Globant Piscine

Project 5

Summary: Globify

Version: 1

Contents

I	**A word about this Project**	**2**
II	**Introduction**	**3**
III	**General instructions**	**4**
IV	**Mandatory part**	**5**
V	**Submission**	**8**

Chapter I

A word about this Project

We will create a web-based music player using Spotify's APIs! It will be built entirely in vanilla—no external libraries of any kind—just JavaScript (TypeScript is optional but recommended) and native CSS. The project will be developed from scratch all the way to production deployment.

Why? The goal is to ensure you understand the foundational concepts. The project will focus solely on frontend development, with no backend involved. Teams will consist of three members, who will self-manage and distribute predefined tasks, though they can also create their own.

It is recommended to have three distinct roles within the team: CSS/Rendering, API/Deployment, and a "Controller."

This project is designed to strengthen your core skills and provide practical experience with web development.

Chapter II

Introduction

What this Project will show you:

- Building a fully functional web-based music player using TypeScript and native CSS.
- Integrating with the Spotify API to retrieve and play music.
- Understanding the fundamentals of frontend development without relying on external libraries.
- Self-managing tasks in a team of three members with distinct roles: CSS/Rendering, API/Deployment, and a "Controller".
- Publishing a complete project from scratch to production.
- Gaining hands-on experience in developing a modern web application with an API-based architecture.

Chapter III

General instructions

Unless explicitely specified, the following rules will apply for every project of this Piscine.

- This subject is the one and only trustable source. Don't trust any rumor.
- This subject can be updated up to one hour before the turn-in deadline.
- A README.md documenting the project, its installation, and usage will be required during evaluation.
- The assignments in a subject must be done in the given order. Later assignments won't be rated unless all the previous ones are perfectly executed.
- Be careful about the access rights of your files and folders.
- Your assignments WON'T be evaluated by your Piscine peers.
- You must not leave in your turn-in your workspace any file other than the ones explicitly requested by the assignments. If the assignment don't precise them, put only the necessary ones to run your Project.
- Using some API Key or Token? Keep them for you! Do not push them on your repository.
- You have a question? Ask your left neighbor. Otherwise, try your luck with your right neighbor.
- Every technical answer you might need is available in the `man` or on the Internet.
- You must read the examples thoroughly. They can reveal requirements that are not obvious in the assignment's description.
- By Thor, by Odin! Use your brain!!!

Chapter IV

Mandatory part

Globant	Exercise 00
Globify	
Turn-in directory : $ex00/$	
Files to turn in : `All needed files to run your Project and nothing else`	
Allowed functions : `None`	

- Docker is mandatory to submit the project. Please provide a Dockerfile and a docker-compose.yml file to run the project.
- **Login/Logout**
  - Ensure that login and logout functionalities work using Spotify's OAuth2.0 API (or equivalent) client ID.
  - Verify that users can successfully log in and log out, and that the authentication integrates with the Spotify API.
- **Layout**
  - Create a basic layout with strong colors that is fully responsive, with a breakpoint at 430px wide.
  - The header and footer should be fixed, especially on mobile.
  - Connect the logout button to allow users to log out.
  - Design does not need to be pixel-perfect but should be functional and responsive.
- **Menu**
  - The menu should always be fixed and display home, favorites, and playlists options.
  - On mobile, the menu should overlay and toggle visibility with a footer button.

- When a menu option is clicked on mobile, the menu should hide after selection.

- **Home**
  - Display a list of categories, and when a user clicks on a category, show related playlists.
  - Load categories when the app starts and whenever the user navigates to the home from the menu.
  - On desktop, the first three categories should be highlighted with larger thumbnails.

- **Profile**
  - Display dinamically user information as defined in the project template, ensuring the user sees their data clearly and accurately.

- **My Favorites/Saved**
  - Display a list of saved tracks in the same format as search results.
  - Clicking on a track in this list should immediately start playback, updating the player at the bottom.

- **Playlists**
  - Show a list of the user's playlists (or playlists based on a genre).
  - Clicking on a playlist should navigate the user to that playlist's details page.

- **Playlist Page**
  - Display the playlist hero section (with image and text) and associated tracks.
  - The play button in the hero section should send the playlist to the player for playback.
  - Clicking on any track in the playlist should also trigger playback.

- **Player**
  - Implement basic play/pause functionality, ensuring the player reflects the current state of playback.
  - The player should display information about the currently playing track on the left.
  - Functions like random, loop, next, and previous should not be implemented at this stage.

- **Search**
  - Allow users to search for tracks only, and display results in a list format.

- Clicking on a search result should start track playback, updating the player at the bottom.

- **General Functionality**
  - All lists (categories, playlists, tracks) should be limited to 50 items, with no pagination.
  - The design does not need to be pixel-perfect, but the app should be user-friendly and functional.
  - Clicking on any track in any view (home, favorites, search, playlist) should start playback and update the player.

- **Team Leader Recommendation**
  - A lead developer is recommended within each team to help guide and coordinate efforts, but the final decision rests with the team.

You don't want or can't use Spotify API? Checkout Deezer API, Apple Music API or even Soundcloud API for alternatives.

You may want to take a look at every technologies referenced in the project description before starting.

This is a group project, remember to communicate with your team members and split the work together. Git Branch are your friends. Use them to split the work and merge it together.

Chapter V

Submission

- Create a git repo (Github, Gitlab, Bitbucket, etc) and add your project files to it.
- Copy the link to your repository and paste it in the project submission form.
- Link to the correction form: Google Form

Please note, no modifications made on the repo after the form is sent will be taken into account for the evaluation.

No Peer evaluation for this Piscine, but feel free to share your project with your peers and get feedback.