Po-Do

Project Team: Yannis Panou,

Mecarah Son, David Amaya

Table of Contents

[**1. Project Definition (200 - 300 words) 3**](#_vy40458qcs8)

[**2. Project Requirements Analysis 4**](#_6i6ad34rzrhv)

[**3. Project Specification 5**](#_wymty2tcqvp8)

[**4. System – Design Perspective 6**](#_o2ypudtf76yf)

[**5. System – Analysis Perspective 10**](#_xx2g79ryv0o8)

[**6. Project Scrum Report 13**](#_6hje8h7f6ubv)

[**7. Subsystems 16**](#_jujnctfvph61)

[7.1 Authentication 16](#_euj5wfz5eogm)

[7.2 Database 18](#_3018xujkylta)

[7.3 Music Player 20](#_azjaraao2cui)

[7.4 Timer 23](#_wvcdkg8h7vgn)

[7.5 Calendar 25](#_dfn74rrij0u9)

[7.6 To-Do List 29](#_8bryh24j3whm)

[**8. Complete System 32**](#_6badixmpzan0)

#### **1. Project Definition (**200 - 300 words**)**

* Why
  + We believe that many people these days struggle with concentration and being able to study. Especially now where we live in a time where working online has become a norm it's easy to get distracted in accomplishing the task you set out to do. There are plenty of study websites already available on the internet but we believe that they are all missing something that can better help people achieve their goals and finish their tasks.
* What
  + We want to create a Pomodoro study website that allows people to be able to set timers for work time and breaks, set and create tasks, add deadlines and due dates on a calendar and listen to their favorite music or playlist all in one place. Pomodoro is basically a type of management system that allows the person to do one task for a set amount of time. After that time period is up they can either switch to another task or take a break. This technique allows users to not get burnt out while doing a task and the website that we will create will ensure that the task is completed.
* How
  + By creating a six week schedule and forming teams of two, we will create a fully functional website that will help the users finish the tasks that they want. Two people will work on the front-end development and two will work on the back-end. Finishing deadlines for certain tasks will be our priority to getting a fully functional website up and running.

#### 2. Project Requirements Analysis

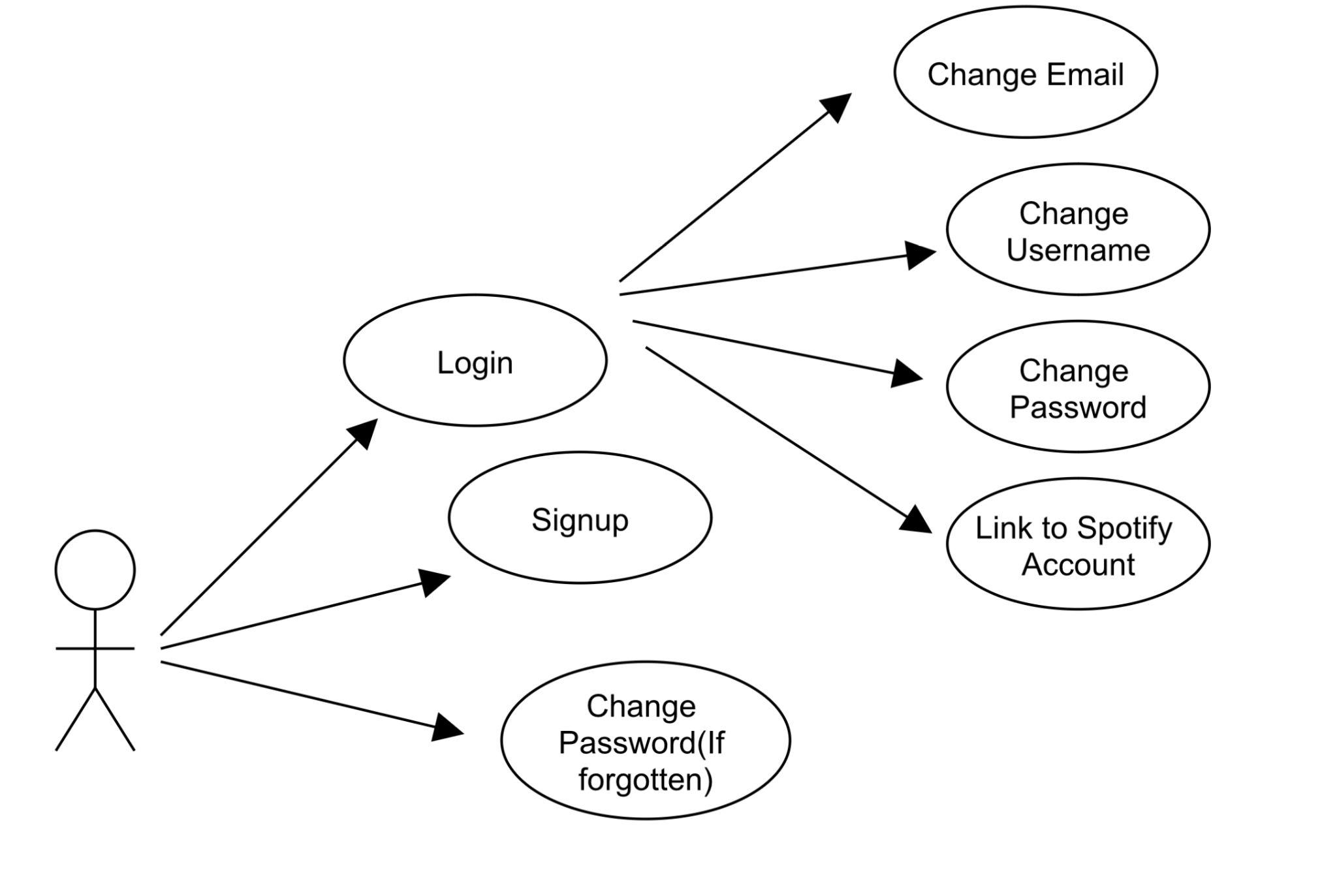
* Functional
  + Signing Up and Logging In
  + Setting Study Session Time
  + Setting Break Periods
  + Creating Tasks
  + Deleting Tasks
  + Setting Deadlines
  + Updating Time Periods
  + Notifications for Time Periods
  + Announcements for Deadlines
  + Music Overlay
  + Calendar of Tasks/Deadlines
  + Dropdowns/Pop-outs/UX
  + Disallow “non-members” access to user settings and music
* Usability
  + User interface
    - The user interface should be intuitive to allow for maximum productivity. It will have a simplistic design to limit distractions and optimize organization.
  + Performance
* System
  + Hardware
  + Software
    - Javascript
    - CSS
    - HTML
    - Nodejs
    - NoSQL
    - React
  + Database
    - Stores Users' music accounts
    - Store Username, Password, and User Details
* Security
  + Spotify AuthO2
  + Token

#### 3. Project Specification

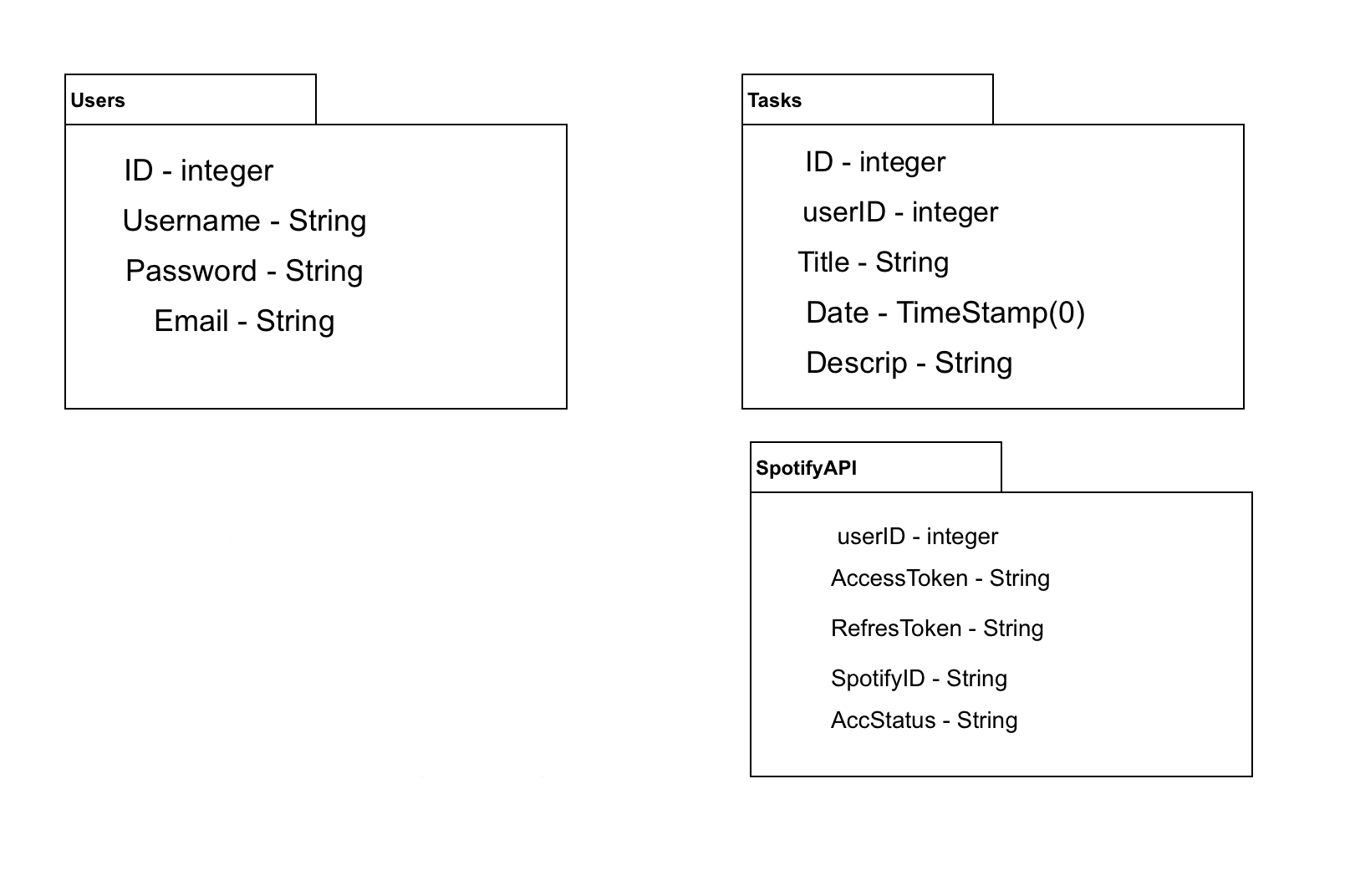
* Focus / Domain / Area
  + Studying / Productivity
* Libraries / Frameworks / Development Environment
  + Oracle
  + ExpressJS
  + Nodejs
* Platform (Mobile, Desktop, Gaming, Etc)
  + Desktop, laptop
* Genre (Game, Application, etc)
  + Web Application
* Teamwork division
  + David
    - Connectivity of Task to users and guest
    - Signup/Login Authentications
    - Settings
    - Spotify Player
  + Yannis
    - Final Calendar
    - Task
    - Timer
  + Mecarah
    - Styling
    - Initial Calendar
    - Light/Dark Mode

#### 4. System – Design Perspective

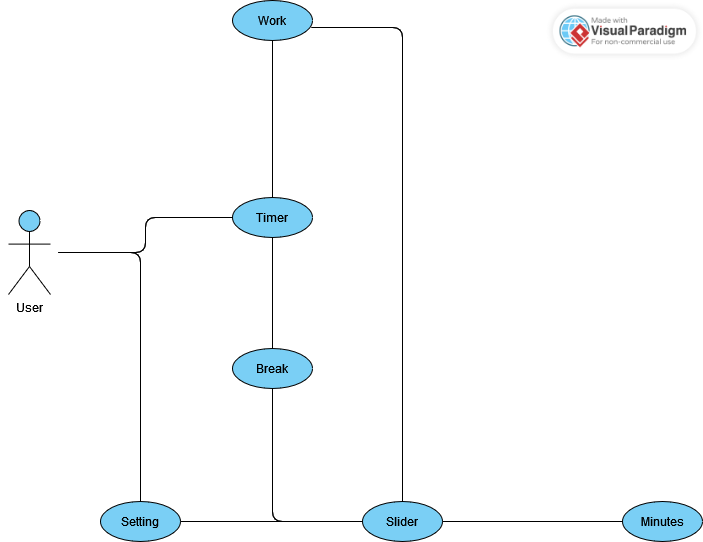
* Identify subsystems – design point of view
  + Authentication

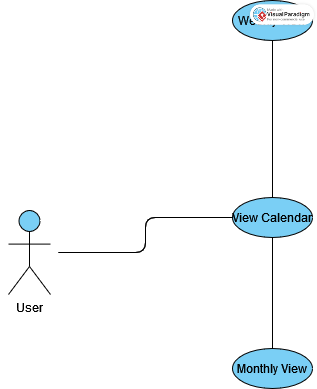


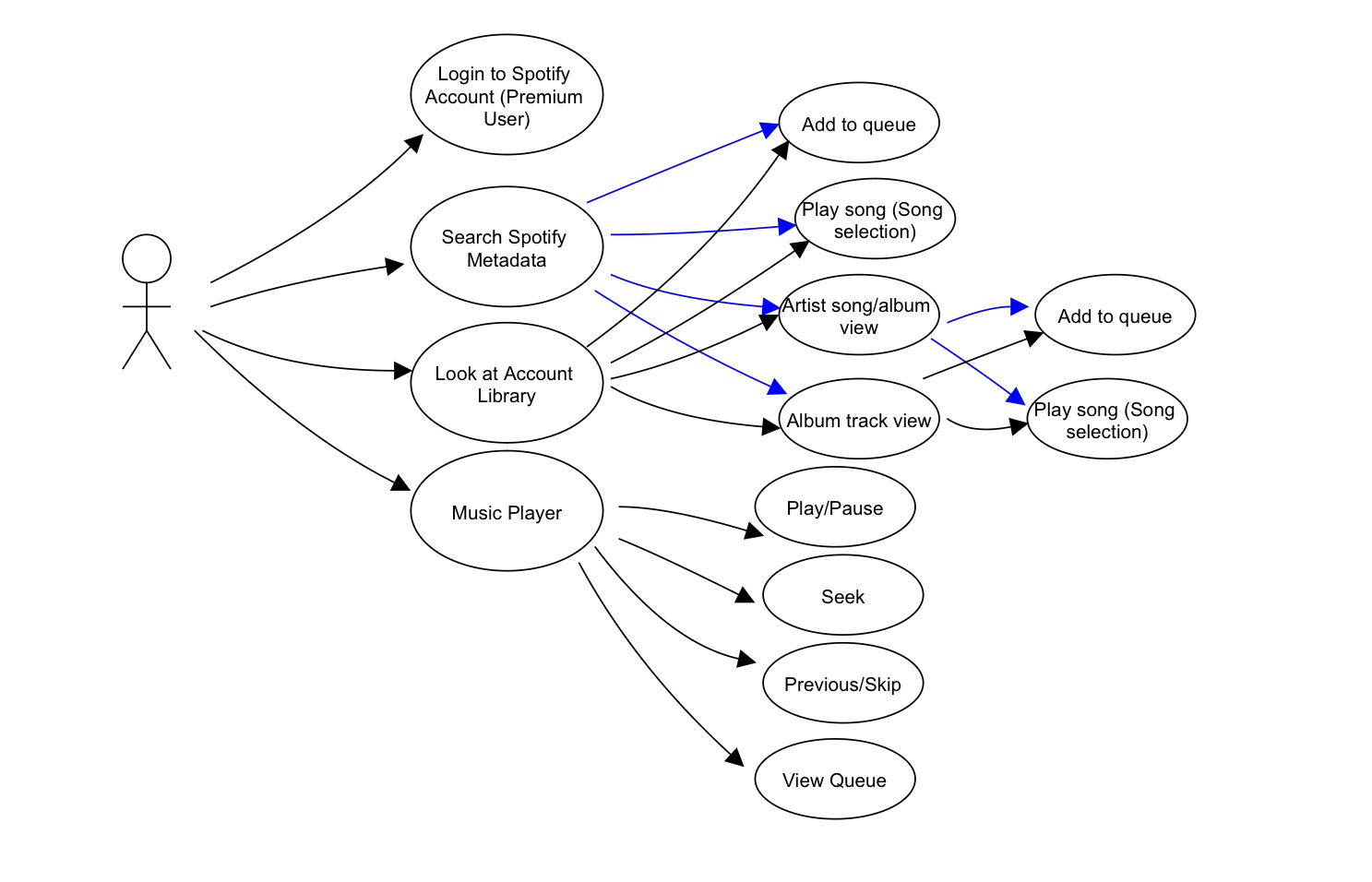
* + Database

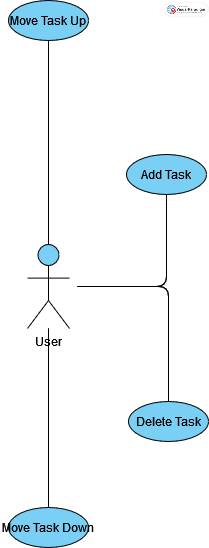


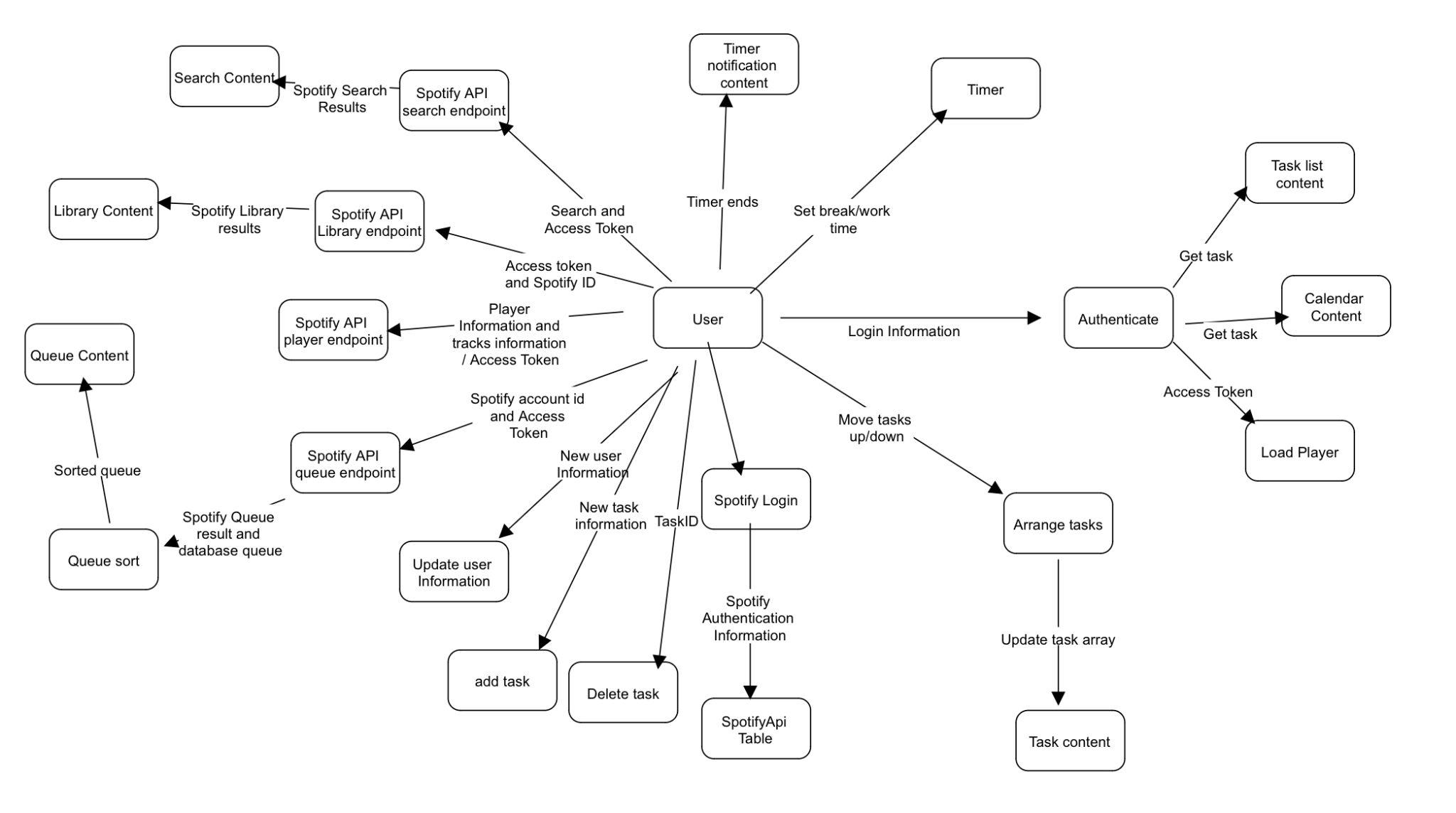
* + Timer



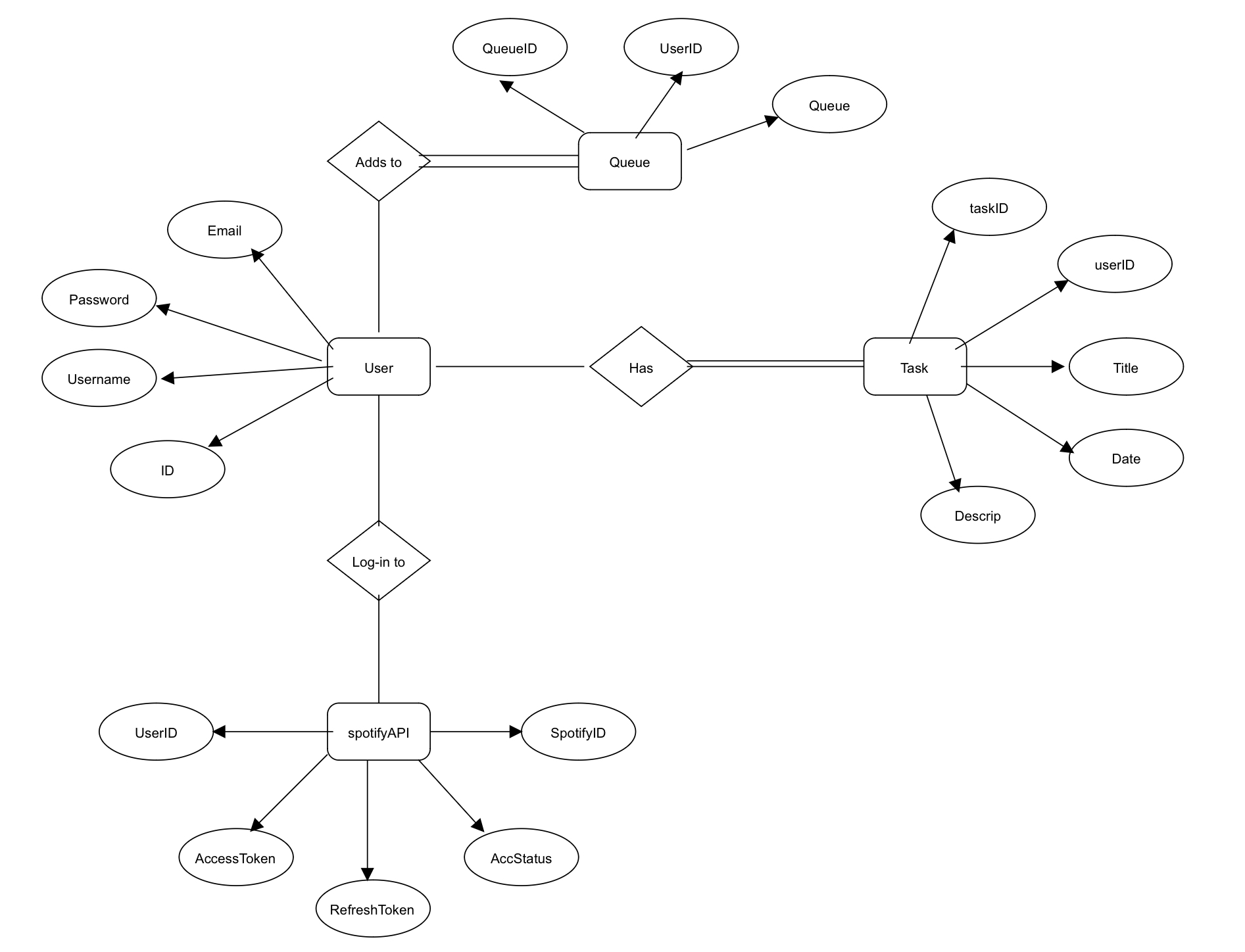
* + Calendar
    - 
  + Music Player



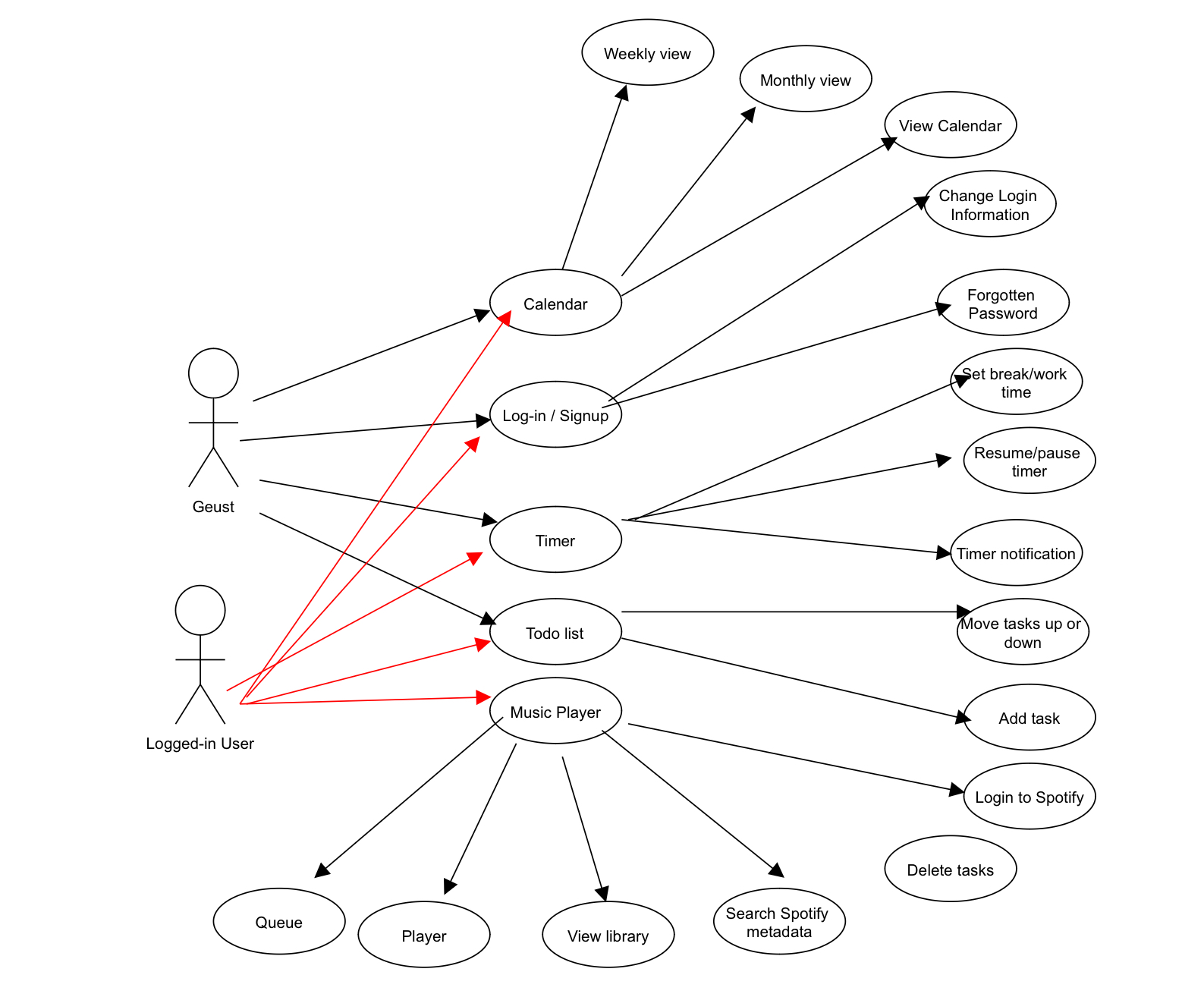
* + To-Do list
    - 
* Sub-System Communication (Diagram and Description)



* Entity Relationship Model (E-R Model)

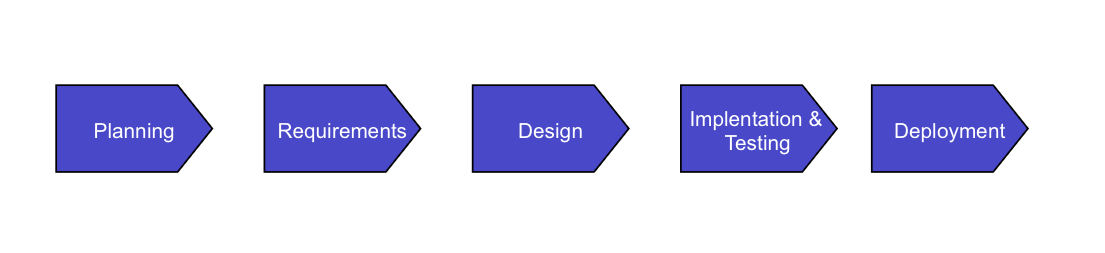


* Overall operation - System Model
  + Simplified Sub-system to System interaction



#### 5. System – Analysis Perspective

* Identify subsystems – analysis point of view
  + Authentication
  + Database
  + Timer
  + To-Do List
  + Calendar
  + Music Player
* System (Tables and Description)
  + Data analysis
    - Data dictionary
      * [Authentication](#gzfgdhxxelt6)
      * [Database](#tdkvupwkh8za)
      * [Timer](#benk42w1a2y2)
      * [To-Do List](#syqxotpmrrnz)
      * [Calendar](#ojxyvwh1qnjx)
      * [Music Player](#xoczew5d0tk)
  + Process models



* Algorithm Analysis
  + Authentication
    - Forms toggles O(1)
    - Forms Submission O(n)
    - Registration/login O(n)
    - Update User O(n)
  + Database O(n)
  + Timer
    - Slider O(n)
    - Countdown O(n)
    - Break/Work switch O(1)
    - Render O(1)
  + To-Do List
    - Tasks up/down O(n)
    - adding/deleting task O(n)
    - Get tasks O(n)
    - State management O(1)
    - Local storage task and sort O(log n)
    - Render O(1)
  + Calendar
    - Get tasks O(n)
    - adding task O(n)
    - State management O(1)
    - Local storage tasks O(n)
    - Date manipulations O(log n)
    - Forms O(1)
    - Event handling O(1)
    - Render O(1)
  + Music Player
    - Token management O(1)
    - Spotify SDK Player O(1)
    - Get player Information O(1)
    - Playback O(1)
    - Player state and Track Information O(1)
    - Volume control O(1)
    - Seeking O(1)
    - Get library O(1)
    - Get Search O(1)
    - Render search and Library O(n)
    - Renders O(n)

#### 6. Project Scrum Report

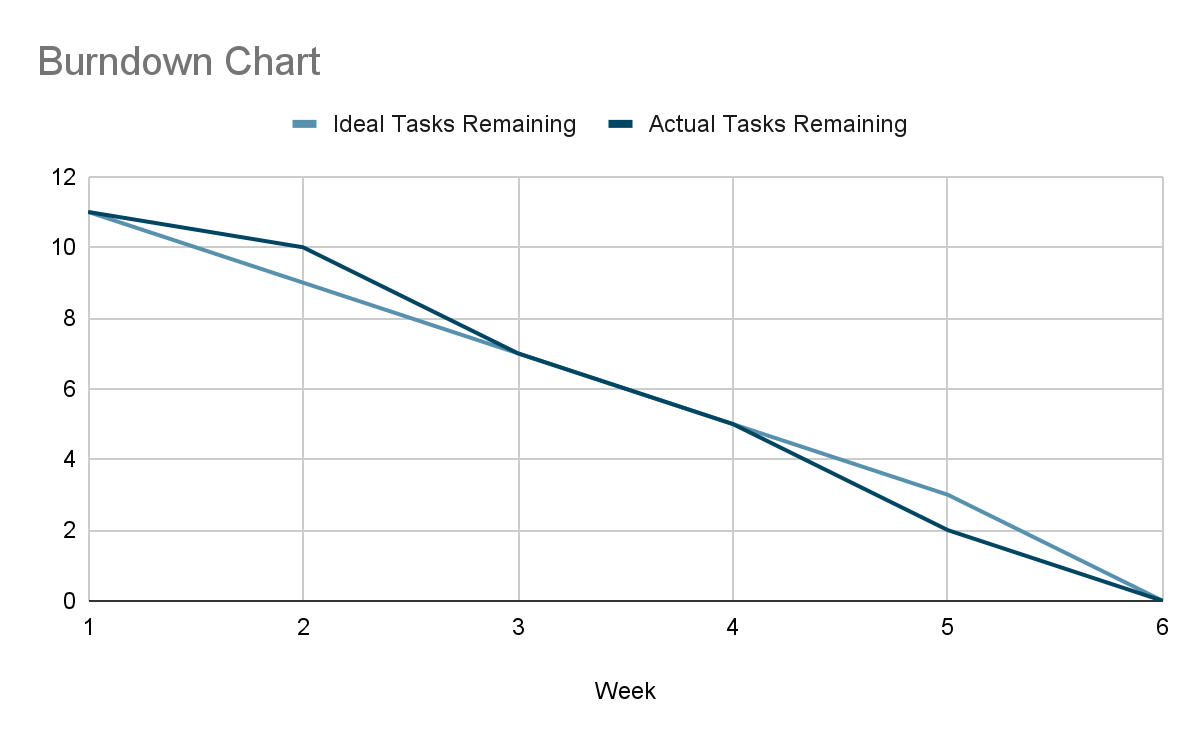
* Product Backlog (Table / Diagram)

| ID | Task Name | Description | Priority | Assigned To |
| --- | --- | --- | --- | --- |
| 1 | User Registration and Login | Allow users to create accounts and log in. | High | David |
| 2 | User Authentication | Implement user authentication logic | High | David |
| 3 | Study Session Time and Break Period Settings | Allow users to set study session times and break periods | High | Yannis |
| 4 | Task Creation and Deletion | Allow users to create new tasks with title, description, and date | High | Yannis |
| 5 | Task Prioritization | Allow users to give priority to different tasks. | Medium | Yannis |
| 6 | Task Calendar | Display a calendar of tasks with weekly or monthly view | Medium | Yannis |
| 7 | Forgot Password Functionality | Implement a way for users to reset their password | High | David |
| 8 | Music Styling | Implement UI for music controls | Low | David |
| 9 | Music Integration | Allow users to play music | High | David |
| 10 | Light and Dark Mode | Implement light and dark mode choices | Medium | Mecarah |
| 11 | Page Layout and Navigation | Design the page layout and navigation | Medium | Mecarah |
| 12 | Buttons Styling | Style buttons with consistency | Low | Mecarah |

* Sprint Backlog (Table / Diagram)

| ID | Task | Assigned To | Week Completed |
| --- | --- | --- | --- |
| 1 | User Registration and Login | David | 3 |
| 2 | User Authentication | David | 5 |
| 3 | Study Session Time and Break Period Settings | Yannis | 5 |
| 4 | Task Creation and Deletion | Yannis | 3 |
| 5 | Task Prioritization | Yannis | 4 |
| 6 | Task Calendar | Yannis | 6 |
| 7 | Forgot Password Functionality | David | 3 |
| 8 | Music Styling | David | 6 |
| 9 | Music Integration | David | 5 |
| 10 | Light and Dark Mode | Mecarah | 4 |
| 11 | Page Layout and Navigation | Mecarah | 1 |
| 12 | Buttons Styling | Mecarah | 2 |

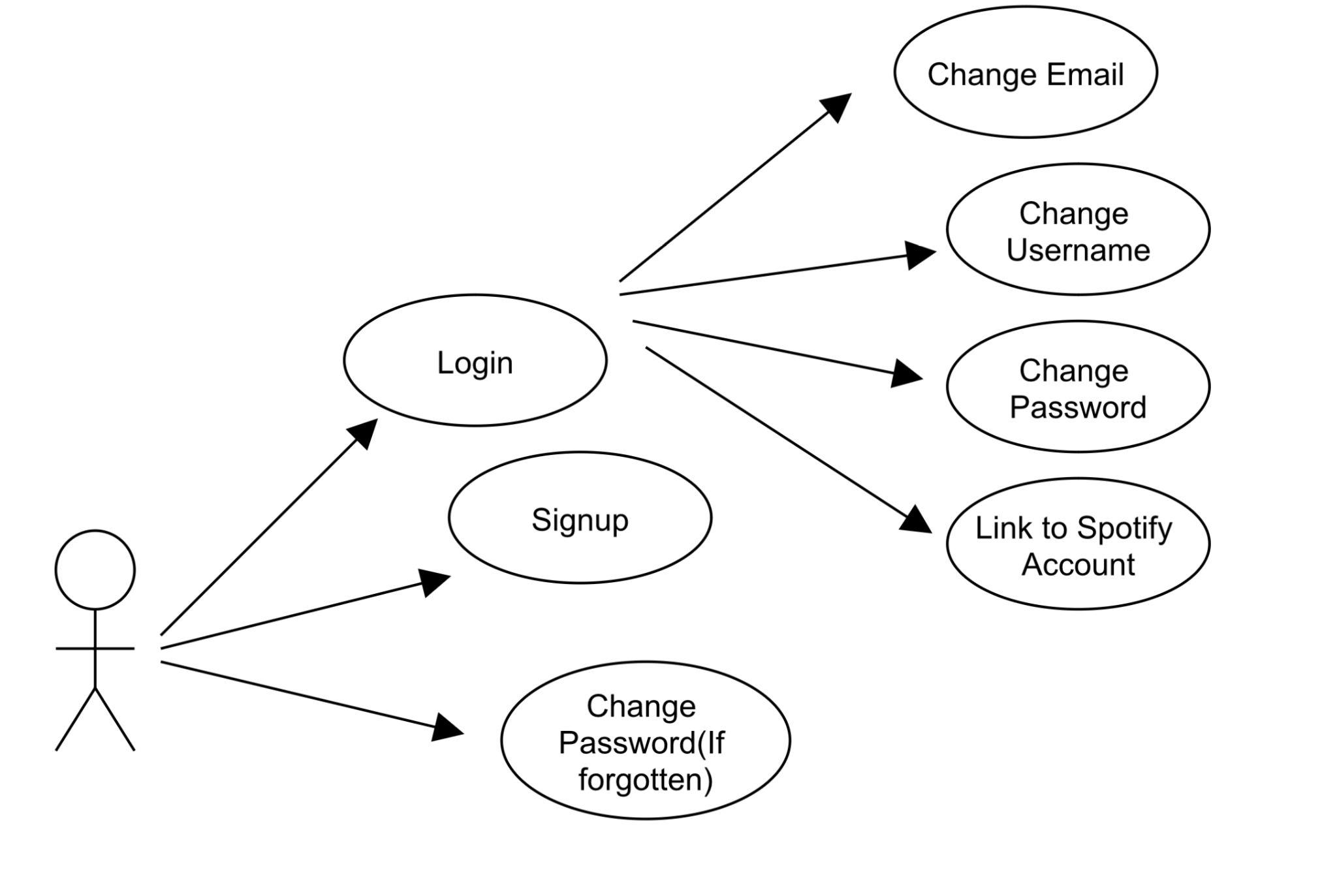
* Burndown Chart



#### **7. Subsystems**

##### 7.1 Authentication

* Initial design and model
  + Use-Case



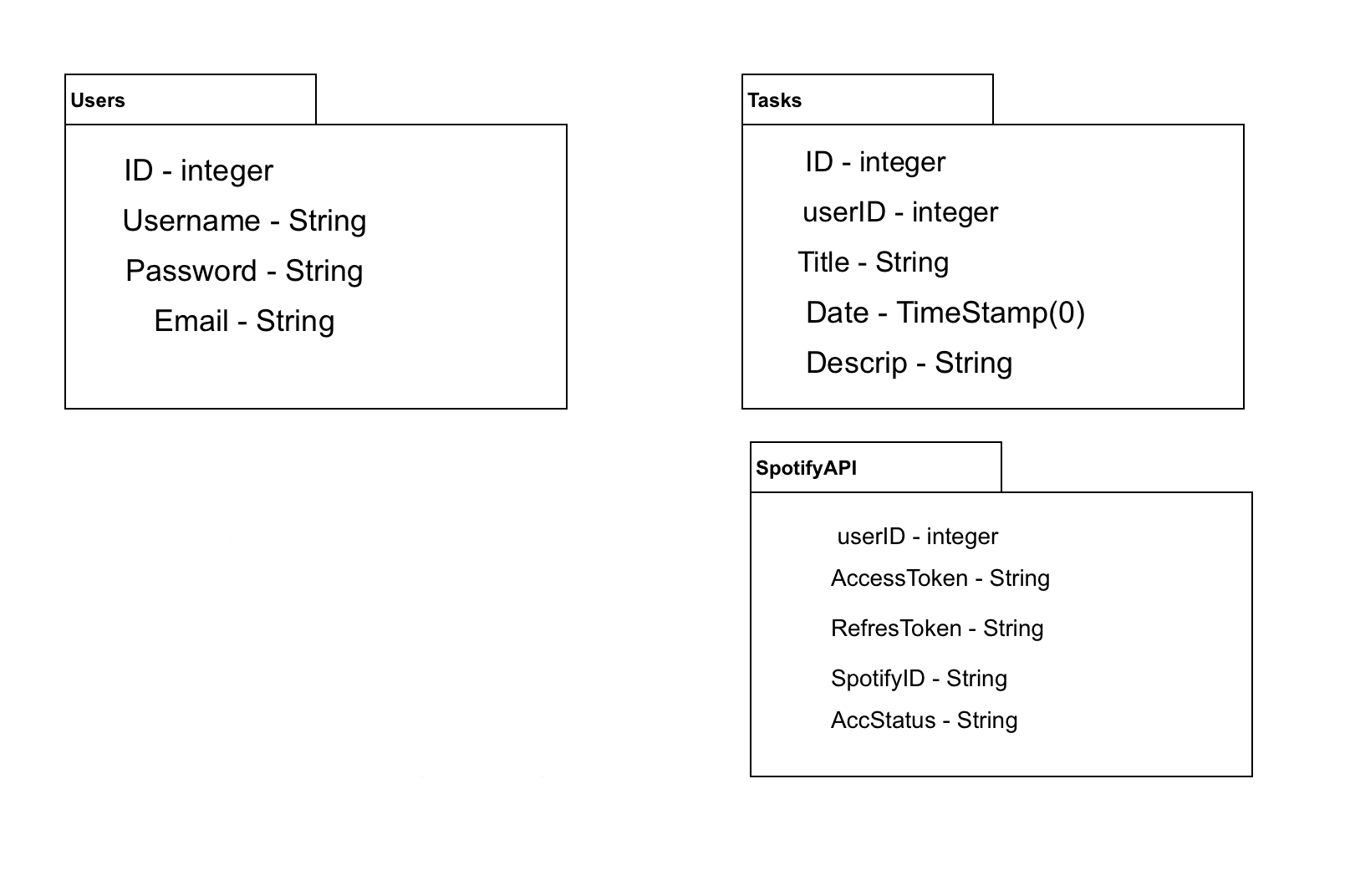
* + Design choices
    - Login
      * JSON Web Token(JWT) to create a user login session
      * Store the JWT in cookies for the login session until the user logs out
    - Signup
      * Check if the email address and username exist in the database
      * Hash new user password and store to database
    - Forgotten Password
      * Change the user password by emailing the user allowing them to change the password
    - Change Email
      * Logged-in user can change their email address to a non-existing email address
    - Change Username
      * Logged-in user can change their username to a non-existing username
    - Change Password
      * Logged-in user can change their password if they know their old password
    - Link to Spotify Account
      * Allow user to link their Premium Spotify account to their account for music player accessibility
* Data dictionary

| Field | Type | Descriptions |
| --- | --- | --- |
| Id/loginStatusID/userID | integer | User ID in the database table |
| Username | String | User username in the database table |
| Password/hashPass | String | User password in the database table |
| Email | String | User email in the database table |
| Token | String | Generated token for users' login session |
| Auth | Boolean | To verify user Logged-in and to load their content |
| spotifyAccessToken | String | User Spotify Account Access token |
| spotifyRefreshToken | String | User Spotify Account Refresh token |

* [Scrum Backlog (Product and Sprint - Link to Section 6)](#_6hje8h7f6ubv)
* Coding
  + Approach
    - Object-Oriented Programming and Functional
  + Language
    - JavaScript
* User training
  + Training / User manual (needed for final report)
* Testing
  + Integration Testing

##### 7.2 Database

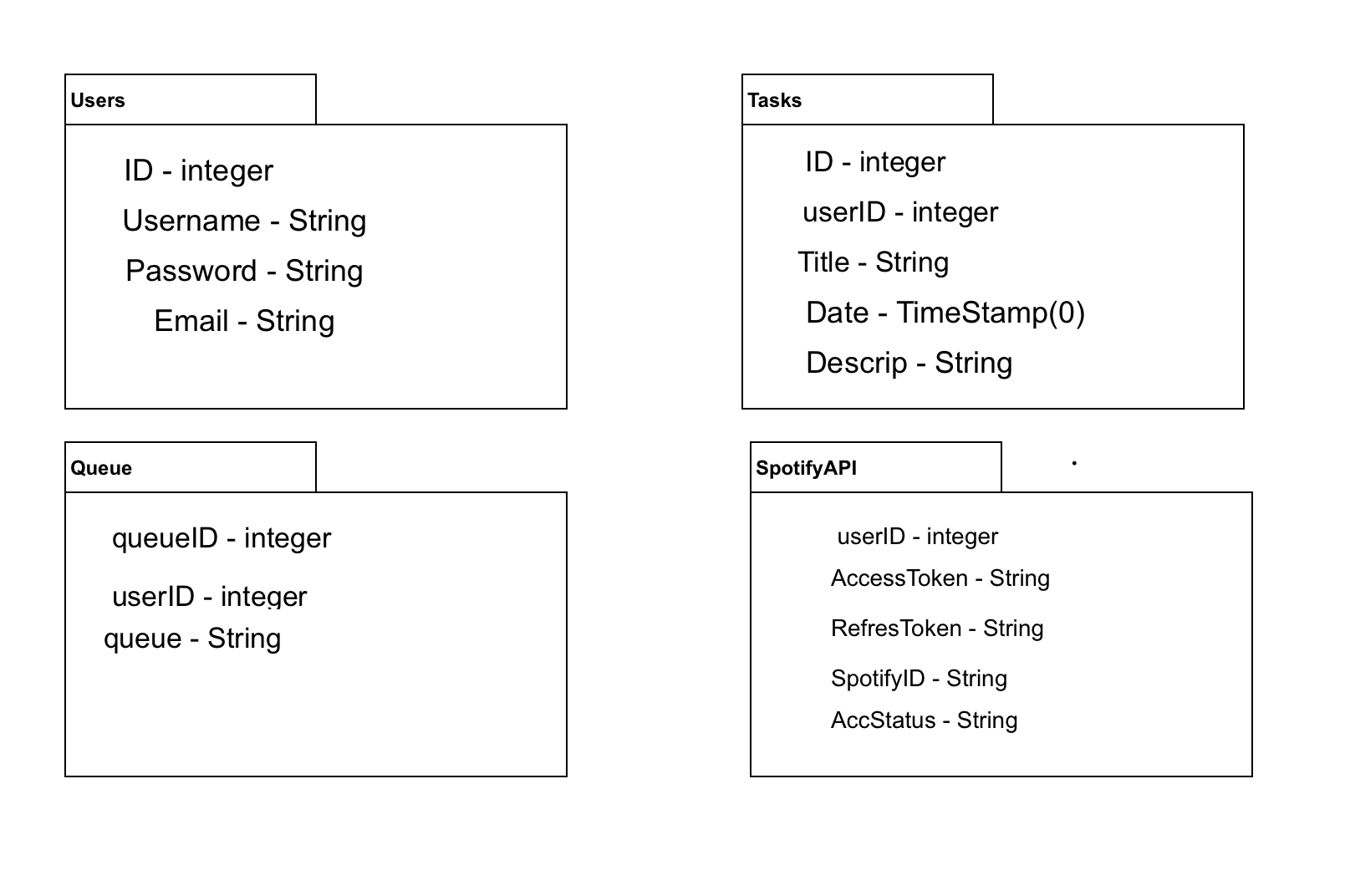
* Initial design and mode



* Data dictionary

| Field | Type | Definitions |
| --- | --- | --- |
| ID | Integer | User Identity integer(PK) |
| username | String | User unique username |
| password | String | Users password |
| email | String | Users unique email |
| taskID | Integer | Task identity for any CRUD options(PK) |
| userID | Integer | Keeps the user information aligned(FK) |
| title | String | Task title |
| date | Timestamp(0) | Date of the task |
| decrip | String | Tasks descriptions |
| Access Token | String | Spotify account access token to allow access |
| Refresh Token | String | Spotify account token to refresh the access token |
| SpotifyID | String | Spotify username |
| AccStatus | String | Free/Premium Spotify User |
| queueID | Integer | Queue position (PK) |
| queue | String | Spotify track URI |

* If refined
  + Reason for refinement
    - Pros
      * The user queue is not being shown 10 tracks of the same track they played at first
      * Shows user queue properly
    - Cons
      * Sometimes the user queue is not deleted from the table correctly
  + Changes from the initial model

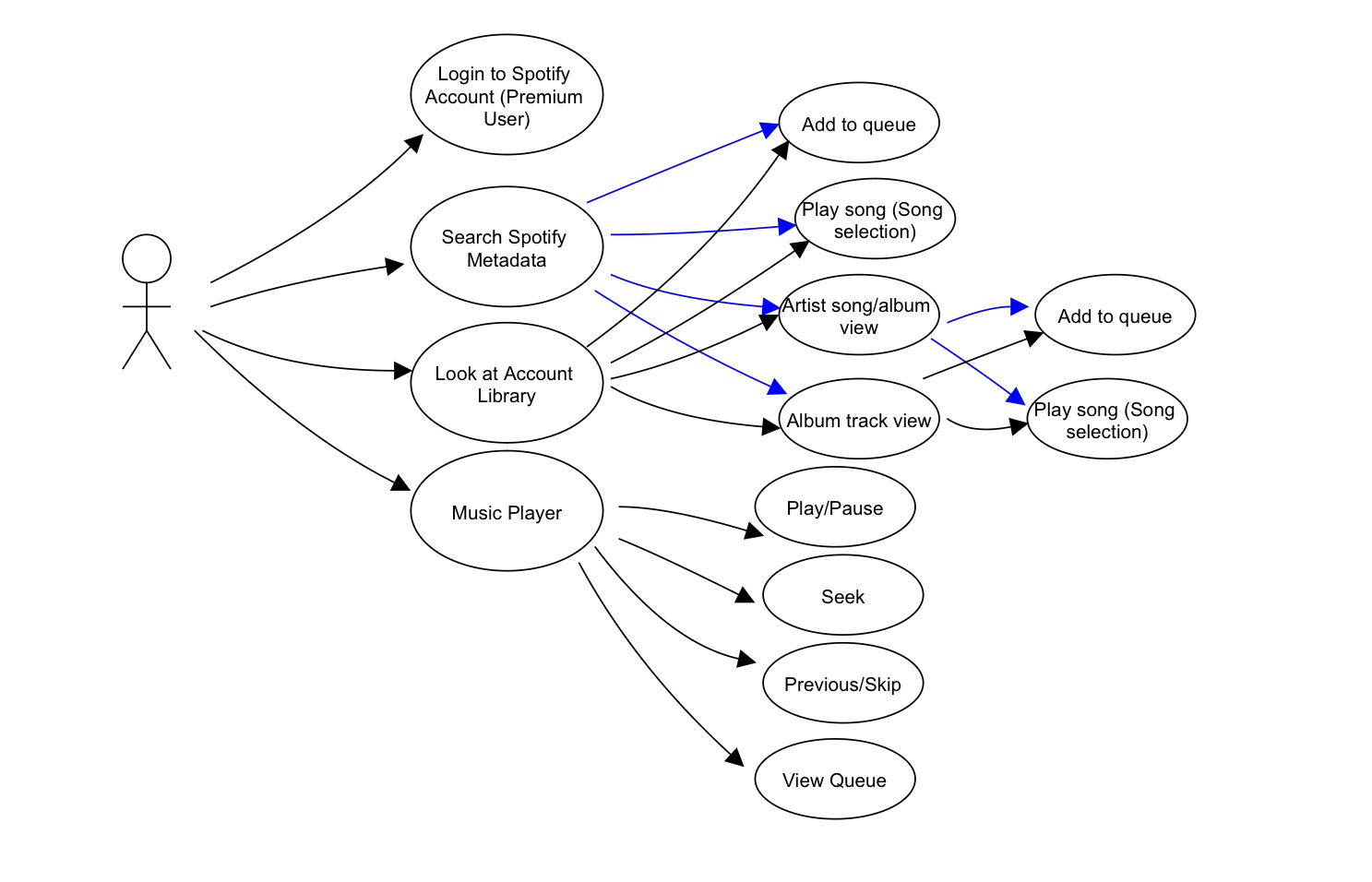


* + Refined model analysis
  + Refined design (Diagram and Description)
* [Scrum Backlog (Product and Sprint - Link to Section 6)](#_6hje8h7f6ubv)
* Coding
  + Approach
    - Object-Oriented Programming and Functional
  + Language
    - JavaScript and Oracle NoSQL
* User training
  + Training / User manual (needed for final report)
* Testing
  + Integration Testing

##### 

##### 7.3 Music Player

* Initial design and model



* + Design choices
    - Log in to Spotify Account
      * Allow user to login to their Spotify premium Account
    - Search Spotify Metadata
      * show/hide menu
      * Search Spotify for albums, artists, and tracks
      * Returns Search results
        + Allow users to select albums, artists, and tracks
        + Album selection shows its tracks
        + Artists selection shows its albums and top tracks
        + Track selection plays a track
      * Allow any track to be added to the queue
    - Spotify Account Library
      * Returns user Spotify Account Library
        + show/hide menu
        + Allow users to select albums, artists, and tracks
        + Album selection shows its tracks
        + Artists selection shows its albums and top tracks
        + Track selection plays a track
    - Music Player
      * Play/Pause tracks
      * Skip the track and go back to the Previous track
      * Seek Track
    - Queue
      * show/hide queue
      * Output user queue

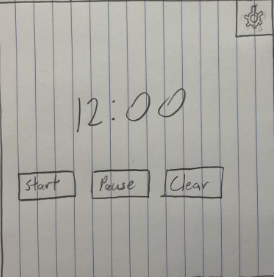
* Data dictionary

| Field | Type | Definitions |
| --- | --- | --- |
| spotifyAccessToken/token | String | User Spotify account access Token |
| spotifyRefreshToken | String | User Spotify account refresh Token |
| spotID | String | User Spotify account username |
| loading | Boolean | Turns false when sdk player is created |
| device\_id/playerID | Integer | Created Player ID |
| player/deviceInfo | Object Array | Created Player information |
| userID | Object Array | Logged-in user ID |
| showQueue | Boolean | Shows queue based on user |
| trackInfo/songInfo | Object Array | Track information on Spotify |
| isPlaying | Boolean | Set true if song is playing or false if not |
| songPlaying | Object Array | Playing track information in Spotify |
| qIndex | Integer | Queue index for database and to sort the queue based on the value |
| volume | Integer | Set the player volume |
| stillLoading | Boolean | Stays loading until the sdk player is fully transferred |
| songName | String | Playing song Name |
| albumImage | String | Album image URL link |
| songArtist | String | Artist name of the song |
| showSideMenu | Boolean | If true show side menu or false hide side menu |
| selectedTrack | Object Array | Hold information of the user track selection from the library or search |
| seekPos | Integer | Seek player of the current playing track |

* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
  + Changes from initial model
  + Refined model analysis
  + Refined design (Diagram and Description)
* [Scrum Backlog (Product and Sprint - Link to Section 6)](#_6hje8h7f6ubv)
* Coding
  + Approach (Functional, OOP)
  + Language
* User training
  + Training / User manual (needed for final report)
* Testing

##### 7.4 Timer

* Initial design and model
  + Drawn Diagram



* + Design choices

Wheel

* + - * Work Timer
        + Color is Green

As time depreciates so does the color of the wheel until Gray.

* + - * Break Timer
        + Color is Red

As time depreciates so does the color of the wheel until Gray.

Setting

* + - * Work Slider
        + Move to desired minutes
      * Break Slider
        + Move to desired minutes
* Data Dictionary

| Fields | Type | Definitions |
| --- | --- | --- |
| settingsInfo | Object | Contains timer settings informations |
| isPaused | Boolean | True if pause and false if timer is going |
| mode | String | Shows the timer modes work/break |
| secondsLeft | Integer | Seconds left in timer |
| isPausedRef | Object | Object to reference boolean for isPaused |
| modeRef | Object | Object to reference string for mode |
| secondsLeftRef | Object | Object to reference integer for secondsLeft |

* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)

Pros:

* + - * More Interactive
      * Visually appealing to Users

Cons:

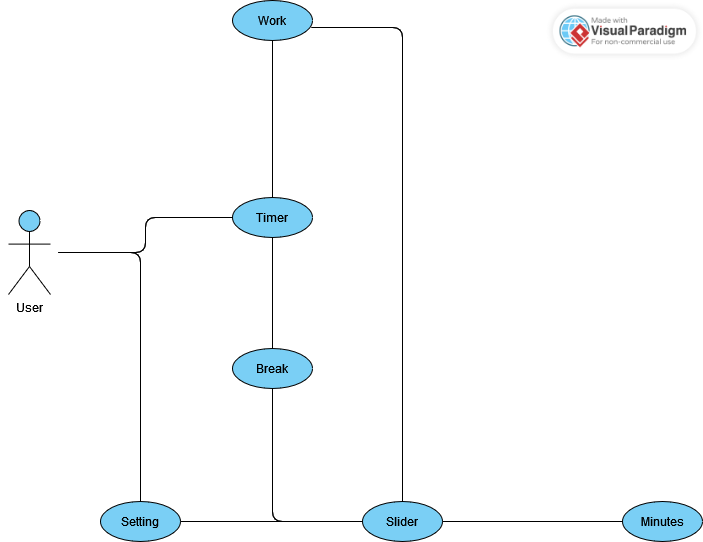
* + - * Depending on the mode(dark or light) you are in the colors of the timer could be difficult to see initially.
  + Changes from initial model

Timer Wheel

Depending on if you are in Work or Break mode the color of the timer is different.

Settings button that has two sliders to choose time for each Work and Break time.

* + Refined design (Diagram and Description)



* [Scrum Backlog (Product and Sprint - Link to Section 6)](#_6hje8h7f6ubv)
* Coding
  + Approach (Functional, OOP)

Functional

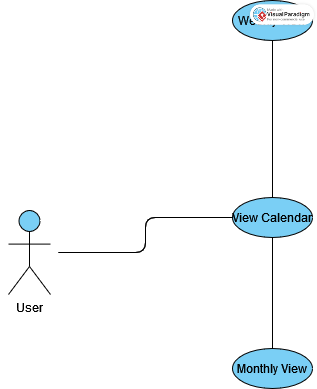
* + Language

JavaScript

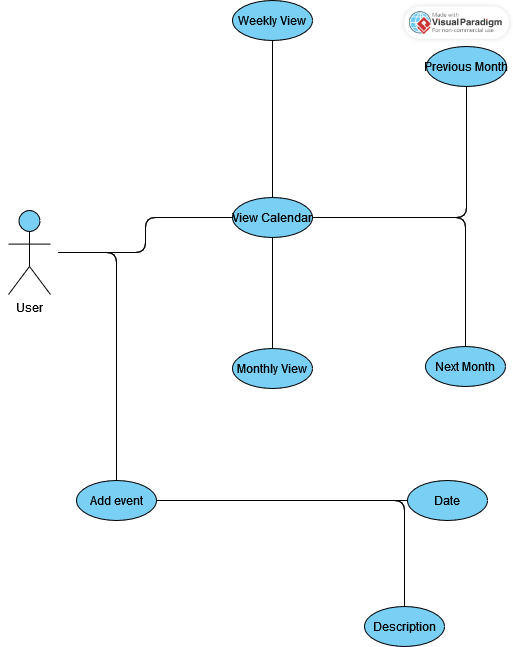
CSS

* User training
  + Training / User manual (needed for final report)
* Testing
  + Integration

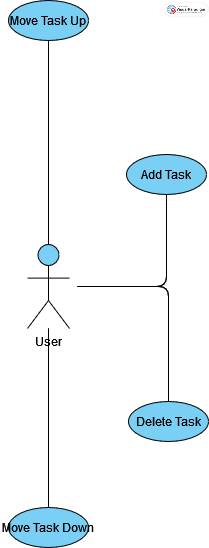
##### 7.5 Calendar

* Initial design and model
  + Use-case
    - 
  + Design choices
    - View
      * Weekly
      * Monthly
    - Month
      * Previous
      * Next
    - Event
      * Add event
        + Date
        + Description
      * Can look at the day of the event and see the event is there based on user input of adding the event.
    - Local storage guest Task
      * Save guest users task to local storage and delete after the time limit for the guest has been reached
* Data dictionary

| Field | Type | Description |
| --- | --- | --- |
| TaskID | Integer | Every task is assigned a unique ID to accurately monitor and manage the task within the database. |
| UserID | Integer | Every user is assigned a distinct ID that can be associated with different fields to effectively track their account in connection with the database. |
| Title/eventName | String | The name of the event. |
| Date/eventDate | TIMESTAMP(0) | Select the date on which the event is scheduled to appear on the calendar. |
| Descrip/eventDescription | String | Users can provide a description of their activities for the event. |
| Time | TIMESTAMP(0) | The date and time are combined and then entered into the database. |
| showEventDescription | Boolean | True show description, and false hide |
| showEventModal | Boolean | True show event modal, and false hide |
| showFullMonth | Boolean | True show month, and false hide |
| currentDate | Date | Used to create moment for the calendar |

* If refined (changed over the course of project)
  + Reason for refinement (Pro versus Con)
    - Pros:
      * Allowed to see previous and future months
      * Allowed for addition of events to the calendar
  + Changes from initial model
    - Previous and Future Months view
    - Events
    - Mini Calendar to choose the Day
  + Refined design (Diagram and Description)
    - 
* [Scrum Backlog (Product and Sprint - Link to Section 6)](#_6hje8h7f6ubv)
* Coding
  + Approach (Functional, OOP)
    - Functional
  + Language
    - JavaScript
    - CSS
* User training
  + Training / User manual (needed for final report)
* Testing
  + Integration

##### 7.6 To-Do List

* Initial design and model
  + Use Case
    - 
  + Design choices
    - Task
      * Add Task
      * Delete Task
      * Move Task Up
      * Move Task Down
    - Local storage guest Task
      * Save guest users task to local storage and delete after the time limit for the guest has been reached

* Data dictionary

| Field | Type | Description |
| --- | --- | --- |
| TaskID | Integer | Every task is assigned a unique ID to accurately monitor and manage the task within the database. |
| UserID | Integer | Every user is assigned a distinct ID that can be associated with different fields to effectively track their account in connection with the database. |
| Title/newTask | String | The name of the event. |
| Date | TIMESTAMP(0) | Select the date on which the event is scheduled to appear on the calendar. |
| Descrip | String | Users can provide a description of their activities for the event. |
| newTime | TIMESTAMP | The newTaskDate and newTime are combined and then entered into the database. |
| newTaskDate | Date | The newTaskDate and newTime are combined and then entered into the database. |
| TIME\_LIMIT | Integer | Time limit to delete guest task in local storage |
| CHECK\_INTERVAL | Integer | Check the interval of the guest task in local storage to remove after time limit is reach |

* [Scrum Backlog (Product and Sprint - Link to Section 6)](#_6hje8h7f6ubv)
* Coding
  + Approach (Functional, OOP)
    - Functional
  + Language
    - JavaScript
    - CSS
* User training
  + Training / User manual (needed for final report)
* Testing
  + Integration

#### 

#### 

#### 

#### 

#### 8. Complete System

* Final software/hardware product
  + http://129.213.68.135/
* Source code and demonstration video – screenshots as needed - Technical report
  + GitHub Link - <https://github.com/Briyannis/Po-Do>
  + <https://youtu.be/3m5ogaKPqgA>
* Evaluation by client and instructor
* Team Member Descriptions
  + Mecarah
    - Frontend
  + David
    - Backend
  + Yannis
    - Frontend