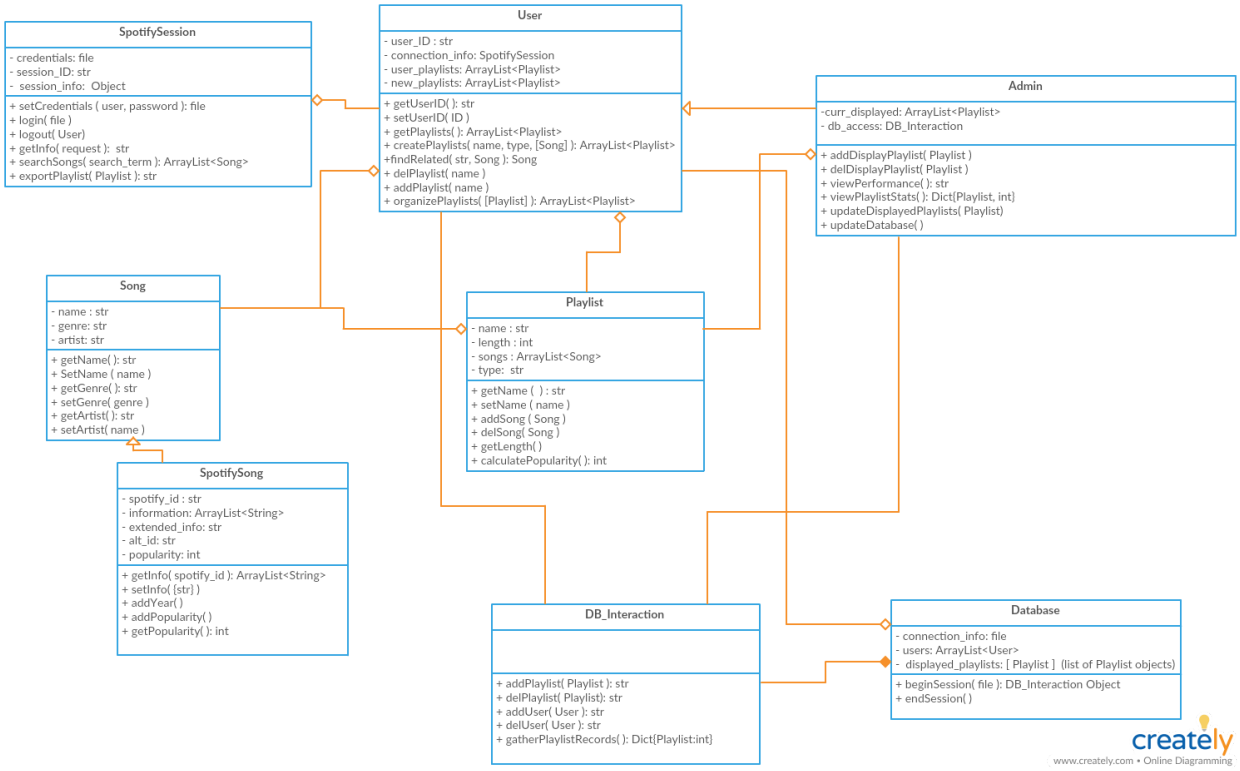
1. **Project Name**: MusicGeneration

2. **Team Member**: Nicholas Montoya
Yifan Li

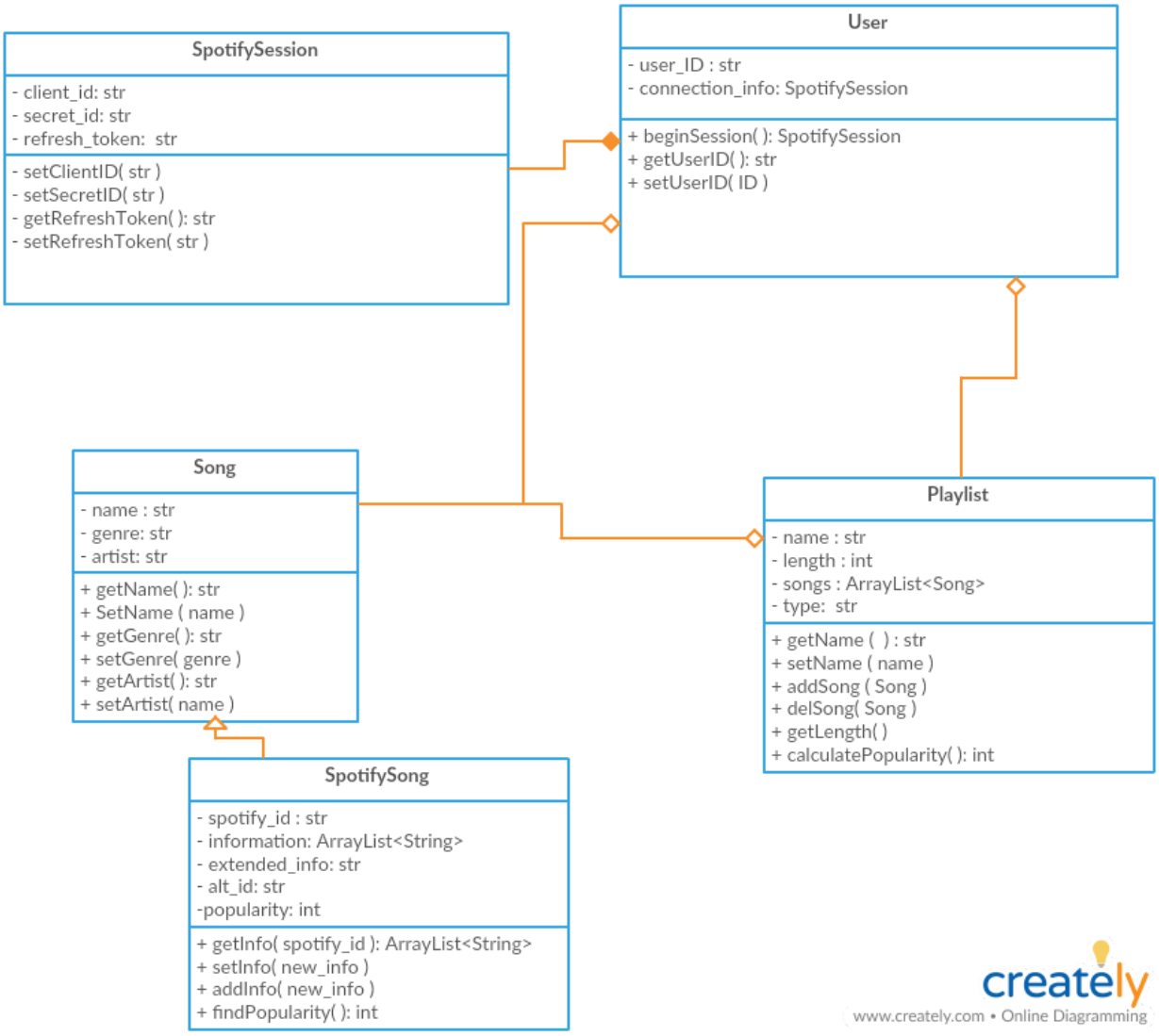
3. **Vision**: Allow users to create, find, and publish auto-generated playlists based on factors of their choice

4. **Project description**: We allow users to log into their Spotify account. Users are able to search for a song and modify their personal Spotify playlist. Users can also get an auto-generated song list based on either songs genre or by the time period.

5. **Previous class diagram from Part 2** (with fixes based on feedback)



6. **Completed Class diagram**:
A class diagram that shows the classes that have been implemented so far and their relationships to one another. (In other words, this diagram will not show the complete system you designed in Part 2 but rather the classes actually implemented since you submitted Part 2.). Be sure to fix issues identified in feedback from Part 2.



7. **Summary:** Textual description of the work done the past two weeks on your project.

So far we have been getting a lot of the base of our web application set up. As this is the first time either of us have engaged in a full project with Django, there were a few road bumps along the way. Fortunately, we were able to get our servers running locally and begin working on the fun stuff. We created the classes in the diagram above in order to store everything in a single session. Using the Spotify API, we have been able to get access to all public information (songs, albums, featured playlists, other featured content, etc.). We also have integrated a

button to login to your Spotify account (using their services and web pages) which allows us to access your personal playlists along with modify them. We are still working on refreshing user tokens automatically in order to maintain a constant connection however to continue, we need to be able to provide visual feedback to users. For the front-end part, we have get basic format page ready. The rest of work is to make tiny change correspond to the purpose of each page on the website This format is very close to the one that we draw on UI mock-up. We use grid for formatting all the element on the homepage. This is very helpful for future development.

8. **Breakdown of work** across team members. Be specific - list out each task done and who.

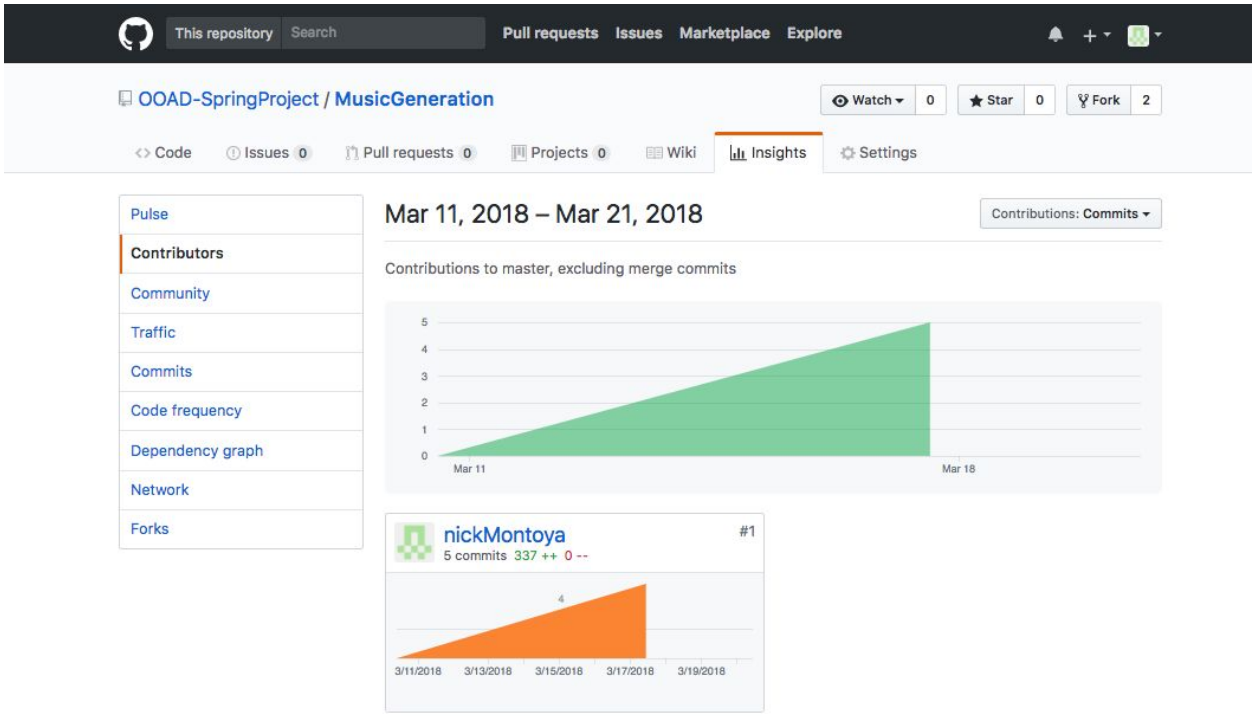
Yifan Li:

1. Made a template homepage for the website. From now on, we can change the layout a little bit for other pages on the website but most HTML templates will inherit from this one.
2. Placed some placeholder for playlist and songs cover art, which will be implemented later on
3. Created an animation search bar for searching songs on main page
4. Change font size and background color as closer as the UI mock
5. Used grid on most of css code for layout purpose, after, I can add placeholders much easier
6. Combine the login button created by Nick within html page

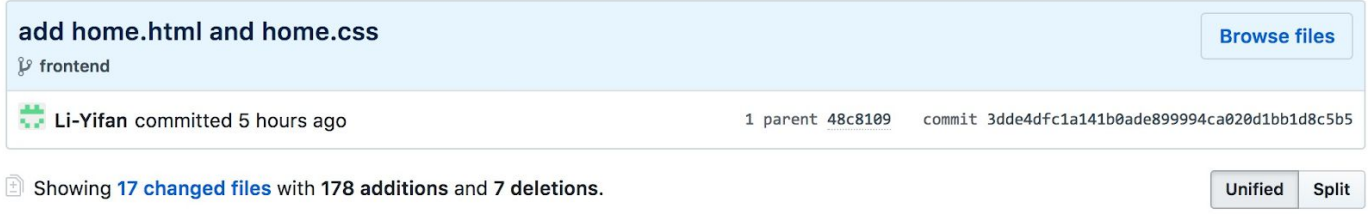
Nicholas Montoya:

1. Got initial Django server setup
2. Made basic HTML pages to continue working on backend (just buttons and a title)
3. Registered application with Spotify to receive developer credentials for API access
4. Tried to find various necessary endpoints to request all information
5. Made SpotifySession class to hold relevant information for the current user's connection to API and various requests
6. Made a general Song class because Spotify provides a lot of information from the request that isn't formatted for easy use and there are also other sources to get songs from online
7. Extended the Song class with SpotifySong in order to more neatly wrap up all relevant information retrieved
8. Made Playlist class in order to hold various Song objects.
9. Began working on PostgreSQL implementation to hold information for all users and not just a single instance
10. Registered application with Facebook in order to achieve easier login to Spotify

9. **GitHub Graph**: Graph in Github of each person's contribution.



Yifan Li pushes his code on a branch called 'frontend' for testing purpose, haven't yet merged them together



10. **Estimate Remaining Effort:**
Provide an estimate of how much more work needs to be done for you to have implemented the design that you presented in Part 2 (or describe how your design is changing as a result of making progress in implementing your system). ○ If you have actual screenshots of your system in action, please include them.
○ If you have developed test cases for your system, please describe them.

We expect the remaining effort to be on pace with our schedule from the start. The main difficulties we were facing having been mostly wrapped up. We still need to implement a scalable database but that is something we have experience with and don't expect too much difficulty. Beyond that, we are still figuring out how we want to auto-generate playlists for the users. The main benefit of our application is to save users time from searching and adding highly rated songs to playlists they have to manually create. However, we think it would be nice to throw in some songs that maybe aren't rated as high as wild cards. Maybe a user will like it regardless of its overall popularity and it will be like finding a hidden gem.

11. **Next Iteration**:
Plans for next iteration: What are your plans for the next iteration?

- Fully implement PostgreSQL database
- Store the playlists users select from their own accounts
- Create pages and functionality to allow users to search for similar songs from their existing library
- (Possibly) Allow users to search for similar songs not in their library (need to guarantee spotify has the song so selecting from their existing library takes care of this)
- Auto-generate multiple playlists from search
- Display those playlists back to users
- Modify users private playlists to add full playlist
- Modify existing playlists to add individual songs
- Finish every pages on the website
- Making sure the font-end can be connected with backend
- Making sure css files are able to load into every html files
- Make our website looks more nicer
- Get every button working with each other
- Stretch Goal: Implement a Spotify player on our page so users can hear the songs before they choose to add anything