# CEN4010 Principles of Software Engineering

### Spring 2021

# Milestone 3 Project Proposal and High-level Description

## Group 2: Team Rocket

## Project name: *Apollo Melodies*

### Team Members

Ivan Bruno-Gaston (Product Owner) - [ibrunog@fau.edu](mailto:ibrunog@fau.edu)

Craig Peroni (Scrum Master) - [cperoni2020@fau.edu](mailto:cperoni2020@fau.edu)

Sharmada Iyer (Developer) - [siyer2018@fau.edu](mailto:siyer2018@fau.edu)

Belal Mahmood (Developer) - [bmahmood2018@fau.edu](mailto:bmahmood2018@fau.edu)

Isabel Tait (Developer) - [itait2017@fau.edu](mailto:itait2017@fau.edu)

About Us

<https://lamp.cse.fau.edu/~cen4010_s21_g02/>

Test Site

(Note: Only profile creation, profile login, profile edit, and logout functionality is currently active)

<https://lamp.cse.fau.edu/~cen4010_s21_g02/projectdemo>

### Date: 4/6/2021

|  |  |
| --- | --- |
| Revision Number | Date |
| 1.0 | 2/16/2021 |
| 2.0 | 4/6/2021 |

1. Executive Summary

**1.1 - Introduction**

Listening to music is a hobby to many people. Many music platforms allow users to enjoy music uploaded by several artists. However, most of these platforms rarely use music as a fun and competitive interaction. The COVID-19 pandemic has restricted outdoor fun activities so it would be good to have more activities for music lovers inside their homes. The internet consists of games that are fun and competitive but very few incorporate music. The purpose of this project is to use music in the form of a game to unleash a competitive environment between users. With Apollo Melodies, users can listen to clips of music, answer multiple choice questions about those clips, and compare their knowledge with other users and friends.

**1.2 - Objectives**

* To provide a fun game-based platform that unleashes a competitive environment with the use of music.
* Allows users to create an account and keep track of their high scores while giving them the option to compete with other users on the platform.
* Provide a variety of genres and difficulty levels for the user to choose from.

**1.3 - Target Customers**

* People who listen to music as a hobby - Individuals can put their music knowledge to the test by taking the music quiz on any genre they so desire.
* Families - Especially due to the COVID-19 Pandemic, families have not been able to engage in outdoor activities. This platform can be a great game to play on family game nights or occasions.
* Individuals looking to expand knowledge on music - There is no required knowledge necessary to play the game. This platform is a great way to expand knowledge on music and even receive recommendations.

**1.4 - Value Proposition**

Apollo Melodies fills in the need of having a game-based platform that is tailored towards music. It is a platform that can be played by individuals of all ages and different music tastes. Users will have the option of either playing a single player, single player competing with other users, or multiplayer to play on game nights or other occasions.

**1.5 - Application Features and Descriptions**

Games can be played without having to sign-in. However, in order to be on the leaderboard and compete with other people, users are required to sign in. The user can select the genre as well as the range of difficulty. The game will play snippets of audio and the user will be asked multiple choice questions on the audio snippet (e.g. Who is the artist of this song?).

2. Competitive Analysis

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Single Player | Add Friends | Music player(Specifically for questions) | All Ages | Free | Fast Paced |
| Apollo Melodies | ✅ | ✅ | ✅ | ✅ | ✅ |  |
| Kahoot | ✅ |  |  | ✅ | ✅ | ✅ |
| Quizlet Live | ✅ |  |  | ✅ |  | ✅ |

Kahoot and Quizlet have been used countless times over the years by many different people, whether it be for fun or studying. Instead of questioning social knowledge in Quizlet or Kahoot, Apollo Melodies brings a spin to the competition by testing music knowledge. Quizlet Live is Quizlet’s newest feature that allows for multiplayer platforms. Most quiz platforms have a primary goal of teaching you information instead of focusing on having fun and listening to music.

Kahoot is one of the most widely used applications for interactions in school between professors and students. It has been known to help students with retaining knowledge for years. While Kahoot is widely used it has been said that it can be a bit too fast paced for students to answer questions; leaving some students, who aren’t so sure, feeling negative. Kahoot also doesn’t use actual music on the website except for background music. In Apollo Melodies, there is a music player that plays a small snippet for the user to listen to and answer a question about the snippet afterwards.

Quizlet Live is a new feature in Quizlet that allows users to remotely engage with other users to play games. Quizlet has been around for many years and has been used in numerous different surroundings from school to companies. Quizlet allows users to use premade cards or make new cards. Quizlet Live is a fast paced environment that tests user social knowledge. In order to have a Quizlet Live account you have to be either a teacher or parent and also pay after a 7-day free trial. Apollo Melodies doesn’t charge for usage, allows any institutions to use it, and is meant to have the user feel like they are having fun rather than learning focused.

3. Data Definitions

**3.1 - Music Quiz**

* **Alias**: None
* **Description:** A series of multiple choice questions related to the series of songs that are played on the webpage.
* **Related Data Items:** None
* **Range of Values:** Not applicable

**3.2 - Score**

* **Alias:** None
* **Description:** The total amount of points a user has achieved on the quiz which is stored in their profile database.
* **Related Data Items:** Score must always be equal to the total number of points the user has accumulated during one quiz.
* **Range of Values:** 0 - 100

**3.3 - Point**

* **Alias:** mark
* **Description:** When the user answers a question correctly, they accumulate a certain number of points. When they don’t answer the question correctly, they either do not gain points or their score may be lowered.
* **Related Data Items:** The total number of points must be equal to the user’s score.
* **Range of Values:** 0 - 100

**3.4 - User**

* **Alias:** participant, contestant, competitor, user
* **Description:** An actor who is actively engaging with a music quiz.
* **Related Data Items:** A player takes a music quiz and gains points that contributes to their score if they answer any questions correctly.

**3.5 - API (Application Programming interface)**

* **Description:** Any programming tool provided by third-party to allow a developer to use that third party’s resources on the developer’s site.

**3.6 - LAMP Server**

* **Description:** A web server which implements Linux, Apache, MySQL, and PHP (L.A.M.P).

**3.7 - “Eye-Level” Graphical User Interface (GUI or UI)**

* **Description:** This is the front end (A.K.A client facing) of the site which describes all of the elements of the application that the user can see and use.

**3.8 - Site Admin**

* **Description:** A site user with administrative rights

4. Overview, Scenarios, Use Cases

**4.1 - Scenario 1**

John tested positive for COVID-19 so he began his quarantine. He has a strong interest in music and decided to check out Apollo Melodies to test his knowledge on music. He scrolled through and picked the genre he was most comfortable with and began to play a game that tested his ability to guess the song using song lyrics and audio snippets.

**4.2 - Scenario 2**

Kate and Jill were having a competition about who knew the most songs. They go to Kate’s house and go to Apollo Melodies to play. They listen to quick snippets of songs and answer the questions that follow. They keep track and see who wins.

**4.3 - Scenario 3**

Jake recently started taking classes at Florida Atlantic University. He has already met with his advisor and gave him a list of classes he will need to take this semester. One of the courses taken is “Music Appreciation 101”. Professor Smith, Jake’s “Music Appreciation 101” wanted a new and more interactive way to test students, he heard about Apollo Melodies through the teacher grape vine. He goes to the site, creates a login, and some tests for the class. Monday comes and they begin the class with a short quiz on music knowledge. The students make an account then Professor Smith chooses a genre, classical, and then displays the questions on the screen and while the students answer on their mobile devices or computers. The class had the best time.

**4.4 - Use Case 1: User Login**

1. The user opens the web application
2. The user selects the option for account login.
3. The user populates the Username field.
4. The user populates the Password field.
5. The user clicks the Login button.

**User Login Variant: The user enters wrong username/password combination**

1. The user opens the web application.
2. The user selects the option for account login.
3. The user populates the Username field.
4. The user populates the Password field with the wrong password.
5. A warning appears telling the user they have entered the wrong user/password combo.

**4.5 - Use Case 2: User Signup**

1. The user opens the web application.
2. The user selects the option to create an account.
3. The user populates the Username field.
4. The user populates the Password field.
5. The user clicks the Create Account button.

**User Signup Variant: The user tries to sign up with an existing account**

1. The user is on the Create Account page.
2. The user populates the Username field.
3. The user populates the Password field.
4. The user clicks the Create Account button.
5. A warning is thrown telling the user the account already exists.

**Use Case 3: User creates a quiz**

1. The user successfully logs in.
2. The user selects the Create Quiz option.
3. The user adds quiz questions along song URLS from Soundcloud.
4. When done, the user selects Finish Quiz.

**Use Case 4: User participates in a quiz**

1. The user successfully logs in.
2. The user selects a quiz to participate in.
3. The user is navigated to the first question of the quiz.

5. Initial High-Level Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement ID | Requirement Statement | Must/Want | Comments |
| FR001 | A user must be able to log into the software and arrive at a personalized home page. | Must |  |
| FR002 | A user must be able to set up an account if the user does not already have one. | Must | A user does not have to make an account to take a quiz if they don’t want to. |
| FR003 | A user should be able to create a sound quiz using the site user interface (UI). | Want | A user can take pre-made quizzes without being required to make their own. |
| FR004 | A user must be able to complete a sound quiz and receive a score for quiz completion. | Must |  |
| FR005 | A user should be able to search for and send friend requests to other users on the site. | Want |  |
| FR006 | A user should be able to change his/her profile status so that they would not be searchable and cannot receive random friend requests on the site. | Want |  |
| FR007 | Site admins should be able to add, remove or modify site user accounts. | Want |  |
| FR008 | Users should be able to log out of the site so that they can sign in with another account if desired. | Must |  |

6. List of Non-Functional Requirements

**6.1 - Security**

Since the application is not expected to contain any Personally Identifiable Information (P.I.I.) and will not facilitate any financial transactions, security risk to the site is considered to be low. Therefore a moderate level of password encryption will be used to encrypt password data. Passwords must contain at least 9 characters.

Validation of passwords based on password rules will be accomplished with JavaScript on the front end of the app. The encryption algorithm will be executed in PHP on the server-side of the application.

**6.2 - Usability**

The target customer for this application will be a user with moderate computing skills. The user should not need advanced computing skills in order to use the product. GUI design should be simple and screen hints should be available to guide users through the process (no training).

**6.3 - Reliability**

User volume on this application is not expected to be high and, thus server up-time is considered to be moderately critical. The expected downtime (other than hardware failures) will likely be due only to pushing new implementations of the product from the development environment to the production environment. Dev-to-Prod pushes are expected to bring the site offline no more than 5 minutes. Dev-to Prod pushes will be off-cycle throughout the duration of the project. However, these pushes will take place after hours.

**6.4 - Maintainability**

Admin users will be created for this site. Admin users will have the ability to make basic configuration changes to the site (i.e. style color), and manage regular users on the site. An admin user should be able to make basic configuration changes to the site through the front end GUI without the need to directly program to an API.

**6.5 - Performance**

* 6.5.1 - **Page Loading Time** - Page loading times should take no longer 90 seconds on a computer or mobile device with a current browser and internet speed of 4G or more
* 6.5.2 - **Concurrent usage** - The site should be able to support at least 1000 concurrent users operating on the site
* 6.5.3 - **UI Responsiveness** - The display should adapt to the size of the viewing port within 10 seconds of full page load.

7. High-Level System Architecture

**7.1 - Overview**

In this section of the proposal,we will detail the technology stack used to implement the software solution. All tools from the hardware to the client-facing GUI will be detailed. The supporting technology will be configured as an ecosystem where this or multiple applications (or application components) can share real time access to the underlying databases

**7.2 - Hardware**

The hardware which will host the application is a LAMP server provisioned by Florida Atlantic University for our group to use. This Linux server implements Apache, MySQL, and PHP Application development environments. No other hardware will be used.

**7.3 - Database Utility**

To build the application databases, we will use the MySQL implementation on the LAMP server. One database will be developed and tables will be created for member data, activities, and administrative needs. This database will be relationally implemented. The unique key and relational key for all database tables will be the unique user ID.

**7.4 - Database Schema - tables and columns:**

* DEV\_ACTIVITY\_LOG (Logs all user activity on the site)
  + ACTIVITY\_NUMBER - varchar(255)
  + ACTIVITY\_TYPE - varchar(20)
  + DOER\_MEMBER\_NUMBER - varchar(255)
  + RECIPIENT\_MEMBER\_NUMBER - varchar(255)
  + QUIZ\_NUMBER - varchar(255)
  + MESSAGE\_NUMBER - varchar(255)
  + BLOCK\_NUMBER - varchar(255)
  + BLOG\_NUMBER - varchar(255)
  + MEDIA\_NUMBER - varchar(255)
  + ACTIVITY\_DETAIL - varchar(3000)
  + DATE\_COMPLETED - timestamp
* DEV\_BLOCK\_LIST (Records when a user blocks another user)
  + BLOCK\_RECORD\_NUMBER - varchar(255)
  + BLOCKER\_PROFILE\_NUMBER - varchar(255)
  + BLOCKEE\_PROFILE\_NUMBER - varchar(255)
  + IS\_DELETED - tinyint(1)
  + DATE\_CREATED - timestamp
  + DATE\_DELETED - timestamp
* DEV\_BLOG\_POST (Store blog post data)
  + BLOG\_NUMBER - varchar(255)
  + BLOG\_CREATOR\_NUMBER - varchar(255)
  + BLOG\_TYPE - varchar(20)
  + BLOG\_MEDIA\_LINK - varchar(255)
  + BLOG\_TEXT - varchar(3000)
  + IS\_DELETED - tinyint(1)
  + DATE\_CREATED - timestamp
  + DATE\_DELETED - timestamp
* DEV\_FRIEND\_LIST
  + FRIENDSHIP\_NUMBER- varchar(255)
  + FRIENDOR\_MEMBER\_NUMBER- varchar(255)
  + FRIENDOR\_SHOW\_REAL\_NAME\_FLAG- tinyint(1)
  + FRIENDOR\_SHOW\_MEDIA\_FLAG- tinyint(1)
  + FRIENDOR\_SHOW\_QUIZ\_OWNED\_FLAG- tinyint(1)
  + FRIENDOR\_SHOW\_QUIZ\_HISTORY\_FLAG- tinyint(1)
  + FRIENDEE\_MEMBER\_NUMBER- varchar(255)
  + FRIENDEE\_SHOW\_REAL\_NAME\_FLAG - tinyint(1)
  + FRIENDEE\_SHOW\_MEDIA\_FLAG- tinyint(1)
  + FRIENDEE\_SHOW\_QUIZ\_OWNED\_FLAG- tinyint(1)
  + FRIENDEE\_SHOW\_QUIZ\_HISTORY\_FLAG- tinyint(1)
  + FRIENDSHIP\_REQUEST\_STATUS- varchar(50)
  + DATE\_REQUESTED- timestamp
  + DATE\_ACCEPTED- timestamp
  + DATE\_REJECTED- timestamp
  + DATE\_DELETED- timestamp
* DEV\_MEDIA\_LIST
  + MEDIA\_NUMBER - varchar(255)
  + MEDIA\_NAME - varchar(255)
  + CREATOR\_NUMBER - varchar(255)
  + MEDIA\_TYPE - varchar(20)
  + MEDIA\_LINK - varchar(255)
  + IS\_PRIVATE - tinyint(1)
  + IS\_DELETED - tinyint(1)
  + DATE\_CREATED - timestamp
  + DATE\_DELETED - timestamp
* DEV\_MEMBER\_PROFILE
  + MEMBER\_NUMBER- varchar(255)
  + MEMBER\_ID- varchar(255)
  + ENCRYPT\_PASSWORD- varchar(3000)
  + FIRST\_NAME- varchar(120)
  + LAST\_NAME- varchar(120)
  + DOB- date
  + EMAIL\_ADDRESS- varchar(100)
  + SEARCHABLE\_PROFILE\_FLAG- tinyint(1)
  + IS\_ONLINE\_FLAG- tinyint(1)
  + MEMBER\_BAN\_FLAG- tinyint(1)
  + LOGICAL\_DELETE\_FLAG- tinyint(1)
  + AUTHORITY\_LEVEL- int(11)
  + DATE\_CREATED- timestamp
  + DATE\_DELETED- timestamp
* DEV\_MESSAGE\_HISTORY
  + MESSAGE\_ID - varchar(255)
  + SENDER\_NUMBER - varchar(255)
  + RECIPIENT\_NUMBER - varchar(255)
  + MESSAGE\_STATUS - varchar(20)
  + MESSAGE\_BODY - varchar(3000)
  + IS\_ADMIN\_MESSAGE - tinyint(1)
  + IS\_SENDER\_DELETED - tinyint(1)
  + IS\_RECIPIENT\_DELETED - tinyint(1)
  + DATE\_CREATED - timestamp
  + DATE\_SENT - timestamp
  + DATE\_SENDER\_DELETED - timestamp
  + DATE\_RECIPIENT\_DELETED - timestamp
* DEV\_QUIZ\_HEADER
  + QUIZ\_NUMBER - varchar(255)
  + QUIZ\_NAME - varchar(120)
  + QUIZ\_CREATOR\_ID - varchar(255)
  + QUIZ\_GENRE - varchar(120)
  + DATE\_CREATED - timestamp
  + DATE\_LAST\_EDITED - timestamp
  + PASS\_PERCENTAGE - decimal(10,10)
  + IS\_DELTED - tinyint(1)
  + DATE\_DELETED - timestamp
  + HIGHEST\_QUESTION\_NUMBER - int(11)
* DEV\_QUIZ\_HISTORY\_HEADER
  + QUIZ\_HISTORY\_NUMBER- varchar(255)
  + QUIZ\_NUMBER- varchar(255)
  + QUIZ\_CREATOR\_NUMBER- varchar(255)
  + QUIZ\_COMPLETER\_NUMBER- varchar(255)
  + DATE\_COMPLETED- timestamp
  + TOTAL\_QUESTIONS- int(100)
  + TOTAL\_QUESTIONS\_CORRECT- int(100)
  + SCORE- decimal(10,10)
  + IS\_PASSING- tinyint(1)
* DEV\_QUIZ\_HISTORY\_QUESTION
  + QUIZ\_HISTORY\_NUMBER - varchar(255)
  + QUIZ\_NUMBER - varchar(255)
  + QUESTION\_NUMBER - int(100)
  + MUSIC\_SOURCE\_URL - varchar(255)
  + ANSWER\_1 - varchar(255)
  + ANSWER\_2 - varchar(255)
  + ANSWER\_3 - varchar(255)
  + ANSWER\_4 - varchar(255)
  + ANSWER\_CHOSEN - int(11)
  + CORRECT\_ANSWER - int(11)
  + DATE\_COMPLETED - timestamp
* DEV\_QUIZ\_QUESTION
  + QUIZ\_NUMBER- varchar(255)
  + QUESTION\_NUMBER- int(100)
  + MUSIC\_SOURCE\_URL- varchar(255)
  + ANSWER\_1- varchar(255)
  + ANSWER\_2- varchar(255)
  + ANSWER\_3- varchar(255)
  + ANSWER\_4- varchar(255)
  + CORRECT\_ANSWER- int(11)
  + HINT- varchar(255)
  + DATE\_CREATED- timestamp
  + IS\_DELETED- tinyint(1)
  + DATE\_DELETED- timestamp
* DEV\_SYSTEM\_ID\_NUMBER\_COUNTER
  + COUNTER\_NAME- varchar(100)
  + COUNT\_VALUE- int(11)

**7.5 - Server-Side Scripting**

In order for the application info interface between the client-facing GUI and the database, PHP will be used to perform the server-side scripting. This includes validating and encrypting the user login, creating new users, and adding/modifying content at the user level. Other features which emerge during the progression of the project will also use PHP.

**7.6 - Client-Side Scripting**

Any client-side scripting will be done using JavaScript and CSS. This includes animation, front-end form validation, and any other user experience elements.

**7.7 - “Eye-Level” Graphical User Interface (GUI)**

The front end GUI for the application will be Coded in HTML5 and CSS by implementing a mobile-responsive Bootstrap template acquired from <https://startbootstrap.com/>. The actual final template chosen will be decided later in the project. Proper credit for the template used will be listed on the site. Bootstrap is compatible with most available browsers

***7*.8 - SoundCloud API**

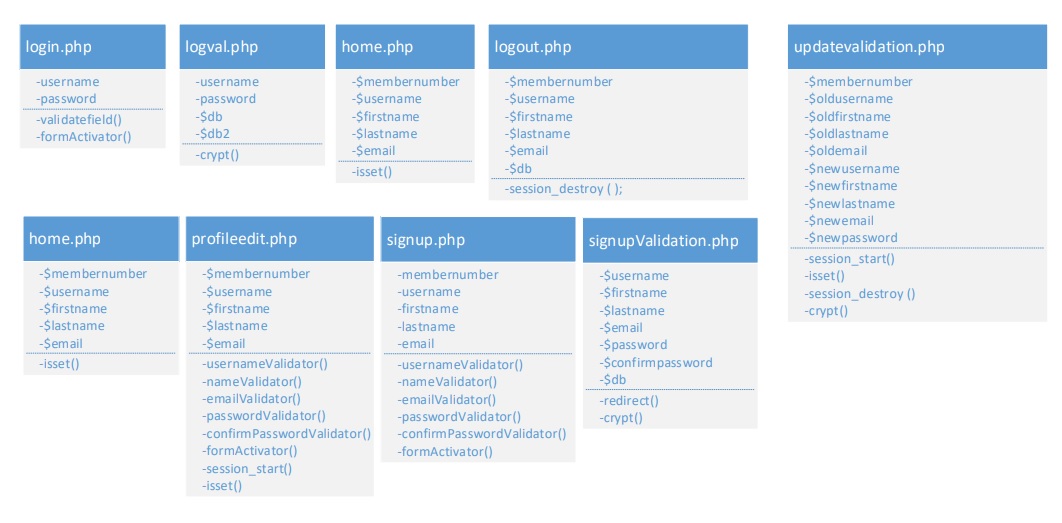
Soundcloud provides an external API to allow other web applications to access music resources and display them with their site. This code is found at <https://developers.soundcloud.com/docs/api/html5-widget>.

**7.9 - Search Algorithms**

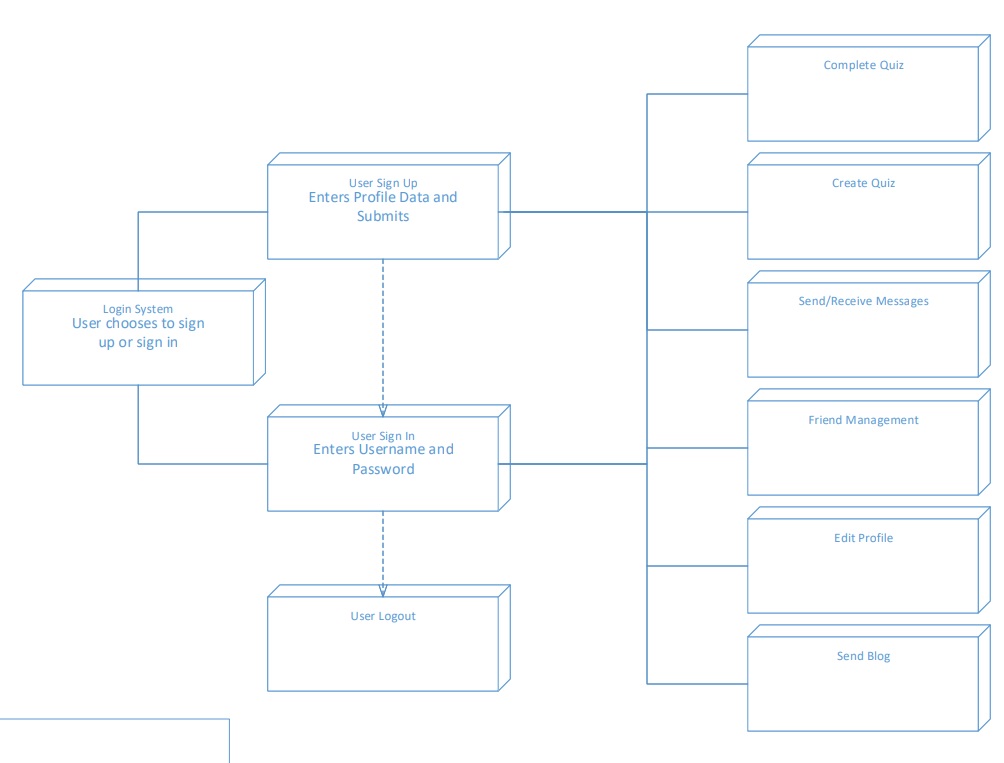
Basically, searches were completed through select queries sent from the PHP server into the Database and the results were captured in Array objects in PHP.

8. High-Level UML Diagrams

**8.1 - UML Class Diagram (All code on the site is executed as procedural code so there are no object classes. Therefore the pages of the site written as objects)**

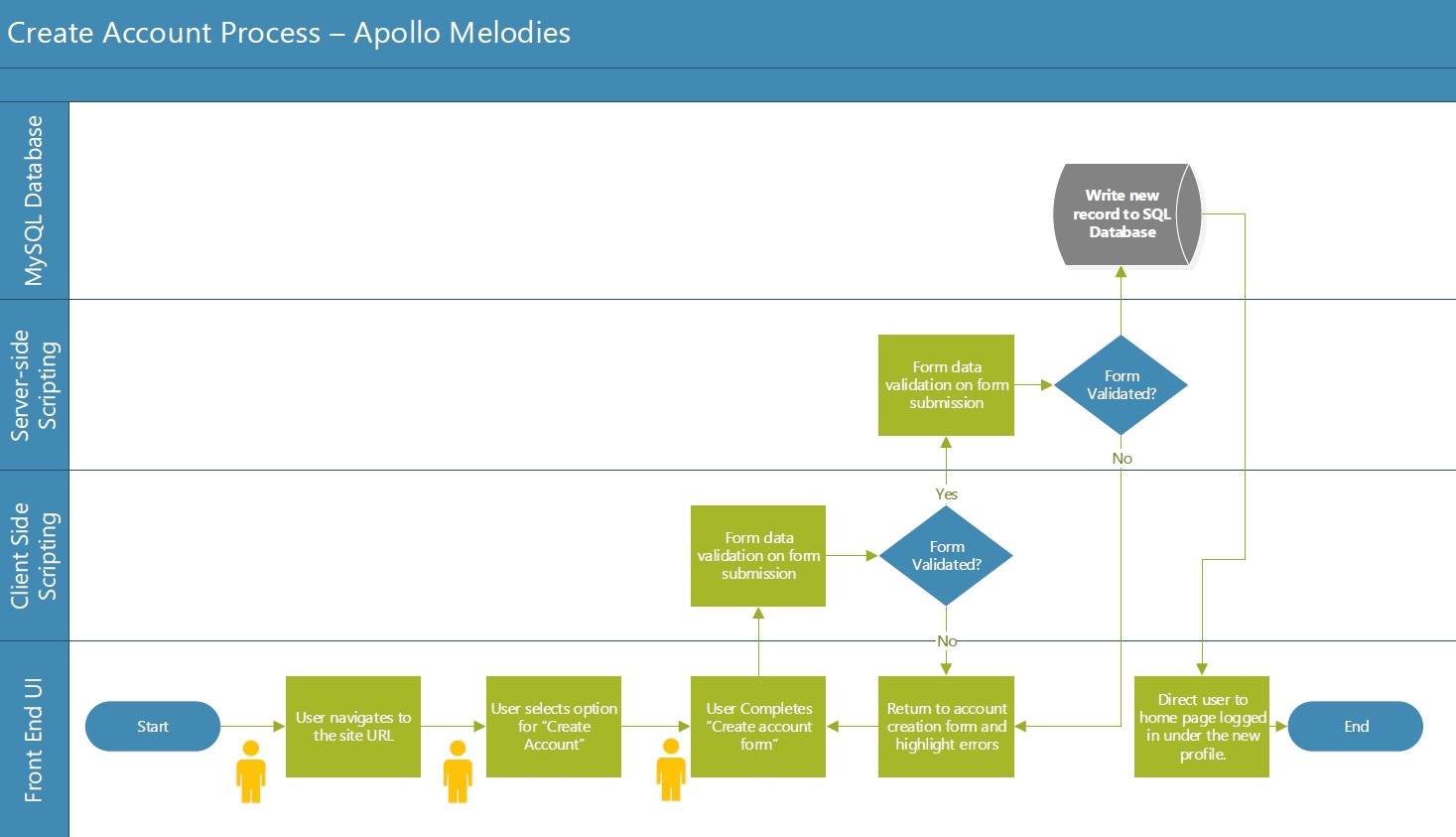


**8.2 - UML Deployment Diagram**

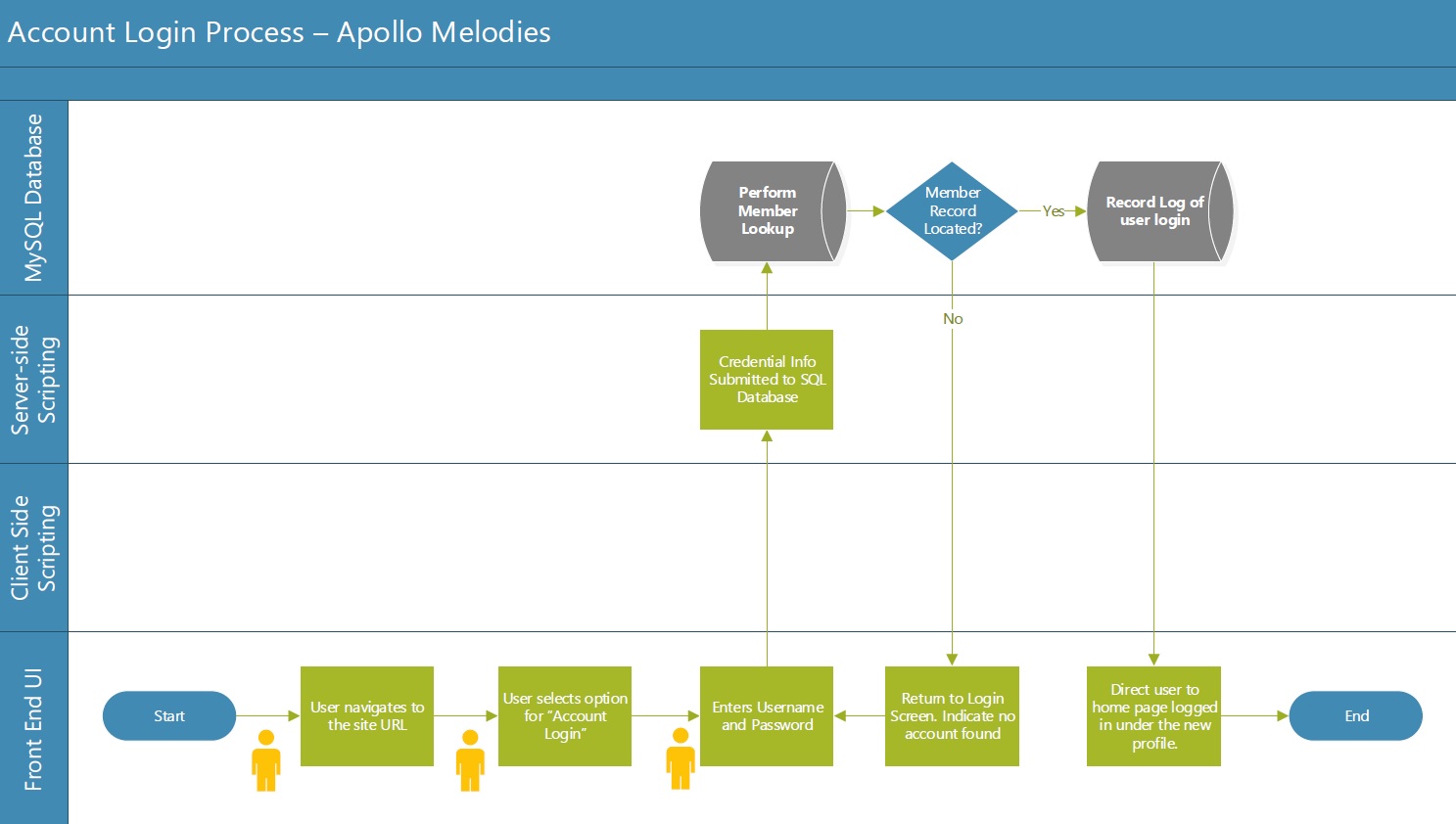


9. Appendix

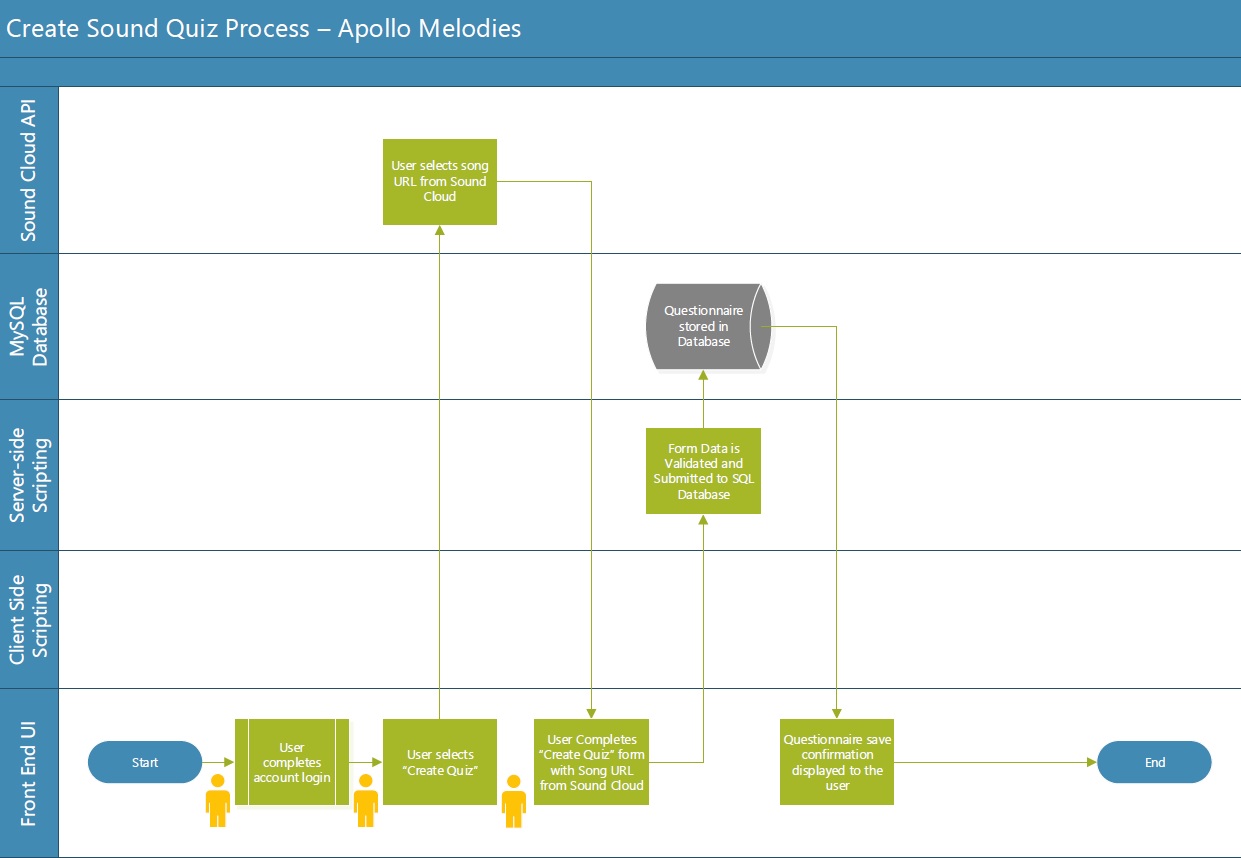
**9.1 - Appendix A - Account Creation Process**

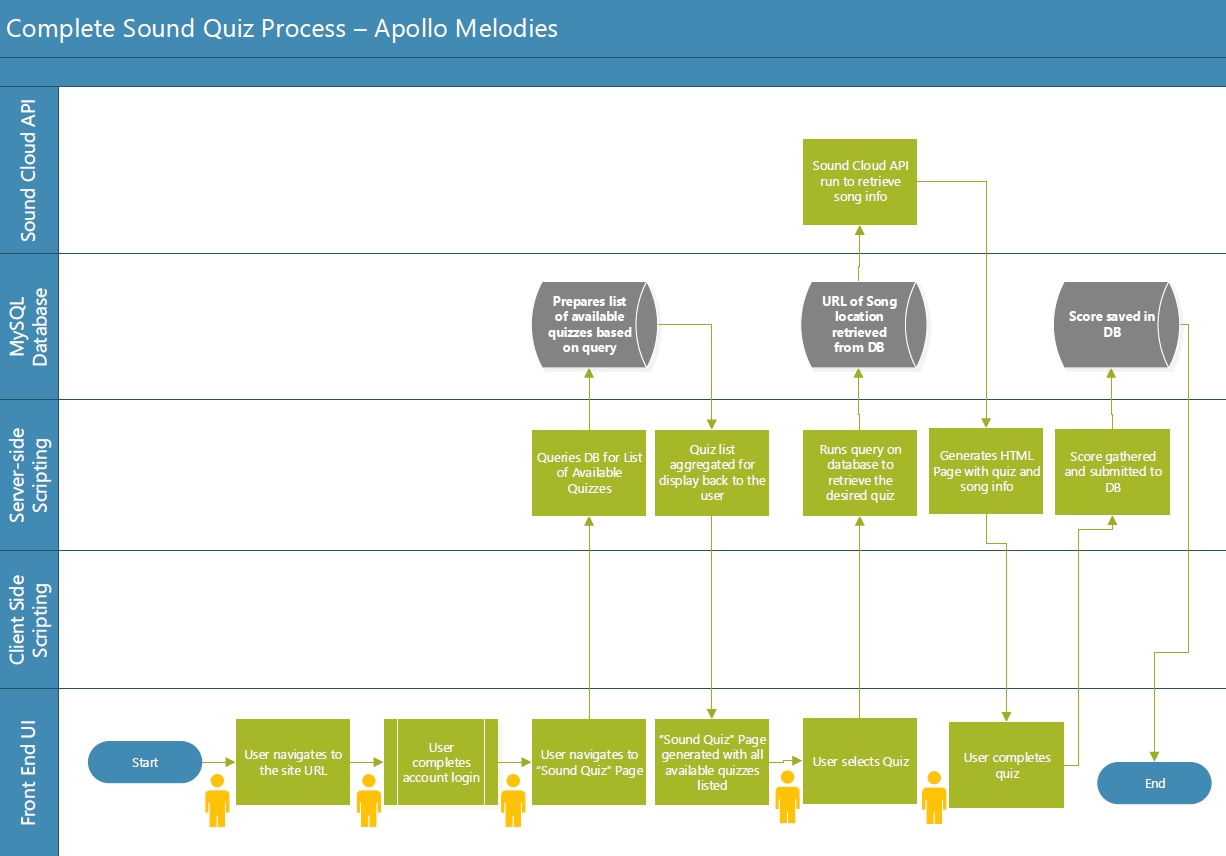
**

**9.2 - Appendix B - Account Login Process**

**

**9.3 - Appendix C - Quiz Creation Process**

**

**9.4 - Appendix D - Quiz Completion Process**

10. Key Risks

**10.1 Skills Risk**

Many members of our team do not have much experience working with the backend such as with stored procedures and the database. This issue is being addressed by our team leader who has a lot of experience in this subject taking charge of the backend. Those who are helping out with the backend will learn from that team member how the backend works. There are also members of our team that have no experience with frontend web development. This issue is being addressed by those of us who have little to no experience in web development by using the internet as a resource to learn about web development.

**10.2 Schedule risks**

Some team members are taking a full course load as well as working full time jobs. Some have as many as 4 group projects to complete for their classes as the semester comes to an end. This leaves very little time for development of a full stack web service. We believe that it’s possible to complete, but it’s going to be a very close call.

**10.3 Technical Risks**

We are unsure what the exact limitations and boundaries are of our system due to the LAMP server that the university provides. However, we believe that it will be able to handle the small amount of resources that our project requires as long as there isn’t a huge influx of users.

We are going to conduct testing on the software so that we find bugs and can determine what exactly the boundaries are of our software. We will come up with test cases for our system to determine the presence of bugs and ensure alignment with requirements.

**10.4 Teamwork Risks**

Fortunately this team works very well together and regularly meets to keep on schedule and divide tasks. There are very few teamwork risks.

**10.5 Legal risks**

Initially we were worried about music licensing for our quizzes, however we are going to be using Soundcloud embedded widgets and/or the Soundcloud API, which Soundcloud allows to be used on other websites without licensing. Since none of the music is hosted by us on our website, we don’t incur any legal risks.