**CST-341 Design Report Template**

|  |  |  |
| --- | --- | --- |
| **Topic:** | *CLC Milestone 2* | |
| **Date:** | *September 23, 2018* | |
| **Revision:** | *2.0* | |
| **Team:** | 1. *William Bierer* | |
| 2. *Brendan Brooks* | |
| 3. | |
| 4. | |
| **Weekly Team Status Summary:** | |  |  |  |  | | --- | --- | --- | --- | | **User Story** | **Team**  **Member** | **Hours**  **Worked** | **Hours Remaining** | | *Create Views, forms, CSS Stylesheets* | *Brendan Brooks* | *6* | *0* | | *Create Models, Controller, data validation, update documentation* | *William Bierer* | *6* | *0* | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | |  |  |  |  | | |
| **Screencast Link** | <https://youtu.be/Yfe4z_1YY6w> | |
| **GIT URL:** | *https://bitbucket.org/Wbierer/cst-341\_bierer\_brooks* | |
| **Peer Review:** | *Y* | We acknowledge that our team has reviewed this report and we agree to the approach we are all taking. |

**Planning Documentation**

**Overview**

For our application, we will be using Agile Scrum methodology to plan, manage, and divide work between each other. We will use our Sprint backlog to plan tasks for each Sprint as well as divide out work. A Burn Down chart will be utilized to manage the amount of time needed to complete each task. Our Product Backlog will allow us to prioritize future tasks based on their importance. Finally, we will hold retrospectives at the end of each sprint to determine what went well and what we can improve on.

**Agile Scrum Product Backlog, Sprint Backlog, and Burn Down Chart:**

[*https://docs.google.com/spreadsheets/d/1jgQ6v3Dwx\_81837OPo\_dOBWQVx2jjOyza0obkbfcC50/edit?usp=sharing*](https://docs.google.com/spreadsheets/d/1jgQ6v3Dwx_81837OPo_dOBWQVx2jjOyza0obkbfcC50/edit?usp=sharing)

**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** |
| **We finished the login/registration modules and started the main module.** |
|  |
|  |

*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** |
| **We could not get 2 forms to work on a single login/reg form** | **Figure out how to pass two models into a view** |  |
|  |  |  |
|  |  |  |

**Design Documentation**

**Application Domain & High Level features description**

The domain of the products is the digital media industry. This application will serve as almost like a “cloud” for people to upload their music to and stream it from there. It is a fairly straight-forward site that will have a library of songs on one page, and another page where users can upload songs to the library. Although users don’t need an account to listen to audio, they need an account to upload songs to the library.

**Install Instructions:**

*Include 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:**

*In your own words describe your approach and design here. You should also summarize any meeting notes, brain storming sessions, and so forth that you want to retain thru the design of your project.*

* Audio Share Web Application
  + Login/Registration System
    - Registration includes: First Name, Last Name, Email, and Password
    - Cannot upload mp3 files unless the user is logged in
  + Audio Player
    - User can play uploaded audio files in the application
    - User needs to have an account to listen to audio
    - User can only listen to audio that they upload to the application
  + Audio File Uploader
    - User can upload audio files; location in filesystem will be stored in the DB
    - User must be logged in to upload audio files
    - User can categorize uploaded songs by album

**Key Technical Design Decisions:**

*Any final technical design decisions, (e.g., framework decisions) should be documented here. List the technology/framework, its purpose in the design, and why it was chosen.*

· Bootstrap CSS Library: Bootstrap is a powerful open-source CSS framework that will be used to build our front-end with a responsive design

· MySQL database will be used; it will be a remote server so modifying the project as a group will be simpler

· Jquery will be used to switch forms in single login/registration view

**Known Issues:**

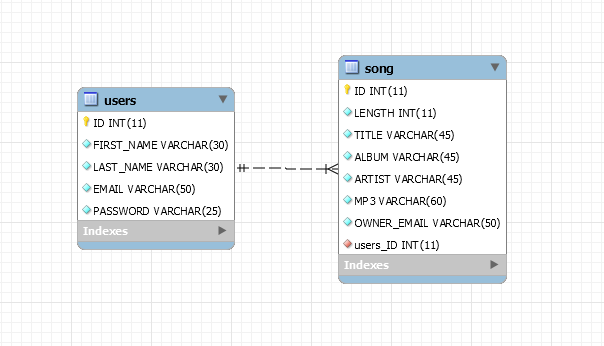
* Data Validation messages go off the form
* Logo is in wrong position on login view due to css issue
* Currently using an ArrayList of users instead of MySQL DB

**Risks:**

· Preventing overwrites; may have to auto-generate names for the files so nothing is overwritten

·

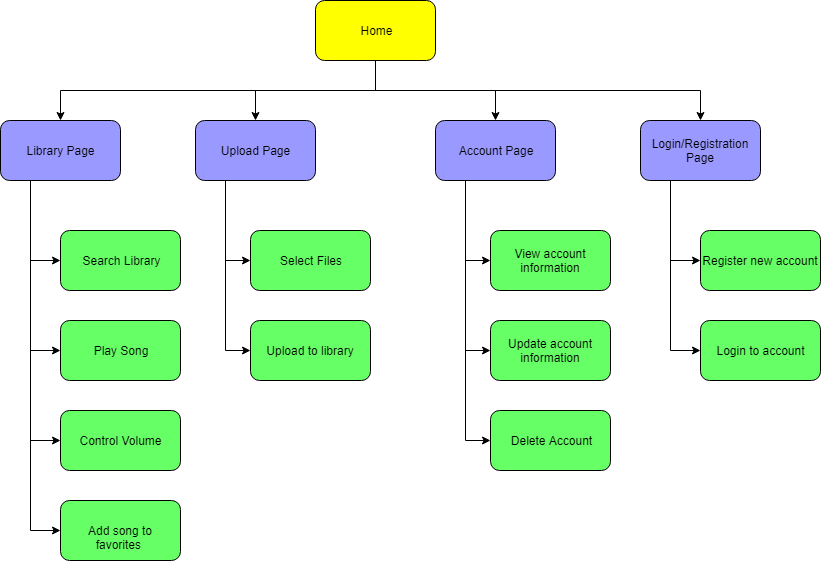
**ER Diagram:**



**DDL Scripts:**

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

**Sitemap Diagram:**

**

**User Interface Diagrams:**

**Login Wireframe**

A screenshot of a cell phone

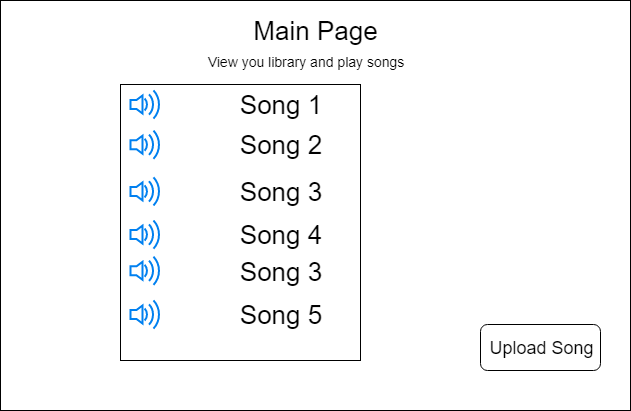
Description generated with high confidence

**Registration Wireframe**

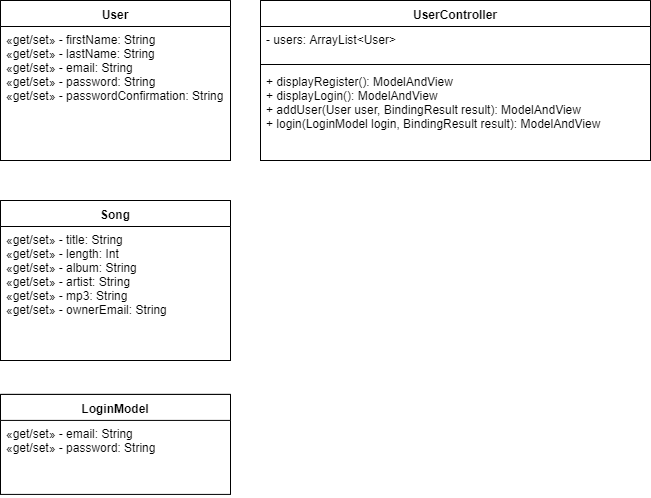
**A screenshot of a cell phone

Description generated with very high confidence**

**Main Page Wireframe**

****

**Class Diagrams:**



**Service API Design:**

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

**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.*

**Other Documentation:**

*You should insert any additional drawings, storyboards, white board pictures, project schedules, tasks lists, and so forth that support your approach, design, and project. If you have no supporting documentation, please explain the rationale for laveling this section N/A.*