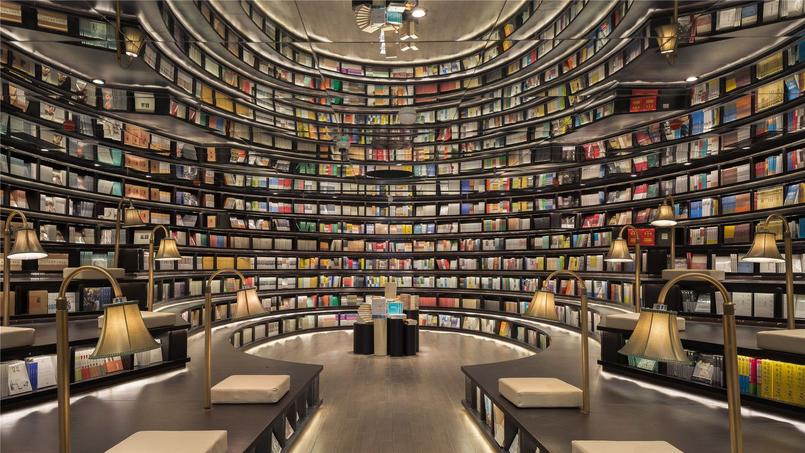
Bibliothèque Studi

Examen certifiant du bloc - 1er dépôt - Le développement d'une solution digitale



POSADAS RODRIGUEZ Jesus Humberto

20/10/2022

# Document technique

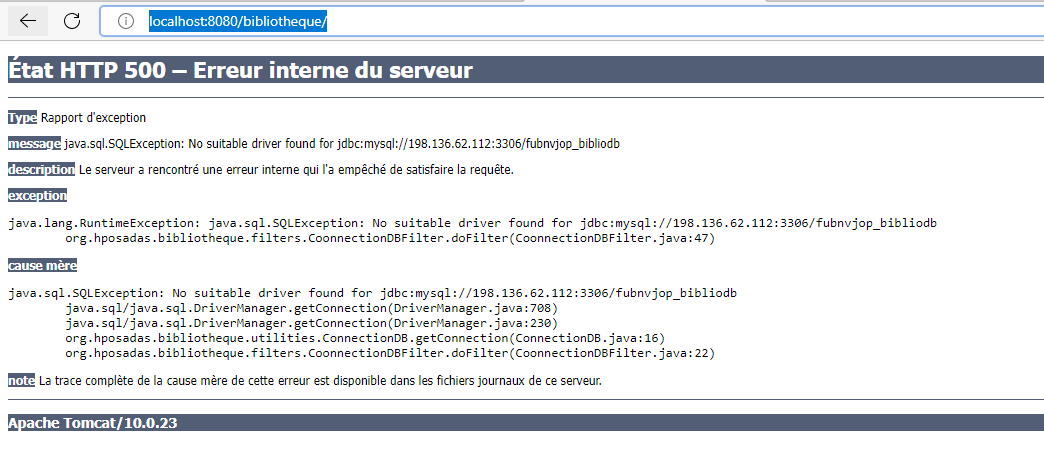
## Lien du projet déployé

Le projet est en ligne et peut être consulté dans le lien suivant :

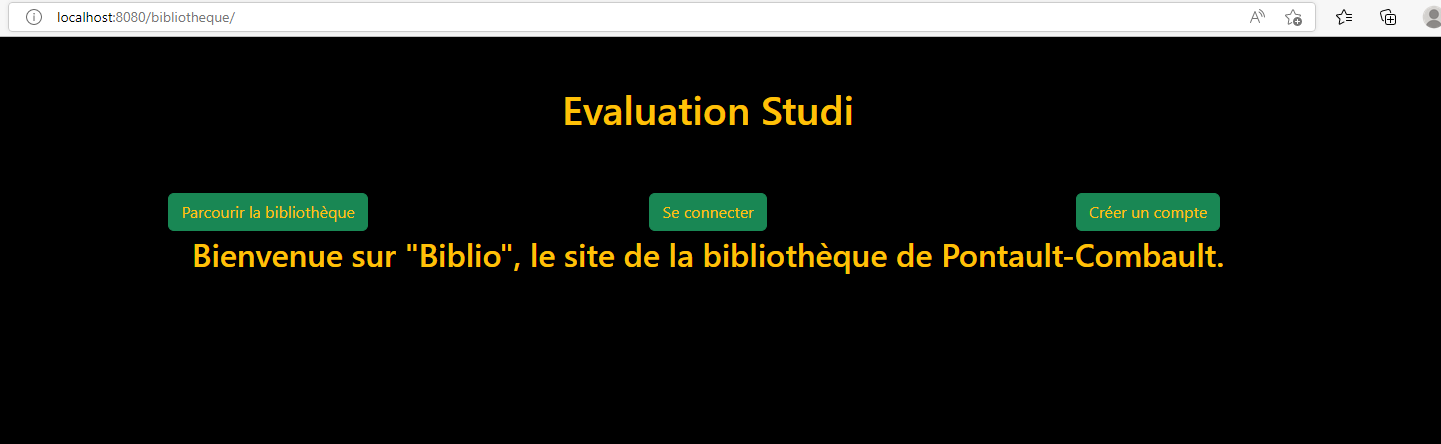
<http://bibbliostudi.sh2.hidora.net/>

Attention, depuis la dernière version du projet, j’ai implémenté une Remote Database pour rendre mon application plus facile a installer pour le formateur en locale, mais cela m’a occasionné un petit problème que j’imagine se trouve dans le fait que l’hébergeur de ma base de données Remote se trouve en Amérique (c’est Banahosting, l’hébergeur de l’entreprise qui m’accueil pour faire mon stage de fin de formation) et l’hébergeur de mon application se trouve en Europe.

Le problème c’est que lorsque je charge la page la première fois une erreur 500 de driver mysql avec le message suivant :



Ce problème va se résoudre en rechargeant la page. Alors si vous faites recharger la page vous allez avoir l’application qui apparait soit en local une fois l’application installe dans votre ordinateour ou bien sur le lien d’acces au projet deployé.



## Couleurs

![Shape

Description automatically generated with low confidence](data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAALUAAAB3CAIAAADKEZZBAAAAAXNSR0IArs4c6QAAATBJREFUeF7t0IEAAAAAw6D5Ux/khVBhwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBgwIABAwYMGDBg4Gdg/OAAAXzkAA8AAAAASUVORK5CYII=) #000000

Shape, square

Description automatically generated #ffc107

Shape, rectangle

Description automatically generated#0d6efd

* Pour le choix des couleurs j’ai pris le couleur noir qui représente l’environnement ombre nécessaire pour une bonne visibilité de l’écran en empêchant la fatigue visuelle.
* La couleur jaune vient du logo Studi.
* La couleur bleue représente ma couleur préférée

## Contexte

Il s’agit d’une application capable de résoudre un besoin en spécifique de la part du client, la gestion de la bibliothèque qui n’avait pas de système de gestion informatique car tout était géré en papier.

Ce système, déjà commencé par le chef de projet, devait respecter les standards opquast accessibles dans le lien suivant : <https://checklists.opquast.com/>

## Ecrans

### Index :

Graphical user interface, website

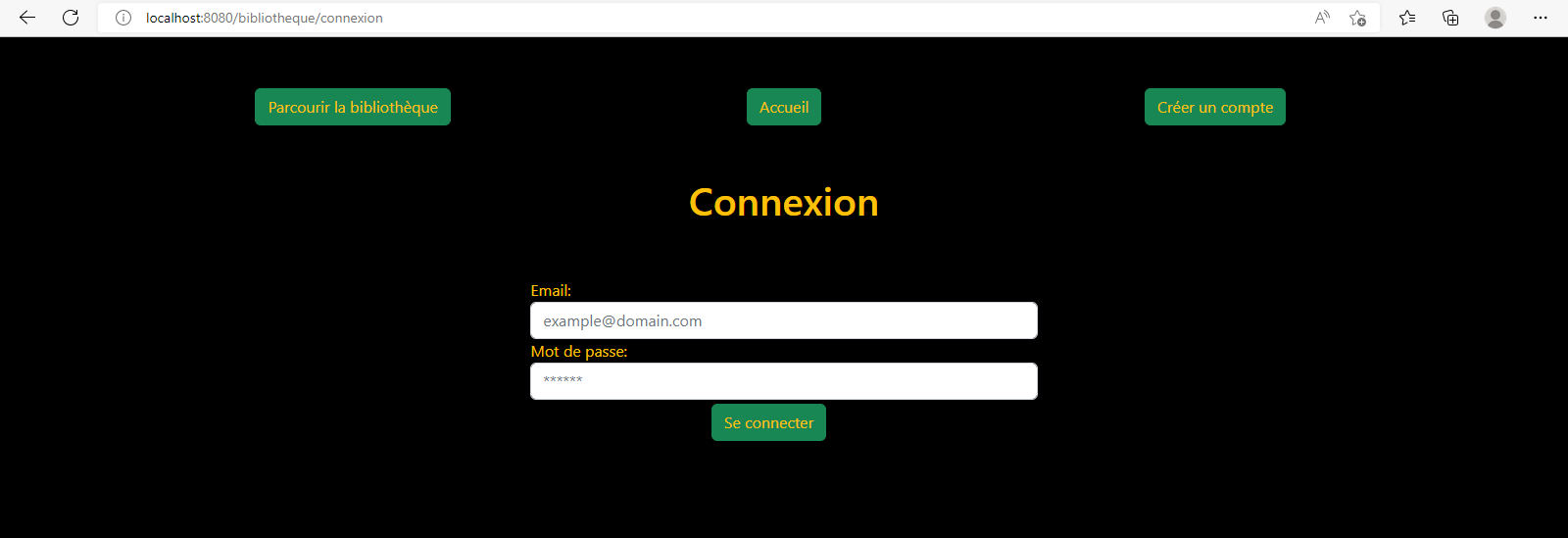
Description automatically generated

### Créer un compte :

Graphical user interface, text

Description automatically generated

### Connexion :



### Modifier informations :

Graphical user interface, application, Teams

Description automatically generated

### Rechercher des livres :

A screenshot of a computer

Description automatically generated with medium confidence

### Suivre les prêts en cours :



## Fonctionnalités

Les fonctionnalités présentes dans le projet sont les suivantes.

### Créer un compte utilisateur

Depuis l’écran d’accueil il est possible de créer un compte utilisateur en cliquant sur le bouton « Créer un compte ».

Text

Description automatically generated

### S’identifier

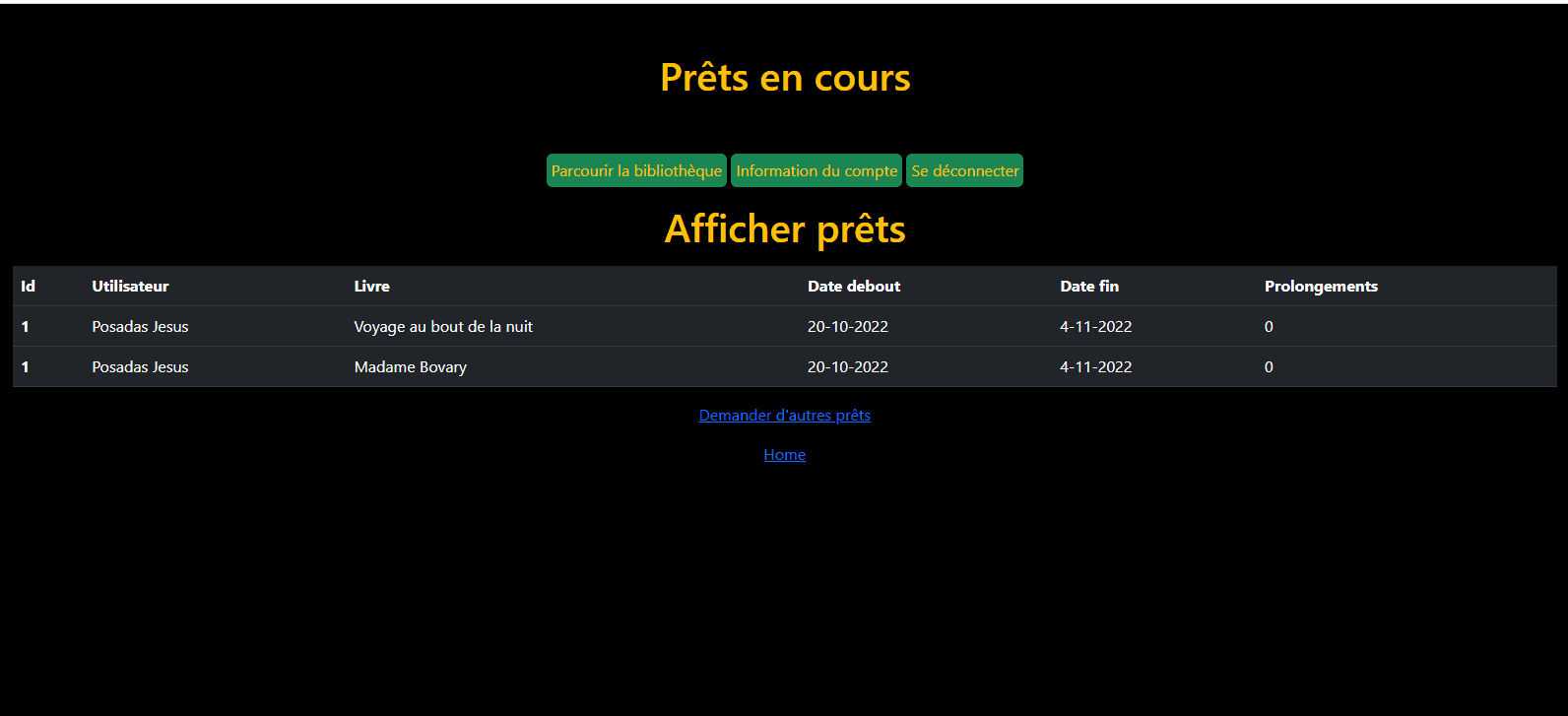
Depuis l’écran d’accueil il est possible de s’identifier en cliquant sur le bouton « Se connecter ».

Text

Description automatically generated

### Afficher / Modifier les informations du compte utilisateur

Depuis l’écran de prêts en cours, l’utilisateur authentifié est capable d’afficher ou bien modifier les informations de son compte en cliquant sur le bouton « Information du compte ».



### Se déconnecter

Depuis l’écran de prêts en cours et l’écran pour se connecter il existe un bouton pour se déconnecter.

A screenshot of a computer screen

Description automatically generated with medium confidence

### Afficher des livres dans la bibliothèque

Depuis l’écran d’accueil en cliquant sur le bouton Parcourir la bibliothèque.

Text

Description automatically generated

### Chercher des livres

Il est possible de chercher des livres par titre depuis l’écran liste de livres.

Text

Description automatically generated

### Demander un livre en prêt

Il faut impérativement que l’utilisateur soit authentifié pour voir apparaitre l’option Demander en prêt. Il suffit de se diriger vers l’option « parcourir la bibliothèque » et la liste des livres va s’afficher avec l’option demander en prêt.

Text

Description automatically generated

### Suivre les prêts en cours

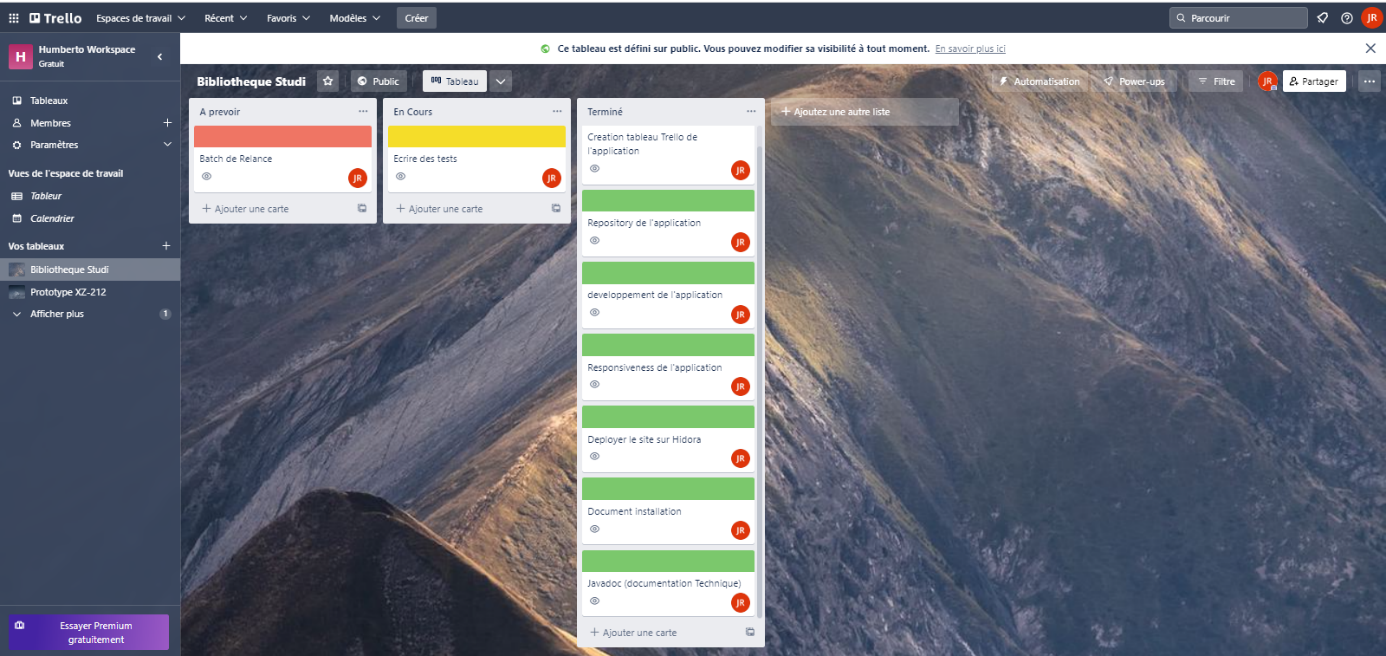
Une fois qu’on a demandé le prêt d’un livre automatiquement l’utilisateur sera dirigé vers l’écran « prêts en cours ».

Text

Description automatically generated

## Tableau Trello – Etapes du projet

Pour décrire les étapes du projet ainsi que son état vous pouvez visiter le lien <https://trello.com/b/RR69Srdb/bibliotheque-studi>



## Version Responsive

L’application propose une adaptation sur tous les écrans pouvant interagir sur toutes les plateformes.

L’application est très légère d’un point de vue visuel, c’est pour cette raison que c’est très simple de la rendre responsive, il suffit d’établir le point de rupture pour le menu supérieur de 3 boutons pour apparaitre en block quand le point de rupture l’indique.

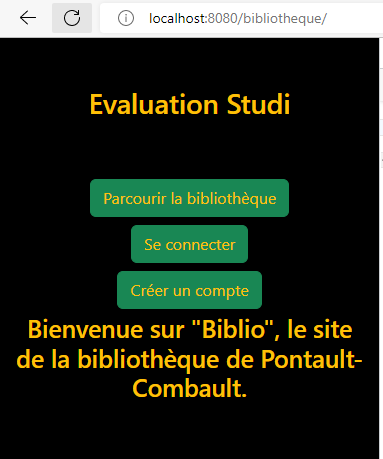
Pour obtenir cette version responsive j’ai utilisé Bootstrap. Je vous présente quelques exemples des écrans en différents tailles pour montrer l’adaptation aux tailles comme vous allez le voir en suite :

### Accueil en taille Large

Graphical user interface, text

Description automatically generated

### Accueil en taille Small



### Connexion en taille Large

Graphical user interface, text, chat or text message

Description automatically generated

### Connexion en Taille Small

Graphical user interface, text, chat or text message

Description automatically generated

### Création de compte en Large

Graphical user interface, text

Description automatically generated

### Création de compte en Small

Graphical user interface, text, application, chat or text message

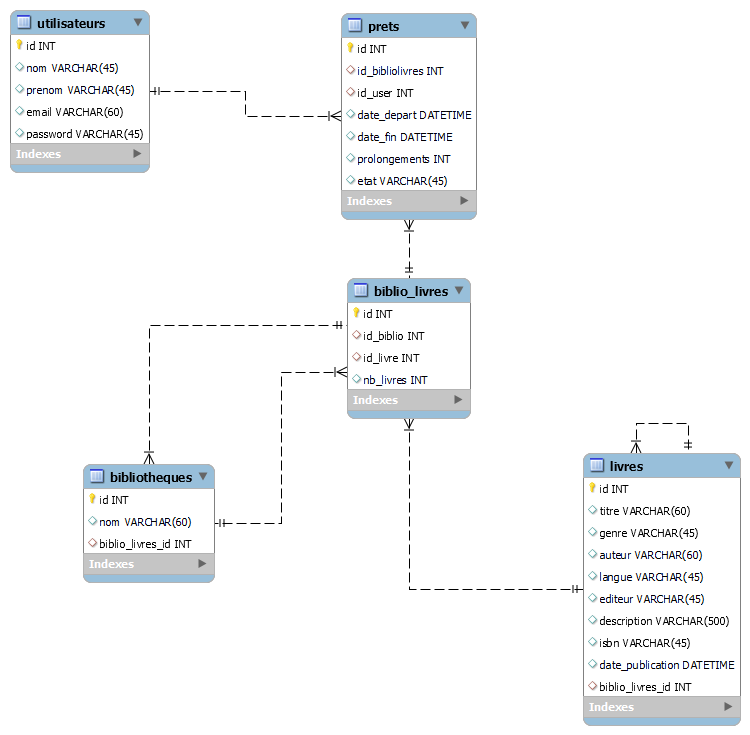
Description automatically generated

## Base de données

L’application fonctionne dans sa dernière version avec une Remote Database a l’aide d’un hébergeur fourni par mon entreprise ou je suis en train d’effectuer mon stage de fin de formation.

J’ai réussi à créer cette base de données a l’aide du Cpanel de l’hébergeur. De cette manière l’utilisateur n’a pas a créer la base de données par lui-même, mais la seule chose qu’il a a faire se d’installer le projet, qui possède déjà les paramètres nécessaires pour accéder a la base de données.

Vous trouverez en suite le diagramme de ma base de données :



## Installation du projet

Pour installer le projet il faudra suivre les étapes suivantes :

### Installer Apache Tomcat 10

Se rendre dans la page <https://tomcat.apache.org/download-10.cgi> et télécharger apache tomcat 10 pour votre Système Opératif.

### Rajouter le Driver JDBC pour mysql

Dans le repository

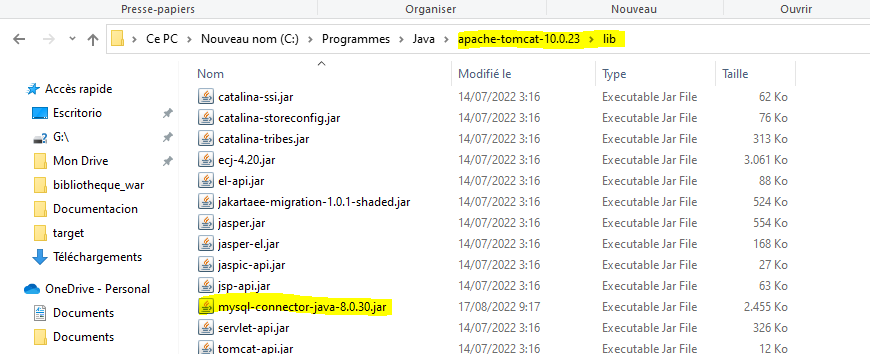
<dependencies>

<dependency>  
 <groupId>jakarta.platform</groupId>  
 <artifactId>jakarta.jakartaee-api</artifactId>  
 <version>9.0.0</version>  
 </dependency>  
 <dependency>  
 <groupId>com.fasterxml.jackson.core</groupId>  
 <artifactId>jackson-databind</artifactId>  
 <version>2.13.3</version>  
 </dependency>  
</dependencies>  
  
<build>  
 <finalName>${project.artifactId}</finalName>  
 <plugins>  
 <plugin>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.1</version>  
 </plugin>  
 <plugin>  
 <groupId>org.apache.tomcat.maven</groupId>  
 <artifactId>tomcat7-maven-plugin</artifactId>  
 <version>2.2</version>  
 <configuration>  
 <url>http://localhost:8080/manager/text</url>  
 <username>admin</username>  
 <password>12345</password>  
 </configuration>  
 </plugin>  
 <plugin>  
 <artifactId>maven-war-plugin</artifactId>  
 <version>3.2.2</version>  
 <configuration>  
 <failOnMissingWebXml>false</failOnMissingWebXml>  
 </configuration>  
 </plugin>  
 </plugins>

Ensuite il faut créer la base de données mysql en utilisant le fichier du script que j’ai mis sur le repository github dans le repository suivant :

<https://github.com/PosadasHumberto/BibliothequeMaven.git>

A l’intérieur du dossier Documentation il faut télécharger le fichier « [mysql-connector-java-8.0.30.jar](https://github.com/PosadasHumberto/BibliothequeMaven/blob/main/Documentacion/mysql-connector-java-8.0.30.jar) » et en suite se diriger dans le dossier lib de notre installation du Tomcat, en suite il faudra déposer le driver a l’intérieur pour que notre serveur Apache Tomcat soit capable d’établir la connection vers la base de données.



### Ouvrir le projet avec Intellij IDEA

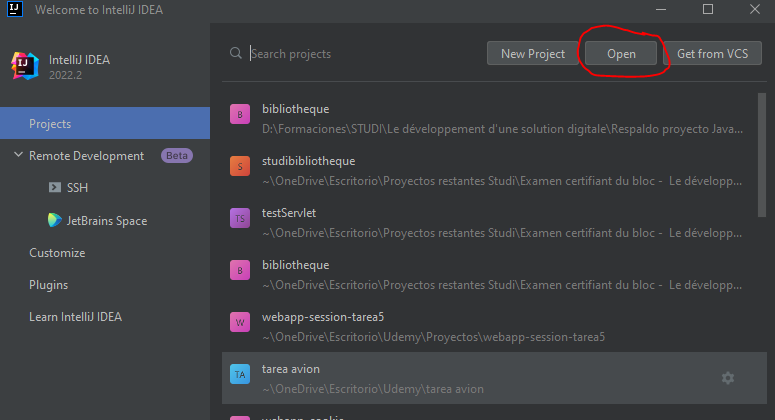
Dans le même repository Github se trouve le projet compressé nommé bibliotheque.rar, il faudra le télécharger pour ensuite le décompresser à l’emplacement souhaité.

Graphical user interface, text, application, email

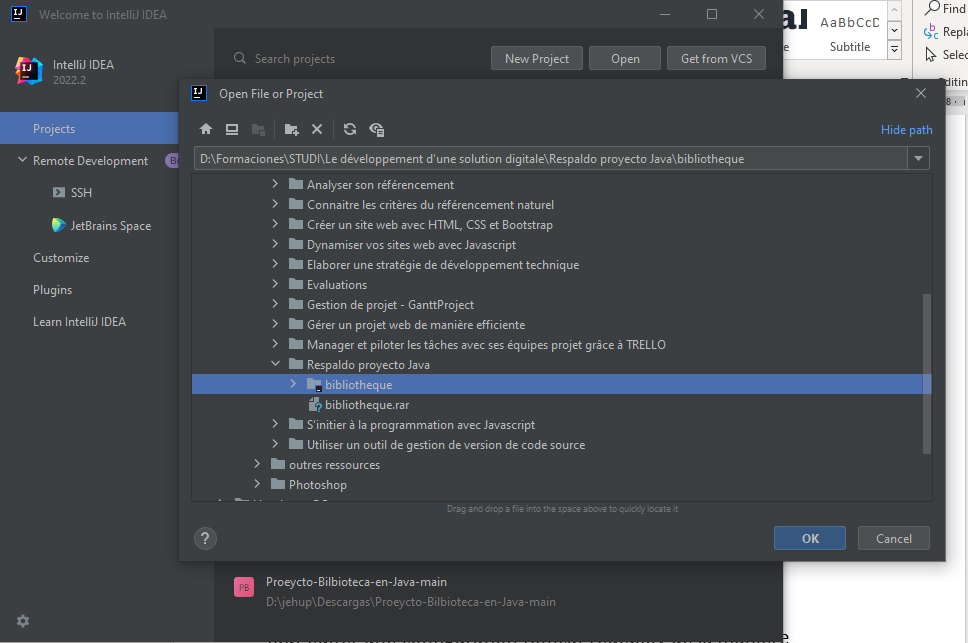
Description automatically generated

Il faudra en suite ouvrir ce projet avec votre IDE Intellij IDEA de la façon suivante :

Open :

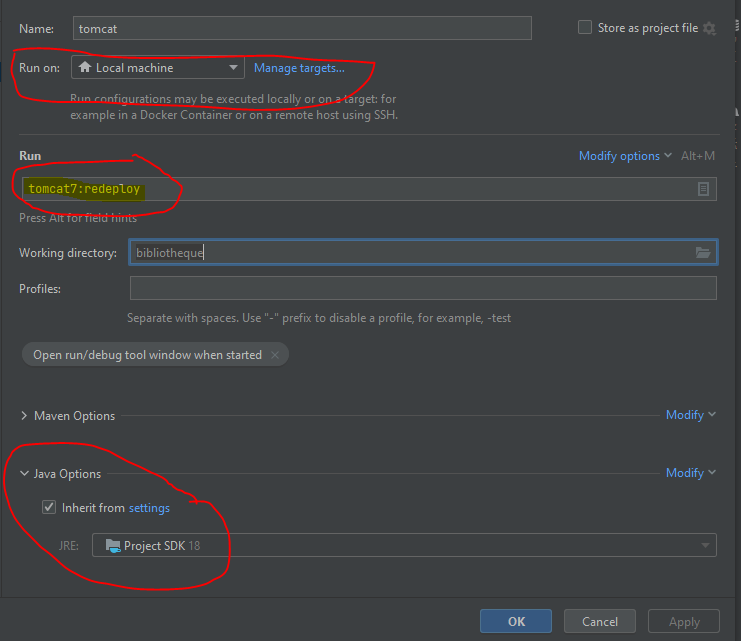


Se rendre sur l’emplacement du projet :



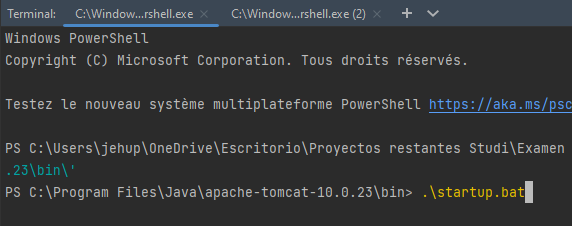
### Editer la configuration pour le déploiement en local

La configuration du run dans l’ide doit se faire de la façon suivante :



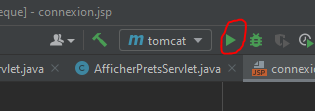
### Démarrer le serveur Apache

A l’aide de la terminal de votre IDE ou bien du cmd.exe ou powershell il faut se rendre a l’emplacement de notre dossier de notre Apache tomcat 10 et se positionner dans le dossier /bin. Il faudra en suite lancer la commande /startup.bat, cette commande va démarrer notre serveur. Le projet est désormais apte pour son exécution.

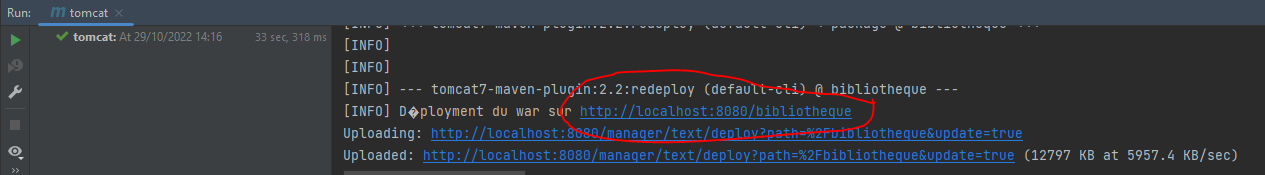


### Lancer l’exécution de l’application

Il est désormais possible d’exécuter notre application, pour cela il faut faire run sur la configuration que nous avons effectué précédemment de la façon suivante :



Il suffit d’attendre quelques secondes pour obtenir le lien pour pouvoir utiliser l’application sur votre navigateur préféré.



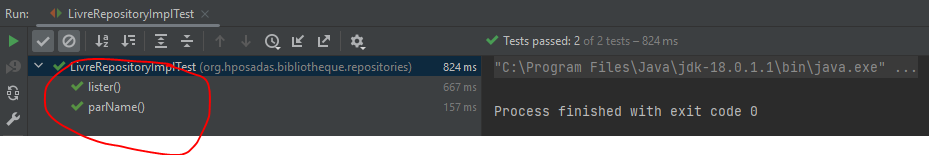
Text

Description automatically generated with low confidence

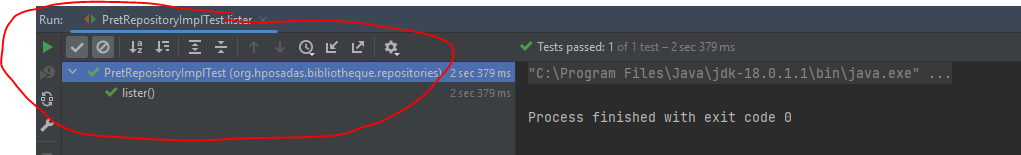
## Tests

Pour l’instant j’ai effectué les tests unitaires suivants avec JUnit

import java.sql.SQLException;  
import java.util.List;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
class LivreRepositoryImplTest {  
  
 Connection conn = ConnectionDB.*getConnection*();  
  
 LivreRepositoryImplTest() throws SQLException {  
 }  
  
 @Test  
 void lister() {  
 LivreService service = new LivreServiceImpl(conn);  
 List<Livre> livres = service.lister();  
 assertNotNull(livres);  
 }  
  
 @Test  
 void parName() {  
 LivreService service = new LivreServiceImpl(conn);  
 List<Livre> livre = service.byName("Humberto");  
 assertNotNull(livre);  
 }  
}



package org.hposadas.bibliotheque.repositories;  
  
import org.hposadas.bibliotheque.models.Livre;  
import org.hposadas.bibliotheque.utilities.ConnectionDB;  
import org.junit.jupiter.api.Test;  
  
import java.sql.Connection;  
import java.sql.SQLException;  
import java.util.List;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
class PretRepositoryImplTest {  
 Connection conn = ConnectionDB.*getConnection*();  
  
 PretRepositoryImplTest() throws SQLException {  
 }  
  
 @Test  
 void lister() throws SQLException {  
 Repository repository = new PretRepositoryImpl(conn);  
 List<Livre> livres = repository.lister();  
 assertNotNull(livres);  
 }  
}



Ces tests permettent de s’assurer que la ou les requêtes SQL nous rendent des résultats, c’est-à-dire que lorsque j’essaye d’accéder a l’information de la base de données, plus particulièrement aux tables utilisateurs et livres il y a un retour d’information.

## Vidéo explicatif du projet

J’ai effectué un vidéo ou j’explique au détail les étapes pour l’élaboration de ce projet ainsi que le tutoriel d’utilisation a l’adresse suivante :

### 1ère version du projet

<https://drive.google.com/file/d/1-XWw3_f3PN7-hldRXefv6m0blOU5dc8i/view?usp=sharing>

### Version finale du projet

<https://drive.google.com/file/d/1Tnd5zZmXa-pCOSJ8Fl0fVq30HGnFgyKS/view?usp=sharing>