CS 30700

**MelodyMunk**

Team 24: Sprint 1 Retrospective

horizontal line



Matt Twardowski, Brooks Van Buren, Elizabeth Stallter, Tristan Pickett, Shrikar Peri, Avnish Bablani

Table of Contents

[**What went well?**](#_2s43f6gsz9gi) **3**

[User Story #1](#_omtulsoem8me) 3

[User Story #2](#_jolm8ubv7h4) 3

[User Story #3](#_jgmsdqsfg2p) 4

[User Story #4](#_4y1r6zh8jp8j) 5

[User Story #5](#_9lnwikyko0l) 6

[User Story #7](#_lkm49uqm5kci) 6

[User Story #8](#_iniqdmytfhj9) 7

[User Story #9](#_kgcz64ny5qyp) 8

[User Story #11](#_khoi02rbbo74) 9

[User Story #14](#_yk4m6hl0h1fg) 10

[**What did not go well?**](#_gjp3kv3xn821) **11**

[User Story #6](#_1c31cod66uk3) 11

[User Story #10](#_hdsdee3pmy88) 12

[User Story #12](#_up8xfjs2ak5l) 13

[User Story #13](#_ym60qwe6o2iy) 13

[User Story #15](#_dv1a3fk9y9je) 14

[**How should we improve?**](#_qs7gdxxomf8t) **15**

[Underestimation of hours needed for tasks](#_5af7grmj1qrx) 15

[Underestimation of time required for learning new technologies](#_r02b07glwptc) 16

[Resolution of conflicts for user stories given to multiple members](#_tkha19qwfit0) 16

[Better testing methods](#_h1ddd1mjj2g0) 16

[Better communication](#_9t98pi4tzxo) 17

[Git commit rate](#_b67fj9fhebi6) 17

[Accountability](#_txfhpksxeq2j) 17

[Informal meetings more often](#_ebyrdfsudgqz) 18

[Formalized plans on how object interactions will occur](#_73e386yj3the) 18

# What went well?

## User Story #1

As a user, I would like to see a landing page when I visit MelodyMunk, that gives me information about the webapp and see links to login and registration pages.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Create static homepage | 3 hours | Shrikar |
| **2.** | Link to other webpages | 2 hour | Shrikar |
| **4.** | UI for Error Handling | 2 hour | Shrikar |

## 

Complete: We have functioning UI for all of these tasks. The UI pages have been designed and connected to some backend components.

## 

## User Story #2

As a developer, I would like to set up an AWS instance and MySQL database to use in future development.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Setup development EC2 instance on AWS | 3 Hours | Matt |
| **2.** | Setup production EC2 instance on AWS | 3 Hours | Matt |
| **3.** | Install MySQL database on dev and prod instances | 2 Hours | Matt |
| **4.** | Set up SSH access for dev and prod instances | 2 Hours | Matt |
| **5.** | Allow outside connection to dev and prod MySQL databases | 3 Hours | Matt |

## 

Complete: We have all the instances set up in AWS. All of the needed permissions have been set up for these instances. All needed databases have also been initialized.

## 

## User Story #3

As a user, I would like to create an account on MelodyMunk.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Create User Entity | 3 Hours | Matt |
| **2.** | Create RegistrationController to handle user registration | 5 Hours | Matt |
| **3.** | Create Registration Page UI | 3 Hours | Elizabeth |

## 

Complete: Users can create their own MelodyMunk account and the user accounts are stored in the database.

## User Story #4

As a user, I would like to log into an account on MelodyMunk.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Create AuthController to handle the creation of new users | 3 Hours | Matt |
| **2.** | Create Login Page | 3 Hours | Matt |
| **3.** | Unit testing to make sure new user is created | 1 Hour | Elizabeth |
| **4.** | Implement login throttling for each user to prevent brute force | 3 Hours | Brooks |

## 

Complete: Users can login into the MelodyMunk account they previously created.

## 

## User Story #5

As a user, I would like to customize my account and link my Spotify account to it.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Create UserController to handle features that relate to the user | 5 Hours | Matt |
| **2.** | Create SpotifyController to handle requests to Spotify’s Third-Party API | 5 Hours | Matt |
| **3.** | Create UserAPIToken Entity | 3 Hours | Avnish |
| **4.** | Create User settings page UI | 4 Hours | Shrikar |
| **5.** | Implement ability to link Spotify account to a user | 4 Hours | Avnish |

Complete: Users can login into their Spotify account and make requests to the Spotify API. We are able to get refresh tokens so users don’t have to reauthenticate.

## User Story #7

As a user I would like to be able to search for songs.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Connect with Spotify API to be able to search for songs and make queries to search songs. | 5 hours | Avnish |
| **2.** | Create a UI to search through songs and allow users to select the correct songs. | 3 hours | Shrikar |
| **3.** | Play proper song from Spotify API by calling it to get the song. | 2 hours | Avnish |
| **4.** | Unit testing to verify song requested is correctly called and with the right song | 1 hour | Avnish |
| **5.** | Error handling for Spotify API | 3 hours | Avnish |

## 

Complete: Users can search Spotify for the items they are looking for by track, album and artist. If something goes wrong, errors are properly handled.

## User Story #8

As a host I would like to generate a QR code that can be displayed on a screen or printed to share with guests

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Create module to generate QR code for given URL | 3 hours | Brooks |
| **2.** | Allow party host to view QR code in web page | 2 hours | Brooks |
| **3.** | Make QR code easily printable | 1 hour | Brooks |
| **4.** | Add UI button to get sharable link or QR code | 3 hours | Brooks |

## 

Essentially Completed: A QR code in PNG format can be easily created by sending a request to a url with a room code. Ex. melodymunk.com/generateqr/roomcode will respond with a code to link to the room represented by “roomcode.” Although this functionality isn’t really connected in the UI, it would be an easy task as soon as the room UI is done.

## User Story #9

As a host, I would like to be able to specify a list of actions guests and admins have permission to take

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Create a module to specify a list of permissions associated with a room | 5 hours | Brooks |
| **2.** | Allow module to be queried against in order to check if an action is permitted | 2 hours | Brooks |
| **3.** | Interface module with database to save and retrieve preferences across sessions | 3 hours | Brooks |
| **4.** | Interface module with UI to allow host to specify permissions | 3 hours | Elizabeth |
| **5.** | Unit testing to ensure that permissions work properly | 1 hour | Avnish |

## 

Essentially Completed: This module is functioning properly and tested. The only thing missing is integration with the UI, because much of the UI work is missing and not connected with the backend.

## User Story #11

As a general user I would like to suggest songs.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Take song from Spotify API | 4 hours | Avnish |
| **2.** | Add song to suggested queue | 3 hours | Avnish |
| **3.** | Unit Test to verify that songs are properly suggested | 1 hours | Avnish |

## 

Completed: Users can find a song in the spotify and take that song and add it to a playlist.

## 

## User Story #14

As a user I would like a ranking system for voted on suggested songs in a room.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Allow users to vote on suggested songs | 5 hours | Tristan |
| **2.** | Store votes on songs in the room | 4 hours | Tristan |
| **3.** | Rank songs based off of number of upvotes, time passed since suggestions, and current rate of upvotes | 3 hours | Tristan |
| **4.** | Unit tests to make sure songs are properly rated and ranked | 1 hours | Tristan |

Completed: Users can vote on a song and the rank of the song is updated and stored in the database. The rank of the song is calculated from a variety of factors including number of upvotes, time passed since suggestions, and current rate of upvotes.

# What did not go well?

## User Story #6

As a party guest I want to be able to join a room.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Designing a room module that handles creating a room and updating attributes | 6 hours | Elizabeth |
| **2.** | Create module to find room identified by URL | 5 hours | Elizabeth |
| **3.** | Create UI to join and create rooms | 4 hours | Elizabeth |
| **4.** | Create UI for URL input. Design error message dialog box if input is invalid. | 4 hours | Shrikar |

Partially Completed: UI for joining a room has been started. The other tasks were not finished due to underestimating time to learn new skills beforehand.

## User Story #10

As a host, I would like to specify users as admins.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Create module that contains info for users in a room and which role they are | 5 hours | Brooks |
| **2.** | Interface module with UI to allow host (or admin) to specify a user as an admin | 3 hours | Tristan |
| **3.** | Interface module with database to save and retrieve preferences across sessions | 3 hours | Tristan/Avnish |
| **4.** | Unit test to verify promotion to admin | 1 hour | Avnish |

Partially Completed: There is code that provides a structure to connect users to rooms, but not to check whether they are permitted in the room. Interfacing with UI and ability to promote users to admin level permissions was not implemented.

## User Story #12

As an admin, I would like to be able to restrict room access to specific users

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Adding to the module that manages users in a room, create a whitelist and blacklist of users | 5 hours | Brooks |
| **2.** | Allow the room to be switched between blacklist mode, whitelist mode, or neither | 4 hours | Tristan |
| **3.** | When attempting to add a user into a room, verify the user is allowed based on the current restriction mode | 3 hours | Tristan |
| **4.** | Unit testing to ensure restrictions hold | 1 hour | Avnish |

Not Completed: Some conceptual work and notes were done for these tasks, but no implementation.

## 

## User Story #13

As a host I would like to be able to create a room for others to join

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Creation of distinct URLs when created for requests for a new room | 3 hours | Tristan |
| **2.** | Keeping track of which URLs are currently taken for other rooms | 3 hours | Tristan |
| **3.** | After rooms are no longer in use allow URL to eventually be recycled back into potential URLs for new rooms | 4 hours | Tristan |
| **4.** | Ensure distinct URLs and recyclable URLs through unit testing | 1 hour | Avnish |

## 

Partially completed: Rooms are given distinct urls however they are just based off room ID in the database currently. This technically meets criteria but easily guessable URLs are not acceptable when room invites can be link based. A hashing function needs to be implemented to create distinct URLs that are hard to guess but still relatively simple on the user side.

## User Story #15

As a user I would like to have a user-friendly website interface.

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task Description** | **Estimated Time** | **Owner** |
| **1.** | Design login page | 3 hours | Elizabeth |
| **2.** | Design room info page | 4 hours | Shrikar |
| **3.** | Design playlist page | 8 hours | Shrikar |
| **4.** | Design chat page | 5 hours | Elizabeth |

Partially Completed: Went over final design ideology in previous meeting, just requires implementation.

# How should we improve?

After looking back on the sprint we noticed many areas ripe for improvement that can help us to achieve our goals.

## Underestimation of hours needed for tasks

* + Estimating time needed for work is always a difficult process, however now that we understand better the constraints we have and rate of potential progress we should be able to develop more accurate timelines on the number of hours needed for work.

## Underestimation of time required for learning new technologies

* + Similar to the underestimation on hours needed for tasks we did not realize how much time needed to be spent to properly understand some of the tools we were using. Though not formally counting for any large portion of hours this is something that we will individually consider more respectfully when approaching any new technologies in the future sprints.

## Resolution of conflicts for user stories given to multiple members

* + Any user stories that involved multiple members working on them became muddled as the interdependence of systems so closely related constantly delayed work or confused members on which exact part of the process they were responsible for. Where we thought this might increase accountability it just slowed work and needlessly complicated things. Going forward minimizing closely related work to as few members as acceptable is the best approach to improve this, along with better definitions of work.

## Better testing methods

* + Testing methods were rarely properly implemented in this sprint. To improve, we will develop some tests side by side with development of each piece of functionality to not let testing become an afterthought in the design process.

## Better communication

* + This is an area that has already radically improved from the beginning of the first sprint. Communication had been a bit of an issue, especially with a group of 6. We now have better communication processes in place and as we continue working and getting familiar with each other those should continue to improve.

## Git commit rate

* + Throughout the sprint we weren’t committing near as often as we should have been. This next sprint it will be a goal to drastically improve this so that we have a continuous history of work done.

## Accountability

* + Since we’re all college students with a host of other things we need to take care of it can become easy to put the project on the backburner. Implementing systems to make sure work is getting done often is a major goal of sprint 2. One of those ways is to restructure our group meetings slightly to put more emphasis on showing off work done since the last meeting.

## Informal meetings more often

* + Our group already has 2 formal meetings every week with a third informal meeting however working together is an integral part in making sure everyone is on the same page. The goal for the next sprint is whenever someone starts developing, they do so with the mindset that it's an opportunity to develop with other members.

## Formalized plans on how object interactions will occur

* + A major problem developed in Sprint 1 where there was ambiguity on where exactly different interactions would occur, specifically to how the database would be implemented and the effect that would have on how to code for that interaction.