Projet ClassGen

Université Paris-Est Marne-la-Vallée

October 17, 2024

1 Objectif du projet

Le but du projet **ClassGen** est d'écrire une application Web qui interagit avec un LLM pour aider un utilisateur à créer une classe Java valide à partir d'indications en anglais. L'application ClassGen est composée d'un back-end écrit en Java offrant différents services REST permettant de créer, visualiser et compiler/exécuter des classes Java.

2 Fonctionnalités de l'application

L'application doit permettre :

- De créer une nouvelle classe Java à partir d'un prompt utilisateur écrit en anglais. Le prompt utilisateur et votre prompt système sont envoyés à un LLM local (qui tourne sur le serveur) pour créer une classe Java. Cette classe est ensuite compilée. Si des erreurs de compilation surviennent, celles-ci sont résolues automatiquement en demandant au LLM de les corriger. Une fois les erreurs résolues, l'utilisateur voit la discussion complète avec le LLM et la classe résultante. Il peut ensuite, grâce à un nouveau prompt, modifier la classe générée toujours à travers le LLM.
- D'afficher la liste de toutes les sessions de discussion avec un utilisateur. Les discussions sont horodatées (jour/heure) et classées de la plus récente à la plus ancienne. Chaque discussion de la liste doit afficher sa date, le nom du LLM utilisé et le premier prompt sous forme d'un lien. Si l'utilisateur clique sur ce lien, la discussion correspondante s'affiche et l'utilisateur peut continuer cette discussion (dans ce cas, l'horodatage est mis à jour).
- La page affichant la liste des discussions doit aussi avoir un gros bouton "+" qui permet de démarrer une nouvelle discussion avec le LLM. Il doit également être possible de choisir le LLM avec lequel on va avoir une discussion plutôt que d'utiliser le LLM configuré par défaut.
- Une page doit permettre de choisir le LLM courant que l'on veut utiliser parmi une liste de 3 LLMs. Pour chaque LLM, on doit avoir ses différentes caractéristiques.

3 Interface graphique

Votre application doit comporter trois écrans :

- La première page sert d'écran d'accueil et affiche la liste des discussions ainsi que le bouton "+".
- Une page de discussion qui affiche une alternance de questions (prompts) et de réponses.
- Une page de sélection du LLM par défaut.

4 Technologies à utiliser

- Vous devez utiliser **Maven** comme outil de build et **IntelliJ** comme IDE, ainsi que Java 23.
- Les tests unitaires doivent être réalisés avec **JUnit 5.11.0**. Chaque classe Java doit avoir une classe de test correspondante pour au moins 80% du projet.
- Pour la sérialisation/désérialisation JSON des requêtes, utilisez l'API **Jackson 2.17.2**.
- Pour interagir avec des LLMs locaux, utilisez la bibliothèque **LangChain4J 0.34.0**.

5 Implémentation des services REST

Votre application doit utiliser l'une des technologies suivantes pour les services REST :

- Spring Boot 3.3.3
- Quarkus 3.14.2
- Micronaut 4.6.1
- Helidon 4.1.1 SE ou MP

6 Base de données

L'application a besoin d'une base de données, mais une base de données "embedded" suffira. Utilisez l'une des technologies suivantes :

- **Sqlite JDBC 3.46.1.0**
- **H2 Database 2.3.232**
- **HyperSQL 2.7.3**
- **Apache Derby 10.17.1.0**

7 Front-end Web

Pour le front-end, vous devez utiliser l'un des frameworks suivants :

- Vue.js 3.5.3
- Svelte 4.2.19
- Solid.js 1.8.22
- React.js 18.3.1
- Angular 18.2.3

Pour améliorer la partie CSS, vous pouvez utiliser l'une des bibliothèques suivantes :

- Bootstrap 5.3.3
- Tailwind CSS 3.4.10
- Bulma CSS 1.0.2
- MUI 6.0.2
- NextUI 2.4.6
- React-UI 0.56.0

8 Sécurité

- Les entrées des services web doivent être validées et les sorties doivent être sécurisées pour éviter les injections de code.
- Il n'est pas demandé d'identifier les utilisateurs.
- Vous ne devez pas utiliser de requêtes CORS.

9 Documentation de l'API REST

Vous devez documenter votre API REST en utilisant le format Open API 3. Utilisez une des bibliothèques suivantes :

- Springdoc OpenAPI
- Quarkus - OpenAPI
- Micronaut - OpenAPI
- Helidon SE - OpenAPI
- Helidon MP - OpenAPI