

Guide de De developpement

I. Technologies Requises

- **VS code:** C'est un éditeur de codesource léger et puissant développé par Microsoft. Il est conçu pour être hautement personnalisable et adaptable à différents langages de programmation et flux de développement, telechargeable à travers le site officiel <https://code.visualstudio.com/>. Comme alternative, on peut utiliser IntelliJ, Eclipse, Spring tools suite.
- **Docker :** Pour créer les images et conteneur du service <https://www.docker.com/>.
- **Redis Insight :** Pour visualiser les données de la base de données <https://redis.io/>.
- **Java JDK 17 :** kit de développement pour Java telechargeable à travers le lien officiel <https://www.oracle.com/java/technologies/javase-jdk17-downloads.html>.
- **Cassandra :** Apache Cassandra est un système de gestion de base de données distribué telechargeable a travers le site officiel: <https://cassandra.apache.org/>

II. Prise en Main du projet

Pour mettre en place l'environnement de développement, Il faudra suivre rigoureusement les differentes etapes suivantes

1. Cloner le Projet

Pour cloner le projet, il faut :

- se rendre sur le depot Github du microservice qui se trouve à l'adresse <https://github.com/alexdiffio/Chatbot-Backend.git> La dernière version qui est encore au stade de developpement se trouve sur la Branche Dev.

← → ↻ github.com/alexdiff0/Chatbot-Backend/tree/dev

COCO ARGEN... Emails from sc... unichange.me... Marijuana - A... Install Git on U... How to Install... (322) Support... Install and Set...

🐙 Search or jump to... Pull requests Issues Codespaces Marketplace Explore

alexdiff0 / Chatbot-Backend Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

dev 3 branches 0 tags Go to file Add file <> Code

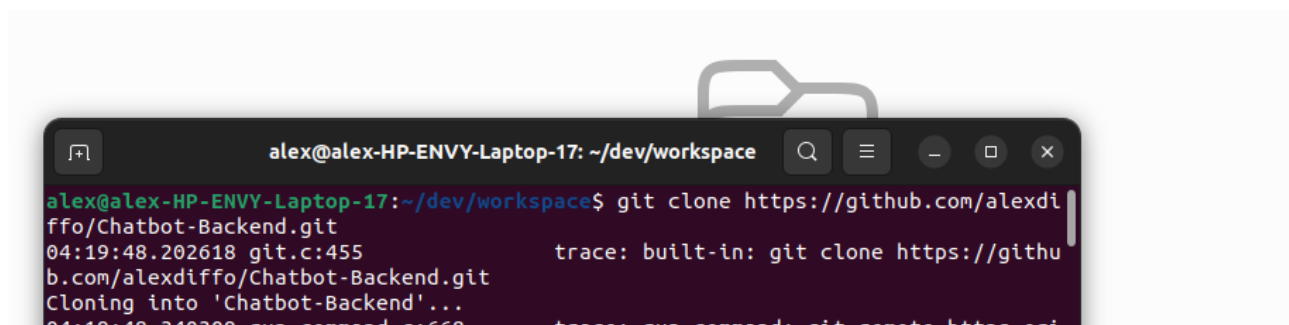
This branch is 5 commits ahead of master. Contribute

alexdiff0 refractoring code and switching to cassandra 68899dc 14 hours ago 6 commits

..mvnw/wrapper	chatbot backend including springboot startercode	3 months ago
src	refractoring code and switching to cassandra	14 hours ago
.gitignore	refractoring code and switching to cassandra	14 hours ago
categories.txt	refractoring code and switching to cassandra	14 hours ago
locations	new features added with category	4 days ago
mvnw	refractoring code and switching to cassandra	14 hours ago
mvnw.cmd	refractoring code and switching to cassandra	14 hours ago
pom.xml	refractoring code and switching to cassandra	14 hours ago

Help people interested in this repository understand your project by adding a README. Add a README

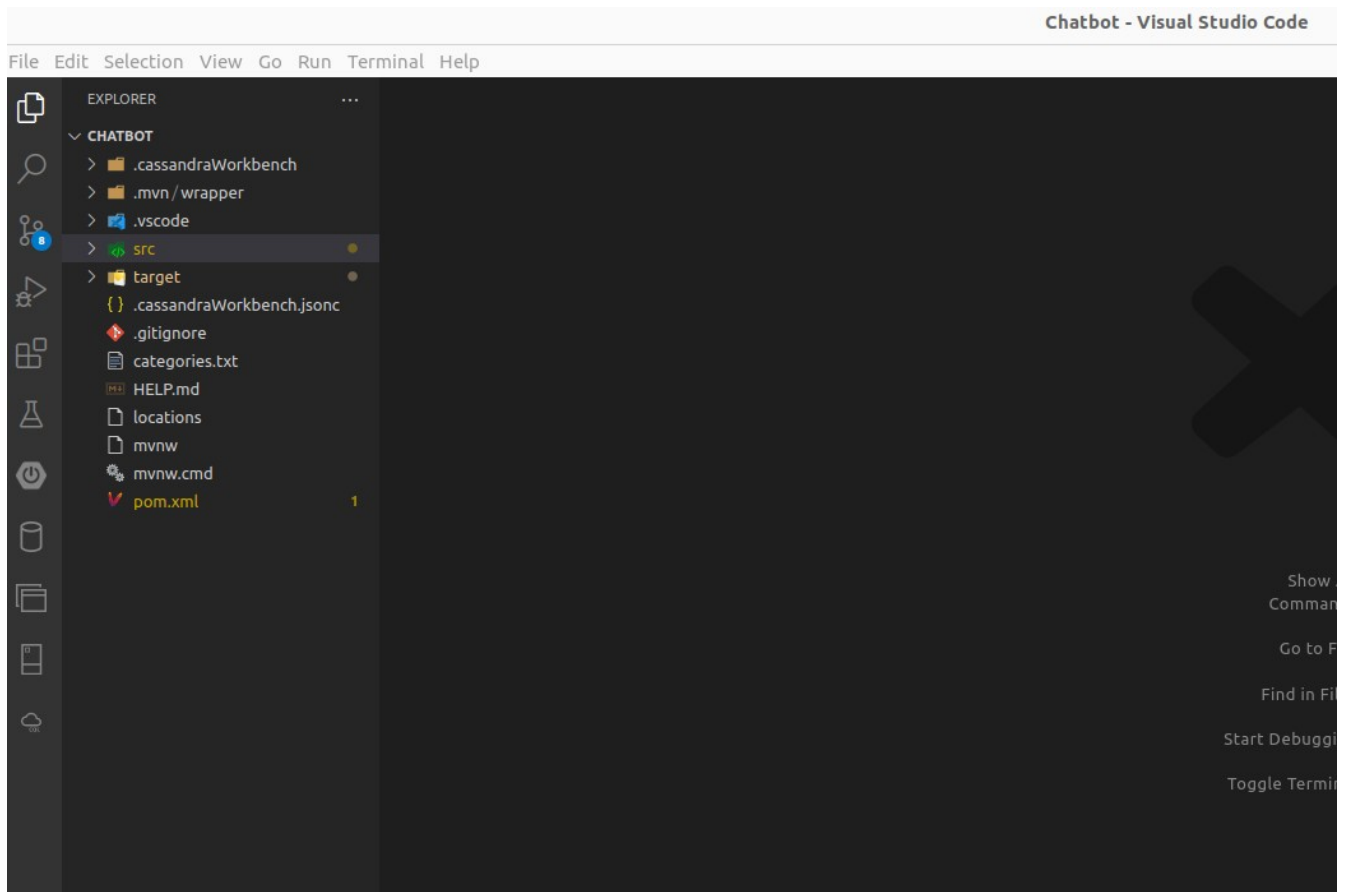
- Ouvrir le terminal en se positionnant sur le repertoire local du workspace et Cloner le projet projet localement à travers la commande: “git clone <https://github.com/alexdiff0/Chatbot-Backend.git>”



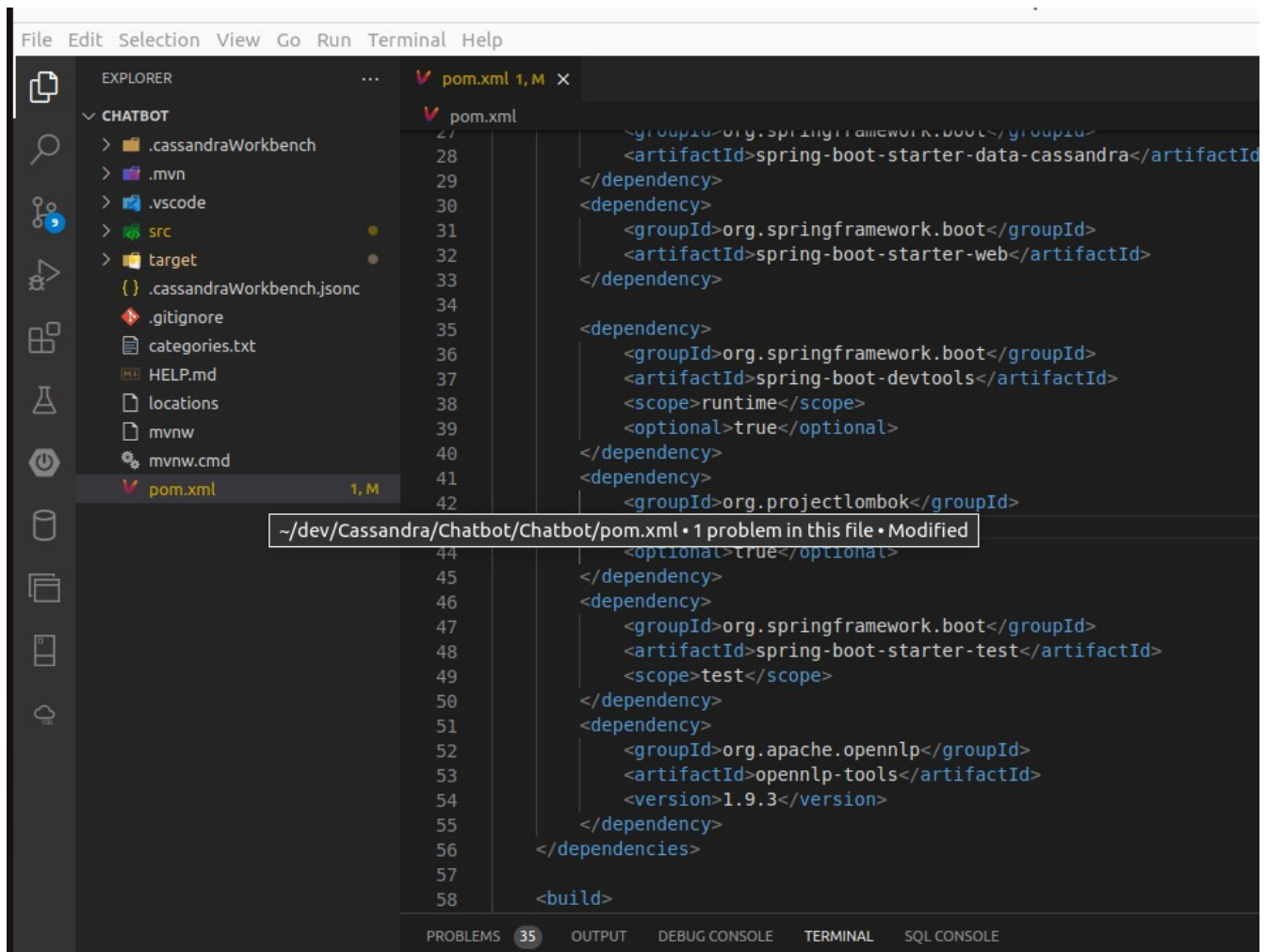
```
alex@alex-HP-ENVY-Laptop-17: ~/dev/workspace
alex@alex-HP-ENVY-Laptop-17:~/dev/workspace$ git clone https://github.com/alexdiff0/Chatbot-Backend.git
04:19:48.202618 git.c:455 trace: built-in: git clone https://github.com/alexdiff0/Chatbot-Backend.git
Cloning into 'Chatbot-Backend'...
```

2. Mettre en place l'environnement de developpement

Une fois le clonage terminé, Naviguer pour se positionner à la racine du dossier du projet à travers la commande **“cd chatbot “** puis ouvrir lancer l’IDE Vscode à travers la commande **“code . “** , ce qui aura pour effet d’ouvrir VS code en chargeant la repository du projet.



- Actualiser le fichier POM xml pour telecharger les dependance Maven. Au préalable, il faut se rassurer d'être connecté à internet:



The screenshot shows an IDE interface with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'CHATBOT' with various files and folders. The code editor displays the content of 'pom.xml', which includes several Maven dependencies. A tooltip is visible over the file, indicating a problem.

```
File Edit Selection View Go Run Terminal Help
EXPLORER
CHATBOT
  .cassandraWorkbench
  .mvn
  .vscode
  src
  target
  .cassandraWorkbench.jsonc
  .gitignore
  categories.txt
  HELP.md
  locations
  mvnw
  mvnw.cmd
  pom.xml 1, M
pom.xml 1, M x
pom.xml
27 <groupId>org.springframework.boot</groupId>
28 <artifactId>spring-boot-starter-data-cassandra</artifactId>
29 </dependency>
30 <dependency>
31 <groupId>org.springframework.boot</groupId>
32 <artifactId>spring-boot-starter-web</artifactId>
33 </dependency>
34
35 <dependency>
36 <groupId>org.springframework.boot</groupId>
37 <artifactId>spring-boot-devtools</artifactId>
38 <scope>runtime</scope>
39 <optional>true</optional>
40 </dependency>
41 <dependency>
42 <groupId>org.projectlombok</groupId>
43 <artifactId>lombok</artifactId>
44 <optional>true</optional>
45 </dependency>
46 <dependency>
47 <groupId>org.springframework.boot</groupId>
48 <artifactId>spring-boot-starter-test</artifactId>
49 <scope>test</scope>
50 </dependency>
51 <dependency>
52 <groupId>org.apache.opennlp</groupId>
53 <artifactId>opennlp-tools</artifactId>
54 <version>1.9.3</version>
55 </dependency>
56 </dependencies>
57
58 <build>
```

~/dev/Cassandra/Chatbot/Chatbot/pom.xml • 1 problem in this file • Modified

PROBLEMS 35 OUTPUT DEBUG CONSOLE TERMINAL SQL CONSOLE

3. Installer la base de donnée

- Installer et demarrer docker
- Lancer l'image docker de cassandra dans un conteneur à travers la commande docker

“**docker run --name cassandra2 -p 127.0.0.1:9042:9042**” pour caster sur le port 9042 en local (où cassandra2 est le nom du conteneur.

```
alex@alex-HP-ENVY-Laptop-17:~/dev/workspace$ sudo docker run --name cassandra2  
-p 127.0.0.1:9042:9042 cassandra
```

- Verifier à travers la commande “**docker ps**” que le serveur est bien lancé; L’ID du conteneur de cassandra doit s’afficher explicitement ainsi que son nom.

```
alex@alex-HP-ENVY-Laptop-17:~/dev/workspace$ sudo docker ps  
[sudo] password for alex:  
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS  
e2c79d163cff   cassandra  "docker-entrypoint.s..." 2 days ago    Up 24 hours   7000-7001/tcp, 7199/tcp, 9160/tcp, 127.0.0.1:9042->9042/tcp  
alex@alex-HP-ENVY-Laptop-17:~/dev/workspace$
```

- Se connecter au shell CQLSH de cassandra pour effectuer des requetes sur la base de donnée à travers la commande: “**sudo docker exec -it cassandra2 cqlsh**”

```
alex@alex-HP-ENVY-Laptop-17:~/dev/workspace$ sudo docker exec -it cassandra2 cqlsh  
Connected to Test Cluster at 127.0.0.1:9042  
[cqlsh 6.1.0 | Cassandra 4.1.2 | CQL spec 3.4.6 | Native protocol v5]  
Use HELP for help.  
cqlsh>
```

- Activer le kespace a travers la commande **“use chat_bot”** pour pouvoir effectuer des requetes.
- creer le Kespace correspondant au nom de la base de donnée que vous souhaitez utiliser à travers la requete

**“CREATE KEYSPACE chat_bot
WITH replication = {'class': 'Strategy', 'replication_factor': N};”.**

Dans ce cas, chat_bot est le keyspace name choisi.

- Creer la table **message** à travers la requete :

-- change table name and structure

**“CREATE TABLE chat_bot.Message(
 messageid text PRIMARY KEY,
 messagebody text,
 responsebody text,
 senderusername text,
 category text,
 event_time TIMESTAMP**

);”

- Confirmer à travers la commande CQLSH **“describe tables”** que la table a bien été créé; le nom de la table doit s’afficher explicitement

```
cqlsh:chat_bot> describe tables;

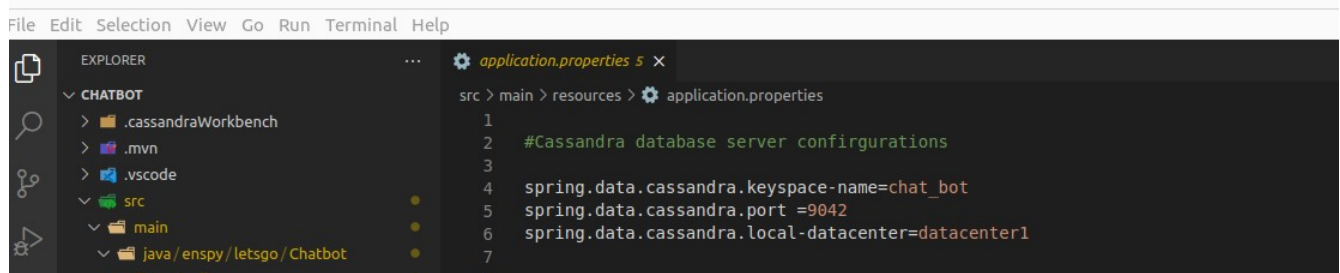
message

cqlsh:chat_bot> 
```

4. configurations de base de l’application

a) Les configuration pour l'accès à la base de donnée

Créer le fichier `application.properties`(s'il n'existe pas) dans le repertoire ressources `src/main/ressources` et renseigner les informations sur le keyspace, ainsi que le port correspondant au port de casting cassandra:

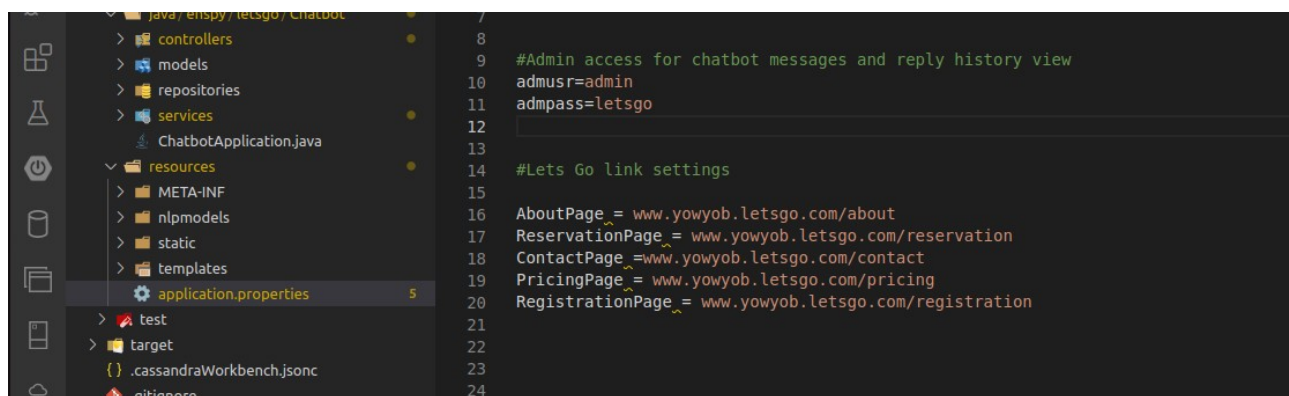


```
File Edit Selection View Go Run Terminal Help
EXPLORER
CHATBOT
  .cassandraWorkbench
  .mvn
  .vscode
  src
    main
      java/enspy/letsgo/Chatbot
        application.properties
application.properties
src > main > resources > application.properties
1
2 #Cassandra database server configurations
3
4 spring.data.cassandra.keyspace-name=chat_bot
5 spring.data.cassandra.port =9042
6 spring.data.cassandra.local-datacenter=datacenter1
7
```

b) Configuration d'exploitation de l'application

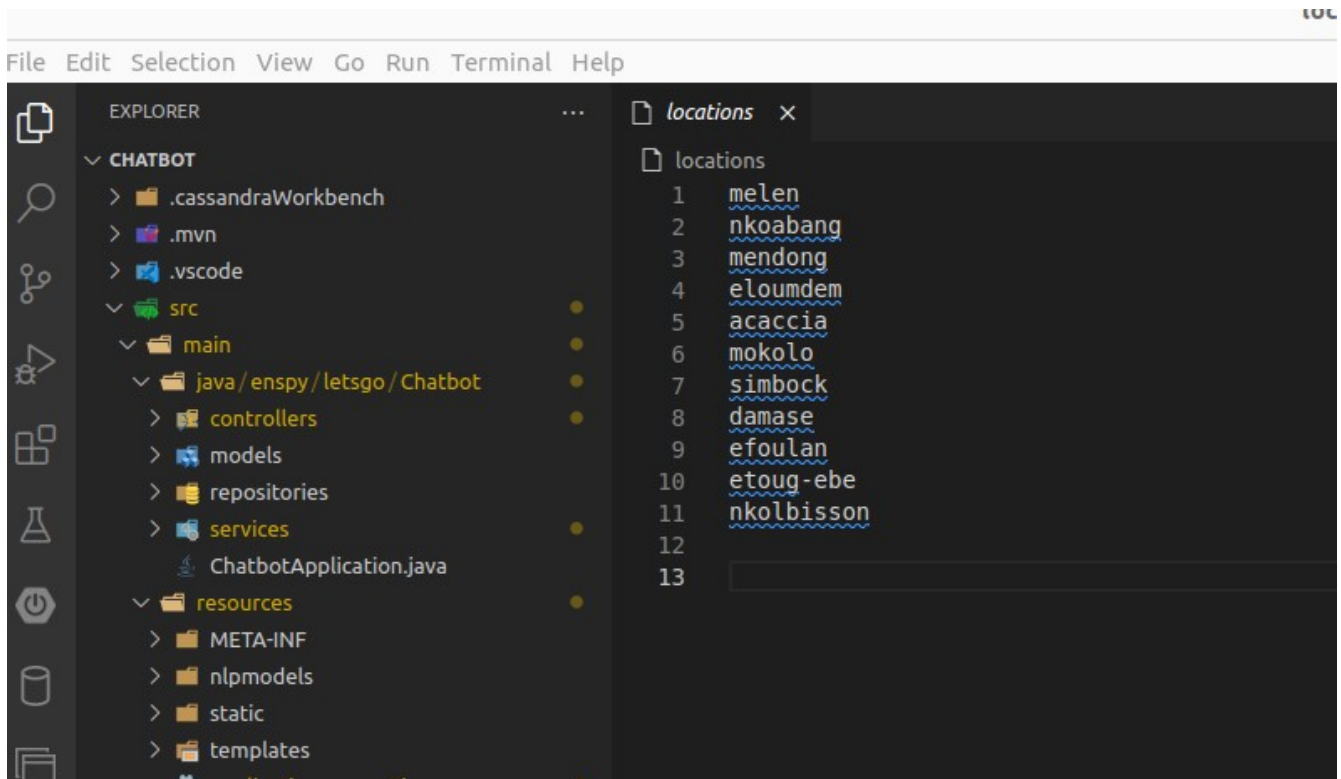
Il faut renseigner les informations suivantes dans le fichier `application.properties`:

- **admusr** et **admpass** doivent correspondre respectivement au username et password de l'administrateur qui souhaite visualiser les messages enregistrés dans la base de donnée.
- **AboutPage**, **ReservationPage**, **ContactPage**, **PricingPage**, **RegistrationPage** doivent respectivement contenir les liens public internet qui pointent vers toutes ces pages.



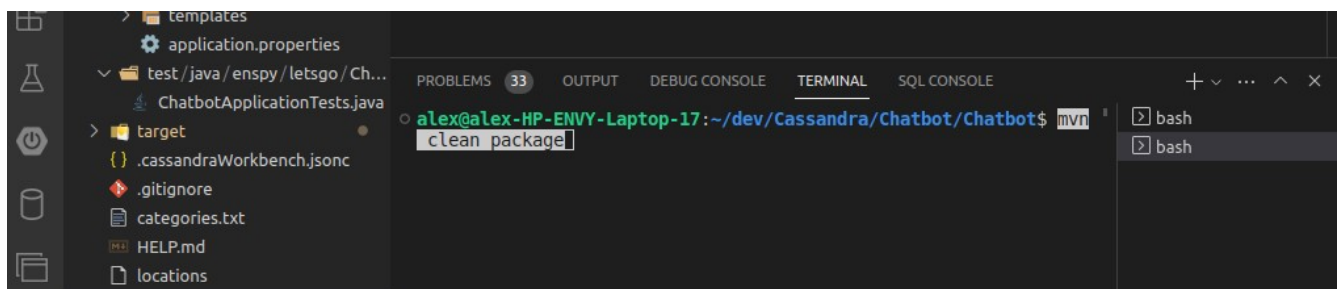
```
java/enspy/letsgo/Chatbot
  controllers
  models
  repositories
  services
  ChatbotApplication.java
resources
  META-INF
  nlpmodels
  static
  templates
  application.properties
test
target
.cassandraWorkbench.jsonc
.gitignore
7
8
9 #Admin access for chatbot messages and reply history view
10 admusr=admin
11 admpass=letsgo
12
13
14 #Lets Go link settings
15
16 AboutPage_ = www.yowyob.letsgo.com/about
17 ReservationPage_ = www.yowyob.letsgo.com/reservation
18 ContactPage_ =www.yowyob.letsgo.com/contact
19 PricingPage_ = www.yowyob.letsgo.com/pricing
20 RegistrationPage_ = www.yowyob.letsgo.com/registration
21
22
23
24
```

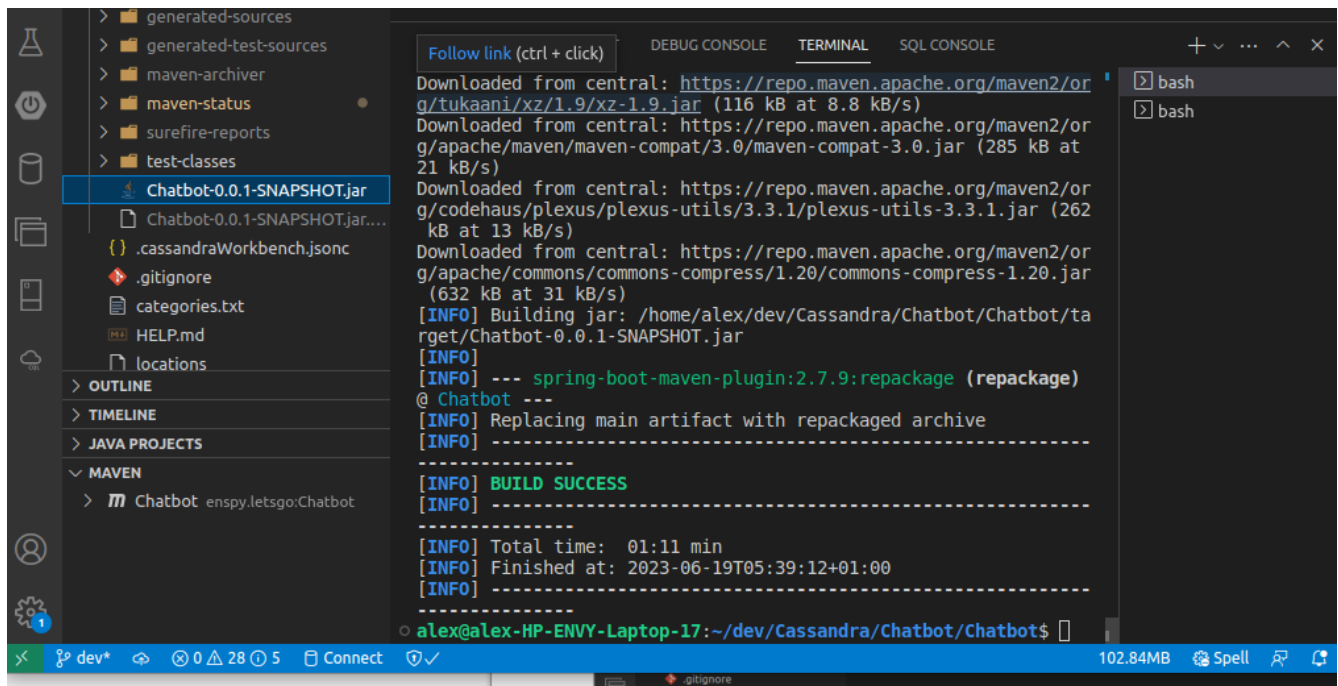
Il se trouve également à la racine du projet, Il faut le compléter avec les lieu de tous les points qui peuvent correspondre aux destinations des clients de Letsgo



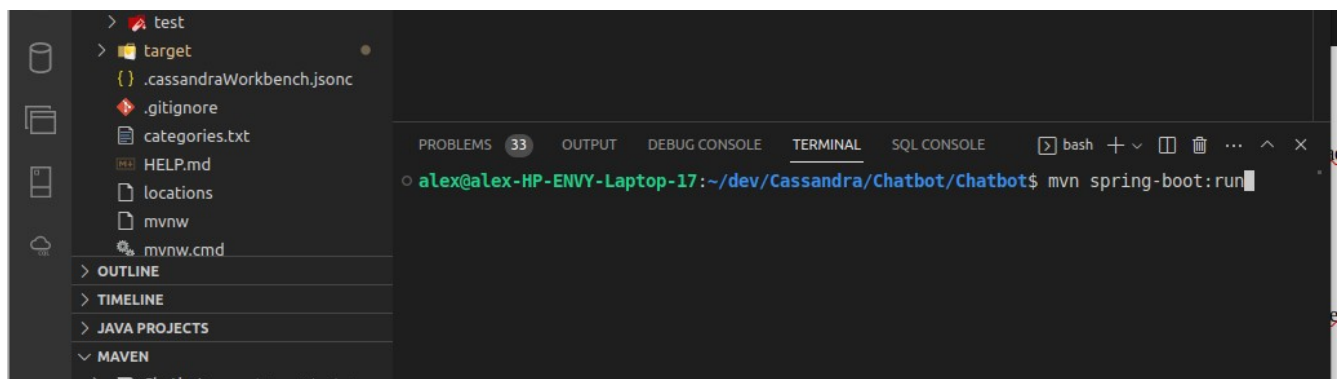
6. Demarrer l'application

- Effectuer le build à travers la commande mvn **“clean package”** ce qui aura pour effet de generer le Jar de l'application dans le repertoire et vous recevrez un message de succès si le build reussi. En cas d'echec du build, verifier les configurations.





- L'application peut être démarrée à travers la commande **"mvn spring-boot:run"** exécutée dans le terminal de l'IDE



- Une fois l'application lancée vous devez recevoir dans le terminal les messages confirmant que les modèles préentraînés ont bien été chargés et que le modèle de catégorisation a été bien entraîné

```

90: ... loglikelihood=-10.6092479997629      1.0
91: ... loglikelihood=-10.506425425311438    1.0
92: ... loglikelihood=-10.405788703328845    1.0
93: ... loglikelihood=-10.307266976748766    1.0
94: ... loglikelihood=-10.210792415928227    1.0
95: ... loglikelihood=-10.116300059804452    1.0
96: ... loglikelihood=-10.023727666876374    1.0
97: ... loglikelihood=-9.933015575313826     1.0
98: ... loglikelihood=-9.844106571552608     1.0
99: ... loglikelihood=-9.756945766784645     1.0
100: ... loglikelihood=-9.671480480798799     1.0

All pretrained models loaded and categorizer's model trained successfully

2023-06-19 05:48:12.184 INFO 535131 --- [ restartedMain] o.s.b
.a.e.web.EndpointLinksResolver : Exposing 1 endpoint(s) beneath
base path '/actuator'
2023-06-19 05:48:12.234 INFO 535131 --- [ restartedMain] o.s.b
.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s):
8080 (http) with context path ''
2023-06-19 05:48:12.251 INFO 535131 --- [ restartedMain] enspy
.letsgo.Chatbot.ChatbotApplication : Started ChatbotApplication
in 3.649 seconds (JVM running for 3.89)

```

III. Creation de L'image Docker du service

Pour créer l'image Docker du service, Ouvrir le terminal, se rendre dans le dossier du projet et lancer la commande **docker-compose up** (rassurez-vous d'avoir au moins 2Go de data et une connexion stable) pour créer le conteneur et l'image docker du service.