UpNext

Test Plan

Shivan Desai, Adam Johnston, Akhil Agrawal, Brian Duffy

|  |  |
| --- | --- |
| Identification and Classification | Test Case 1: User Log In  Test Case Type: functionality  Severity: 1 |
| Test Steps | 1. User is presented the login form 2. User enters in their username and password 3. User clicks login 4. The login page sends the data to Spotify for validation 5. Spotify sends verification back and the song page is loaded |
| Expected Behavior | 1. If the user’s credentials are correct, the app should redirect to the home page 2. If the user’s credentials are incorrect, the app should display an alert message indicating a failed login |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 2: Song Voting  Test Case Type: functionality  Severity: 2 |
| Test Steps | 1. A user clicks on a voting icon 2. The voting icon is an upvote or downvote |
| Expected Behavior | 1. The amount of votes is updated server side 2. The amount of votes is displayed to all users upon refresh |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 3: Song Voting Equivalence  Test Case Type: equivalence  Severity: 2 |
| Test Steps | 1. A user clicks the upvote or downvote for a particular song 2. This is repeated until the amount of votes reaches 0 or a maximum 3. The user then tries to go beyond the bounds of the voting functionality ie. (under 0 or over a set limit) |
| Expected Behavior | 1. The app should not increment or decrement past the bounds of 0 - max 2. When one of the bounds is reached the corresponding vote button is greyed out and the number is not changed |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 4: Song Voting Boundary  Test Case Type: boundary value  Severity: 2 |
| Test Steps | 1. A user clicks the upvote button of a song multiple times 2. A user clicks the downvote button of a song multiple times |
| Expected Behavior | 1. The app should only allow each user to upvote once per song 2. If a user tries to upvote twice, their upvote will be canceled (toggle behavior) 3. If a user already has upvoted a song, when they downvote it the upvote is removed and a downvote takes its place (viceversa). |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 5: Post Song Functionality  Test Case Type: functionality  Severity: 1 |
| Test Steps | 1. A user types in the song they want to add to the shared playlist 2. The request is sent to the server 3. The server responds and updates web interface and database |
| Expected Behavior | 1. The song should be added to the playlist on the database and should update the web interface. |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 6: Post Song Boundary  Test Case Type: boundary value  Severity: 2 |
| Test Steps | 1. A user tries to add a song that doesn’t exist 2. A user tries to add a song that is already in the playlist 3. A user tries to add more than the allowed limit of songs that can be added per person 4. A user tries to add a song and it will be the 10,000 song in the playlist. |
| Expected Behavior | 1. An error message will appear alerting them of invalid input 2. An error message will appear alerting them that the song is already in the playlist 3. An error message will appear telling them that they have reached their daily post limit 4. An error message alerts user that the playlist is full |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 7: Delete Song Functionality  Test Case Type: functionality  Severity: 2 |
| Test Steps | 1. A user attempts to delete a song in the playlist |
| Expected Behavior | 1. If the user is the poster of that song, the song will be deleted from the database and the web interface will be updated upon refresh 2. If the user is and admin, the song will be deleted from the database and the web interface will be updated upon refresh 3. Any other users, upon reaching the song before refresh, will contact the database before playing the song and skip it 4. Upon deletion of a song by a user, Spotify handles the deletion of concurrent listeners after they finish the song |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 8: Delete Song Boundary  Test Case Type: boundary value  Severity: 3 |
| Test Steps | 1. A user attempts to delete one of the songs that they added to the playlist recently 2. A user attempts to delete one of the songs they added more than a day ago |
| Expected Behavior | 1. If the song was posted that day, the user will be given back one Song Request upon deletion of that song 2. If the song was not posted that day, the user will not be given back one Song Request upon deletion of that song |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 9: Flag Song  Test Case Type: functionality  Severity: 3 |
| Test Steps | 1. A user clicks on the flag as inappropriate button |
| Expected Behavior | 1. If the song has less than a certain limit of flaggings-to-be-removed, then the flagging count is incremented 2. If the song has greater than or equal to the limit of flaggings, then the song is removed from the playlist |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 10: Flag Song Equivalence  Test Case Type: equivalence  Severity: 3 |
| Test Steps | 1. A user clicks on the flag song button |
| Expected Behavior | 1. If the user already flagged that song, then the song becomes unflagged and the counter decrements 2. If the user has not flagged that song, then the song becomes flagged and the counter increments |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 11: Comment on Song  Test Case Type: functionality  Severity: 2 |
| Test Steps | 1. A user clicks on a song in the playlist 2. They then click in a form to type a comment |
| Expected Behavior | 1. Upon submission of a comment, the form is sent to the database to be stored 2. The web interface is updated to reflect that change upon refresh |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 12: Comment on Song Equivalence  Test Case Type: equivalence  Severity: 1 |
| Test Steps | 1. A user clicks on a song in the playlist 2. The user clicks in a form to type a comment |
| Expected Behavior | 1. If the user types a comment that is too long, the textbox should prevent additional text and notify the user it is too long 2. If the user tries to post a comment with no characters, an alert message will appear notifying the user to add characters |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 13: Mark Song to Skip  Test Case Type: functionality  Severity: 2 |
| Test Steps | 1. A user presses the skip button next to the post |
| Expected Behavior | 1. The song id is added to the user’s skip list in the database 2. The user’s player will no longer play this song when it reaches the song (will skip over it) |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 14: Mark Song to Skip Boundary  Test Case Type: boundary value  Severity: 2 |
| Test Steps | 1. A user presses the skip button for all songs in the playlist |
| Expected Behavior | 1. Upon skipping all songs in the list, nothing is played and if a current song is playing it will stop |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 15: Create a Profile  Test Case Type: functionality  Severity: 1 |
| Test Steps | 1. A user presses a button to create a profile on the login page 2. The server redirects them to the Spotify create profile page 3. The server returns the user to the login page |
| Expected Behavior | 1. The user will now be able to login with their newly created Spotify account and use it to comment/upvote/etc 2. If the user did not sign up for Spotify Premium, they will not be able to log in |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 16: Create a Profile Boundary  Test Case Type: boundary value  Severity: 1 |
| Test Steps | 1. A user tries to create a profile that already exists |
| Expected Behavior | 1. Spotify notifies them that the username is already taken, the password is invalid, or any other information is incorrect/invalid |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 17: Remove Songs as Admin  Test Case Type: functionality  Severity: 2 |
| Test Steps | 1. An admin clicks a button next to a song post to remove it |
| Expected Behavior | 1. The song is removed from the database and upon refresh is removed from each users song list |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 18: Remove Songs as Admin Boundary  Test Case Type: boundary value  Severity: 2 |
| Test Steps | 1. An admin removes all songs in the list 2. An admin removes a song being played by other users in the list |
| Expected Behavior | 1. The song and it’s comments are removed from the database 2. Spotify is told to delete that song and the song will continue playing for the user until finish |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 19: Ban Users as Admin  Test Case Type: functionality  Severity: 3 |
| Test Steps | 1. The admin enters the username or id of user to ban into a special textbox only visible to admins 2. The admin clicks submit |
| Expected Behavior | 1. Upon refresh the user is returned to the login page with an error message and is prevented from logging in 2. If the user attempts to add or remove songs or post comments, the user is prevented by the database |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 20: Ban Users as Admin Boundary  Test Case Type: boundary value  Severity: 2 |
| Test Steps | 1. An admin attempts to ban another admin 2. An admin attempts to ban an already banned user |
| Expected Behavior | 1. If the user is already banned, the user is unbanned 2. If the user does not exist, nothing is done 3. If the user is an admin, the user is notified that they cannot ban the person |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 21: Playlist Garbage Collector  Test Case Type: functionality  Severity: 2 |
| Test Steps | 1. A user adds a song to the playlist 2. A certain amount of time passes |
| Expected Behavior | 1. If the song that was added reaches an “expiration date” and the rating is not high enough, then the song is removed from the database 2. If the song reaches an “expiration date” and the rating is high enough, then the song’s “expiration date” is extended |

|  |  |
| --- | --- |
| Identification and Classification | Test Case 22: Playlist Garbage Collector Boundary  Test Case Type: boundary value  Severity: 2 |
| Test Steps | 1. A user adds a song to the playlist 2. A certain amount of time passes |
| Expected Behavior | 1. If the song that was added has been up for a month, then the song is removed from the playlist |