Cave à vin (partie 11)

# TP05 du module 04 – Spring Web Service

|  |
| --- |
| Ce TP sera un TP fil rouge du cours   * Ajout d’un Exception Handler * Mise en place de l’internationalisation des exceptions |

|  |
| --- |
| **Durée estimée** |
| 20 minutes |

Contexte

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

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

* Elle permet de gérer le stock des bouteilles de vins par le propriétaire qui est un administrateur de l’application
* Elle permet de gérer l’achat en ligne des bouteilles par un client
  + Les bouteilles de vin peuvent avoir 3 couleurs : Rouge, Blanc ou Rosé
  + Les bouteilles de vin sont associées aux régions françaises
* Un client et un propriétaire doivent s’authentifier
* Un visiteur est un client qui n’a pas de compte
* Il pourra uniquement consulter le stock

Dans les itérations nous réaliserons uniquement le back de l’application.

Cette application se fera avec Spring Boot.

Dans les itérations de ce module, nous allons développer l’API Web.

Solution

## Copier la classe AppExceptionHandler de la démonstration

* Elle correspond aux erreurs qu’il faut capturer pour le moment
  + Exception à pour ne pas laisser d’exception Java passer
  + MethodArgumentNotValidException à pour gérer les exceptions de validation

## Mise en place de l’internationalisation

* + Copier les 3 fichiers properties fournis
  + Configurer application.properties avec l’Anglais par défaut

#i18n

spring.messages.encoding=UTF-8

spring.web.locale=en

* + Appeler les clefs sur les annotations de validation

**package** fr.eni.cave.bo.vin;

…

**public** **class** Bouteille {

…

**@NotBlank(message = "{bottle.name.blank-error}")**

**@Size(max = 250, message = "{bottle.name.size-error}")**

@Column(name = "NAME", length = 250, unique = **true**, nullable = **false**)

**private** String nom;

…

**@Size(max = 100, message = "{bottle.vintage.size-error}")**

@Column(name = "VINTAGE", length = 100)

**private** String millesime;

**@Min(value = 1, message = "{bottle.quantity.min-error}")**

@Column(name = "QUANTITY")

**private** **int** quantite;

**@Min(value = 1, message = "{bottle.price.min-error}")**

@Column(name = "PRICE", precision = 2)

**private** **float** prix;

**@NotNull(message = "{bottle.region.not-null}")**

@EqualsAndHashCode.Exclude

@ManyToOne

@JoinColumn(name = "REGION\_ID")

**private** Region region;

**@NotNull(message = "{bottle.color.not-null}")**

@EqualsAndHashCode.Exclude

@ManyToOne

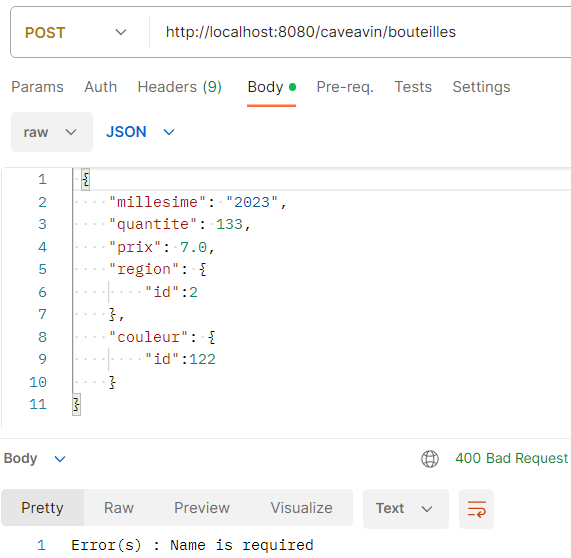
@JoinColumn(name = "COLOR\_ID")

**private** Couleur couleur;

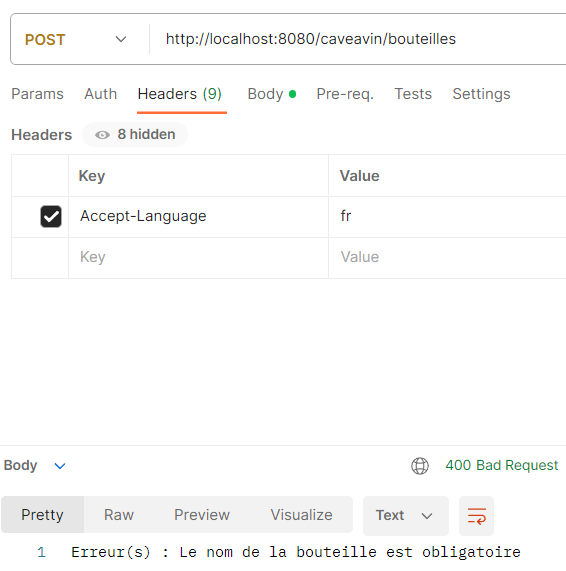
}

## Tester avec Postman

* Test de nom qui est nul :



* + Si on lance avec la langue par défaut, on obtient bien le message en Anglais
* Si on ajoute dans l’entête : Accept-Language à fr



* + On obtient bien le message en Français
* Tous les autres cas sont similaires