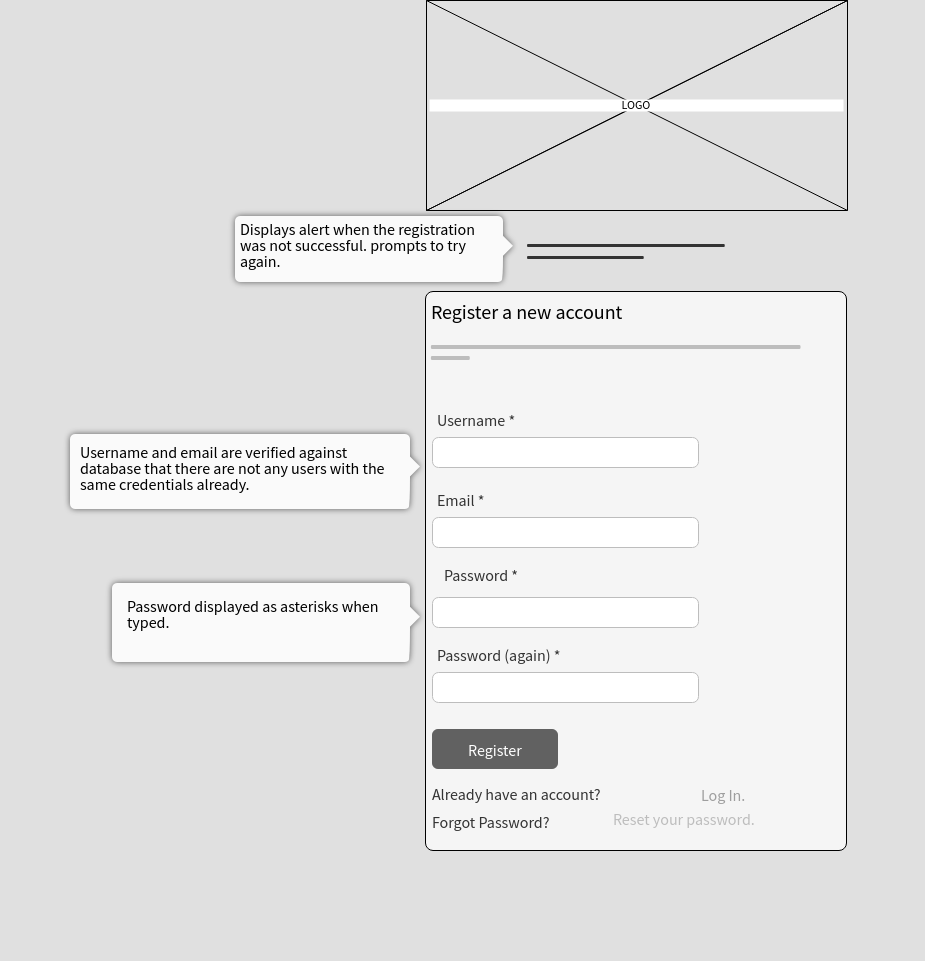
**Third Part Interface Design:**

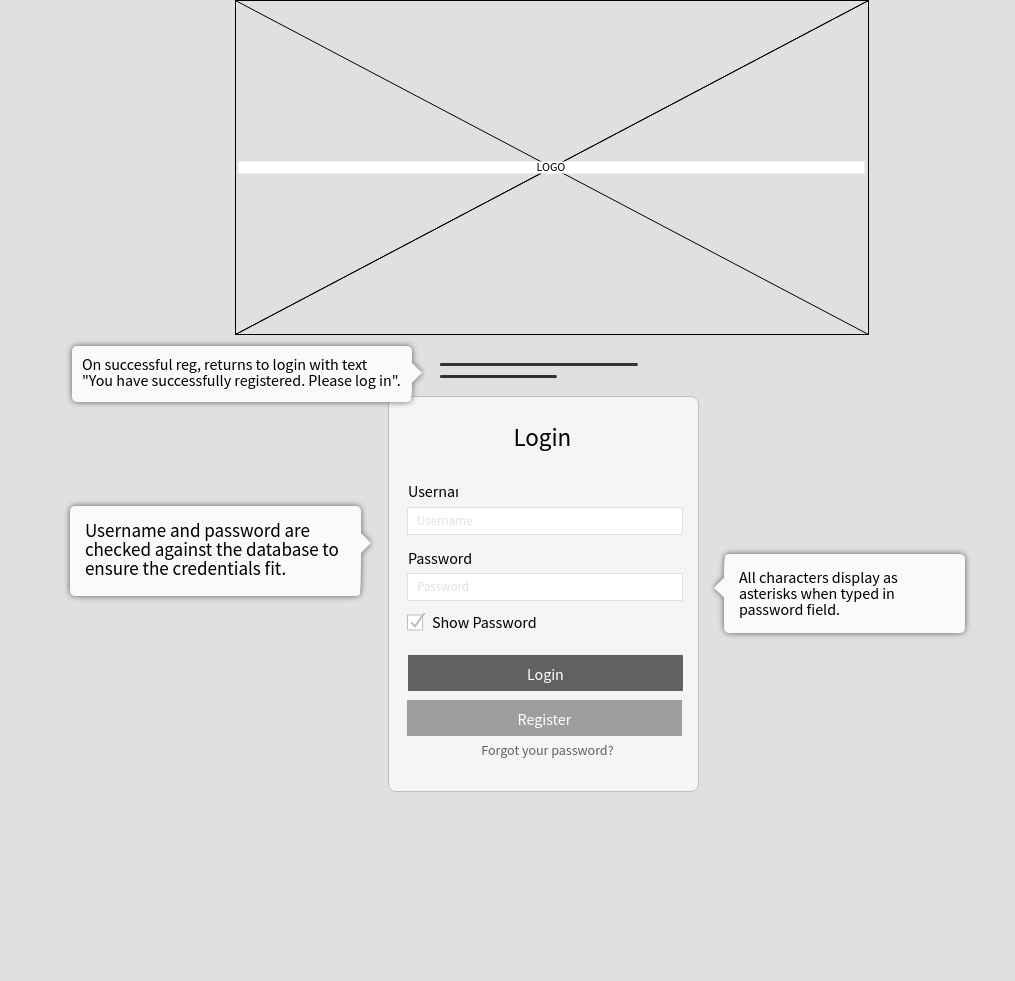
*NA*

**Flow Charts:**

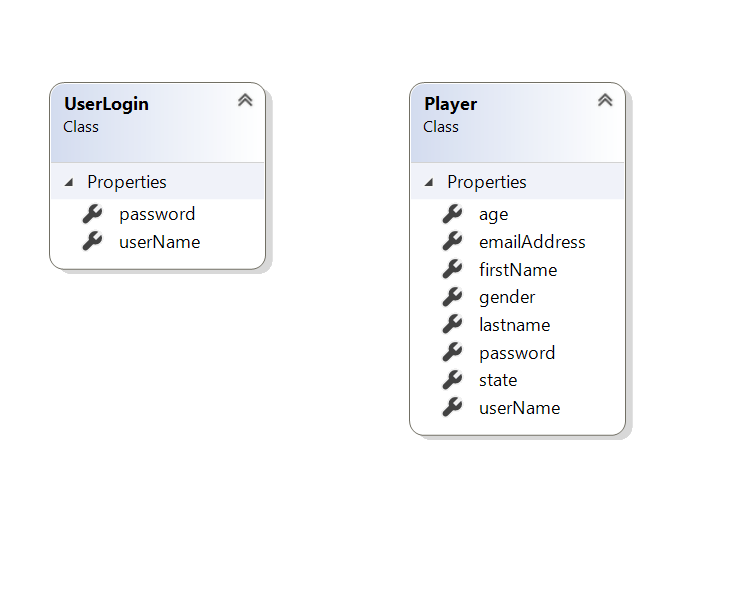
*NA*

**User Interface Diagrams:**





**Class Diagrams:**



**Pseudo Code:**

*N/A*

**Other Documentation:**

*N/A*