**Projet : Wiki Spotify**

**Introduction**

Le projet consiste en la création d'un site web structuré autour de l'API Spotify, développé en Go. Ce site a pour but de fournir une interface claire et intuitive permettant d'explorer et d'afficher des informations sur la musique, les artistes et les albums disponibles sur Spotify, comme un wiki interactif.

**Objectifs**

* **Utiliser l'API Spotify** pour récupérer et afficher des données musicales.
* **Structurer l'application web** en utilisant Go pour la gestion du backend et du serveur.
* **Offrir plusieurs pages dynamiques**, adaptées aux différentes catégories de contenu.
* **Mettre en place un système de favoris** pour sauvegarder certaines ressources.

**Technologies utilisées**

* **Go (Golang)** pour le backend et la gestion des requêtes API.
* **HTML / CSS** pour la structure et le style du site.
* **API Spotify** pour récupérer les données musicales.

**Arborescence du site**

1. **Page d'accueil** : Présente l’univers et le thème du site.
2. **Page collection** : Affiche une liste d’artistes, albums ou playlists.
3. **Page ressource (détails)** : Fournit des informations détaillées sur une ressource donnée (album, artiste, chanson).
4. **Page catégorie** : Classe les ressources selon un genre musical ou une autre caractéristique.
5. **Page favoris** : Permet d'afficher et gérer les ressources favorites.
6. **Page recherche** : Affiche les résultats de recherche en fonction des requêtes de l’utilisateur.
7. **Page à-propos** : Explique le projet et les technologies utilisées.

**Étapes de développement**

**1. Mise en place du projet**

* Initialisation du projet en Go avec go mod init.
* Création d’un serveur HTTP en Go avec net/http.
* Inscription et obtention des clés d’API Spotify.
* Mise en place des structures de données (modèles Go pour les ressources Spotify).

**2. Connexion à l'API Spotify**

* Gestion de l’authentification OAuth 2.0 pour récupérer un token d’accès.
* Création de fonctions en Go pour interroger l’API Spotify et récupérer les données.

**3. Développement des différentes pages**

* Création des routes backend pour chaque page.
* Mise en place des templates HTML dynamiques avec Go (html/template).
* Gestion des erreurs et des cas où les données ne sont pas disponibles.

**4. Fonctionnalités avancées**

* Mise en place d’un système de favoris persistant avec go.
* Optimisation des performances du backend en réduisant les appels API inutiles.
* Ajout d’un système de cache pour améliorer la rapidité d’affichage des pages.

**Conclusion**

Ce projet exploite les capacités de Go pour créer un site web fonctionnel et performant, permettant aux utilisateurs d’explorer l’univers de la musique de manière intuitive et organisée. L'utilisation d'un backend en Go offre une gestion efficace des requêtes et une intégration fluide avec l'API Spotify, garantissant une expérience utilisateur optimale.