SOFTWARE VERIFICATION, VALIDATION AND TESTING

TESTING DOCUMENTATION

Spotify Automated Testing

Prepared by:
Zara Bahtanović

Proposed to:
Samed Jukić, Assist. Prof. Dr.
Aldin Kovačević, Teaching Assistant

21.01.2023

Table of Contents

1. Introduction

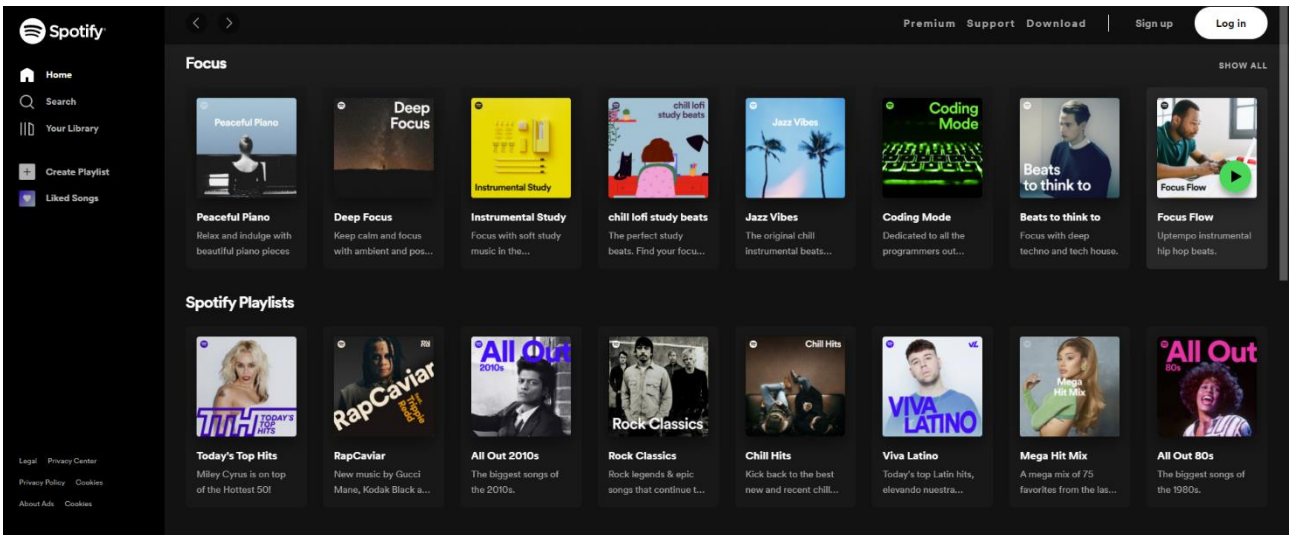
1.1. About the Project

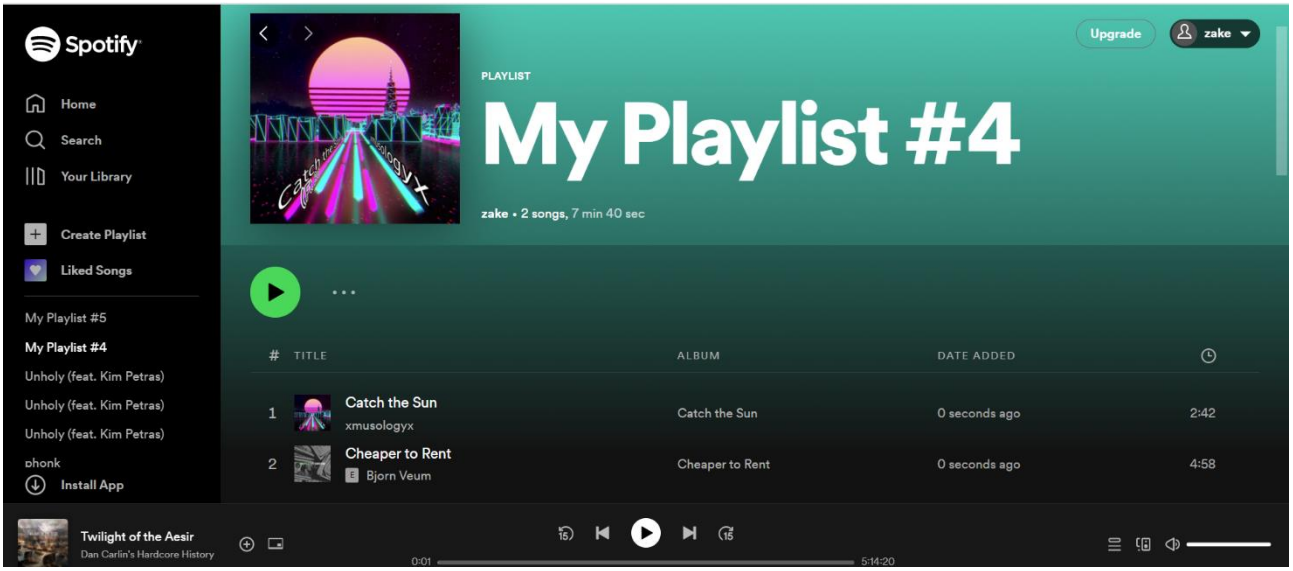
Spotify is a music streaming service that allows users to listen to millions of songs, albums, and playlists on various devices. Founded in 2006, it has over 456 million monthly active users including 196 million paying subscribers. It also allows users to search for music based on artist, album, or genre, to create, edit, and share playlists, follow other users, and discover new music based on their listening habits. The application is available on multiple platforms including web, iOS, and Android. The homepage link is **https://www.spotify.com/**.

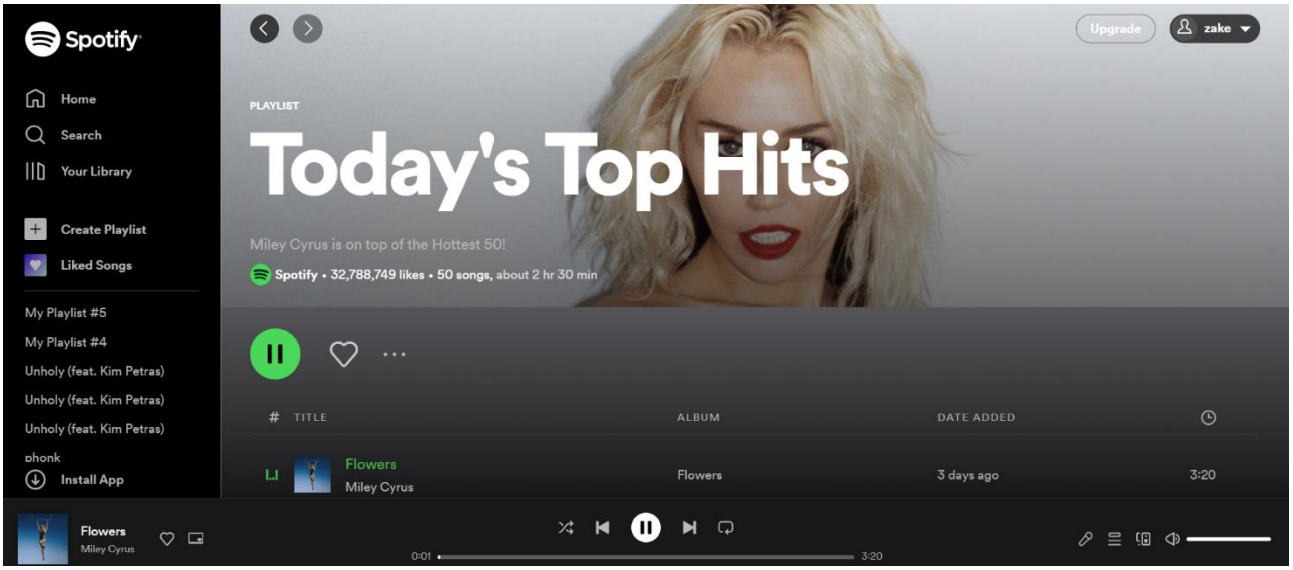
1.2. Project Functionalities and Screenshots

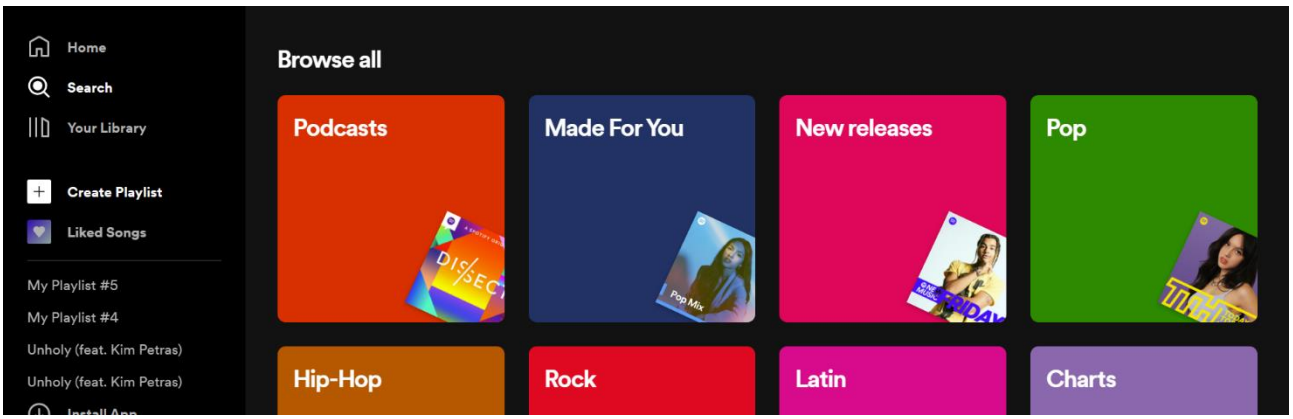
The main features of the Spotify project include:

- Music streaming: Users can listen to a wide variety of music from various genres and artists.
- Playlists: Users can create, edit, and share playlists with other users.
- Search: Users can search for music based on artist, album, or genre.
- Radio: Users can listen to radio stations based on a song, artist, album or genre and save the radio station to access it from the library.
- Podcasts: Users can listen to podcasts and access a variety of podcast shows.
- Social media integration: Users can connect their Spotify account with their social media accounts to share music and playlists with friends.
- Offline listening: Users can download music to listen offline.
- Personalization: Spotify offers personalized recommendations for music and podcasts to its users based on their listening history and preferences.

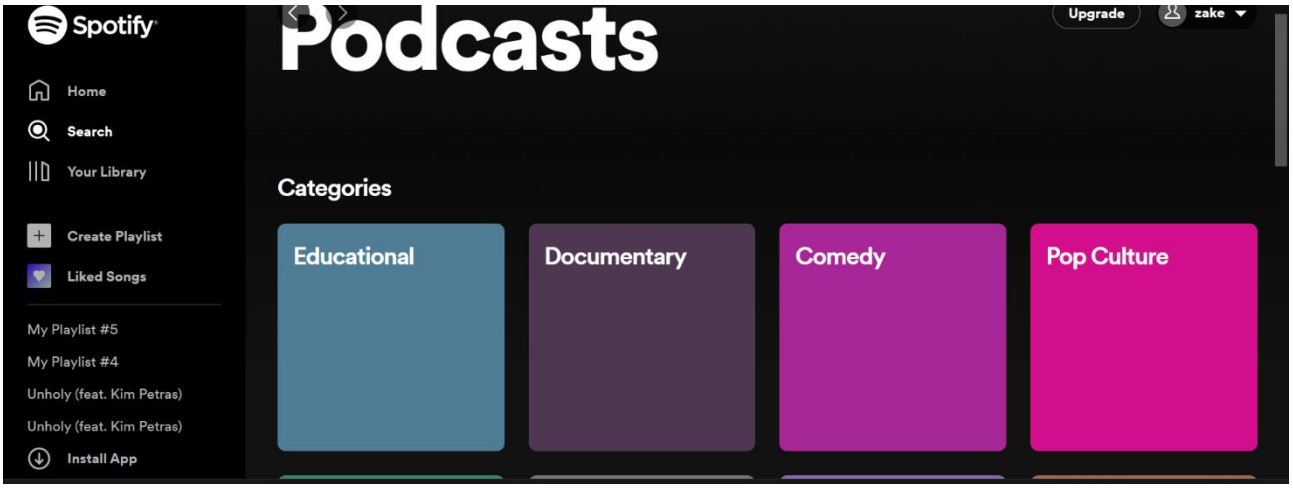

Spotify
Home
Search
Your Library
Create Playlist
Liked Songs
Premium
Support
Download
Sign up
Log in
Focus
SHOW ALL
Peaceful Piano
Relax and indulge with beautiful piano pieces
Deep Focus
Keep calm and focus with ambient and pos...
Instrumental Study
Focus with soft study music in the...
chill lofi study beats
The perfect study beats. Find your focu...
Jazz Vibes
The original chill instrumental beats...
Coding Mode
Dedicated to all the programmers out...
Beats to think to
Focus with deep techno and tech house.
Focus Flow
Uptempo instrumental hip hop beats.
Spotify Playlists
Today's Top Hits
Miley Cyrus is on top of the Hottest 50!
RapCaviar
New music by Gucci Mane, Kodak Black a...
All Out 2010s
The biggest songs of the 2010s.
Rock Classics
Rock legends & epic songs that continue t...
Chill Hits
Kick back to the best new and recent chill...
Viva Latino
Today's top Latin hits, elevando nuestra...
Mega Hit Mix
A mega mix of 75 favorites from the las...
All Out 80s
The biggest songs of the 1980s.
Legal
Privacy Center
Privacy Policy
Cookies
About Ads
Cookies

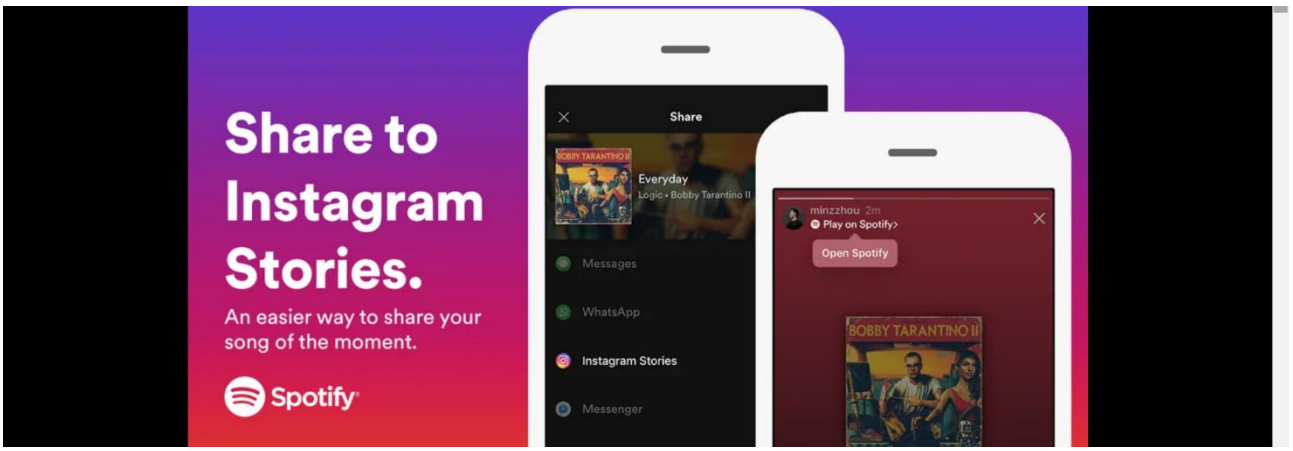

Spotify
Home
Search
Your Library
Create Playlist
Liked Songs
My Playlist #5
My Playlist #4
Unholy (feat. Kim Petras)
Unholy (feat. Kim Petras)
Unholy (feat. Kim Petras)
phonk
Install App
Upgrade
zake
PLAYLIST
My Playlist #4
zake • 2 songs, 7 min 40 sec
TITLE
ALBUM
DATE ADDED
1 Catch the Sun
xmusologyx
Catch the Sun
0 seconds ago
2:42
2 Cheaper to Rent
E Bjorn Veum
Cheaper to Rent
0 seconds ago
4:58
Twilight of the Aesir
Dan Carlin's Hardcore History
0:01
5:14:20

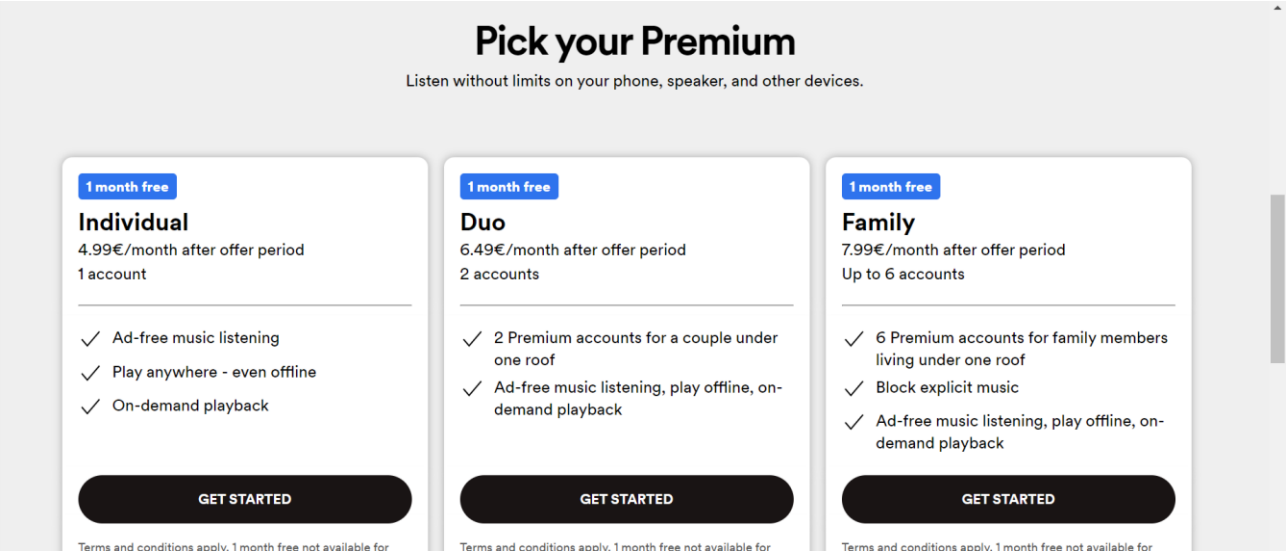

Spotify
Home
Search
Your Library
Create Playlist
Liked Songs
My Playlist #5
My Playlist #4
Unholy (feat. Kim Petras)
Unholy (feat. Kim Petras)
Unholy (feat. Kim Petras)
phonk
Install App
Upgrade
zake
PLAYLIST
Today's Top Hits
Miley Cyrus is on top of the Hottest 50!
Spotify • 32,788,749 likes • 50 songs, about 2 hr 30 min
TITLE
ALBUM
DATE ADDED
Flowers
Miley Cyrus
Flowers
3 days ago
3:20
Flowers
Miley Cyrus
0:01
3:20


Home
Search
Your Library
Create Playlist
Liked Songs
My Playlist #5
My Playlist #4
Unholy (feat. Kim Petras)
Unholy (feat. Kim Petras)
Browse all
Podcasts
Made For You
New releases
Pop
Hip-Hop
Rock
Latin
Charts


Home
Search
Your Library
Create Playlist
Liked Songs
My Playlist #5
My Playlist #4
Unholy (feat. Kim Petras)
Unholy (feat. Kim Petras)
Install App
Miley Cyrus
RADIO
PLAYLIST
Miley Cyrus Radio
Spotify • 86,363 likes • 50 songs, about 2 hr 45 min
TITLE
ALBUM
DATE ADDED


Spotify
Home
Search
Your Library
Create Playlist
Liked Songs
My Playlist #5
My Playlist #4
Unholy (feat. Kim Petras)
Unholy (feat. Kim Petras)
Install App
Podcasts
Upgrade
zake
Categories
Educational
Documentary
Comedy
Pop Culture


Share to Instagram Stories.
An easier way to share your song of the moment.
Spotify
Share
Everyday
Logic • Bobby Tarantino II
Messages
WhatsApp
Instagram Stories
Messenger
minzzhou 2m
Play on Spotify
Open Spotify
BOBBY TARANTINO II



2. Test Plan

2.1. Scope

Our main focus will be on functional testing, ensuring that all features of the application are working correctly and as expected. This includes testing the user's ability to create an account, manage it, browse and search for music, podcasts, as well as the ability to manage their playlists. We will also be performing accessibility testing, using tools such as aXe to verify that the website is accessible to as many users as possible. Additionally, we will be testing the website's localization to ensure that it is available in different languages.

However, we will not be testing the recommendation feature as it is beyond the scope of our testing efforts. nor will we be testing the applications responsive design and features that come with the premium version.

2.2. Testing Environment and Tools

In this project, we will be using Selenium WebDriver, a widely-used open-source tool for automating web browsers, as our main testing tool. We will also be using JUnit, a popular open-source testing framework for Java, to write and run our test cases. Additionally, we will be using the selenium-axe library to perform accessibility testing, which allows us to run accessibility checks using the Axe engine. The programming language we will be using is Java. Furthermore, we will be using Maven, a build automation tool for Java projects, to manage our dependencies. We will also be using the selenium-java dependency to run the selenium web driver. The development environment that we will be using is Eclipse. In addition, we will use Git for version control and GitHub for code hosting.

3. Test Execution

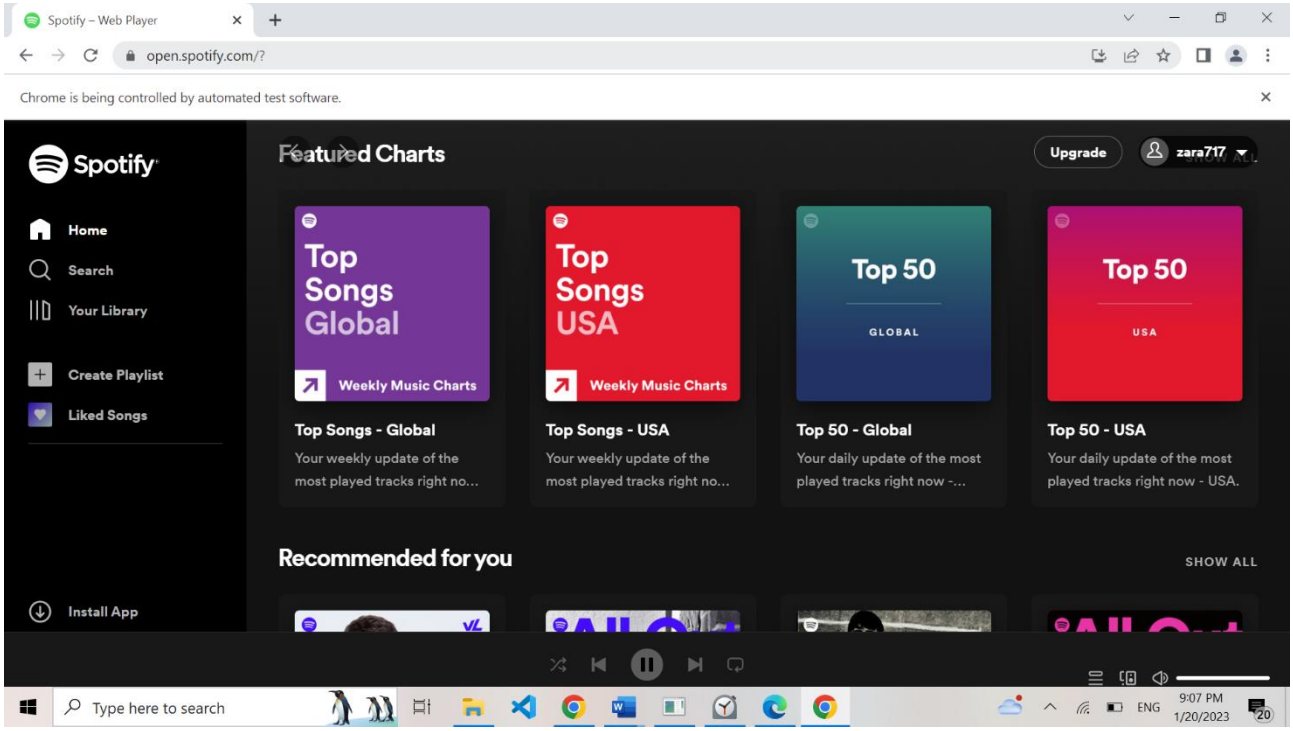
3.1. Sing Up

User wants to create an account for the website.

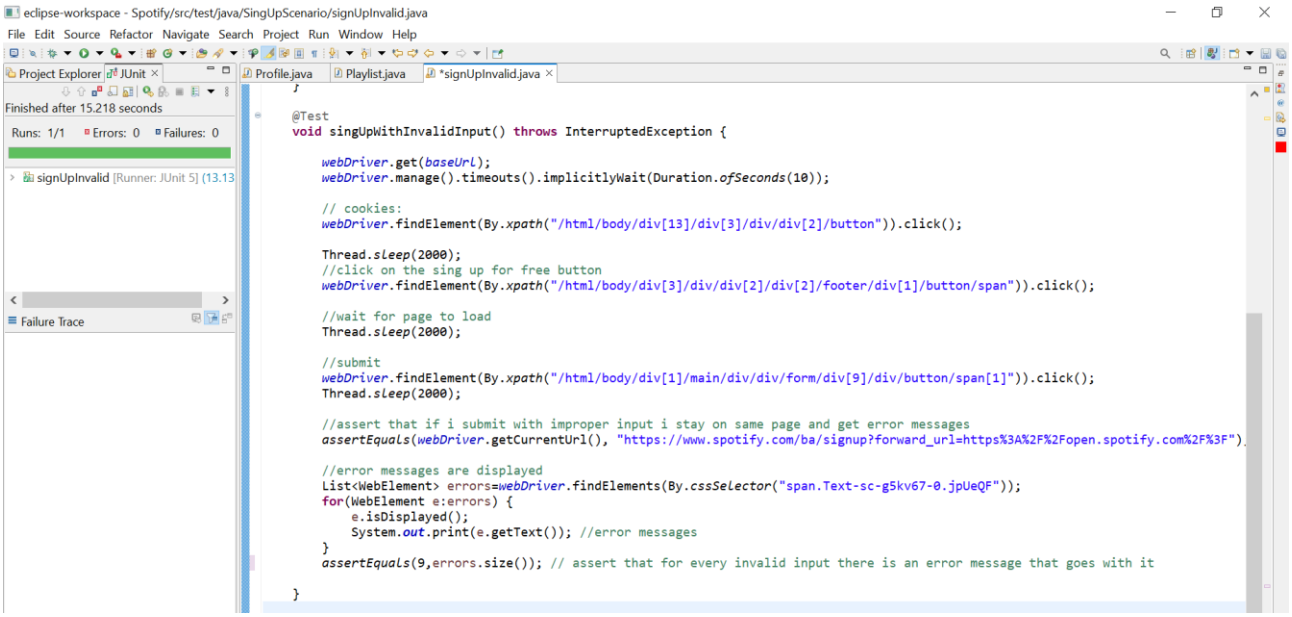
Test Name: Create an Account with Email and Password				
Description: Verify that users can create an account using an email and password				
Pre-condition(s): None				
Test Steps: 1.Open the Spotify website and click on the “Sign up” button. 2. Fill in the required fields. 3. Click on the “Sing up” button. 4. Assert that the user has been logged in and redirected to the home page	**Test Data:** Email=” zaratester717@gmail.com”, Password=uniburch7, Display Name=” zara717”, Date=”02.10.2000”	**Expected Result:** The user successfully creates an account and is redirected to the home page with that account’s information.	**Actual Result:** The user successfully creates an account and is redirected to the home page with that account’s information.	**Status:** PASS
Notes:				

*SingUpTest.java

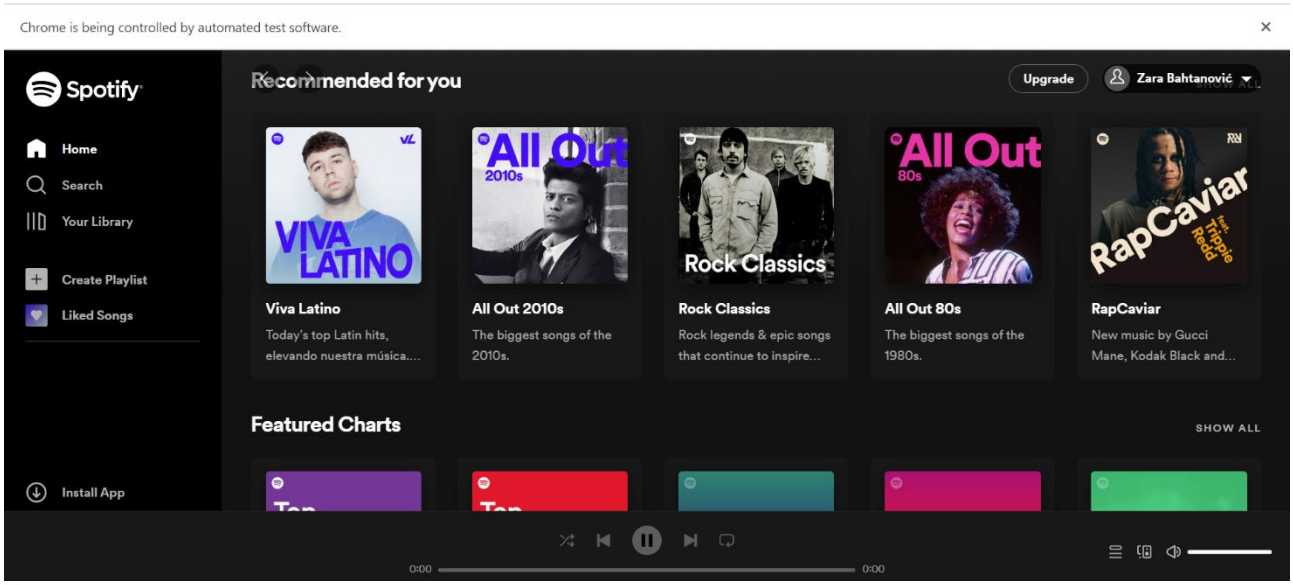
```java
@Test
void singUpTestWithEmailAdress() throws InterruptedException {
    webDriver.get(baseUrl);
    webDriver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));

    // cookies:
    webDriver.findElement(By.xpath("/html/body/div[13]/div[3]/div/div[2]/button")).click();

    Thread.sleep(2000);
    //click on the sing up for free button
    webDriver.findElement(By.xpath("/html/body/div[3]/div/div[2]/div[2]/footer/div[1]/button/span")).click();

    //wait for page to load
    Thread.sleep(2000);
    // asserts that when we click the button we are transfered to the right page
    assertEquals(webDriver.getCurrentUrl(), "https://www.spotify.com/ba/signup?forward_url=https%3A%2F%2Fopen.spotify.com%2F%3F");

    //get all form elements and check if they are intractable
    List<WebElement> formElements=webDriver.findElements(By.className("Input-sc-1gbx9xe-0"));
    for(WebElement element:formElements) {
        assertTrue(element.isDisplayed());
        assertTrue(element.isEnabled());
    }

    //Fill in form
    //email
    webDriver.findElement(By.name("email")).sendKeys("zaratester71732@gmail.com");

    //confirm email
    webDriver.findElement(By.name("confirm")).sendKeys("zaratester71732@gmail.com");

    //password
    webDriver.findElement(By.name("password")).sendKeys("uniburch7");

    //profile name
    String nameString="zara717";
    webDriver.findElement(By.name("displayname")).sendKeys(nameString);

    //day
    webDriver.findElement(By.name("day")).sendKeys("2");

    //month
    Select monthSelect=new Select(webDriver.findElement(By.name("month")));
    monthSelect.selectByVisibleText("October");

    //year
    webDriver.findElement(By.name("year")).sendKeys("2000");


    //assert radioButtons - one is selected then others are not
    List<WebElement> radioButtons = webDriver.findElements(By.cssSelector("input[type='radio'][className='Radio-sc-tr5kfi-0']"));
    for (WebElement radioButton : radioButtons) {
        radioButton.click();
        for (WebElement otherRadioButton : radioButtons) {
            if (!otherRadioButton.equals(radioButton)) {
                assertFalse(otherRadioButton.isSelected());
            }
        }
    }

    //gender
    webDriver.findElement(By.xpath("/html/body/div[1]/main/div/div/form/fieldset/div/div[2]/label/span[1]")).click();


    //last two buttons
    webDriver.findElement(By.xpath("/html/body/div[1]/main/div/div/form/div[6]/div/label/span[1]")).click();
    webDriver.findElement(By.xpath("/html/body/div[1]/main/div/div/form/div[7]/div/label/span[1]")).click();

    //captcha - manual
    Thread.sleep(12000);

    //gender
    webDriver.findElement(By.xpath("/html/body/div[1]/main/div/div/form/fieldset/div/div[2]/label/span[1]")).click();


    //last two buttons
    webDriver.findElement(By.xpath("/html/body/div[1]/main/div/div/form/div[6]/div/label/span[1]")).click();
    webDriver.findElement(By.xpath("/html/body/div[1]/main/div/div/form/div[7]/div/label/span[1]")).click();

    //captcha - manual
    Thread.sleep(12000);

    //submit
    webDriver.findElement(By.xpath("/html/body/div[1]/main/div/div/form/div[9]/div/button/span[1]")).click();

    //after submit wait until page is loaded
    Thread.sleep(2000);
    //assert that we are singedIn with our account - that profile name we choose is displayed
    String profileUsername=webDriver.findElement(By.xpath("/html/body/div[4]/div/div[2]/div[1]/header/button[2]/span")).getText();
    assertEquals(nameString, profileUsername);

    File screenshot = ((TakesScreenshot) webDriver).getScreenshotAs(OutputType.FILE);
    try {
        FileUtils.copyFile(screenshot, new File("screenshot.png"));
    } catch (IOException e) {
        e.printStackTrace();
    }


}
```

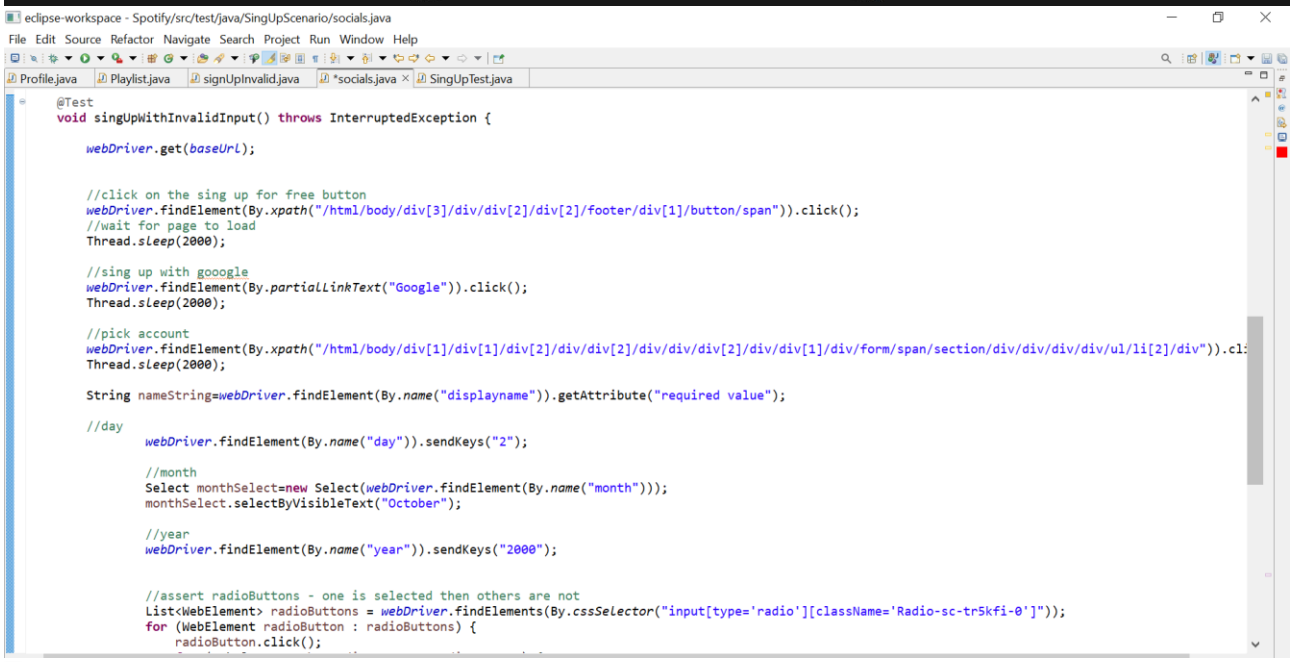


Test Name: Create an Account with Invalid Credentials				
Description: Verify that users cannot create an account using invalid credentials				
Pre-condition(s): None				
Test Steps: 1.Open the Spotify website and click on the “Sign up” button. 2. Click on the “Sing up” button to submit form. 3. Assert that error message is displayed indicating the credentials are invalid.	**Test Data:**	**Expected Result:** Error message is displayed indicating that the credentials are invalid.	**Actual Result:** Error message is displayed indicating that the credentials are invalid.	**Status:** PASS
Notes:				

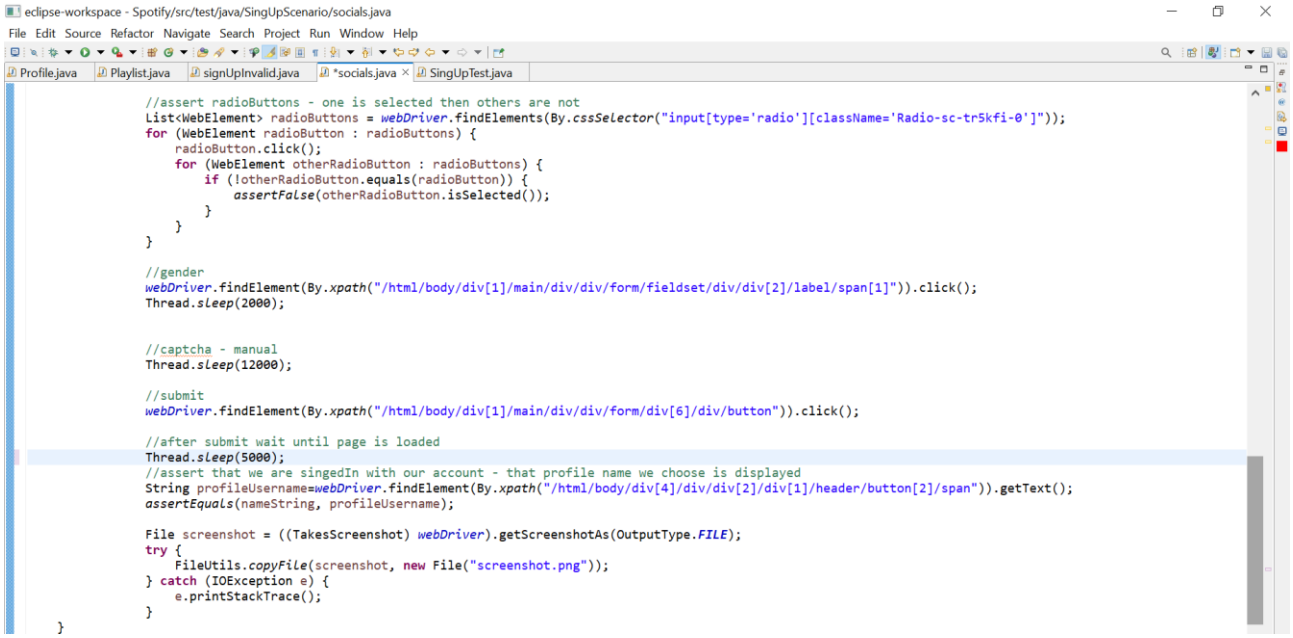


Test Name: Create an Account with Socials				
Description: Verify that users create an account using social media accounts				
Pre-condition(s): User has a valid social media account				
Test Steps: 1.Open the Spotify website and click on the “Sign up” button. 2. Click on the “Sing up with Google” button. 3. Log in to the Google Account 4. Fill in the form 5. Assert that the user has been logged in and redirected to the home page	**Test Data:** Google account credentials	**Expected Result:** The user is able to successfully sign up using their Google account and is redirected to the home page	**Actual Result:** The user is able to successfully sign up using their Google account and is redirected to the home page	**Status:** PASS
Notes: Similar steps for other social media accounts like Facebook				






```java
    @Test
    void singUpWithInvalidInput() throws InterruptedException {

        webDriver.get(baseUrl);


        //click on the sing up for free button
        webDriver.findElement(By.xpath("/html/body/div[3]/div/div[2]/div[2]/footer/div[1]/button/span")).click();
        //wait for page to load
        Thread.sleep(2000);

        //sing up with goooogle
        webDriver.findElement(By.partialLinkText("Google")).click();
        Thread.sleep(2000);

        //pick account
        webDriver.findElement(By.xpath("/html/body/div[1]/div[1]/div[2]/div/div[2]/div/div/div[2]/div/div[1]/div/form/span/section/div/div/div/div/ul/li[2]/div")).cl
        Thread.sleep(2000);

        String nameString=webDriver.findElement(By.name("displayname")).getAttribute("required value");

        //day
                webDriver.findElement(By.name("day")).sendKeys("2");

                //month
                Select monthSelect=new Select(webDriver.findElement(By.name("month")));
                monthSelect.selectByVisibleText("October");

                //year
                webDriver.findElement(By.name("year")).sendKeys("2000");


                //assert radioButtons - one is selected then others are not
                List<WebElement> radioButtons = webDriver.findElements(By.cssSelector("input[type='radio'][className='Radio-sc-tr5kfi-0']"));
                for (WebElement radioButton : radioButtons) {
                    radioButton.click();
```




```java
                //assert radioButtons - one is selected then others are not
                List<WebElement> radioButtons = webDriver.findElements(By.cssSelector("input[type='radio'][className='Radio-sc-tr5kfi-0']"));
                for (WebElement radioButton : radioButtons) {
                    radioButton.click();
                    for (WebElement otherRadioButton : radioButtons) {
                        if (!otherRadioButton.equals(radioButton)) {
                            assertFalse(otherRadioButton.isSelected());
                        }
                    }
                }

                //gender
                webDriver.findElement(By.xpath("/html/body/div[1]/main/div/div/form/fieldset/div/div[2]/label/span[1]")).click();
                Thread.sleep(2000);


                //captcha - manual
                Thread.sleep(12000);

                //submit
                webDriver.findElement(By.xpath("/html/body/div[1]/main/div/div/form/div[6]/div/button")).click();

                //after submit wait until page is loaded
                Thread.sleep(5000);
                //assert that we are singedIn with our account - that profile name we choose is displayed
                String profileUsername=webDriver.findElement(By.xpath("/html/body/div[4]/div/div[2]/div/div[1]/header/button[2]/span")).getText();
                assertEquals(nameString, profileUsername);

                File screenshot = ((TakesScreenshot) webDriver).getScreenshotAs(OutputType.FILE);
                try {
                    FileUtils.copyFile(screenshot, new File("screenshot.png"));
                } catch (IOException e) {
                    e.printStackTrace();
                }
    }
```

3.2. Login Functionality

User want to login to their account.

Test Name: Login with Email and Password				
Description: Verify that users can login to their account using their email and password				
Pre-condition(s): User has an exciting account with Spotify				
Test Steps: 1.Open the Spotify website and click on the “Login” button. 2. Fill in the required fields. 3. Click on the “Login” button.	**Test Data:** Email=” zaratester717@gmail.com”, Password=uniburch7,	**Expected Result:** User is successfully logged in and redirected to their home page.	**Actual Result:** User is successfully logged in and redirected to their home page.	**Status:** PASS
Notes:				

Test Name: Login with Incorrect Credentials				
Description: Verify that users cannot login to their account with incorrect credentials				
Pre-condition(s): User has an exciting account with Spotify				
Test Steps: 1.Open the Spotify website and click on the “Login” button. 2. Fill in the required fields.	**Test Data:** Email=” zaratester717@gmail.com”, Password= “uniburch”	**Expected Result:** An error message is displayed and user is not logged in	**Actual Result:** An error message is displayed and user is not logged in	**Status:** PASS

3. Click on the “Login” button. 4. Verify that an error message is displayed				
Notes:				

Test Name: Login with Socials				
Description: Verify that users can login to their account with proper social media credentials				
Pre-condition(s): User has an account that has been linked to a social media account				
Test Steps: 1.Open the Spotify website and click on the “Login” button. 2. Fill in the required fields. 3. Click on the “Login” button.	**Test Data:** Google account credentials	**Expected Result:** The user is successfully logged in using their Google account and is redirected to the home page	**Actual Result:** The user is successfully logged in using their Google account and is redirected to the home page	**Status:** PASS
Notes: Similar steps for other social media accounts like Facebook				

Test Name: Verify Logout Functionality				
Description: Verify that users can logout from their account				
Pre-condition(s): User is logged in to their account				
Test Steps: 1.Click profile 2. Select “logout” button. 3. Assert that the user has been logged out.	**Test Data:** Google account credentials	**Expected Result:** The user is successfully logged out.	**Actual Result:** The user is successfully logged out.	**Status:** PASS
Notes:				

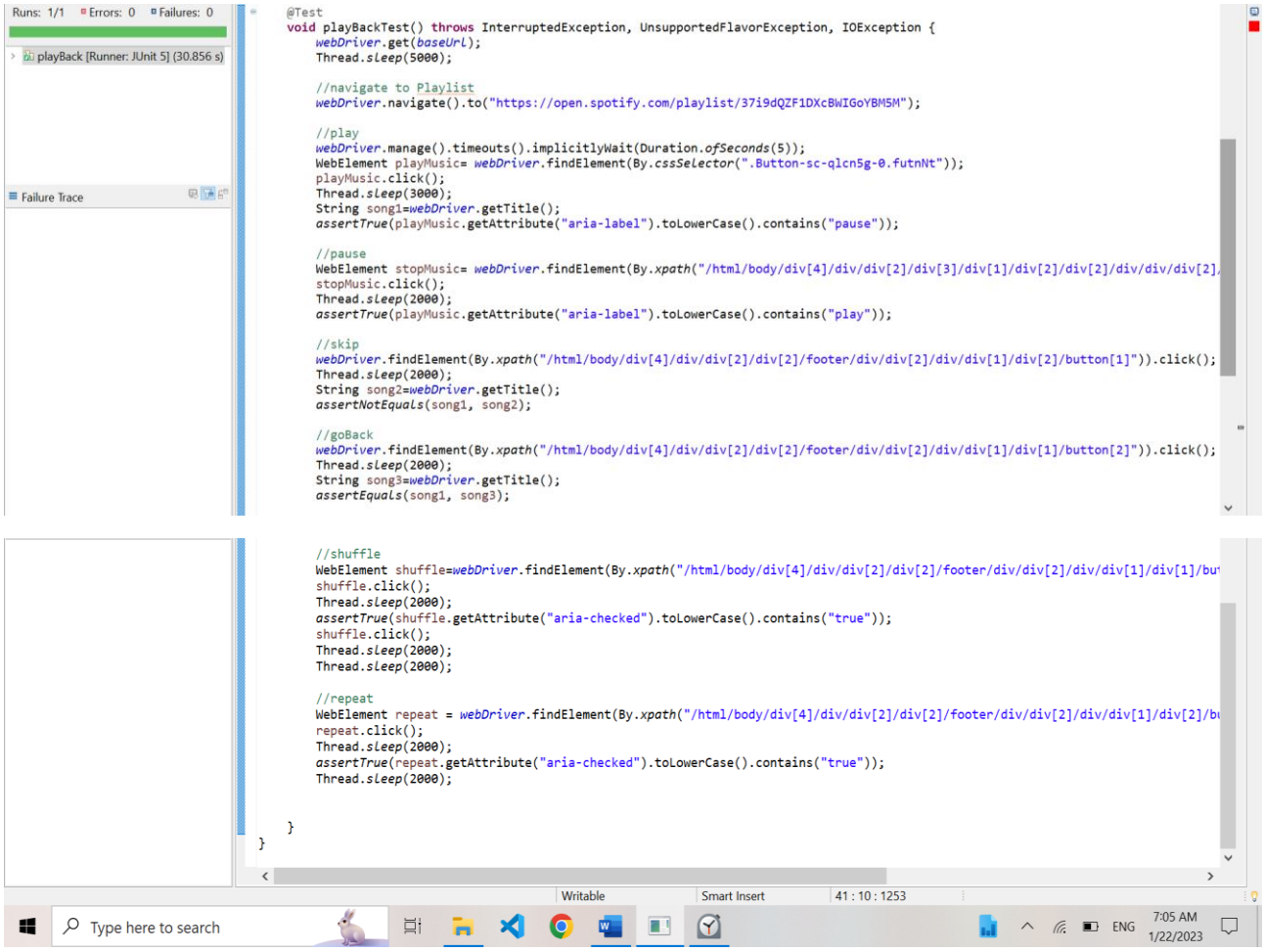
Test Name: Forgotten Password				
Description: Verify that users can reset their password if they have forgotten it.				
Pre-condition(s): The user has an exciting account with Spotify				
Test Steps: 1.Open the Spotify website and click on the “Login” button. 2. Click on the "forgot password" link. 3.Enter a valid email and click the "send” button. 4.Confirm that an email has been successfully send	**Test Data:** Email=” zaratester717@gmail.com”,	**Expected Result:** User is able to reset their password	**Actual Result:** User is able to reset their password	**Status:** PASS
Notes:				

3.3. Playback Functionality

User want to play a song

Test Name: Verify Playback of Playlist
Description: Test the functionality of playing a playlist, pausing a playlist, skipping a song, shuffling songs, and repeating a song
Pre-condition(s): A user has selected a playlist

Test Steps:	Test Data:	Expected Result:	Actual Result:	Status:
1.Click “play” 2.Click “pause” 3.Click “skip” 4.Click “go Back” 5. Click “Shuffle” 5.Click “Repeat”		The song plays, pauses, skips, shuffles and repeats.	The song plays, pauses, skips, shuffles and repeats.	PASS
Notes: Similar steps for playing an album, radio, song…				

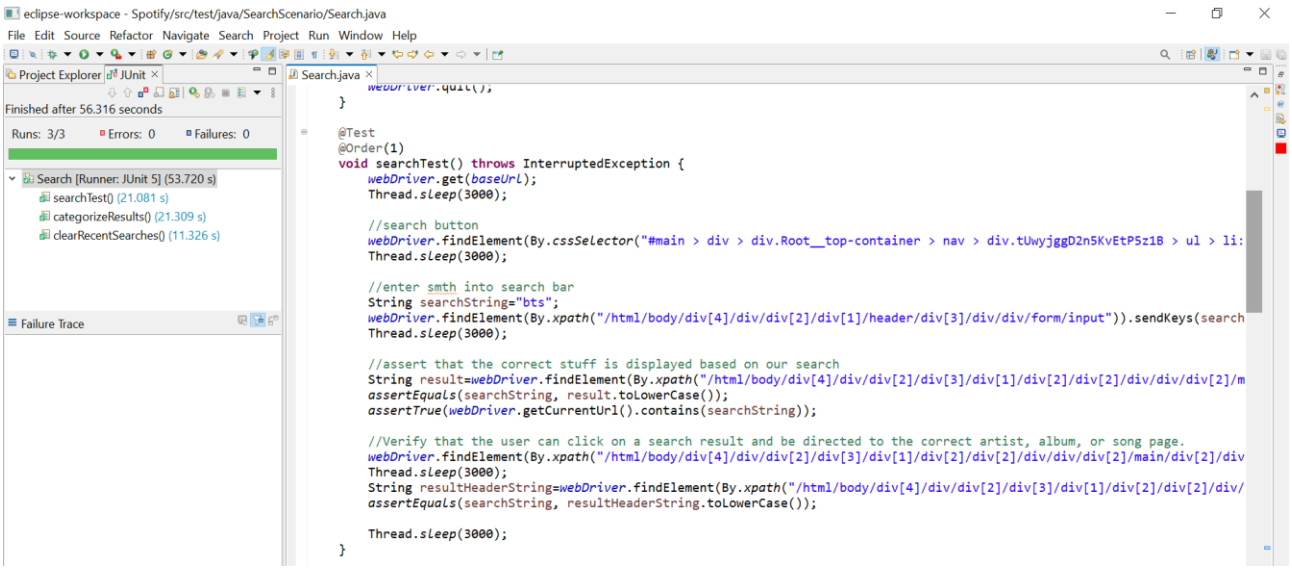


3.4. Search and Browse Functionality

This test scenario aims to verify that a user can search through the music library for a specific artist, album, song or browse the library through genres.

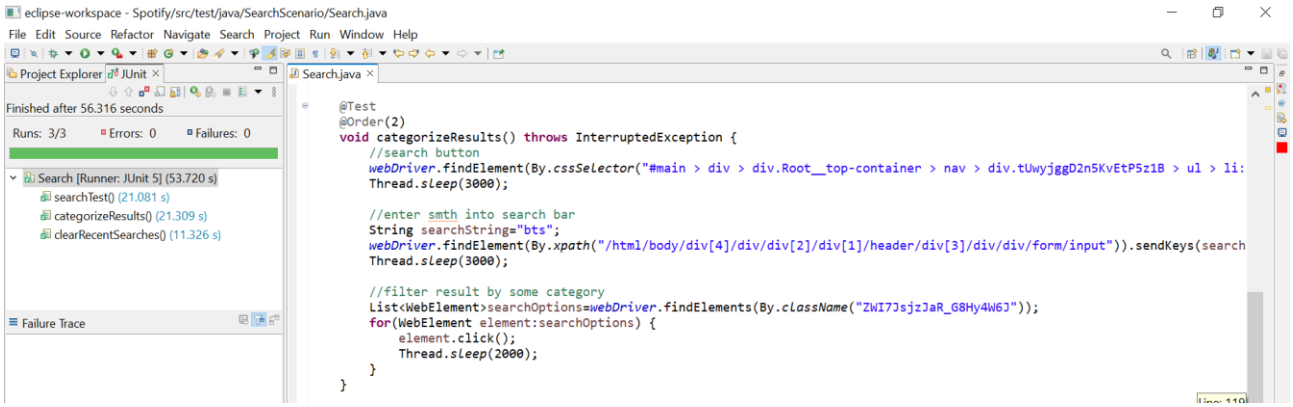
Test Name: Verify Search Functionality

Description: Test the search functionality and verify the correct results are displayed				
Pre-condition(s): A user is logged in to their account				
Test Steps: 1. Click the “search” button 2. Enter data into search bar 3. Verify that the correct results are displayed 4. Verify that the results are clickable and lead to correct path	**Test Data:** Search data = “BTS”	**Expected Result:** The search result should display correct result based on entered data.	**Actual Result:** The search result should display correct result based on entered data.	**Status:** PASS
Notes:				

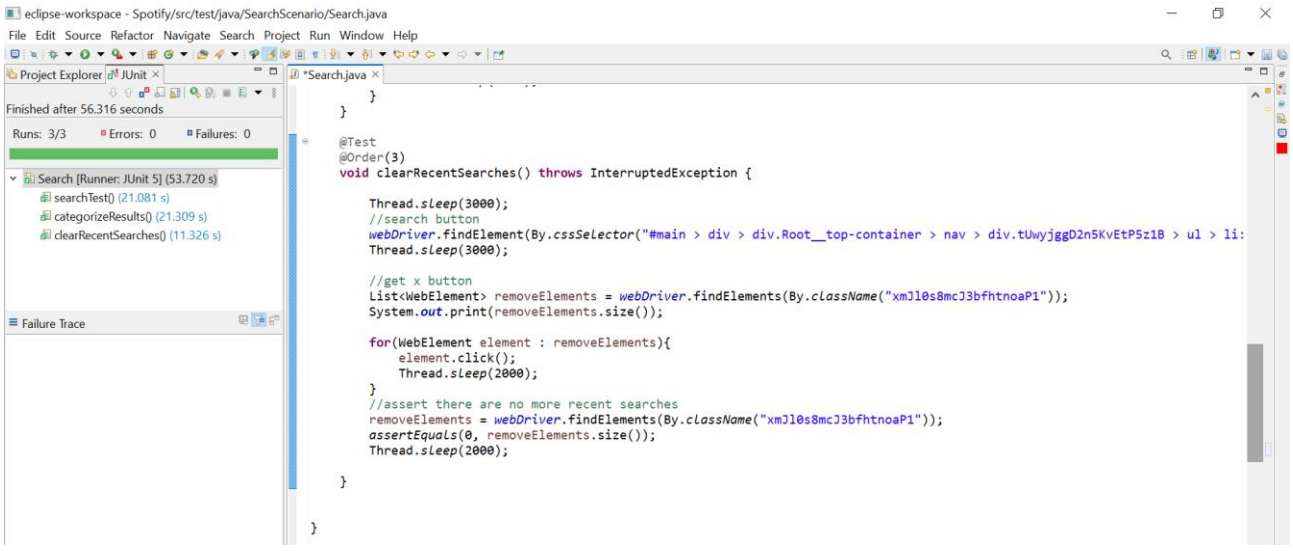


Test Name: Verify Search by Categorizes				
Description: Test the functionality of searching by category (e.g., album, song.)				
Pre-condition(s): A user is logged in to their account				
Test Steps: 1. Click the “search” button. 2. Enter data into search bar	**Test Data:**	**Expected Result:** The searched data should be correctly filtered and displayed by selected category	**Actual Result:** The searched data is correctly filtered and displayed by selected category	**Status:** PASS

3. Test the feature to filter result by category				
Notes:				



Test Name: Verify Clear Search Functionality				
Description: Test the functionality of clearing recent searched				
Pre-condition(s): A user is logged in to their account and has made at least one prior search				
Test Steps: 1.Click on “search” button 2. Click on “remove” button 3.Verify there are no more “recent searches”	**Test Data:**	**Expected Result:** No more “recent searches” displayed	**Actual Result:** No more “recent searches” displayed	**Status:** PASS
Notes:				



Test Name: Verify Browsing Functionality				
Description: Test that user can browse through the music library by some genre/option				
Pre-condition(s): A user is logged in to their account				
Test Steps: 1. Click the search link 2. Verify that each prelisted option for “browsing” is clickable and leads to correct path	**Test Data:** ,	**Expected Result:** The genres are all valid, clickable and lead to a correct path	**Actual Result:** The genres are all valid, clickable and lead to correct a path	**Status:** PASS
Notes:				

3.5. Localization

This test scenario aims to verify that the website is able to display content in different languages and that the user can switch between languages easily

Test Name: Verify Localization for Different Languages				
Description: Test that the website is properly localized for different languages				
Pre-condition(s): The website should be available for different languages and user is logged in.				
Test Steps: 1.Open website 2.Go to user settings 3.Select a different language from the dropdown menu	**Test Data:** Language preference= “de” and “en”	**Expected Result:** Website's language is changed to the selected language.	**Actual Result:** Website's language is changed to the selected language	**Status:** PASS

4. Verify that the website is properly localized for the selected language. 5. Repeat step 3				
Notes:				

3.6. Accessibility

This test scenario aims to check if the website is accessible for users.

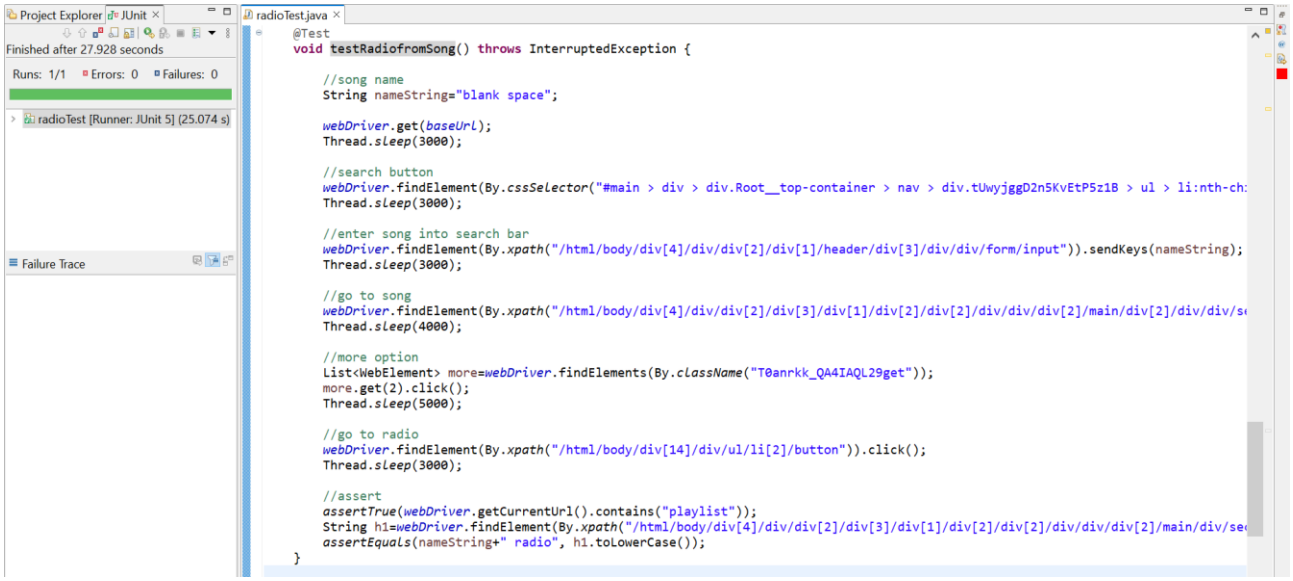
Test Name: Verify Website's Accessibility				
Description: Verify that the website is accessible for users with disabilities / compatible with different screen readers / compatible with keyboard navigation				
Pre-condition(s): None				
Test Steps: 1. Navigate to the website 2. Run the Axe accessibility tool on the website 3. Check the results for any violations	**Test Data:**	**Expected Result:** No accessibility violations found.	**Actual Result:** Accessibility violations found.	**Status:** FAIL
Notes:				

3.7. Radio Functionality

This test scenario aims to verify that user is able to access and use the radio features, and based on a particular playlist/song/artist that is selected, a new playlist can be created with the same vibes and saved to user's library.

Test Name: Verify Radio Creation from a Playlist				
Description: Test the functionality of creating a radio playlist from a certain playlist				
Pre-condition(s): A user is logged in and has a playlist selected				
Test Steps: 1.Select playlist 2. Click the "more" button 3.Click the "go to radio" button. 4.Verify that you are redirected to that playlist's radio playlist	**Test Data:**	**Expected Result:** The user is able create and listen to a playlist based on a specific playlist.	**Actual Result:** The user is able create and listen to a playlist based on a specific playlist.	**Status:** PASS
Notes:				

Test Name: Verify Radio Creation from a Song				
Description: Test the functionality of creating a radio playlist from a certain song				
Pre-condition(s): A user is logged in and has a song selected				
Test Steps: 1. Select song 2. Click the “more” button 3.Click the “go to radio” button. 4.Verify that you are redirected to that song’s radio playlist	**Test Data:** Name of Song = 1989	**Expected Result:** The user is able create and listen to a playlist based on a specific song.	**Actual Result:** The user is able create and listen to a playlist based on a specific song.	**Status:** PASS
Notes:				



Test Name: Verify Radio Creation from an Artist				
Description: Test the functionality of creating a radio playlist from a certain artist				
Pre-condition(s): A user is logged in and has an artist selected				
Test Steps: 1. Select Artist 2. Click the “more” button 3.Click the “go to radio” button. 4.Verify that you are redirected to that artist’s radio playlist	**Test Data:** Name of Artist = John Legend	**Expected Result:** The user is able create and listen to a playlist based on a specific artist.	**Actual Result:** The user is able create and listen to a playlist based on a specific artist.	**Status:** PASS
Notes:				

Test Name: Verify Saving Radio to Personal Playlist				
Description: Test the functionality of saving a radio playlist to user’s personal playlist collection				
Pre-condition(s): A user is logged in and has a radio playlist created.				
Test Steps: 1. Go to radio playlist 2. Click “more” button 3. Click “add to playlist” button	**Test Data:**	**Expected Result:** The user is able save a radio playlist to their personal playlist collection	**Actual Result:** The user is able save a radio playlist to their personal playlist collection	**Status:** PASS

4.Click “create new playlist” button 5. Wait for the confirmation message pop-up.				
Notes:				

3.8. Profile Management

Test scenario to verify that user can view their account and modify it.

Test Name: Verify Account Settings				
Description: Test that a user can manage their account settings, including email and push notifications				
Pre-condition(s): A user has an account and is logged in				
Test Steps: 1. Click on the “user” tab. 2. Click on the “settings” link. 3. Update account information. 4. Click on “save changes” button. 5. Verify that the account setting have been successfully changed.	**Test Data:**	**Expected Result:** The user can view their settings and successfully update it.	**Actual Result:** The user can view their settings and successfully update it.	**Status:** PASS

Notes:

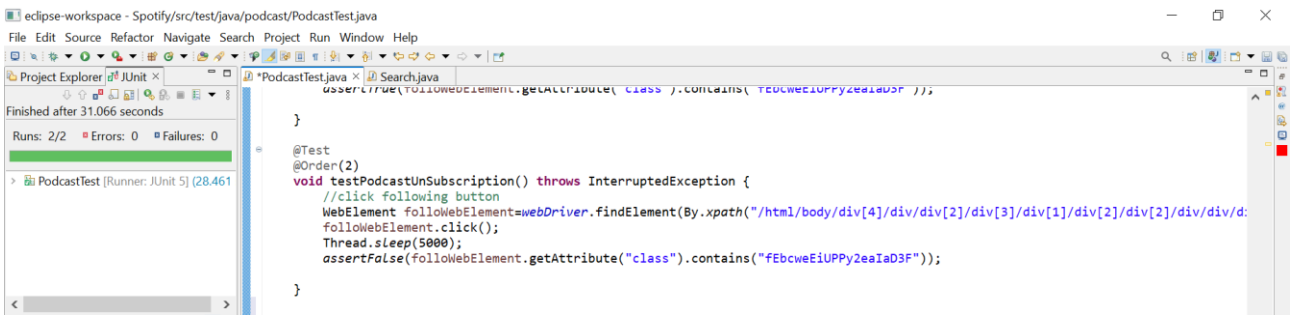
Test Name: Verify Account Settings Update				
Description: Verify that a user can update their account settings				
Pre-condition(s): A user has an account and is logged in				
Test Steps: 1. Click on the “user” tab 2. Click on the “settings” link 3.Click on “edit” link 4. Change gender setting. 5. Verify that the settings have successfully updated.	**Test Data:** Gender = “other”	**Expected Result:** The account settings are updated and saved successfully	**Actual Result:** The account settings are updated and saved successfully.	**Status:** PASS
Notes:				

3.9. Podcast Functionality

This test scenario aims to verify that user can subscribe to podcasts, play episodes, un-subscribe, etc...

Test Name: Verify Podcast Subscription				
Description: Verify that a user can subscribe to a podcast				
Pre-condition(s): A user is logged in				
Test Steps: 1. Navigate to “Podcast” tab. 2. Pick desired genre 3.Select the first podcast. 3. Click on the “follow” button. 4. Wait for the success pop-up notification and assert that user has successfully subscribed by checking class name.	**Test Data:** Genre = “Documentary”	**Expected Result:** The user is successfully subscribed to the podcast.	**Actual Result:** The user is successfully subscribed to the podcast.	**Status:** PASS
Notes:				

Test Name: Verify Podcast Un-Subscription				
Description: Verify that a user can un-subscribe to a podcast				
Pre-condition(s): A user is logged in and is subscribed to at least one podcast				
Test Steps: 1. Select podcast. 4. Wait for the success pop-up notification and assert that user has successfully unsubscribed by checking class name.	**Test Data:**	**Expected Result:** The user is successfully un-subscribed to the podcast.	**Actual Result:** The user is successfully un-subscribed to the podcast.	**Status:** PASS
Notes:				



Test Name: Verify Episode Functionality				
Description: Test the functionality of playing an episode, pausing an episode, adding it to the library				
Pre-condition(s): A user has selected an episode				
Test Steps: 1.Click “play” 2.Click “pause”	**Test Data:**	**Expected Result:** The episode plays, pauses, and can be added to library	**Actual Result:** The episode plays, pauses, and can be added to library.	**Status:** PASS

3. Add to library 4. Verify that the episode has been added to library				
Notes:				

Test Name: Test Podcast Shareability				
Description: Verify that user can generate a link to share podcast				
Pre-condition(s):				
Test Steps: 1.Navigate to the podcast you wish to share 2. Click on the share button. 3. Select the option to share the playlist by copying link 3. Assert that the copied link leads to the episode in question	**Test Data:**	**Expected Result:** The link to podcast has been successfully copied.	**Actual Result:** The link to podcast has been successfully copied.	**Status:** PASS
Notes:				

3.10. Library Functionality

Testing that the user can access their library and see their collection based on certain category

Test Name: Verify Library				
Description: Test the functionality of accessing a library and browsing through categories.				
Pre-condition(s): A user is logged in and has a song/album/playlist selected				
Test Steps: 1.Click “My Library” link 2. Browse depending on category 3.Verify that the correct link is accessed.	**Test Data:**	**Expected Result:** The library is accessed, and the user can see their collection, based on common category.	**Actual Result:** The library is accessed, and the user can see their collection, based on common category..	**Status:** PASS
Notes:				

3.11. Playlist Functionality

Testing the playlist functionalities, creating a playlist, deleting it, sharing it, adding songs, etc.…

Test Name: Verify Playlist Creation				
Description: Verify Playlist creation				
Pre-condition(s): A user is logged in to their account				
Test Steps: 1. Click on the “Create Playlist” button 2. Assert that the library has been created	**Test Data:**	**Expected Result:** A new playlist is created and added to the user's library	**Actual Result:** A new playlist is created and added to the user's library	**Status:** PASS
Notes:				

Test Name: Verify Song Addition to a Playlist				
Description: Test the ability for a user to add songs to an existing playlist				
Pre-condition(s): A user is logged in to their account, and has at least one existing playlist				
Test Steps: 1.Navigate to the playlist where you want to add song. 2. Search for songs you want to add and select them 3. Click on “add” button 4.Verify that the song has been added.	**Test Data:** Search song/artist= “bts”	**Expected Result:** The selected songs are added to the selected playlist	**Actual Result:** The selected songs are added to the selected playlist	**Status:** PASS
Notes:				

Test Name: Verify Song Deletion from Playlist				
Description: Test the ability for a user to delete songs from an existing playlist				
Pre-condition(s): A user is logged in to their account				
Test Steps: 1.Navigate to the playlist from where you want to delete the song. 2. Search for song you want to delete and select it 3. Press delete key 4. Verify that the song has been deleted.	**Test Data:**	**Expected Result:** The selected song is deleted from the selected playlist	**Actual Result:** The selected song is deleted form the select playlist	**Status:** PASS
Notes:				

Test Name: Test Playlist Editing				
Description: Verify that you can edit playlist				
Pre-condition(s): A user is logged in to their account and has at least one playlist				
Test Steps:	**Test Data:**	**Expected Result:**	**Actual Result:**	**Status:**

1. Navigate to your playlist 2. Click on the edit button 3. Once a popup appears, add text 4.Save it	Text= “Selenium”	The playlist has been edited	The playlist has been edited	PASS
Notes:				

Test Name: Test Sharing Link to a Playlist				
Description: Verify sharing link to a playlist				
Pre-condition(s): A user is logged in to their account, and has at least one playlist				
Test Steps: 1.Navigate to the playlist you wish to share 2. Click on the share button located on the playlist. 3. Select the option to share the playlist by copying link 3. Assert that the copied link leads to the playlist in question	**Test Data:** ,	**Expected Result:** The link to playlist has been successfully copied.	**Actual Result:** The link to playlist has been successfully copied.	**Status:** PASS
Notes:				

Test Name: Test Playlist Deletion				
Description: Verify the ability for a user to delete an existing playlist				
Pre-condition(s): A user is logged in to their account and has at least one playlist				
Test Steps: 1.Locate the playlist you want to delete 2.Click on the playlist and press the delete key 3. Click on the confirmation button in the dialog to delete the playlist. 4. Wait for the page to refresh and assert that the playlist is no longer present.	**Test Data:**	**Expected Result:** The selected playlist is deleted from the user's library	**Actual Result:** The selected playlist is deleted from the user's library	**Status:** PASS
Notes:				

4. Conclusion

4.1. Testing Summary

Testing Tool	Total Tests	Passed Tests	Failed Tests
Junit, Selenium Web Driver, aXe	32	31	1– testing Accessibility

4.2. Final Thoughts

In conclusion, the Spotify application was found to be well-implemented, with a user-friendly interface and a wide range of features. However, during testing, some issues were identified in terms of accessibility for users and compatibility with different screen readers, which suggests that some improvements could be made. Recommendations for the site would include conducting further testing in these areas and addressing any issues that are identified.

Overall, the application provides a seamless music streaming experience for users, with the ability to search for music, create playlists, and listen to radio stations.