Gestion des avis (partie 01)

# TP01 du module 03 – Spring Data MongoDB

|  |
| --- |
| Ce TP sera un TP fil rouge du cours   * Création d’un projet Spring Boot avec Spring Data MongoDB * Création d’un Document * Création du premier Repository de Spring Data MongoDB * Utilisation de tests unitaires pour valider |

|  |
| --- |
| **Durée estimée** |
| 45 minutes |

Contexte

En vous appuyant sur l’ensemble des concepts abordés dans le cours, mettez en place une application Spring Boot permettant la gestion des avis d’une cave à vin.

L’application de cave à vin sera une application d’API Web sécurisée à la fin des itérations.

L’application de gestion des avis sera aussi une application d’API Web :

* Elle doit permettre de gérer les avis des clients sur le vin commandé.
  + La date de l’avis
  + L’identifiant de la bouteille commandée, ainsi que de la Région et couleur
  + La quantité commandée par le client
  + Une note (entre 0 et 5), un commentaire et le pseudo du client (permettre de protéger les informations du client RGPD)
* Le but étant de faire des statistiques pour le propriétaire du Vignoble

Dans les itérations de ce module ; nous réaliserons uniquement du back de l’application. En particulier, la couche DAL et les BO pour le moment.

Cette application se fera avec Spring Boot.

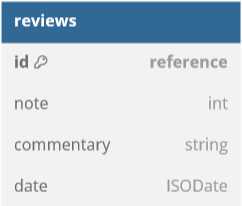
Déclarer la structure des données en Anglais

* Voici le diagramme de classes pour cette itération

Une image contenant texte, Police, reçu, capture d’écran

Description générée automatiquement

* Voici le diagramme de la Collection en base :

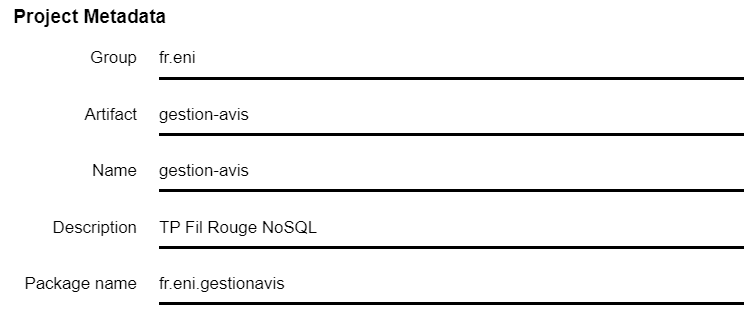


Solution

# Créer une application GestionAvis avec SpringBoot

* Utiliser Spring initializr et sélectionner :
* Gradle (privilégier avec le langage Groovy)
* Java en version minimum 17
* La version courante de Spring Boot (Release)
* Le packaging est en Jar

## Métadonnées :



## Dépendances Spring Boot :

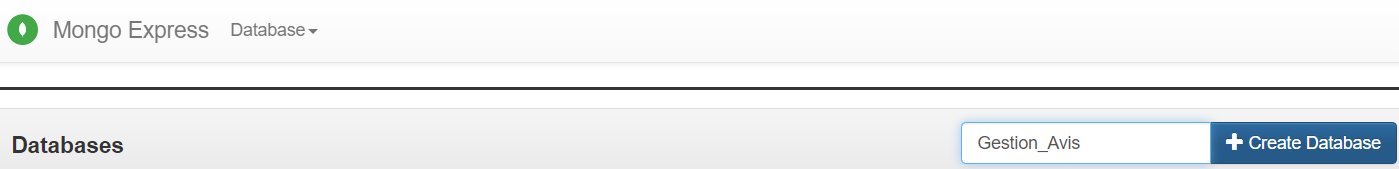
* Lombok
* Spring Boot DevTools
* Spring Data MongoDB
* Importer le projet dans votre IDE

L’ensemble des tests unitaires seront réalisés en base de données comme dans les démonstrations du cours.

# Configurer la base de données

## Dans Mongo Express

* Créer une base de données appelée : Gestion\_Avis sous MongoDB



## Déclaration de la base

* Configurer la source de données associées dans le fichier application.yml
* Renommer le fichier application.properties en application.yml

#Data Source

spring:

data:

mongodb:

username: sa

password: Passw0rd

host: localhost

port: 27017

database: Gestion\_Avis

authentication-database: admin

auto-index-creation: true

#Log

logging:

level:

org:

springframework:

data:

mongodb:

core:

MongoTemplate: DEBUG

* Ajout des informations pour détailler les traces d’exécution de Spring Data MongoDB
  + A supprimer pour mettre en production

# Créer le Document Avis

* Le package de la classe est fr.eni.gestionavis.bo
* Utiliser les annotations de Spring Data MongoDB
* La classe Avis à 4 attributs :
  + id
  + note
  + commentaire
  + et date
* Il faut poser les annotations de Lombok pour respecter le design pattern POJO
  + @Data
  + Et ajouter @Builder, @NoArgsConstructor et @AllArgsConstructor pour créer des instances facilement
* Pour que Avis devienne un Document de MongoDB, il faut
  + @Document + préciser le paramètre collection en reviews
  + @Id sur l’attribut id pour définir l’identifiant
  + @Field sur commentaire pour le renommer commentary

**package** fr.eni.gestionavis.bo;

**import** java.time.LocalDateTime;

**import** org.springframework.data.annotation.Id;

**import** org.springframework.data.mongodb.core.mapping.\*;

**import** lombok.\*;

@Data

@AllArgsConstructor

@NoArgsConstructor

@Builder

@Document(collection = "reviews")

**public** **class** Avis {

@Id

**private** String id;

**private** **int** note;

@Field(name = "commentary")

**private** String commentaire;

**private** LocalDateTime date;

}

# Créer AvisRepository

**package** fr.eni.gestionavis.dal;

**import** org.springframework.data.mongodb.repository.MongoRepository;

**import** fr.eni.gestionavis.bo.Avis;

**public** **interface** AvisRepository **extends** MongoRepository<Avis, String>{

}

# Créer des tests unitaires

* En vous inspirant des tests des démonstrations.
* Créer une classe de tests unitaires qui s’appuie sur le contexte de Spring
  + @Slf4j
  + @SpringBootTest
  + @TestMethodOrder(MethodOrderer.MethodName.**class**)
* Créer 2 tests :
  + Sauver un Avis
  + Remonter tous les Avis

**package** fr.eni.gestionavis.bo;

**import** **static** org.assertj.core.api.Assertions.*assertThat*;

**import** java.time.LocalDateTime;

**import** java.util.List;

**import** org.junit.jupiter.api.\*;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.boot.test.context.SpringBootTest;

**import** fr.eni.gestionavis.dal.AvisRepository;

**import** lombok.extern.slf4j.Slf4j;

@Slf4j

@SpringBootTest

@TestMethodOrder(MethodOrderer.MethodName.**class**)

**class** TestDocument {

@Autowired

AvisRepository avisRepository;

@Test

**void** test01\_save\_avis\_sans\_commentaire\_cours() {

Avis avis = Avis

.*builder*()

.note(4)

.commentaire("Vin délicieux")

.date(LocalDateTime.*now*())

.build();

Avis avisDB = avisRepository.save(avis);

//Vérifier que l'identifiant n'est pas nul

*assertThat*(avisDB.getId()).isNotNull();

*assertThat*(avisDB.getId()).isNotBlank();

//Vérifier que tous les attributs sont remplis et non nuls

*assertThat*(avisDB.getNote()).isNotNull();

*assertThat*(avisDB.getNote()).isEqualTo(4);

*assertThat*(avisDB.getCommentaire()).isNotNull();

*assertThat*(avisDB.getCommentaire()).isNotBlank();

*assertThat*(avisDB.getDate()).isNotNull();

***log***.info(avisDB.toString());

}

@Test

**void** test02\_findAll\_avis() {

List<Avis> listeAvis = avisRepository.findAll();

*assertThat*(listeAvis).isNotNull();

*assertThat*(listeAvis).isNotEmpty();

***log***.info(listeAvis.toString());

}

}

* Les 2 tests sont verts à l’exécution
* Voici le type de trace obtenu :

Inserting Document containing fields: [note, commentary, date, \_class] in collection: reviews

Avis(id=64be3a1672bd0840feb1b6c9, note=4, commentaire=Vin délicieux, date=2023-07-24T10:45:10.618940800)

find using query: {} fields: Document{{}} for class: class fr.eni.gestionavis.bo.Avis in collection: reviews

[Avis(id=64be3a1672bd0840feb1b6c9, note=4, commentaire=Vin délicieux, date=2023-07-24T10:45:10.618)]