Après la semaine de relâche

[RENCONTRE 11](#_Toc69291299)

[TP2 – 31 Mars Sprint 2](#_Toc69291300)

[Scrum Flow \* Voir pdf Processus Scrum](#_Toc69291301)

[DFD](#_Toc69291302)

[Annotations](#_Toc69291303)

[Repository](#_Toc69291304)

[@Temporal](#_Toc69291305)

[@Component](#_Toc69291306)

[@Order(nombre) doit être accompagné de @Component](#_Toc69291307)

[@service](#_Toc69291308)

[@Autowired](#_Toc69291309)

[@Controller (pour un contrôle de servlet)](#_Toc69291310)

[Itext](#_Toc69291311)

[RENCONTRE 12](#_Toc69291312)

[Phase02](#_Toc69291313)

[LocalDate](#_Toc69291314)

[Service](#_Toc69291315)

[JUNIT 5](#_Toc69291316)

[RENCONTRE 13](#_Toc69291317)

[Sprint 2](#_Toc69291318)

[Stratégie de programmation](#_Toc69291319)

[<scope></scope>](#_Toc69291320)

[GenererQR](#_Toc69291321)

[GenererPDF](#_Toc69291322)

[Envoyer mail A REVOIR](#_Toc69291323)

[RENCONTRE 14](#_Toc69291324)

[Phase02](#_Toc69291325)

[@DataJpaTest](#_Toc69291326)

[Environnement = fichier .properties](#_Toc69291327)

[Repositories](#_Toc69291328)

[@RestController](#_Toc69291329)

[RENCONTRE 15](#_Toc69291330)

[CheckUSER](#_Toc69291331)

[requestPermit](#_Toc69291332)

[Puller](#_Toc69291333)

[Test théorique](#_Toc69291334)

[Test Examen](#_Toc69291335)

[RENCONTRE 16](#_Toc69291336)

[TP 2](#_Toc69291337)

[2 types de WebService](#_Toc69291338)

[MVC Fournisseur du WS](#_Toc69291339)

[Exemple Ministere](#_Toc69291340)

[Est-ce qu’une classe a besoin d’un repo ou non?](#_Toc69291341)

[Le cas normal](#_Toc69291342)

[Le cas anormal](#_Toc69291343)

[Le cas improbable](#_Toc69291344)

[RestController](#_Toc69291345)

[RENCONTRE 17](#_Toc69291346)

[Différence entre @Service et @Repository](#_Toc69291347)

[Puller](#_Toc69291348)

[@EnableScheduling](#_Toc69291349)

[@Scheduled(cron = « 0 30 01 ? \* \*)](#_Toc69291350)

[RENCONTRE 19](#_Toc69291351)

[RENCONTRE 20 – TP3 INTÉGRATION WEB](#_Toc69291352)

# RENCONTRE 11

## TP2 – 31 Mars Sprint 2

### Scrum Flow \* Voir pdf Processus Scrum

Sprint 1 était de comprendre le besoin client

Les besoins clients ne seront jamais accompli tant qu’on ne finit pas le projet.

Toujours corriger les erreurs ou les malentendus au prochain livrable

Sprint 2 =- coder les services et Expliquer l’architecture 🡪 31 mars

Sprint 3 = coder interfaces graphiques 🡪 28 Avril

Sprint 4 = Intégrer le tout, déployer l’appli sur AWS et coder appli admin 🡪 21 Mai

Vous DEVEZ le dire lorsque vous aller remettre le livrable en retard pour libérer des ressources sur le projet.

Livrez une première version!

### DFD

Diagramme de flux de données

## Annotations

Définit une méta-data qui est spécial

### Repository

= Un objet spring DAO, il permet de fournir toutes les méthodes nécessaires pour manipuler la table du modele spécifier (Tout ce qu’on a besoin pour le code SQL)

Sert a générer des requetes SQL

### @Temporal

Pour déclarer un type approprier de TemporalType, seulement possible pour un objet Date

@Temporal(TemporalType.*DATE*)

### @Component

Une classe java de type Spring Bean prioritaire

### @Order(nombre) doit être accompagné de @Component

Définit la priorité du Bean(Component) 1 étant le plus prioritaire.

### @service

Sert a exposer des méthodes de service vers l’extérieur comme un Webservice

### @Autowired

L'annotation @Autowired vous permet de sauter des configurations ailleurs de ce qu'il faut injecter et le fait juste pour vous.

Il faut soit instancier le repository avec tous les méthodes OU encore mieux

Injecter les méthodes de l’interface grâce à @Autowired

### @Controller (pour un contrôle de servlet)

## Itext

Librairie Java qui génère des codes QR et des pdf

# RENCONTRE 12

## Phase02

### LocalDate

private LocalDate datePermis;  
private LocalDate dateExpiration;  
  
public Permis () {  
 datePermis = LocalDate.*now*();  
 setDateExpiration(datePermis.plusDays(15));  
}

Toutes les requêtes qui finissent par un WHERE sont des findBy ou deleteBy

### Service

La classe service sert à cacher les méthodes de service.

Ex : Un serveur de restaurant, c’est un obstacle entre le client et le cuisinier pour ne pas voir

### JUNIT 5

Un JUNIT c’est un test

**PAR CONVENTION** : Un test unitaire ne retourne rien!

#### Un Bon test unitaire

Un bon test unitaire, il faut bien choisir ses données

La technique des 3 A est utilisée.

#### @DataJpaTest

Toutes les tests unitaires se feront sur une BD h2 en mémoire qui sera lancé que pour les TESTS .Pour ne pas toucher à ta vrai BD lors des tests!

#### @TestInstance(TestInstance.Lifecycle.PER\_CLASS)

Pour bypasser que BeforeAll doit être static

#### Annotations Cycle de vie

Pour charger les données. Méthode de JUNIT.

* @BeforeEach
* @AfterEach
* @BeforeAll

La méthode doit être statique

Dit que la Méthode est à FORTE PRIORITÉ

Il est bon de nettoyer sa BD lors de cette méthode

* @AfterAll
* @Test => Les méthodes a tester //Act
* @DisplayName pour donner un nom au test exécuté

#### Stratégie des 3 A

* Arrange 🡪 Préparer les données de test
* Act 🡪 Appeler la méthode à tester.
* Assert 🡪 Appel les méthodes statiques de l’objet Assertions.

#### @Disabled

Pour démontrer que le test est déjà passé, au lieu de le commenter ou l’effacer

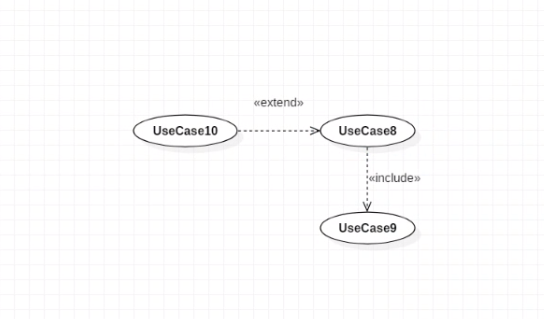
#### TDD(Test Driven Development)

Coder les tests avec les méthodes de service, ensuite, coder les méthodes de service.

# RENCONTRE 13

## Sprint 2

Dans le relevé 2 sprint 2, il faut faire un glossaire (définition des termes, java, Spring, …)



Extend : non obligatoire pour le usecase8

Include : UseCase9 doit être inclus dans UseCase8

### Stratégie de programmation

A mettre dans le cahier des charges

Dans cette phase, on doit coder les services et les tests unitaires. Pour cela, on va ultiser la technique du **TDD**, développement piloté par les tests.

|  |  |  |  |
| --- | --- | --- | --- |
| UseCases | Repositories | Méthodes de Service | Tests Unitaires |
| Login  UC-1 | findUserByCourrielAndPassword(courriel, pwd) | Login(username, password) | testLogin() |
| GenererQR  Uc-2 |  | genererQR(citoyen) | testGenererQR() |
| sendEmail  UC-3 | findUserByCourriel(courriel) | sendEmail(email) | testSendEmail(user.email) |
| GenererPDF  UC-4 |  | genererPDF(citoyen) | testGenererPDF(citoyen) |
| renouvellerQR | findUserByCourriel (courriel) | renouvellerQR(Citoyen) | testRenouvellerQR(Citoyen) |

Librairie ZXing pour les codes à barres de type QR. (EXIGENCE TECHNIQUE)

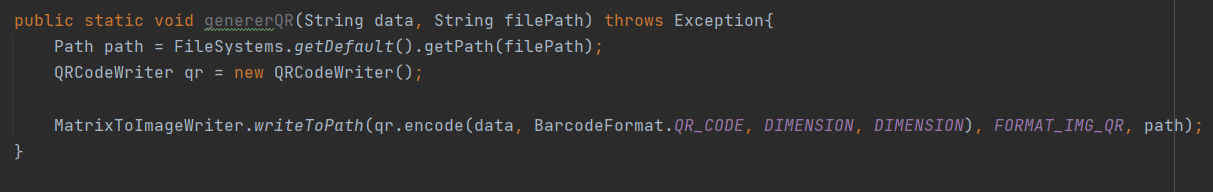
ITest pdf pour les PDF

### <scope></scope>

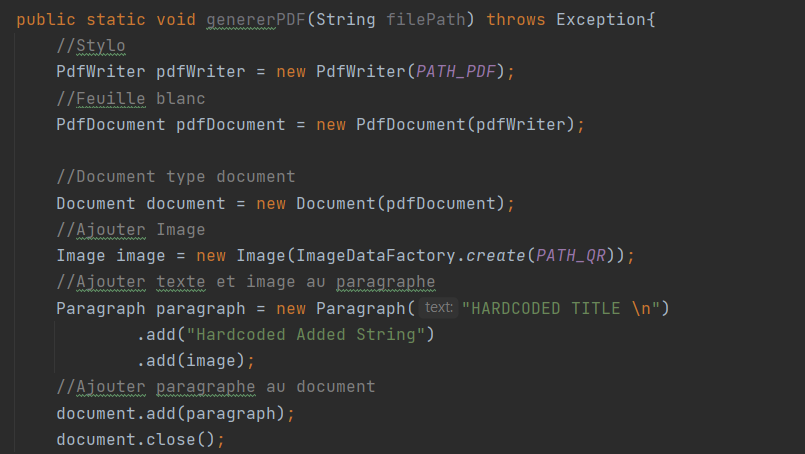
Cycle de vie de la dépendance (ex : seulement pendant les tests unitaire = <scope>test</scope>

<type>pom</type> pour les versions payantes 🡪 type pom pour l’utiliser sans payer

### GenererQR

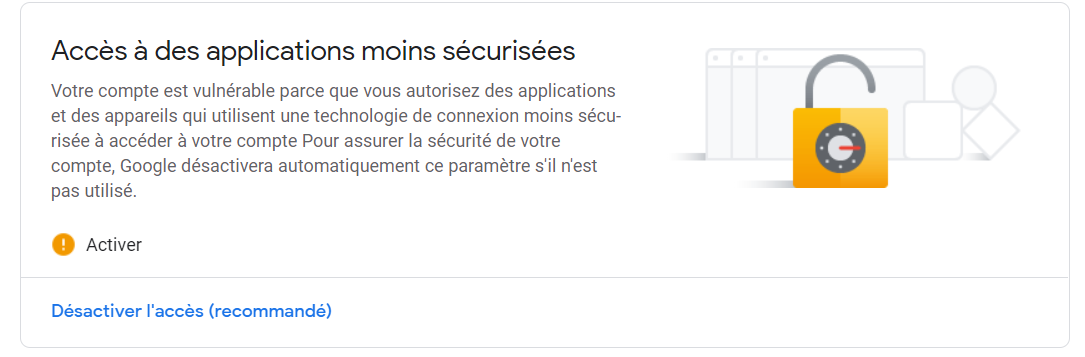


### GenererPDF



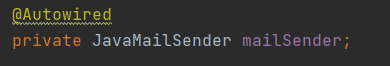
### Envoyer mail A REVOIR

##Envoyer un mail avec le protocole smtp  
spring.mail.host=smtp.gmail.com  
##Port du smtp  
spring.mail.port=587  
##Avant le @gmail.com  
spring.mail.username=maxirus14ans  
##MotDePasse  
spring.mail.password=maxou1234  
  
#Authentification autorisation + changer propriers sur gmail pour autoriser  
spring.mail.properties.mail.smtp.auth=true  
spring.mail.properties.mail.smtp.starttls=enable

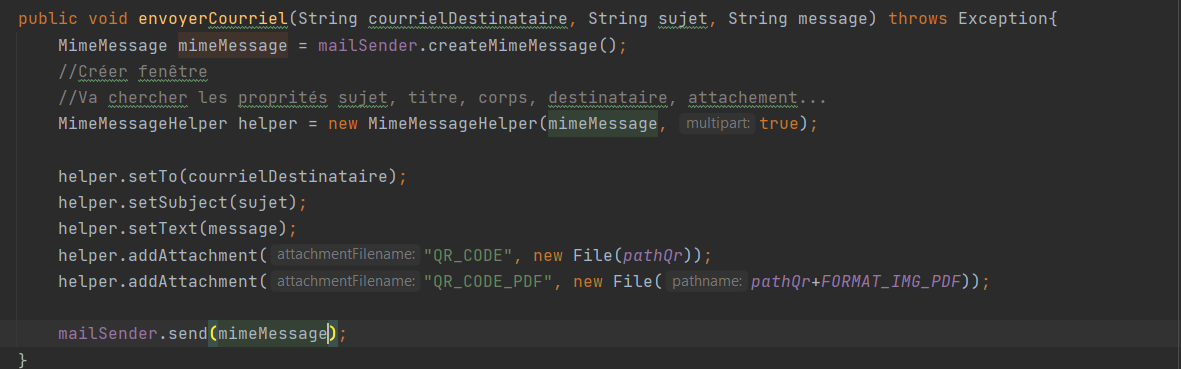


#### Code

Services



Méthode



# RENCONTRE 14

//Va masquer les méthodes du Repository  
@Service

## Phase02

### @DataJpaTest

Lorsqu’on n’a pas actuellement de base de données. Pour tester l’application sans BD

Cycle de vie juste pour le test

@DataJpaTest  
@TestInstance(TestInstance.Lifecycle.*PER\_CLASS*) //POUR BYPASS LE STATIC

public class AppPermisTests {

### Environnement = fichier .properties

Toutes les clefs doivent être différentes

//Vient du fichier properties  
@Autowired  
private Environment environment;

Vient du fichier properties

qrCode.width=300  
qrCode.height=300  
qrCode.format=PNG

### Repositories

Les repo ont par défaut les méthodes find, delete, exists et count

### @RestController

Va faire comme Comme Postman, POUR FAIRE DES CALLS VERS LE WEBSERVICES

Dépendance requises :

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
</dependency>

Ex : vers le ministère

# RENCONTRE 15

TP 🡪 //renouveler permis

## CheckUSER

Call WS(if ok) {

Delete old permis

Generate new QR + send email

Update BD

} else {

Message error

}

## requestPermit

if(getPermis != null){

check(user)

else

check(user)

## Puller

Un thread qui tourne tout seul. Un programme qui va tourner et qui balayer

Idée TP : Puller pour BD de Permis expiré, MAIS AUSSI POUR CODE d’envoi pour citoyen en bas age!

Dans une application autre.

# Test théorique

https://testmoz.com/q/8083810

CAL420455T01 par Reda Hamza

**Reda Hamza**12:48

CAL420455T01

# Test Examen

Système

Écrire les useCases avant les acteurs

Lorsque l’on écrit le diagramme de useCase, écrire ce qui est dit sans rien ajouter pour la v1 du moins.

# RENCONTRE 16

## TP 2

Rest

Angular

Model Service App

Back-end

call HTTP

Citoyen

Permis

callWS

Ministère

Dans un autre schéma

Web Service = call entre BD et classes

APP

Class et Interface

Web service

BD

Toutes les méthodes peuvent aller dans un WS

On doit protéger les classes et les interfaces

### 2 types de WebService

#### WS

Consommateur du WebService = La méthode n’est pas codée par le consommateur

#### Autres

Fournisseur des méthodes du WebService 🡪 Fournisseur a coder les méthodes

### MVC Fournisseur du WS

Request

Ministère

App

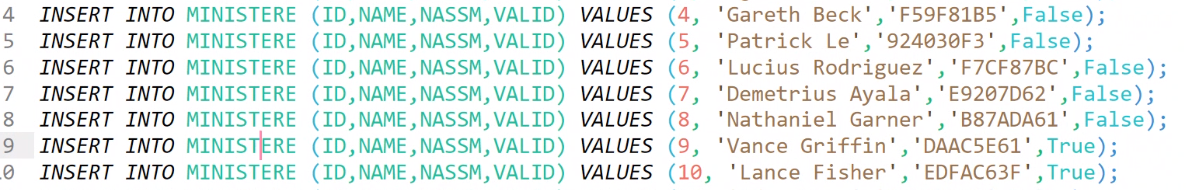
Angular

HTTP

réponse

### Exemple Ministere

Changer nom du ministère par autre chose, car ceci est la liste des citoyens qui ont un nassm.



Pour créer un autre projet dans le projet >new > modules puis lui donner tous les dépendances qu’il a besoin.

Une fois que ca fonctionne, on dump les données dans une bd local

## Est-ce qu’une classe a besoin d’un repo ou non?

Recherche?

Select

Compter?

Select count

Modifier?

Update pas avec JpaRepo, utiliser sa propre méthode dans le repo ou dans le service

Supprimer?

DELETE

LES SERVICES SONT LÀ POUR CACHER LES CHOSES?

### Le cas normal

ex : Je lui donne le permis car numéro valide.

ex : Je ne donne pas le permis car numéro non valide.

### Le cas anormal

ex : Je lui donne le permis même si le numéro est non valide

ex : Je ne lui donne pas le permis même si le numéro est valide.

### Le cas improbable

Le cas où tu dois te creuser la tête ex : regarder les 2 côtés de la rue dans pour une rue en sens simple.

## RestController

Méthode pouvant être exécuter par HTTP

GET pour récupérer

POST pour insérer

PUT pour modifier

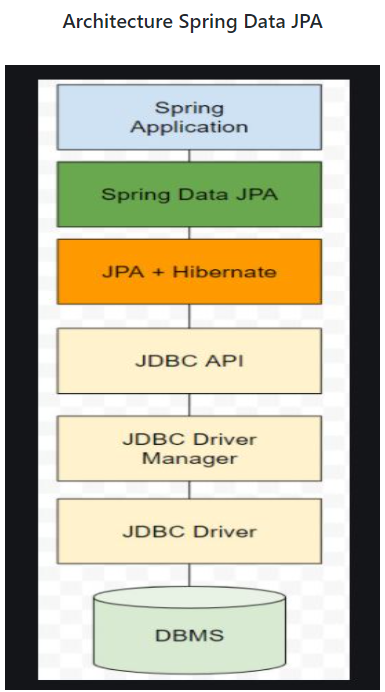
DELETE pour supprimer

# RENCONTRE 17

## Différence entre @Service et @Repository

Un Service vient de Spring et un Repository c’est avec JPA 🡪 JDBC, c’est de bas niveau

Le service sert a cacher les méthodes donné par le JpaRepository



On ne peut pas tester les send email, car le smtp ne nous appartient pas, car on ne reçoit pas le signal ACK(Acknowledge).

## Puller

### @EnableScheduling

sur l’application

### @Scheduled(cron = « 0 30 01 ? \* \*)

sur la métode updateDB()

# RENCONTRE 19

Diagram de Flux de données = Comment naviguer (DFD)

# RENCONTRE 20 – TP3 INTÉGRATION WEB

En web, essayer de faire du code qui peut être réutilisé.

Une méthode doit faire une et une seule chose.

Faire une action à la fois. KISS 🡪 Keep It Simple Stupid

Faire du code facile à utiliser, à reprendre.

# RENCONTRE 21

## Pour recevoir 2 parametres lors du request Mapping du back end de l’appli

@GetMapping (‘’/permisSante/{email}/{password}

Public boolean login(@PathVariable(“email) String email, @PathVariable(“password”) String password){

If(citizenRepository.findByEmailAndPassword(email, password) != null) {

Return true;

}

Return false;

}

# RENCONTRE 22

Puller peut etre fait avec spring ou un script linux

DÉCONSEILLER : Pour faire le cross origin du coter Angular pour connecter spring et angular avec un PROXY

# RENCONTRE 23

## Ligne de commande pour se connecter à l’instance Amazon

### Connection

Ssh -i 14avril.pem @ubuntu [24.201.62.198](https://www.whatismyip.com/24.201.62.198/)

### Installation

**apt-get**

Sudo apt update …

Sudo apt install nodejs

Sudo apt install npm

Sudo apt install angular-cli

Sudo apt-get install mysql-server -y

Sudo apt install java-jdk-11

### Entrée dans mysql

Sudo mysql

Mysql> CREATE DATABASE MINISTERE;

Mysql>SHOW DATABASES;

CREATE USER ‘reda’@’localhost’ IDENTIFIED BY ‘reda1234’;

GRANT ALL PRIVILEGES ON MINISTERE.\* TO ‘reda’@’localhost’ WITH GRANT OPTION;

FLUSH PRIVILEGES; **Il faut toujours flush privileges pour reload.**

Exit; 🡪 Pour sortir

ALTER USER ‘root’@’localhost’ IDENTIFIED WITH mysql\_native\_password BY ‘reda1234’

### Dans l’instance (pas mysql)

SELECT user,authentication\_string, plugin,host FROM mysql.user;

Sudo service mysql status **pour avoir le statut de mysql**

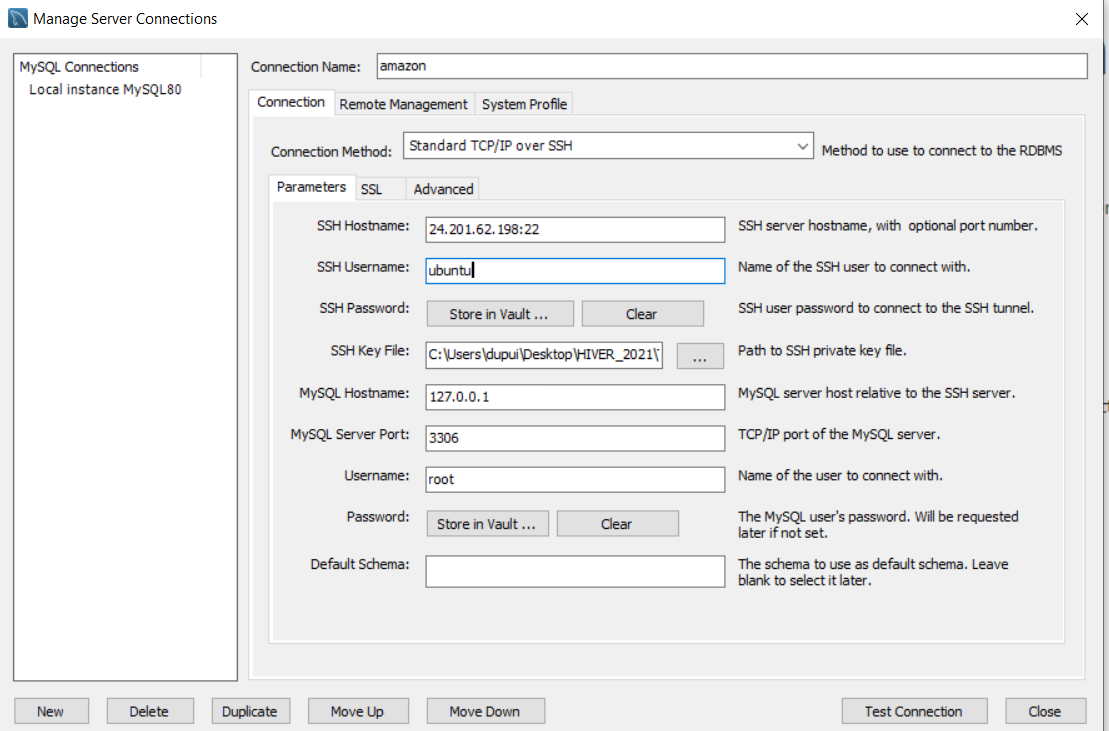
Sudo service mysql start **pour start le service.**

Sudo service mysql stop **pour stop le service.**

Sudo servicemysql restart **pour restart le service**

Sudo mysqladmin -p -u root version **avoir la version de l’user root**

Sudo tail -f /var/log/mysql/error.log **avoir les derniers log** -f pour rester dans le fichier



+ STORE IN VAULT!!!

# RECONTRE 24

Pour bloquer certaines routes :

On créer un Guard

On dit quelle route on veut qui utiliser la méthode du Guard

# RENCONTRE 26

## RÉVISION

Spring, il faut beaucoup de libraire. = Framework qui implémente le concept IoC. Spring instancie les classes à notre place. Spring connait lorsqu’on a besoin des classes.

Scope : = portée

SpringBoot normalise = un outil

**Projet :**

Monter une architecture :

**Spring boot devtool** = Au lieu de faire une modification, permet a Spring de redémarrer sans avoir a redémarrer le projet.

**Lombok** pour éviter d’Avoir des getters et des setters dans les classes

Spring Web = Serveur d’application Java Tomcat using Spring MVC

**Rest Repositories** sert à

1. Structurer le projet

@Component = Cet objet est un objet qui n’a pas besoin d’être instancié

# RENCONTRE EXAMEN

BD

Front-end

Back-end

Model

JPA

WebService

Épreuve Certificative : Présentation orale du TP Final

# Testmoz

Une méthode finale ne peut pas être utiliser de l’extérieur, mais elle peut – être appelé.

## La POO

Tout ton code est structuré en objet, objet configuration, …

Un objet peut-il parler avec un autre objet? Déterminer dans le UML. RELATIONS

KISS

YAGNI

DRY

S.O.L.I.D

Héritage A extends B ou bien B extends A

Polymorphisme : A utilise les méthodes de B puisque A hérite de B.

Abstraction : Interface ou classe abstraite : A implements B

Relations Normal : 1 à 1 (Composition et agrégation, mais aussi association)

1 à \* ou \* à 1 ou \* à \*

Mots clés : encapsulation : Quel objet ou quel méthode ou quel attribut peut être vu ?

Private –

Public +

Protected #

Visibilité :

Package-private default java

Class **final** Calcul{

Une classe final ne pourra pas être appelé dans une autre classe.

}

Surcharge :

Redéfinition : Override 🡪 Que dans le cas de l’héritage ou de l’abstraction.

**Début DÉCLARATION JAVA** : a…z, A…Z, \_, $

Stratégie des 3 A test unitaire :

A Arrange 🡪 Préparer les données

A Act 🡪 Appeler la méthode a tester : Une combinaison entre les données et la méthode a tester.

A Assert 🡪 Vérifier les résultats de test.

Command line runner = C’est un objet thread de forte priorité qui va être exécuté quand l’application est compiler.

Spring boot faciliter la gestion des dépendances

Serveur d’application est un serveur dédié a un langage . Ex : Tomcat de Appache(embarqué)

Pour développer des application incluant le rest controller et le MVC

Weblogic

Websphere

Glassfish

Autowired : IOC 🡪 objet Bean qui vont être instancié

Exprime l’injection de dépendance (DI)

Bean = jouer le rôle d’une table ou le mapping d’une table.

**Méthode et classe finale ne peuvent pas être redéfini!**

# Livre conseillé par Reda

Certification Java OCP

Craig Larman 🡪 Inventeur des GRAPS Pattern

Applying UML and Patterns

Clean coder 2011 uncle bob martin

# Pour Stocker IMG dans SQL

