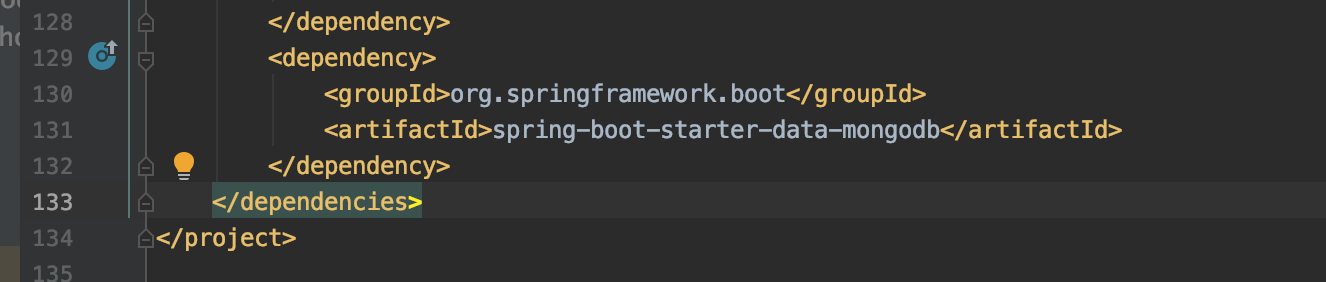
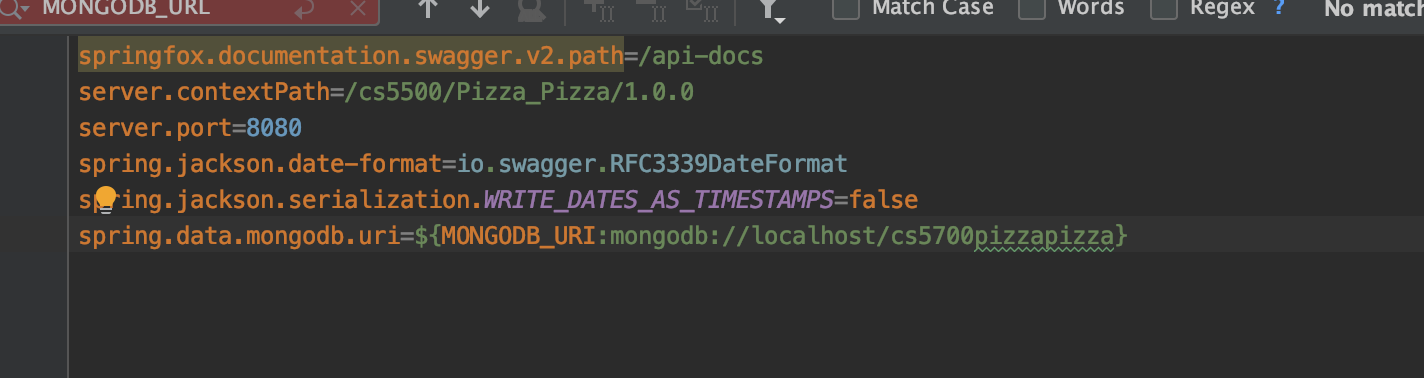
This document describes some hints and instructions for MongoDB set up, development, testing and deployment.

**Setup:**

1. Install MongoDB on your laptop: <https://docs.mongodb.com/manual/tutorial/install-mongodb-on-os-x/>
2. Add dependency spring-boot-starter-data-mongodb to project pom.xml.

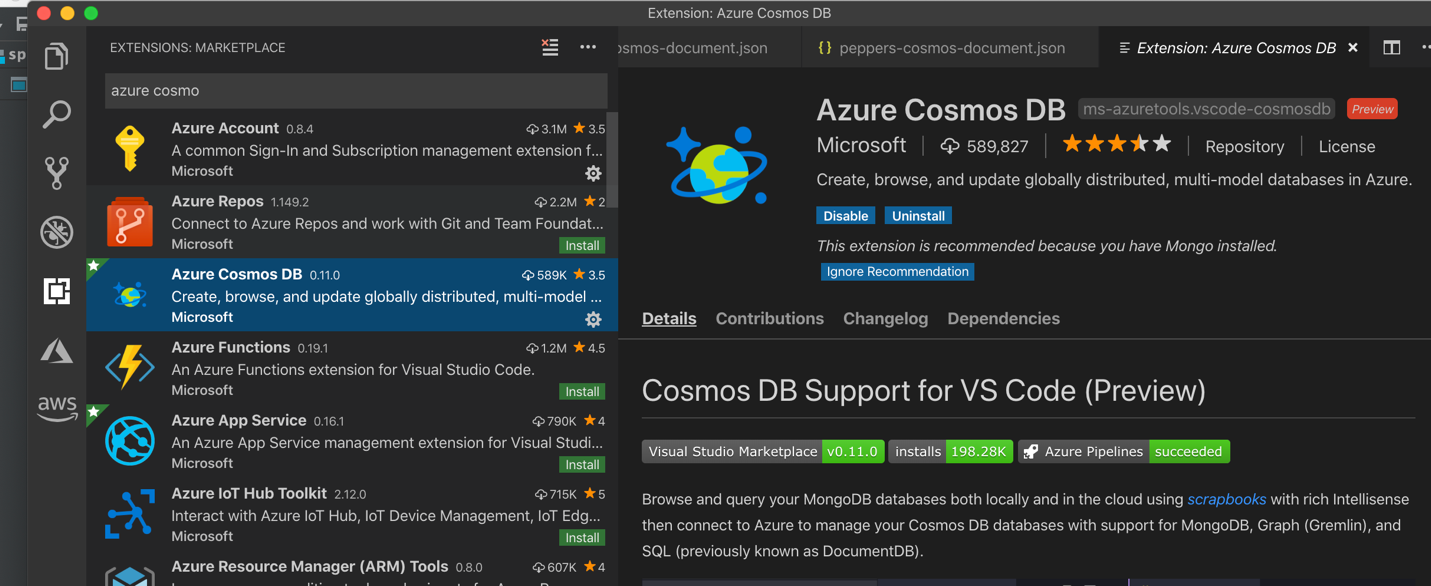


1. Configure the MongoDB URI in resources/application.properties. The value is spring.data.mongodb.uri=${MONGODB\_URI:mongodb://localhost/cs5700pizzapizza}. It means that spring application will first tries using the environment variable MONGODB\_URI as the MongoDB URI. If MONGODB\_URI is not available, the application will connect to the MongoDB locally on your laptop, whose URI is mongodb://localhost/cs5700pizzapizza.

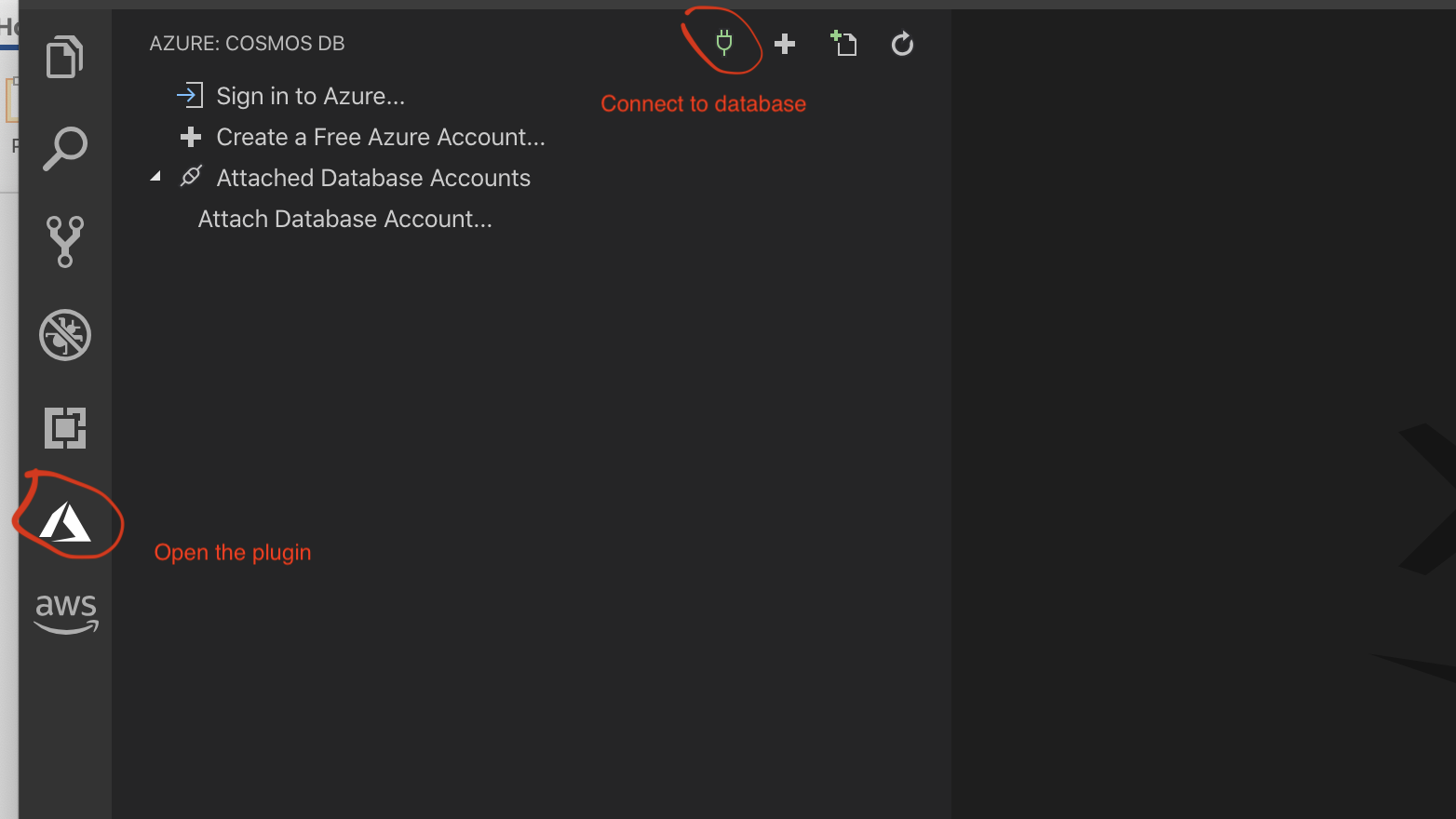


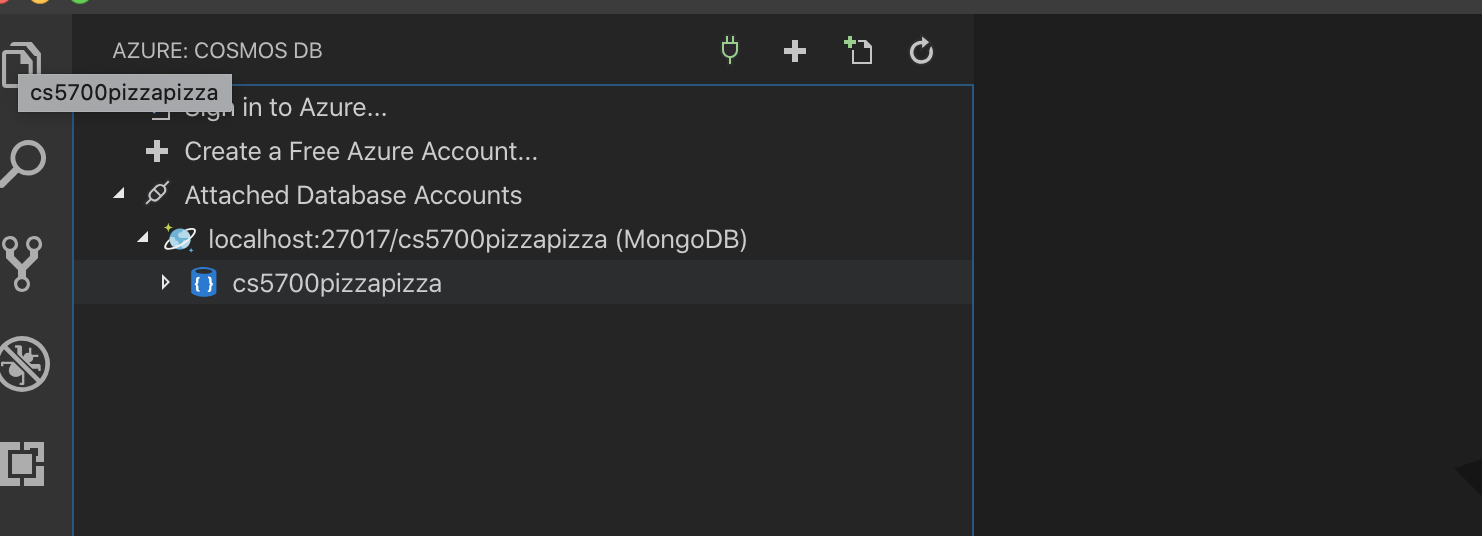
If we just deploy the service locally on laptop, we don’t need to care about the environment variable MONGODB\_URI. We will see the usage of it later when we deploy our application on Heroku.

1. If you want to view the MongoDB content in your laptop, you can install Visual Code and a plugin called “AZURE: Cosmos DB”.
   1. Install Visual Code: <https://code.visualstudio.com/>
   2. Install Azure Cosmos DB plugin:



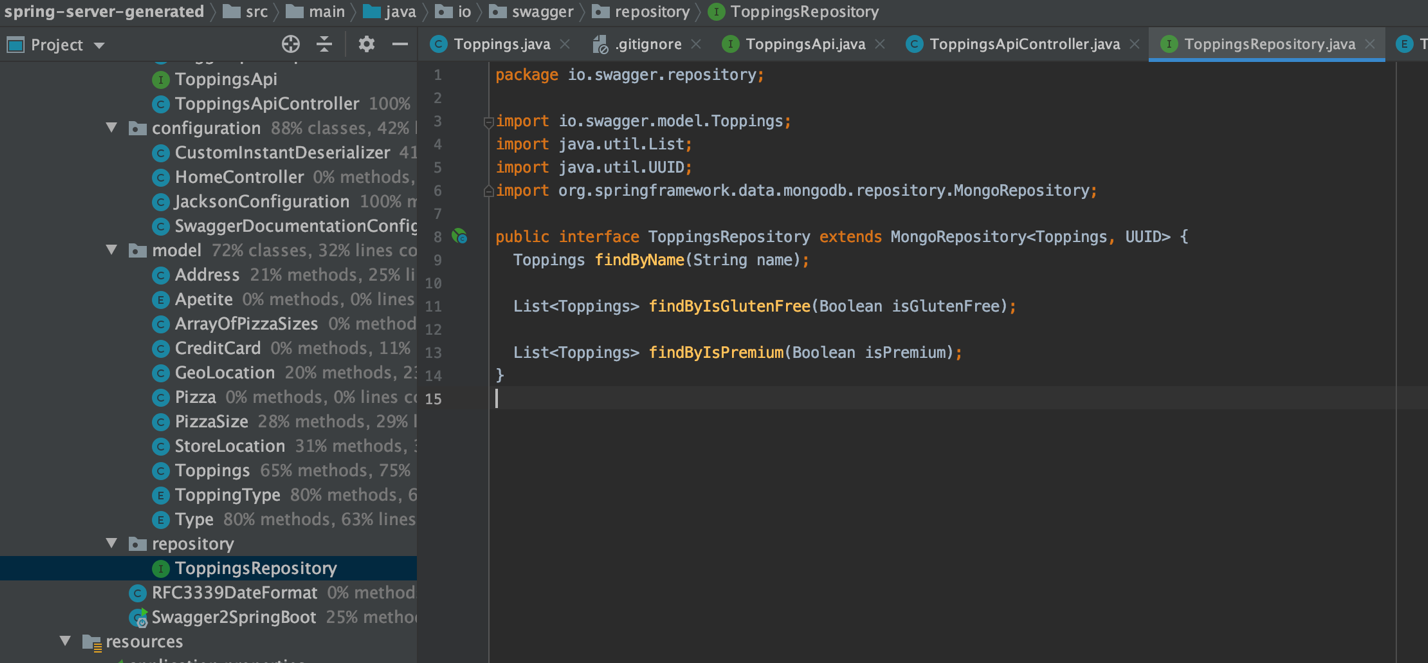
* 1. Open the Azure plugin, and click the “plug” icon to connect to a database.



* 1. Click the “plug” icon and choose the “MongoDB” option, input your MongoDB local URI. Mine is mongodb://localhost/cs5700pizzapizza. 

**Development:**

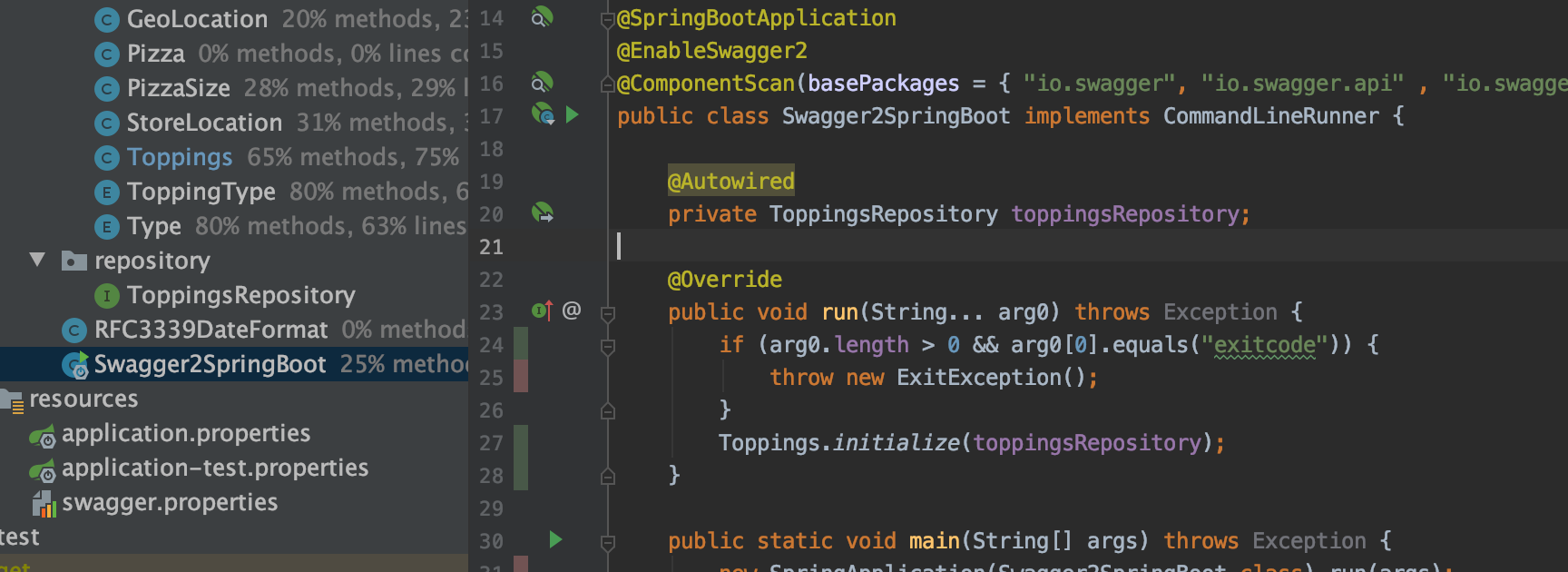
1. Create a repository interface for the data model you want to store in the database. For example, I have a data model Toppings.java, which has attributes “name”, “isGlutenFree” and “isPremium”. And I want to search the database based on the three attributes. Therefore, I create an interface ToppingsRepository.java



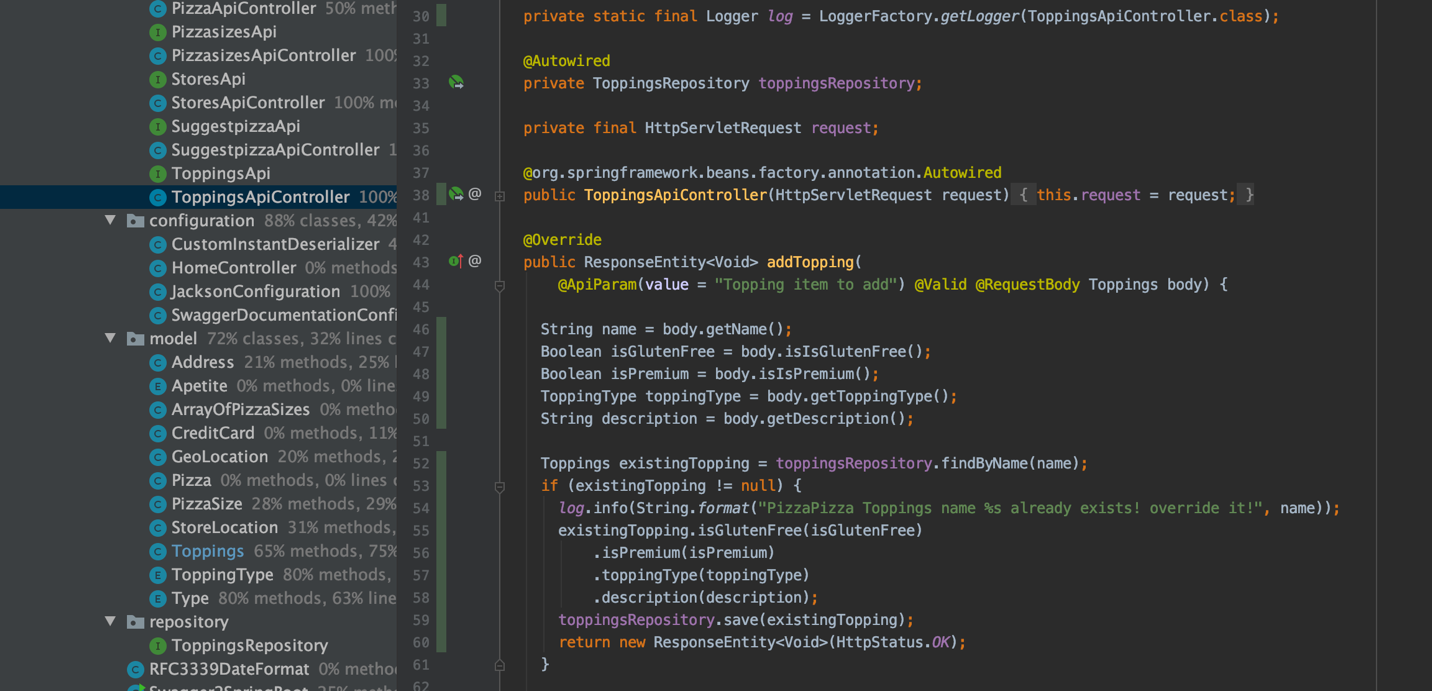
1. If we want to insert into the database with some preloaded data when the application is launched, we can create an initialize method to insert data into the repository.



We need to use @Autowired annotation to get an instance of the repository. And the following code will call the initialize method we just created to preload data into database upon the application is launced.

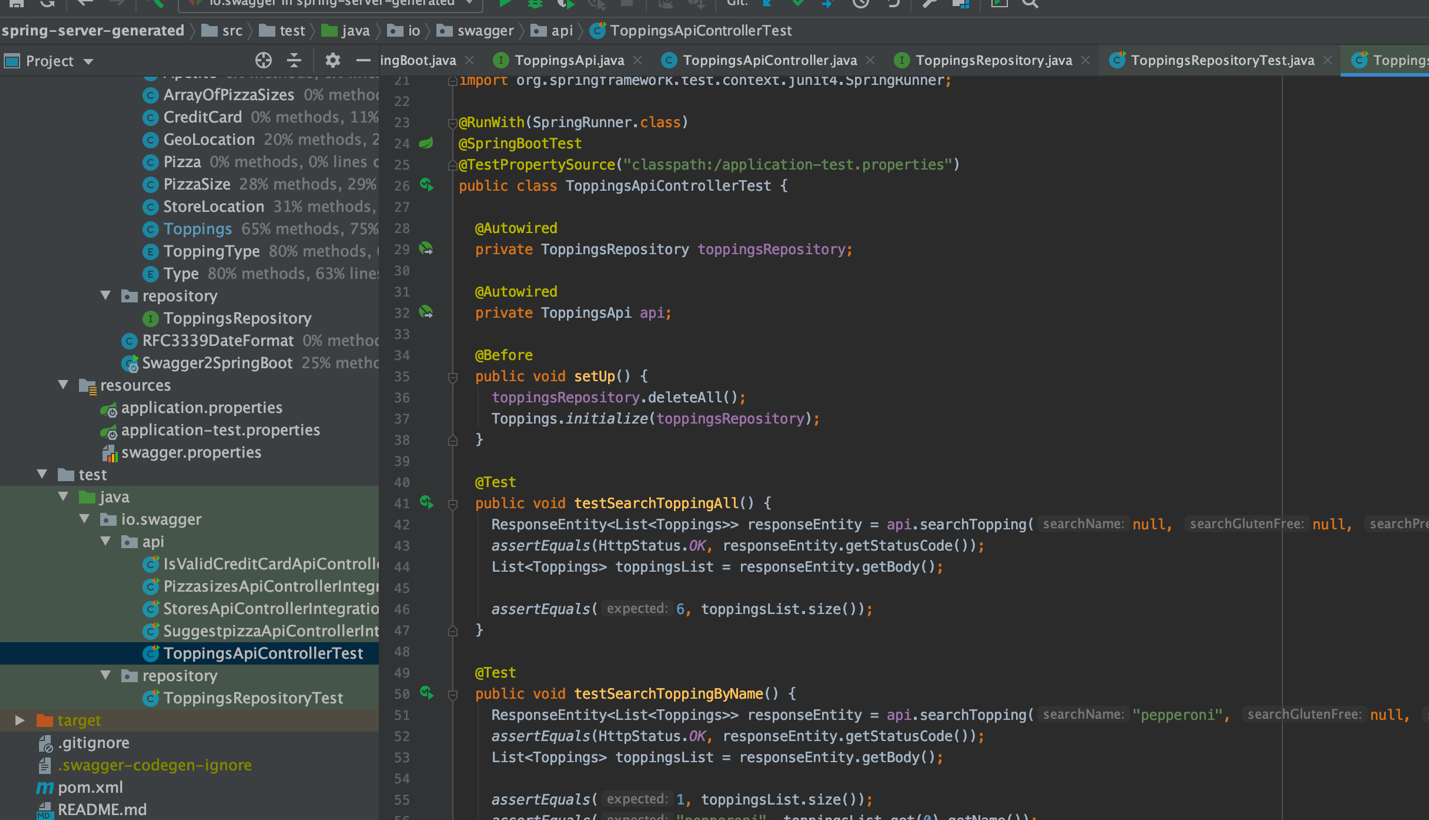


1. In ApiController, we can auto wire an instance of the repository and leverage the methods defined in the repository interface to search the database.

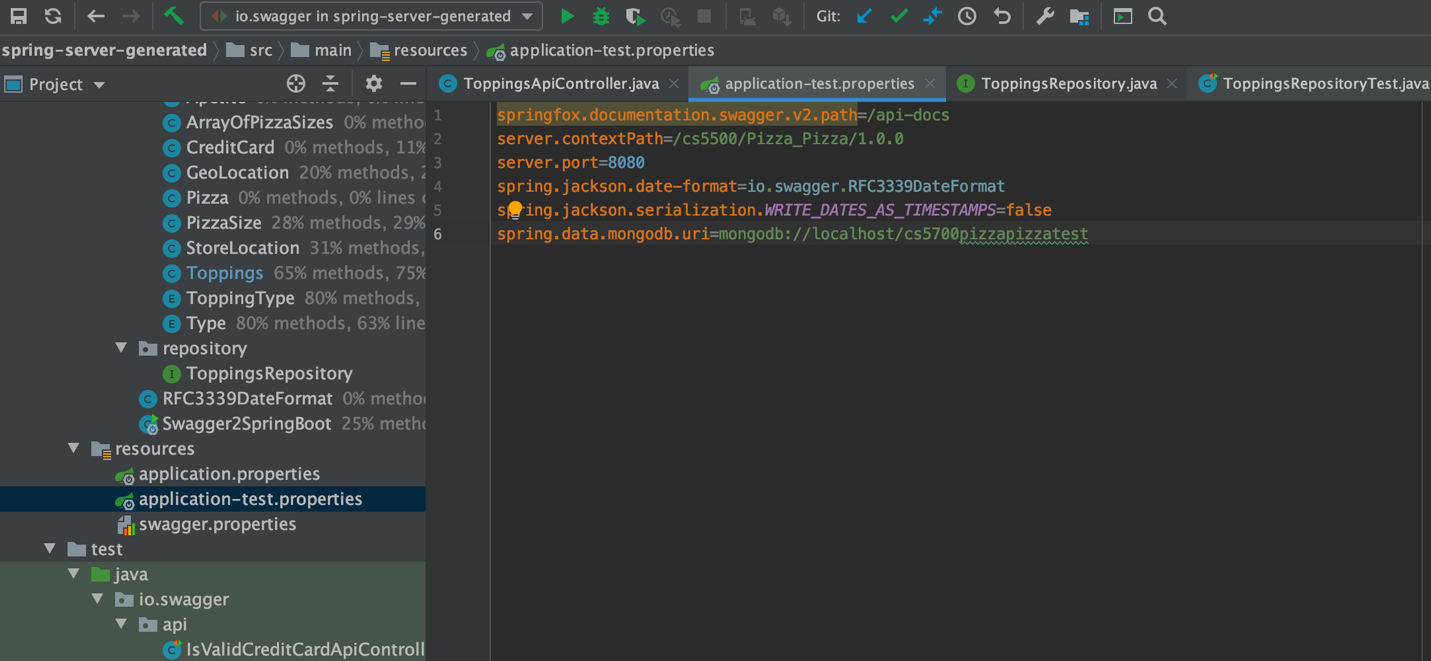


**Test**

1. Similar with development section, we can use @Autowired annotation to create the instance and call the initialize method to load test data.



1. We can have a different application property. It can be specified as @TestPropertySource(“classpath:/application-test.properties”).



**Deployment:**

1. To deploy the service on Heroku, we need to create a mLab MongoDB addon in Heroku. The mLab addon will provide us an environment variable MONGODB\_URI as the database URI on Heroku.

