Introduction à Spring Webflux

1 Mise en place

Nous allons appliquer les principes de la programmation réactive pour la mise en place d'une API REST. Nous allons, pour ce faire, utiliser la librairie Spring Web flux.

⏩ **Travail à faire :**Nous allons travailler sur le projet des TPs précédents.

- Ajouter la dépendance ci-dessous :

```
<dependency>
    <groupId>io.projectreactor</groupId>
    <artifactId>reactor-test</artifactId>
    <scope>test</scope>
</dependency>
```

- Ces dépendances doivent déjà exister :

Ces dépendances doivent déjà être présentes (à vérifier)

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-redis-reactive</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-webflux</artifactId>
</dependency>
<dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <optional>true</optional>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
</dependency>
```

- Suivez ensuite ce tutoriel qui détaille la mise en place d'une petite API REST basée sur redis et la programmation réactive.
- Utilisez cette documentation pour construire des tests unitaires basés sur le client réactif `WebClient`.
- Avec cette documentation, vous pourrez explorer d'autres possibilités de Spring Webflux, notamment l'utilisation des routes et la capacité à définir des contrôleurs avec des lambdas sans la création de classes spécifiques.