Team 15 - *Beefed Up Music Scheduler*

Team members

William Vanschaik (wvanscha@purdue.edu)
Joey Imburgia (jimburgi@purdue.edu)
Santiago Abondano (sabonda@purdue.edu)
Rachel Gully (rgully@purdue.edu)
Gaurav Srivastava (srivast6@purdue.edu)

Problem Statement

Many times, it is too difficult to find and play the best song for the current situation. For example, before an important meeting, we are too busy preparing for the meeting to focus on finding a song to pump us up. Music should also be able to play continuously, but not play the same genre or constantly replay songs. Alarms get old quickly and people dread hearing it, so a specific genre would help them wake up quickly. Your music should be based off your schedule not pandora's or spotify's.

Background information

Many music players focus on making it easier for a user to access music. For example, Spotify allows users to create and share playlists over the internet. This makes music discovery very simple. Despite music discovery being very simple, the user still has to manually play the music. However, they don't consider choosing the right song for the right time of day. Having a music scheduler would reduce the hassle on the user by automatically playing the genre of music. This reduces the user's effort to experience a great song selection.

Environment

We will be writing in Java in order for our application to be able to be run on all operating systems. We also be using local serialization to store the user settings, song preferences, logins, and local songs. We are going to be creating our own scheduling system that is also going to be saved with our serialized data to ensure persistence.

Functional Requirements

Backlog ID	Functional Requirement	Hours
1	As a user, I would like to play a song.	2
2	As a user, I would like to play music from a playlist.	2
3	As a user, I would like to schedule a song to play.	8
4	As a user , I would like to schedule a playlist to play.	4
5	As a user, I would like to create a playlist	2

6	As a user, I would like to pause a song.	6
7	As a user, I would like to snooze the alarm.	2
8	As a user, I would like to remove a song from a playlist.	2
9	As a user, I would like to control the volume directly from the application.	8
10	As a user, I would like to minimize the application and run it in the background.	6
11	As a user, I would like to select a custom alarm sound.	2
12	As a user, I would like to like to customize the length of my snooze.	2
13	As a user, I would like to my Beefed up music schedule to be a startup service.	4
	Total:	50

Non-Functional Requirements

- Low CPU usage
- Doesn’t crash over a long period of time
- Change the background colors based on user preference
- Run on a computer that has java
- Sort music in alphabetic order
- Able to play FLAC (if time allows)
- Able to run on a low powered computer.
- Able to cache information for offline use (if time allows)
- Keyboard shortcut for pause, play, rewind, skip and snooze (if time allows)
- Have a visualizer (if time allows)
- Have an equalizer (if time allows)

Use Cases

Case: Play a song	System Response
1. Press “Play”	2. System connects to local file.
	3. System begins playing song.
	4. “Pause” replaces the “Play” text.

Case: Play music from a playlist	System Response
1. Press “Play Playlist” on specific playlist	2. System connects to local file
	3. System begins to play song from the playlist.

Case: Schedule a song	System Response
1. Open Schedule tab	2. Display schedule window
3. Seek to the time you would like to play song on.	
4. Press “Schedule”	5. Make a event in Google Calendar for that song/genre.

Case: Schedule a playlist	System Response

1. Open Schedule tab	2. Display schedule window
3. Seek to the time you would like to play playlist on.	
4. Press “Schedule”	5. Make a event in Google Calendar for that song/genre.

Case: Create a playlist	System Response
1. Click on add playlist button	2. Opens serialized data
4. Adds what the playlist title will be in textbox.	
5. Place songs or genres into playlist.	6. Write the playlist data to the serialized data

Case: Pause a song	System Response
1. Press “Pause” button	2. Interrupt the current playing song
	3. “Play” replaces the “Pause” text

Case: Snooze the alarm	System Response
	1. Alarm goes off as scheduled
2. User presses spacebar to snooze.	3. Alarm time is extended by snooze time.
	4. Repeat

Case: Remove a song from a playlist.	System Response
1. Click playlist and then press “Edit”	2. Opens playlist, opens up list of songs in playlist.
3. Clicks “Remove” on the unwanted songs.	
4. Press “Save”.	5. Open SQLite database and overwrite the old playlist.

Case: Control volume	System Response
1. Change the slider to a percentage of volume	2. Changes the volume of songs being played

Case: Minimize Application and Run in Background	System Response
1. Click on minimize button	2. Minimizes window and plays music in background

Case: Custom alarm sound	System Response
1. Open App settings	2. Displays the system settings
3. Click on alarm settings	4. Display alarm settings
5. Change alarm sound	6. Drop down of available sounds
7. Choose sound from list	10. Changes the alarm sound in system

Case: Customize length of snooze	System Response
1. Open App settings	2. Displays the systems settings
3. Click on alarm settings	4. Displays alarm settings
5. Input snooze time in input box	6. Update snooze time.

Case: App Starts on startup service	System Response
1. Open App settings	2. Opens Systems settings
3. Check box add to startup list	3. Puts the application in the startup list