DEVOPS PROJECT

Creating Spotify Playlists Using Terraform

Overview:

This project demonstrates how to use **Terraform to create Spotify playlists**. By following this guide, you will learn the fundamental Terraform concepts and use the Spotify provider to manage your playlists programmatically.

Prerequisites

1. **Terraform**: Install Terraform by following these steps:
    - Download Terraform from the official page
    - Install the appropriate version for your system.
    - Verify the installation:

```
1. terraform -version
```

2. **Spotify Account**:
    - Create a free Spotify account if you don't already have one at Spotify.
3. **Spotify Developer Account**:
    - Set up a developer account at Spotify for Developers.
4. **VS Code**:
    - Download and install Visual Studio Code.
5. **Docker**:
    - Ensure Docker Desktop is installed and running.

Step 1: Initialize the Project

1. Create Project Folder:

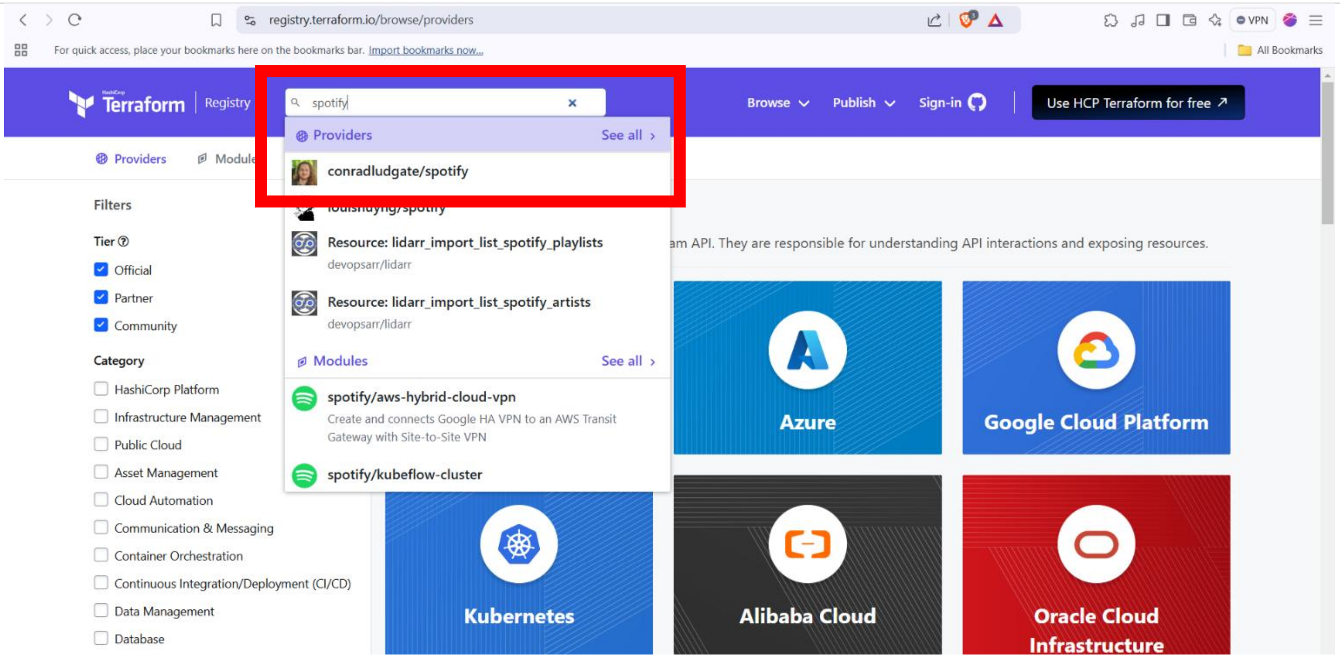
Open **VS Code** and **create a new folder** named **Spotify_TF**.

2. Set Up provider.tf:

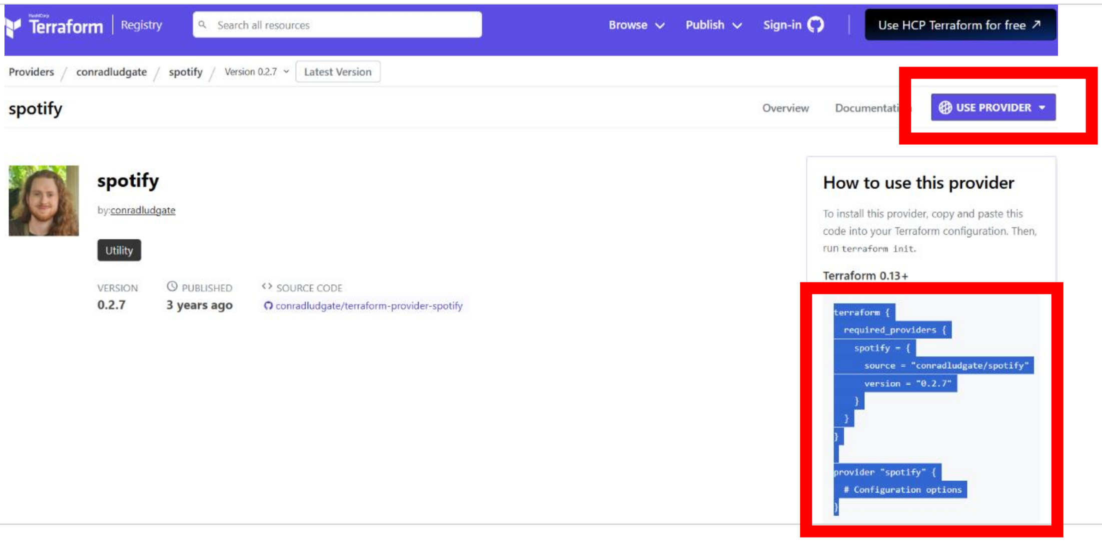
Create a file named provider.tf in the folder.

Go to the [Terraform Provider Registry](https://registry.terraform.io/browse/providers).

Search for "Spotify" and select the provider by conradludgate.


registry.terraform.io/browse/providers
Terraform | Registry
spotify
Providers See all
conradludgate/spotify
louishuyng/spotify
Resource: lidarr_import_list_spotify_playlists
devopsarr/lidarr
Resource: lidarr_import_list_spotify_artists
devopsarr/lidarr
Modules See all
spotify/aws-hybrid-cloud-vpn
Create and connects Google HA VPN to an AWS Transit Gateway with Site-to-Site VPN
spotify/kubeflow-cluster
Browse
Publish
Sign-in
Use HCP Terraform for free
Filters
Tier
Official
Partner
Community
Category
HashiCorp Platform
Infrastructure Management
Public Cloud
Asset Management
Cloud Automation
Communication & Messaging
Container Orchestration
Continuous Integration/Deployment (CI/CD)
Data Management
Database
Azure
Google Cloud Platform
Kubernetes
Alibaba Cloud
Oracle Cloud Infrastructure

Click **Use Provider** and copy the code snippet



To install this provider, copy and paste this code into your Terraform configuration. Then, run `terraform init`.

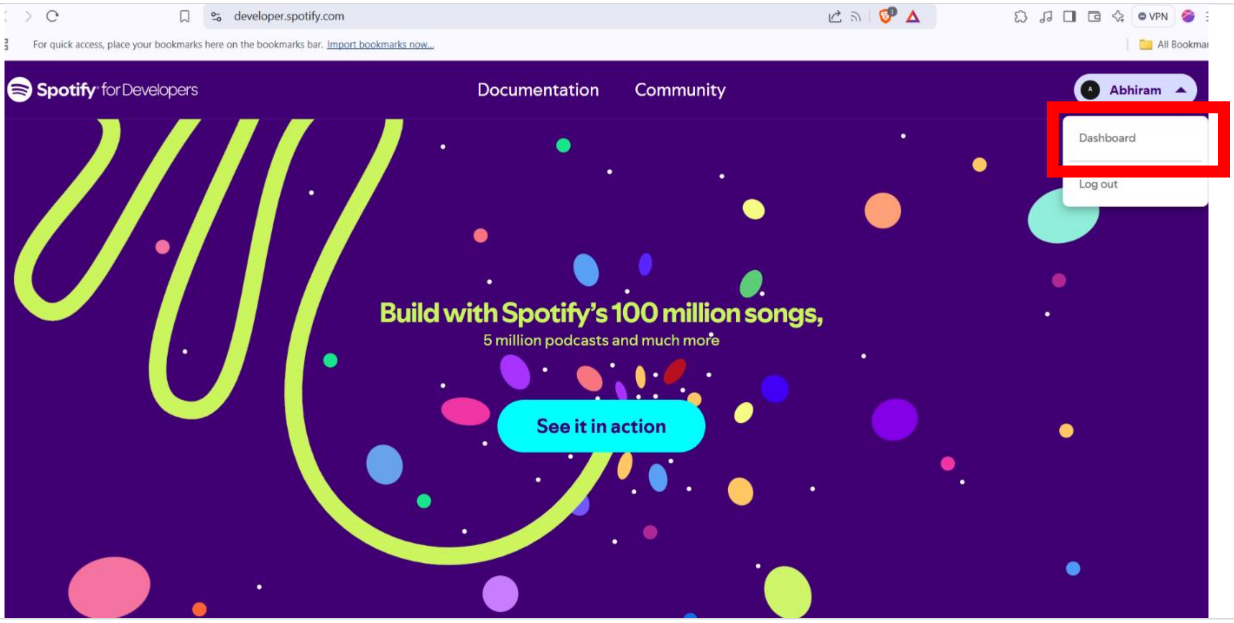
```
terraform {
  required_providers {
    spotify = {
      source = "conradludgate/spotify"
      version = "0.2.7"
    }
  }
}

provider "spotify" {
  # Configuration options
}
```

Paste it into the provider.tf file.

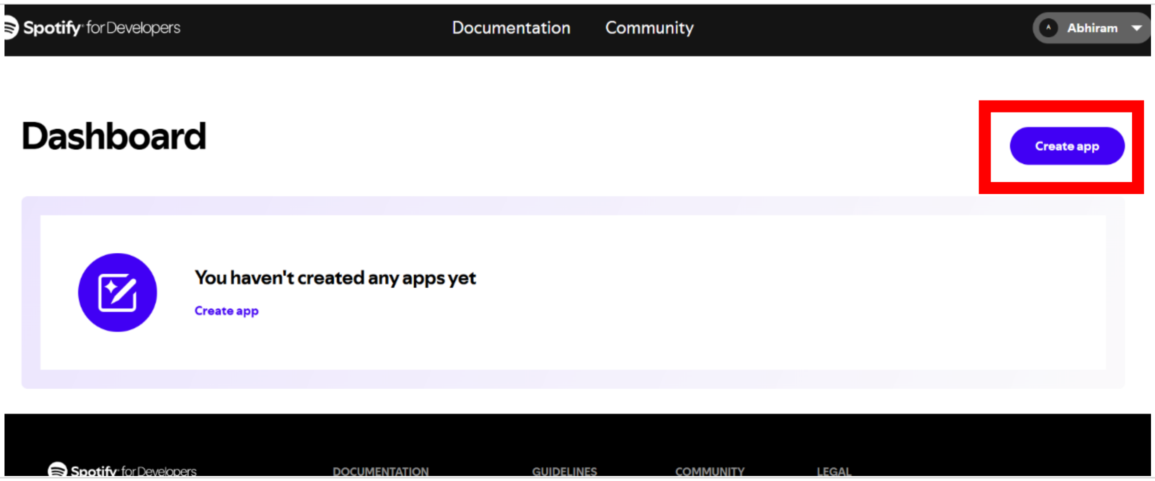
Step 2: Configure Spotify Developer App

1. Create an App:

Log in to **Spotify for Developers** & **Navigate to** the **Dashboard**


developer.spotify.com
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
All Bookmarks
Spotify for Developers
Documentation
Community
Abhiram
Dashboard
Log out
Build with Spotify's 100 million songs,
5 million podcasts and much more
See it in action

Click **Create App**



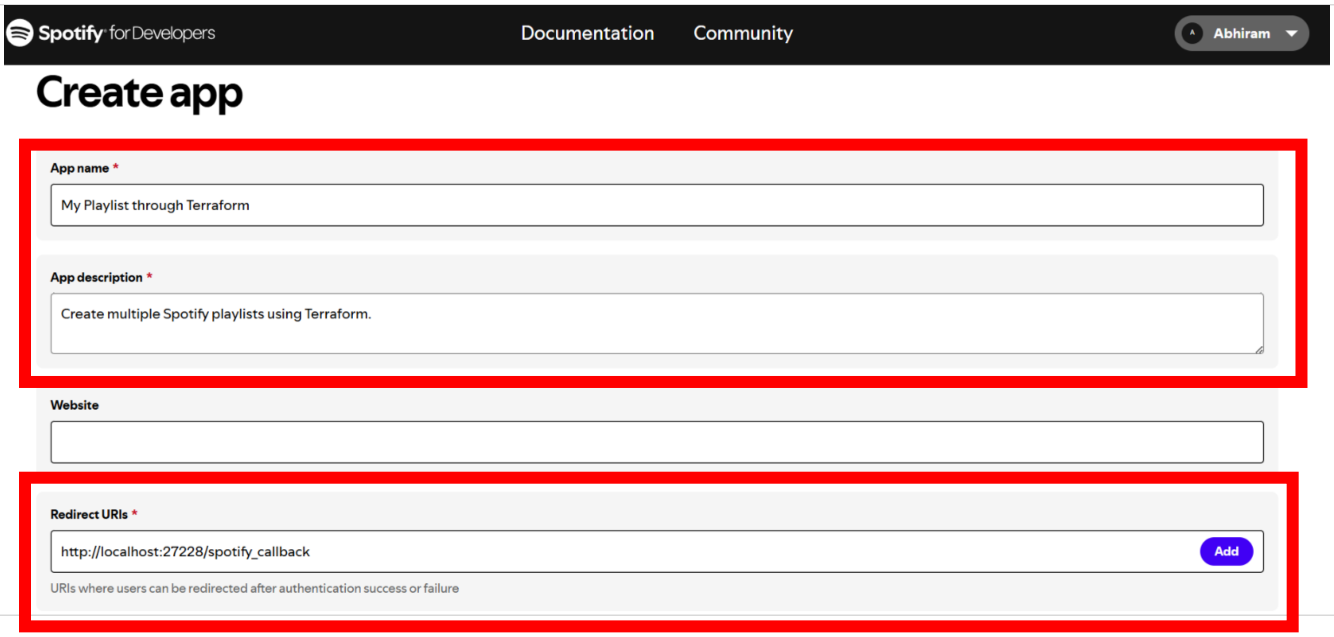
Dashboard

You haven't created any apps yet

Create app

Fill in the following details:

- **Name**: e.g.: My Playlist through Terraform
- **Description**: e.g.: Create multiple Spotify playlists using Terraform
- **Redirect URL**: http://localhost:27228/spotify_callback



Create app

App name *

My Playlist through Terraform

App description *

Create multiple Spotify playlists using Terraform.

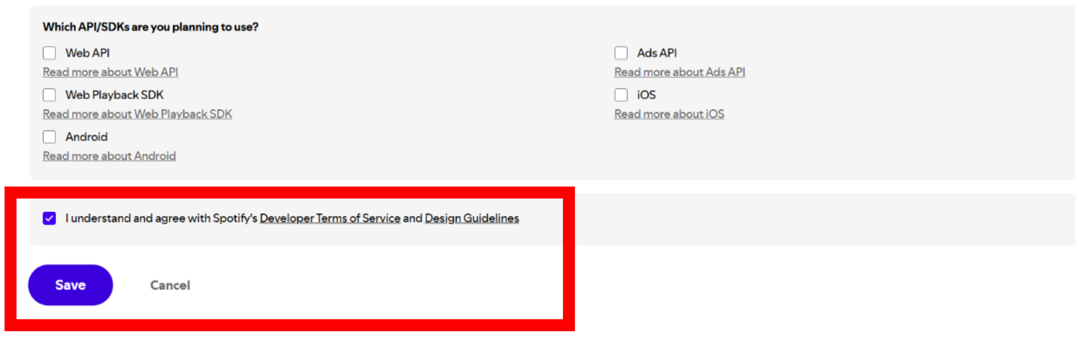
Website

Redirect URIs *

http://localhost:27228/spotify_callback

URIs where users can be redirected after authentication success or failure

Agree to the terms and click **Save**



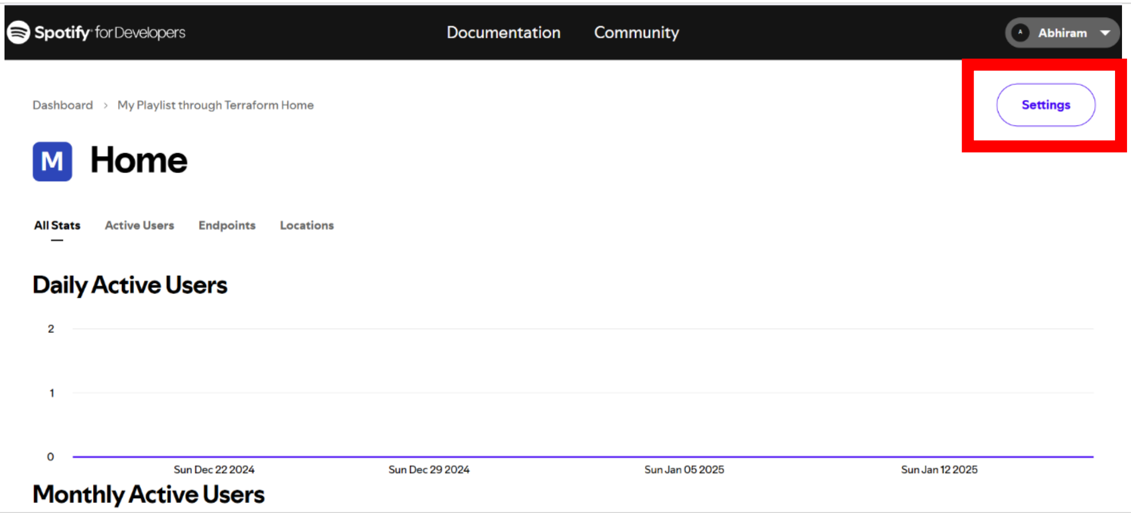
Which API/SDKs are you planning to use?

- [ ] Web API
  Read more about Web API
- [ ] Ads API
  Read more about Ads API
- [ ] Web Playback SDK
  Read more about Web Playback SDK
- [ ] iOS
  Read more about iOS
- [ ] Android
  Read more about Android

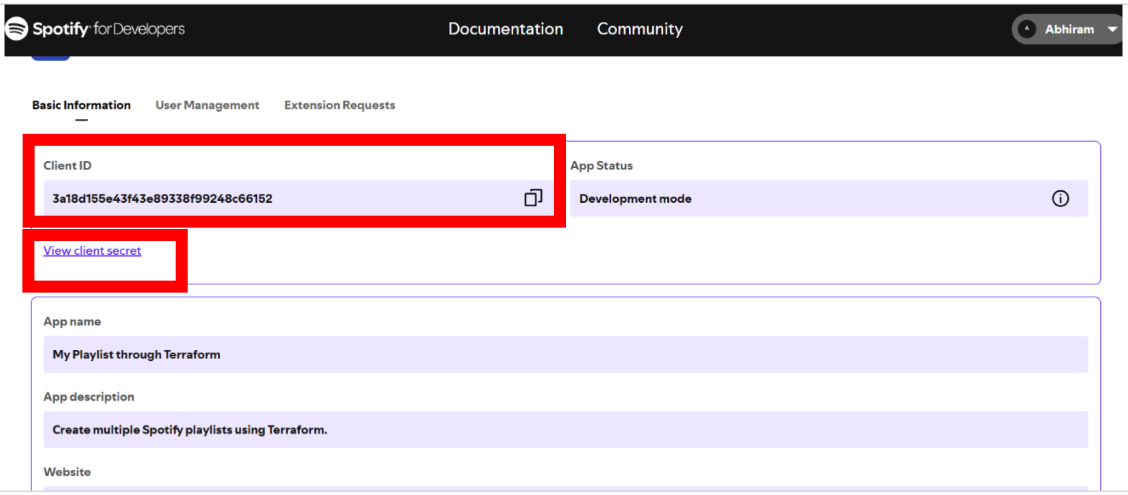
- [x] I understand and agree with Spotify's Developer Terms of Service and Design Guidelines

2. Retrieve Credentials:

Open the app **settings**



Copy the **Client ID** & **Client Secret**



Client ID: 3a18d155e43f43e89338f99248c66152

App Status: Development mode

App name: My Playlist through Terraform

App description: Create multiple Spotify playlists using Terraform.

Website

3. Store Credentials:

Create a new file .env in the Spotify_TF folder.

Add the following content, **replacing placeholders** with **your credentials**:

```
SPOTIFY_CLIENT_ID=<your_spotify_client_id>
SPOTIFY_CLIENT_SECRET=<your_spotify_client_secret>
```

Step 3: Run Spotify OAuth Proxy

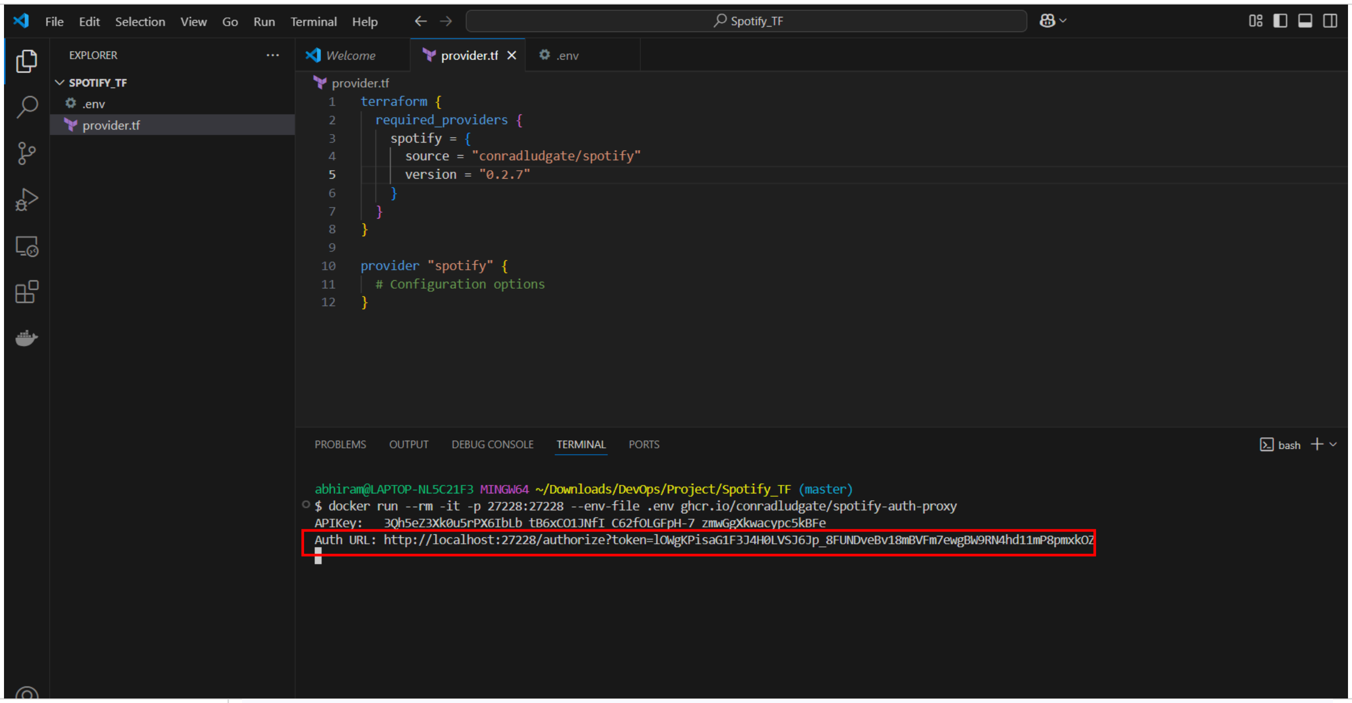
1. Start the Proxy Server:

Open terminal in VS Code and run this command:

```
docker run --rm -it -p 27228:27228 --env-file .env ghcr.io/conradludgate/spotify-
auth-proxy
```

2. Authorize the App:

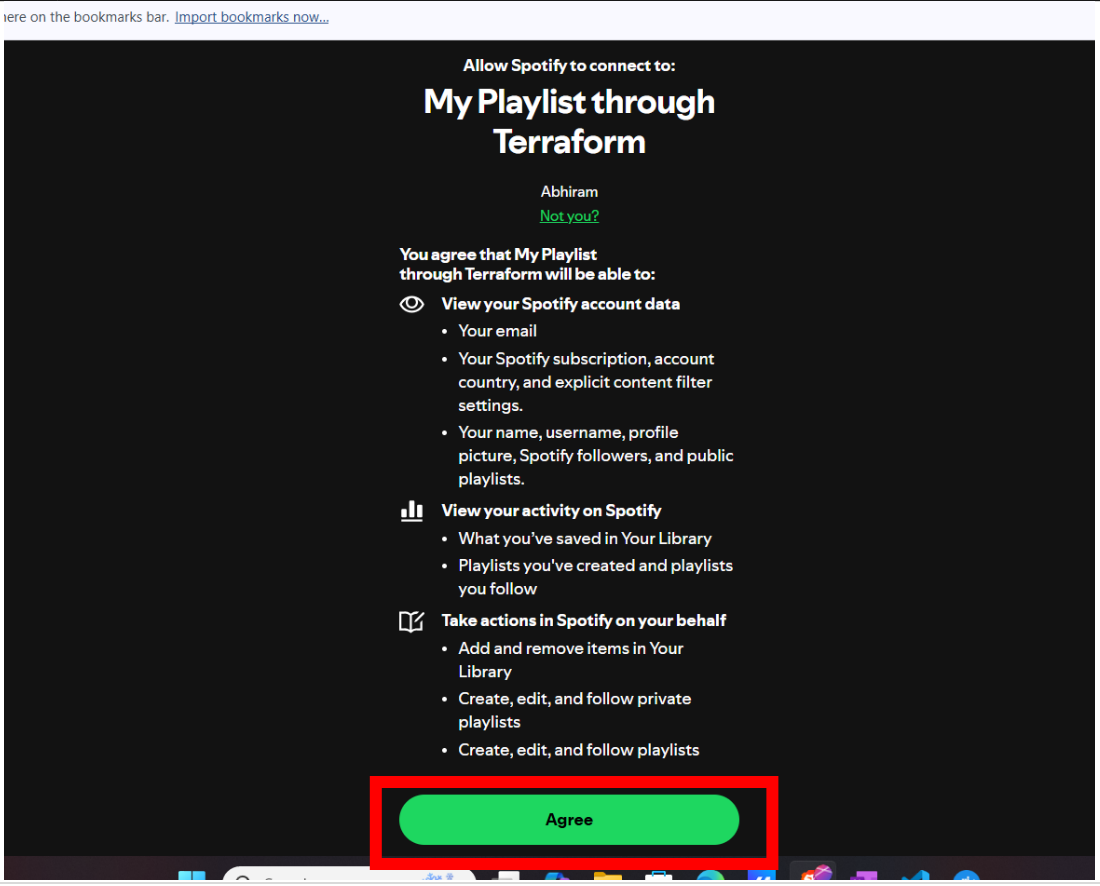
Copy the **Auth URL** from the **terminal output** and **open it in a browser**.



```
terraform {
  required_providers {
    spotify = {
      source = "conradludgate/spotify"
      version = "0.2.7"
    }
  }
}

provider "spotify" {
  # Configuration options
}
```

```
abhiram@LAPTOP-NL5C21F3 MINGW64 ~/Downloads/DevOps/Project/Spotify_TF (master)
$ docker run --rm -it -p 27228:27228 --env-file .env ghcr.io/conradludgate/spotify-auth-proxy
APIKey:   3Qh5eZ3Xk0u5rPX6IbLb_tB6xCO1JNfI_C62fOLGFpH-7_zmwGgXkwacypc5kBFe
Auth URL: http://localhost:27228/authorize?token=lOWgKPisaG1F3J4H0LVSJ6Jp_8FUNDveBv18mBVFm7ewgBW9RN4hd11mP8pmxkOZ
```

Click Agree on the authorization page.



ıere on the bookmarks bar. Import bookmarks now...

Allow Spotify to connect to:

My Playlist through Terraform

Abhiram

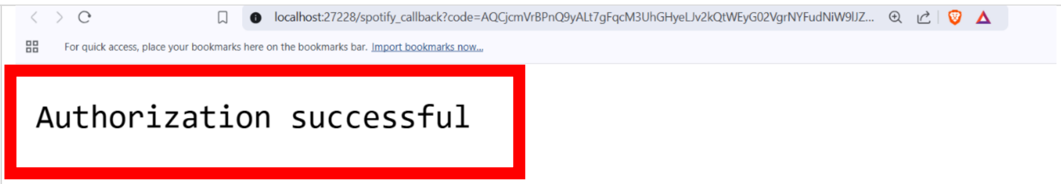
Not you?

You agree that My Playlist through Terraform will be able to:

View your Spotify account data
- Your email
- Your Spotify subscription, account country, and explicit content filter settings.
- Your name, username, profile picture, Spotify followers, and public playlists.

View your activity on Spotify
- What you've saved in Your Library
- Playlists you've created and playlists you follow

Take actions in Spotify on your behalf
- Add and remove items in Your Library
- Create, edit, and follow private playlists
- Create, edit, and follow playlists

Confirm the "**Authorization Successful**" message.	 localhost:27228/spotify_callback?code=AQCjcmVrBPnQ9yALt7gFqcM3UhGHyeLJv2kQtWEyG02VgrNYFudNiW9lJZ... For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now... Authorization successful

3. Retrieve API Key:

Note down the API key from the **terminal output**.

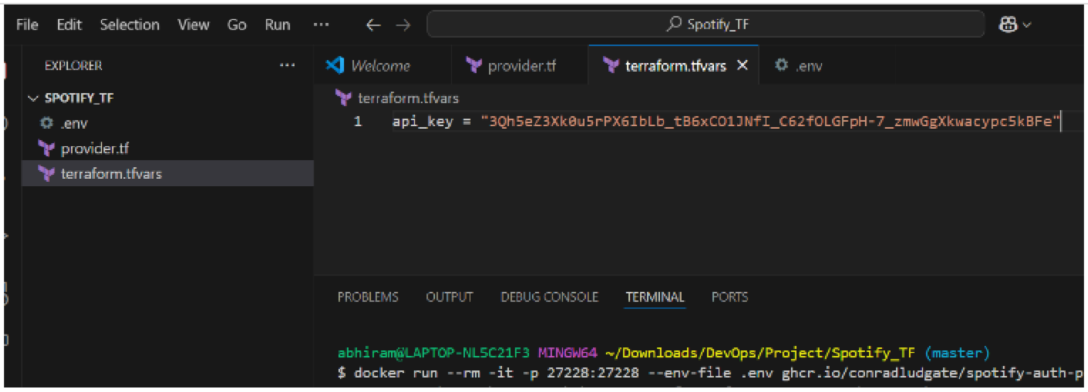
Step 4: Configure Terraform Variables

1. Create terraform.tfvars:

Create a New File named **terraform.tfvars**

Add the API key:

`api_key = "<APIKey>"`



```
api_key = "3Qh5eZ3Xk0u5rPX6IbLb_tB6xCO1JNfI_C62fOLGFpH-7_zmwGgXkwacypc5kBFe"
```

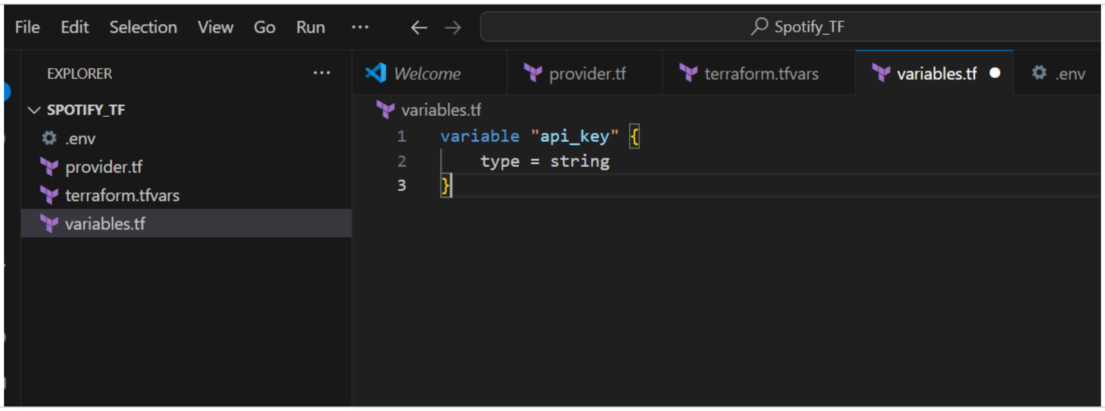
```
abhiram@LAPTOP-NL5C21F3 MINGW64 ~/Downloads/DevOps/Project/Spotify_TF (master)
$ docker run --rm -it -p 27228:27228 --env-file .env ghcr.io/conradludgate/spotify-auth-p
```

2. Create variables.tf:

Create a new file named **variables.tf**

Define the variable:

```
variable "api_key" {
  type = string
}
```

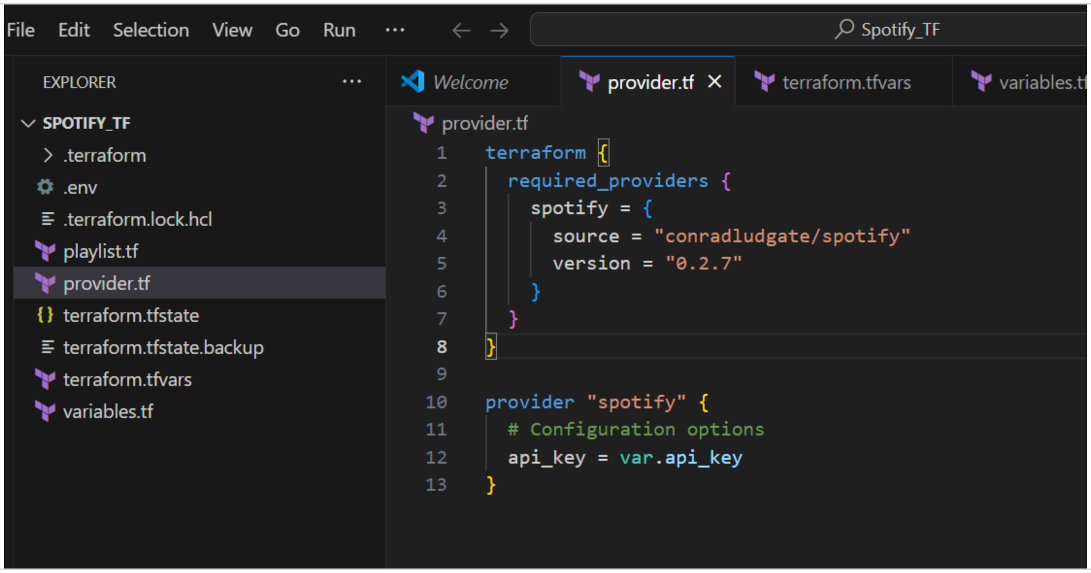


```
variable "api_key" {
    type = string
}
```

3. Update provider.tf:

Configure the provider with the **API key**:

```
provider "spotify" {
  api_key = var.api_key
}
```



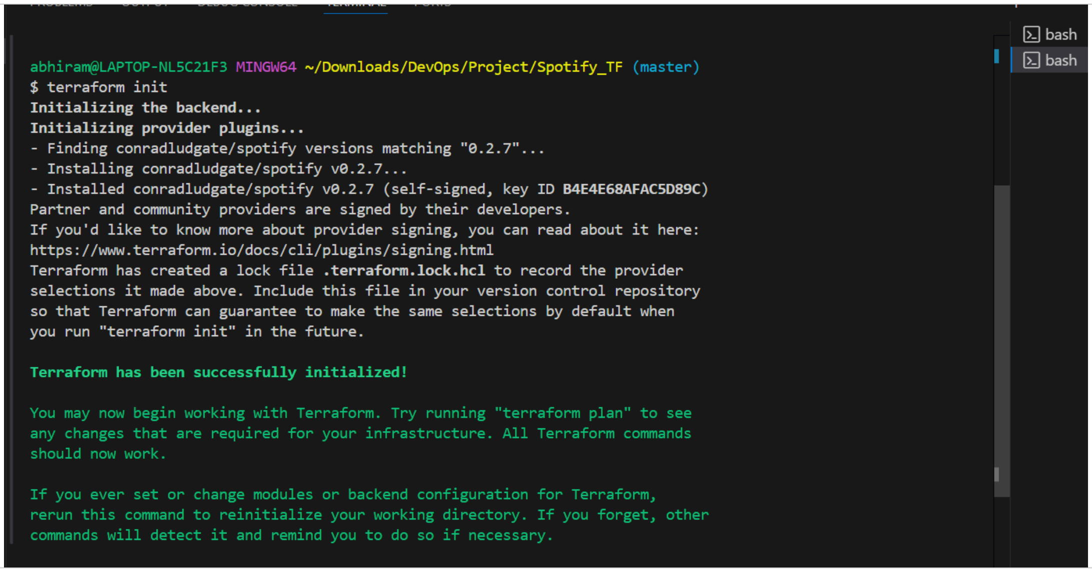
```
terraform {
  required_providers {
    spotify = {
      source = "conradludgate/spotify"
      version = "0.2.7"
    }
  }
}

provider "spotify" {
  # Configuration options
  api_key = var.api_key
}
```

Step 5: Initialize Terraform

1. Run Initialization Command:

Open a **terminal** and **execute**:

`terraform init`



```
abhiram@LAPTOP-NL5C21F3 MINGW64 ~/Downloads/DevOps/Project/Spotify_TF (master)
$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding conradludgate/spotify versions matching "0.2.7"...
- Installing conradludgate/spotify v0.2.7...
- Installed conradludgate/spotify v0.2.7 (self-signed, key ID B4E4E68AFAC5D89C)
Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

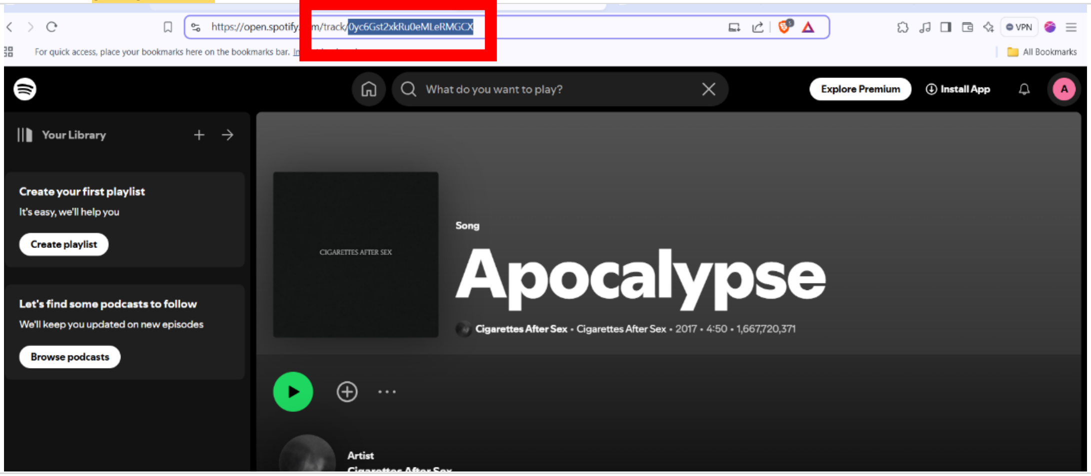
Step 6: Create Spotify Playlists

6.1 Static Playlist

1. Create playlist.tf

Create a New file named **playlist.tf**

Copy the ID of a **track** from **Spotify**



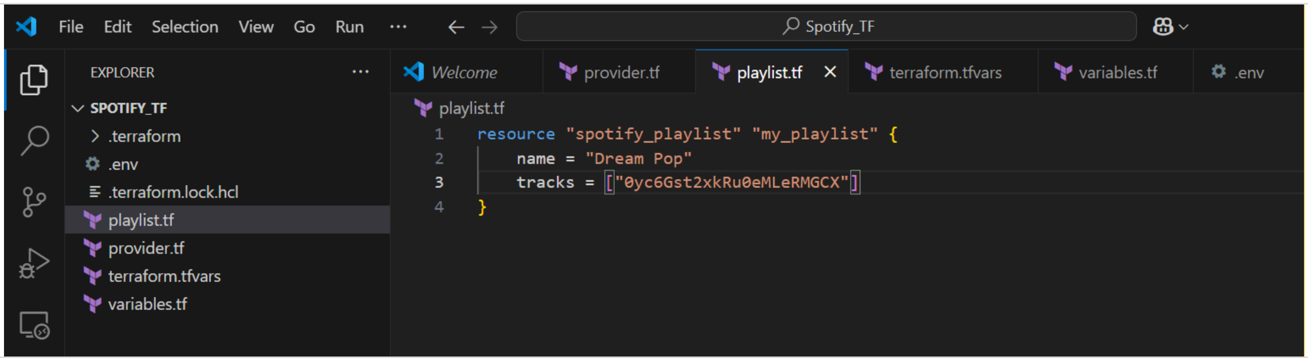
```
https://open.spotify.com/track/0yc6Gst2xkRu0eMLeRMGCX
```

For quick access, place your bookmarks here on the bookmarks bar.
All Bookmarks
VPN
What do you want to play?
Explore Premium
Install App
Your Library
Create your first playlist
It's easy, we'll help you
Create playlist
Let's find some podcasts to follow
We'll keep you updated on new episodes
Browse podcasts
CIGARETTES AFTER SEX
Song
Apocalypse
Cigarettes After Sex • Cigarettes After Sex • 2017 • 4:50 • 1,667,720,371
Artist
Cigarettes After Sex

Create a resource in playlist.tf

Use this code by **replacing** the Playlist name & Track ID

```
resource "spotify_playlist" "my_playlist" {
  name    = "<Playlist Name>"
  tracks  = ["<Track ID>"]
}
```

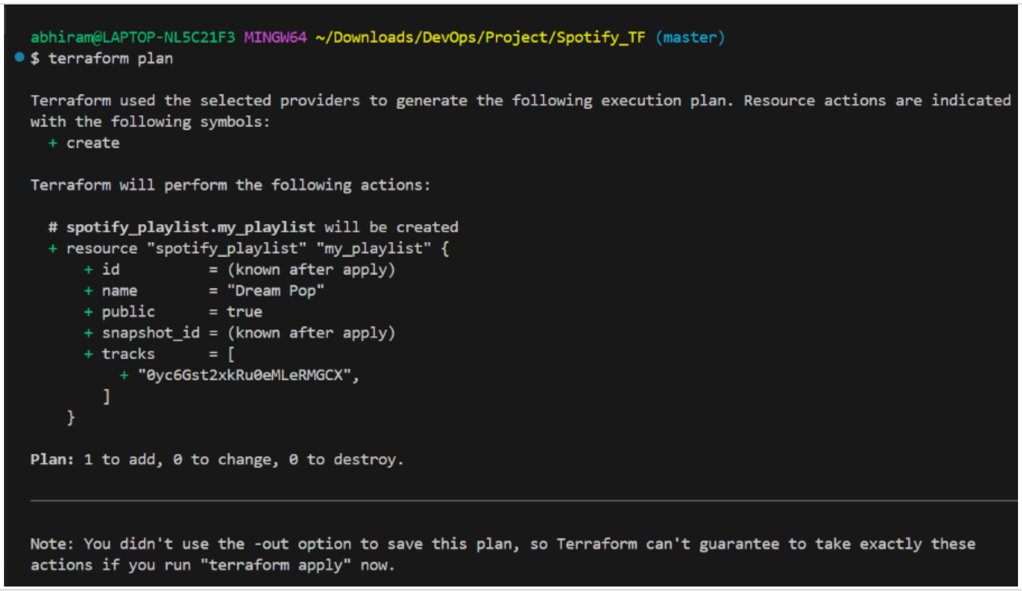


```
resource "spotify_playlist" "my_playlist" {
    name = "Dream Pop"
    tracks = ["0yc6Gst2xkRu0eMLeRMGCX"]
}
```

2. Preview and Apply:

Open a terminal and **execute**:

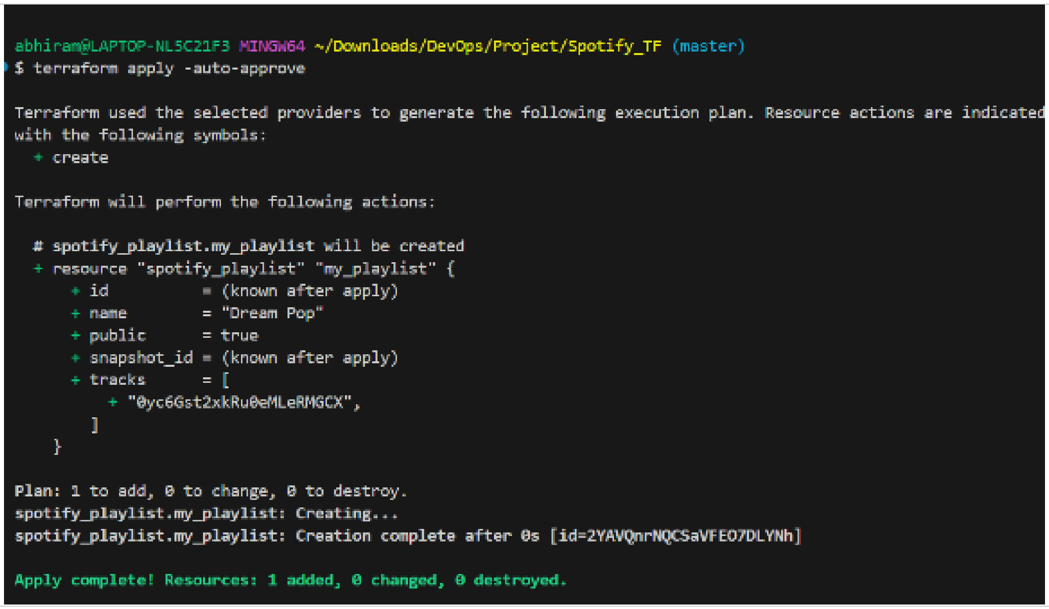
```
terraform plan
```



```
abhiram@LAPTOP-NL5C21F3 MINGW64 ~/Downloads/DevOps/Project/Spotify_TF (master)
$ terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated
with the following symbols:
  + create

Terraform will perform the following actions:

  # spotify_playlist.my_playlist will be created
  + resource "spotify_playlist" "my_playlist" {
      + id          = (known after apply)
      + name        = "Dream Pop"
      + public      = true
      + snapshot_id = (known after apply)
      + tracks      = [
          + "0yc6Gst2xkRu0eMLeRMGCX",
        ]
    }

Plan: 1 to add, 0 to change, 0 to destroy.

─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these
actions if you run "terraform apply" now.
```

Apply the changes:

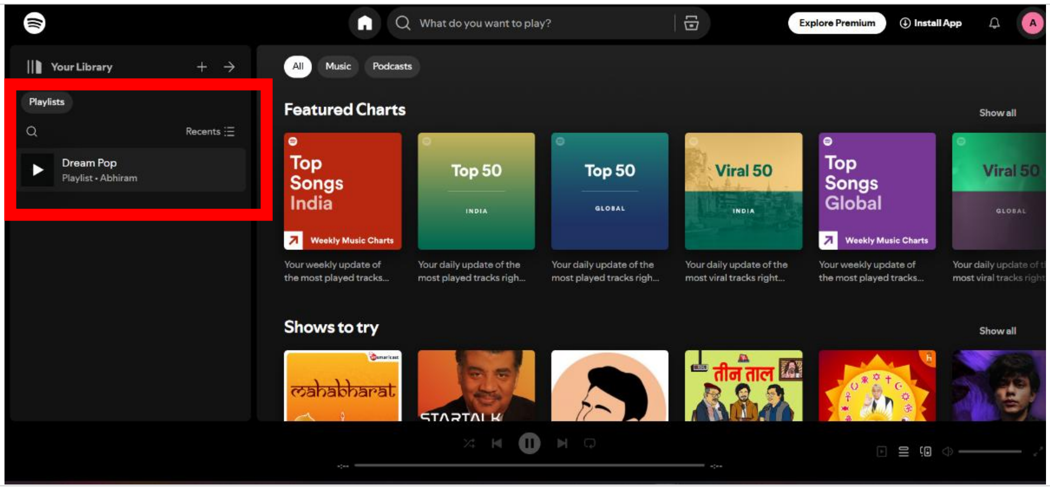
`terraform apply -auto-approve`



```
abhiram@LAPTOP-NL5C21F3 MINGW64 ~/Downloads/DevOps/Project/Spotify_TF (master)
$ terraform apply -auto-approve

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated
with the following symbols:
  + create

Terraform will perform the following actions:

  # spotify_playlist.my_playlist will be created
  + resource "spotify_playlist" "my_playlist" {
      + id          = (known after apply)
      + name        = "Dream Pop"
      + public      = true
      + snapshot_id = (known after apply)
      + tracks      = [
          + "0yc6Gst2xkRu0eMLeRMGCX",
        ]
    }

Plan: 1 to add, 0 to change, 0 to destroy.
spotify_playlist.my_playlist: Creating...
spotify_playlist.my_playlist: Creation complete after 0s [id=2YAVQnrNQCSaVFEO7DLYNh]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

3. Verify the Playlist:

Open your Spotify account to **confirm** the playlist has been created.



6.2 Dynamic Playlist

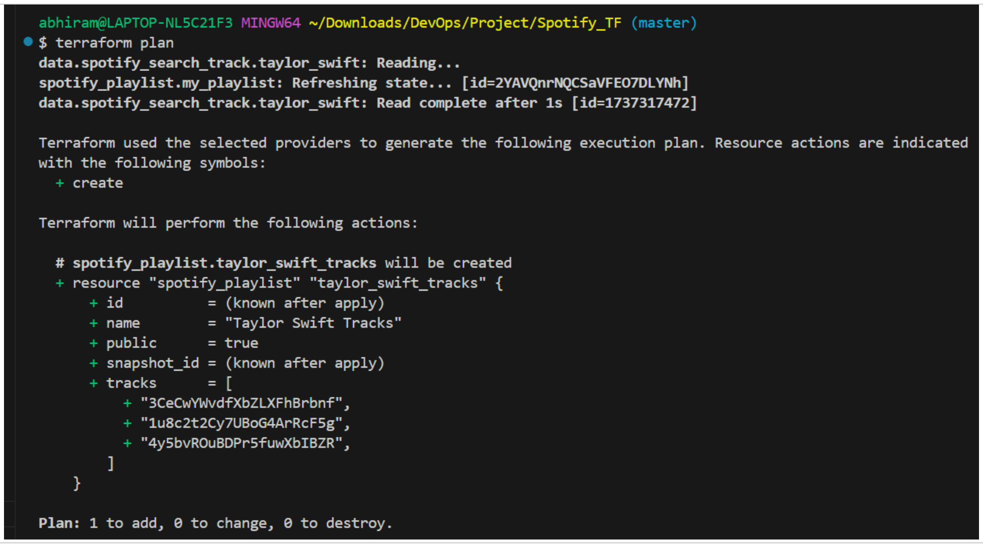
1. Use Data Block:

Modify playlist.tf to include a data block:

```
data "spotify_search_track" "taylor_swift" {
  artist = "Taylor Swift"
}

resource "spotify_playlist" "taylor_swift_tracks" {
  name   = "Taylor Swift Tracks"
  tracks = [
    data.spotify_search_track.taylor_swift.tracks[0].id,
    data.spotify_search_track.taylor_swift.tracks[1].id,
    data.spotify_search_track.taylor_swift.tracks[2].id
  ]
}
```

2. Preview and Apply:

Preview the changes:
`terraform plan`



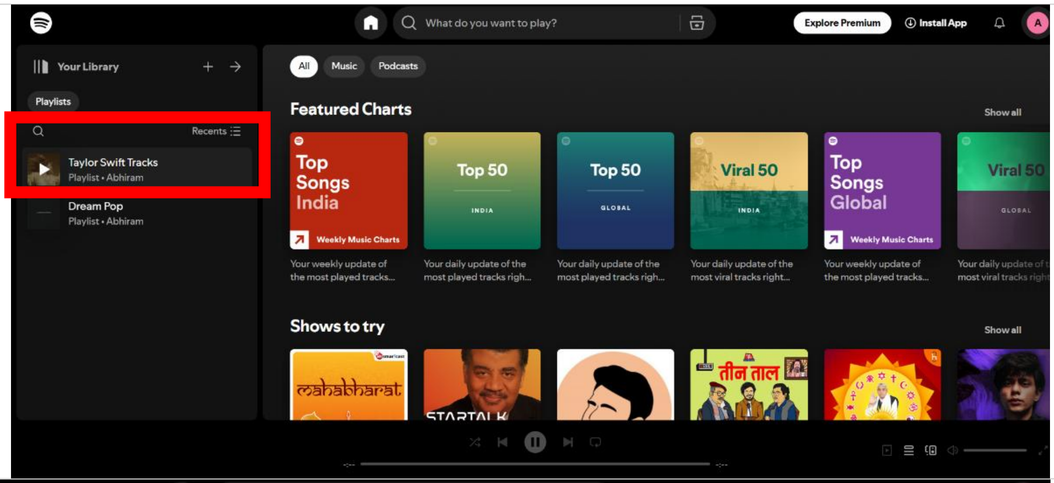
```
abhiram@LAPTOP-NL5C21F3 MINGW64 ~/Downloads/DevOps/Project/Spotify_TF (master)
$ terraform plan
data.spotify_search_track.taylor_swift: Reading...
spotify_playlist.my_playlist: Refreshing state... [id=2YAVQnrNQCSaVFEO7DLYNh]
data.spotify_search_track.taylor_swift: Read complete after 1s [id=1737317472]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated
with the following symbols:
  + create

Terraform will perform the following actions:

  # spotify_playlist.taylor_swift_tracks will be created
  + resource "spotify_playlist" "taylor_swift_tracks" {
      + id          = (known after apply)
      + name        = "Taylor Swift Tracks"
      + public      = true
      + snapshot_id = (known after apply)
      + tracks      = [
          + "3CeCwYWvdfXbZLXFhBrbnf",
          + "1u8c2t2Cy7UBoG4ArRcF5g",
          + "4y5bvROuBDPr5fuwXbIBZR",
        ]
    }

Plan: 1 to add, 0 to change, 0 to destroy.
```

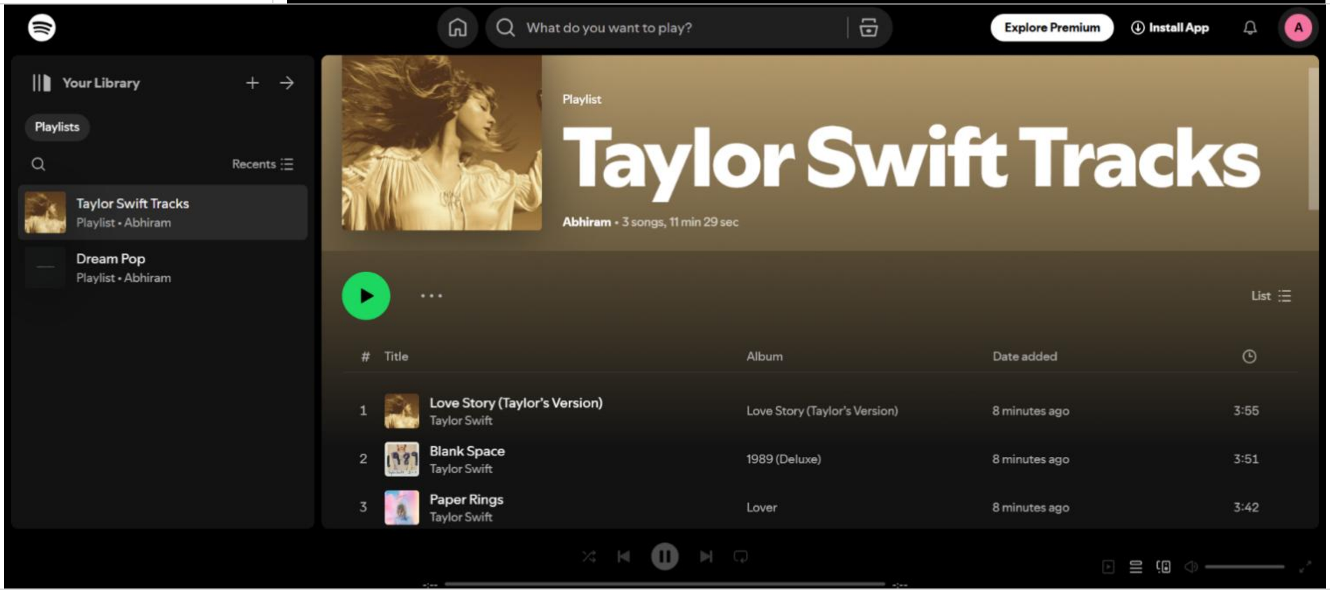
Apply the changes:

`terraform apply -auto-approve`

3. Verify the Playlist:

Check your Spotify account to **confirm** the new playlist has been created.





#	Title	Album	Date added	
1	Love Story (Taylor's Version) Taylor Swift	Love Story (Taylor's Version)	8 minutes ago	3:55
2	Blank Space Taylor Swift	1989 (Deluxe)	8 minutes ago	3:51
3	Paper Rings Taylor Swift	Lover	8 minutes ago	3:42

Conclusion

Through this project, we gained hands-on experience in Terraform, API integration, and automating workflows. The successful creation of dynamic and static playlists on Spotify showcases the versatility of DevOps tools in solving creative challenges.