Rapport : Développement d’une application web esalaf

Encadrée par:

Monsieur El Mokhtar EN-NAIMI

monsieur Lotfi El Aachak

Realiser par :

Sanae Ben Hammadi(grp 1)

Introduction

Dans ce tutoriel, vous apprendrez à créer, exécuter et empaquetter une simple application Java qui imprime "Hello, World !" sur la sortie du système. En cours de route, vous vous familiariserez avec les fonctionnalités d'IntelliJ IDEA pour stimuler votre productivité en tant que développeur : assistance au codage et outils supplémentaires.

Préparation du projet

Création d'un nouveau projet JAVA

Dans IntelliJ IDEA, un projet vous aide à organiser votre code source, les tests, les bibliothèques que vous utilisez, les instructions de construction du projet et vos paramètres personnels en une seule unité.

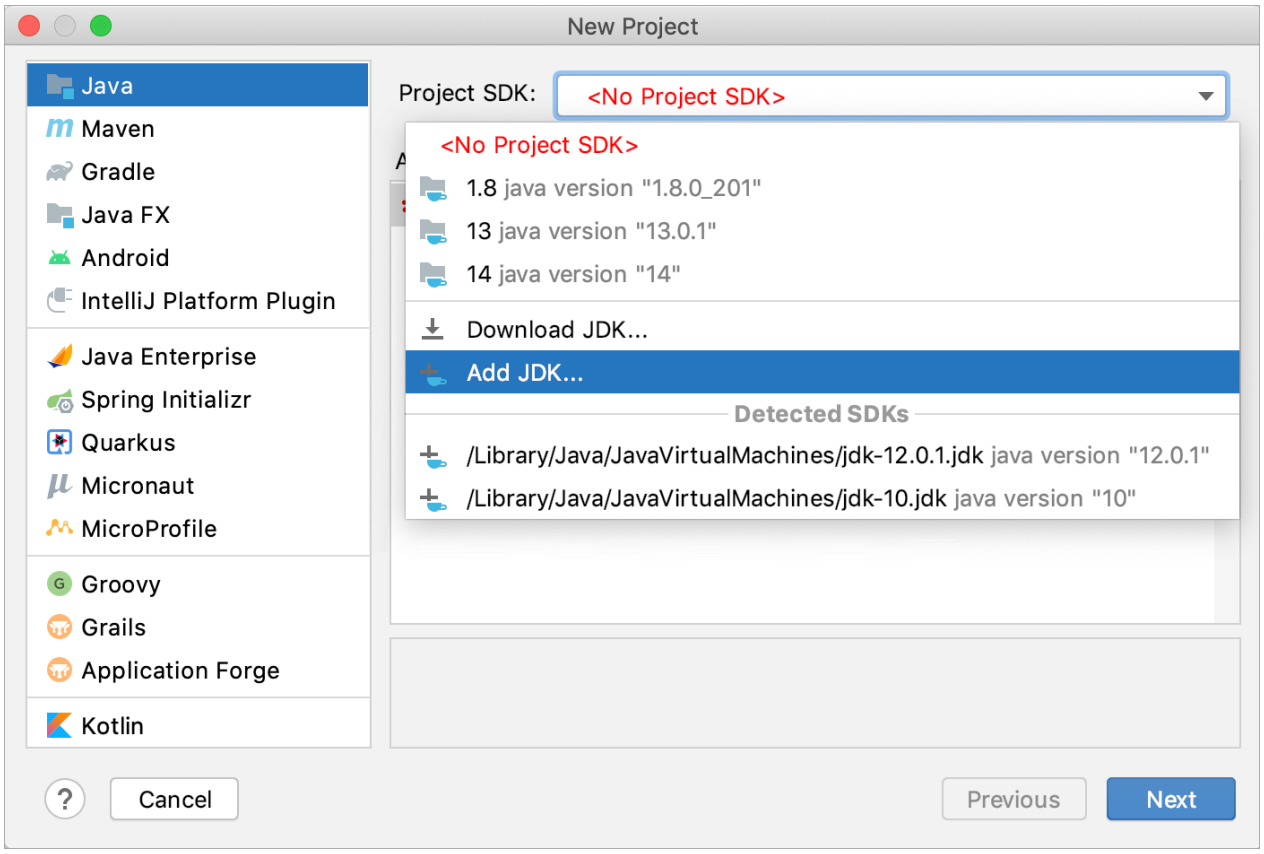
1. Lancer IntelliJ IDEA. Si l'écran de bienvenue s'ouvre, cliquez sur *New Project*.Sinon, dans le menu principal, sélectionnez *File* | *New* | *Project*.

2. Dans l'assistant *New Project*, sélectionnez **Java** dans la liste de gauche.

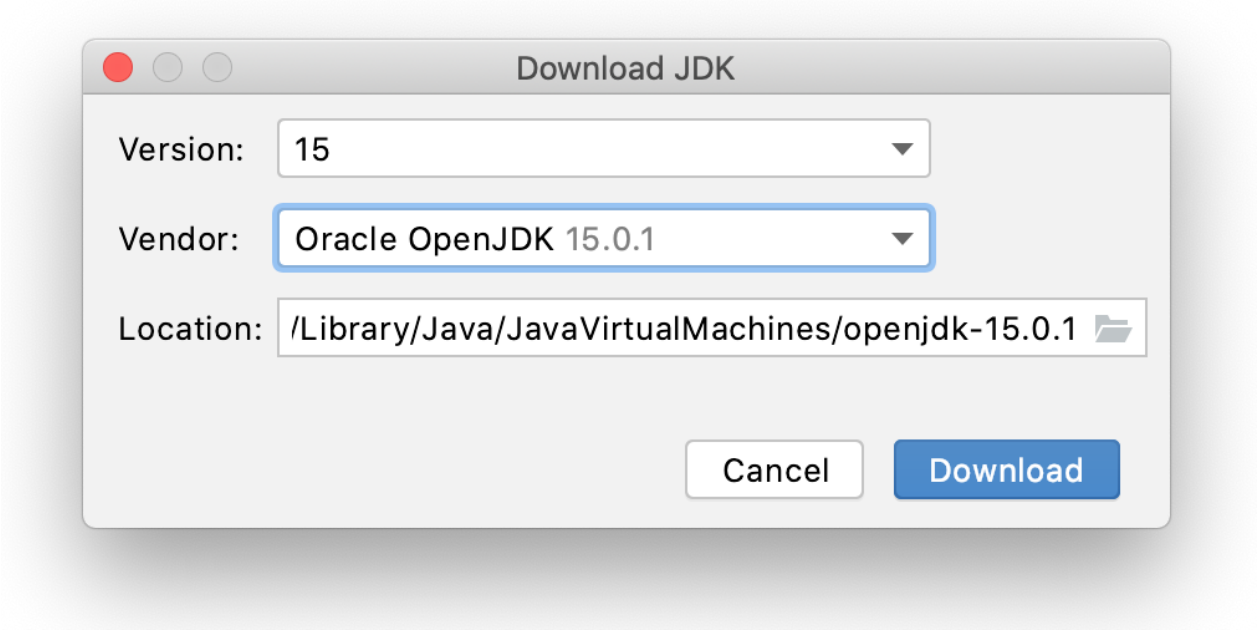
3. Pour développer des applications Java dans IntelliJ IDEA, vous avez besoin du **Java SDK (JDK)**.

Si le JDK nécessaire est déjà défini dans IntelliJ IDEA, sélectionnez-le dans la liste **Project SDK**.

Si le JDK est installé sur votre ordinateur, mais n'est pas défini dans l'IDE, sélectionnez *Add JDK* et indiquez le chemin d'accès au répertoire d'origine du JDK (par exemple, **/Library/Java/JavaVirtualMachines/jdk-13.0.1.jdk**).



Si vous n'avez pas le JDK nécessaire sur votre ordinateur, sélectionnez *Download JDK*. Dans la boîte de dialogue suivante, indiquez le fournisseur du JDK (par exemple, **OpenJDK**), la version, modifiez le chemin d'installation si nécessaire, et cliquez sur *Download*.

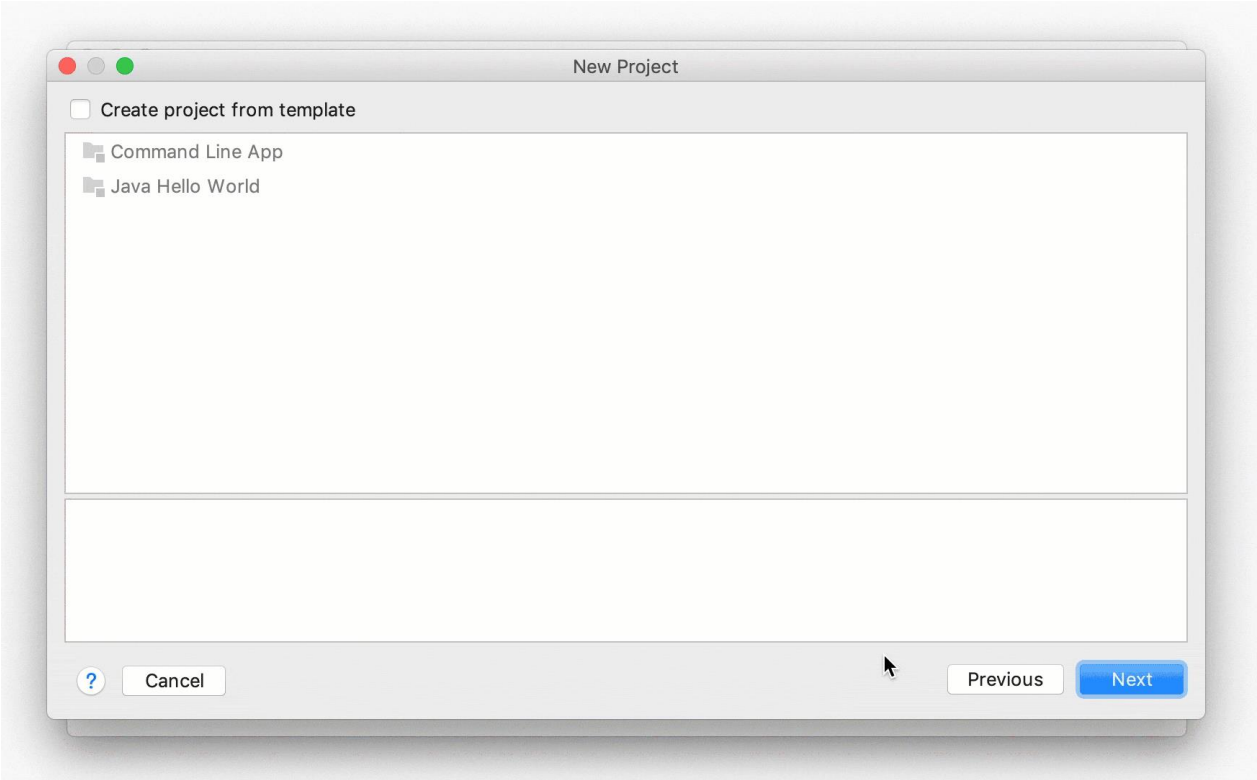


3. Nous n'allons pas utiliser de bibliothèques ou de frameworks supplémentaires pour ce tutoriel, alors cliquez sur *Next*.

5. Ne créez pas de projet à partir d'un modèle. Dans ce tutoriel, nous allons tout faire à partir de zéro, alors cliquez sur *Next*.

6. Donnez un nom au projet, par exemple : **HelloWorld**.

Si nécessaire, modifiez l'emplacement par défaut du projet et cliquez sur **Terminer**.



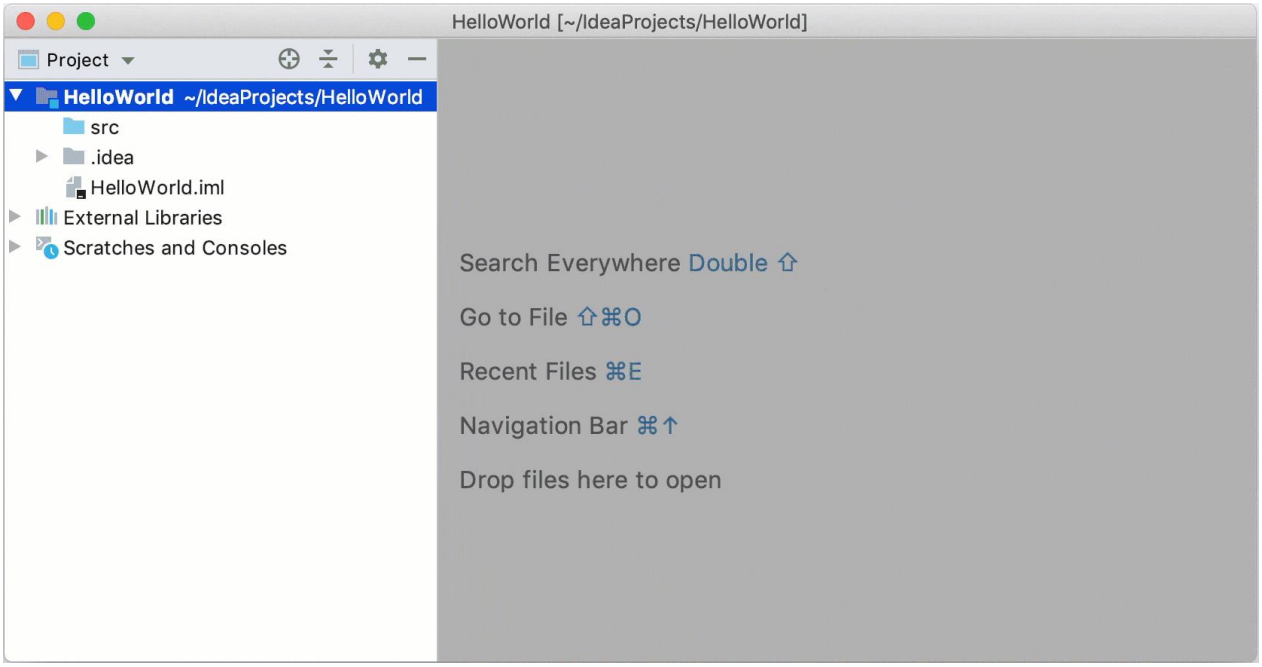
Créer un package et une classe

Les ***packages*** sont utilisés pour regrouper des classes qui appartiennent à la même catégorie ou qui offrent des fonctionnalités similaires, pour structurer et organiser de grandes applications comportant des centaines de classes.

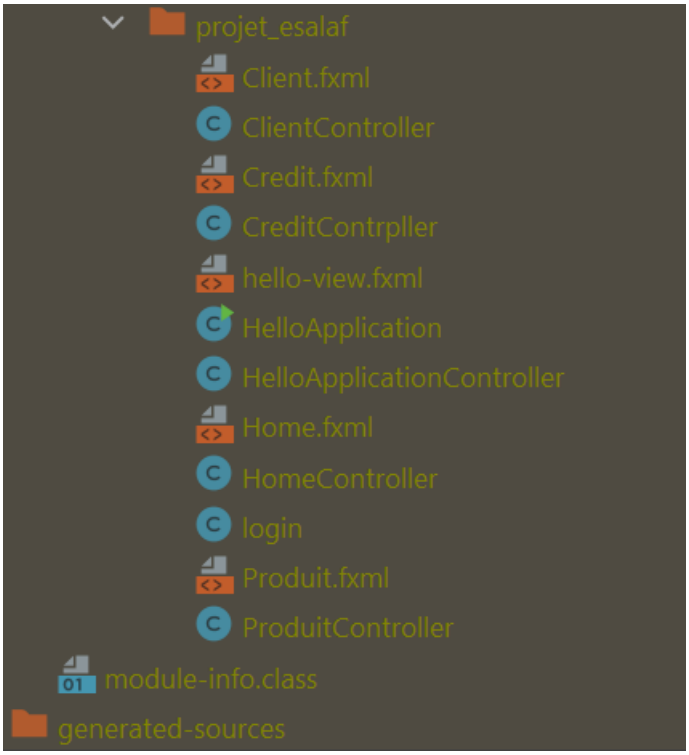
Dans l'arborescence du projet, sélectionnez le dossier **src**, appuyez sur ***Alt+Insert*** et sélectionnez la classe Java.

Dans le champ Name, tapez **com.example.helloworld.HelloWorld** et cliquez sur OK.

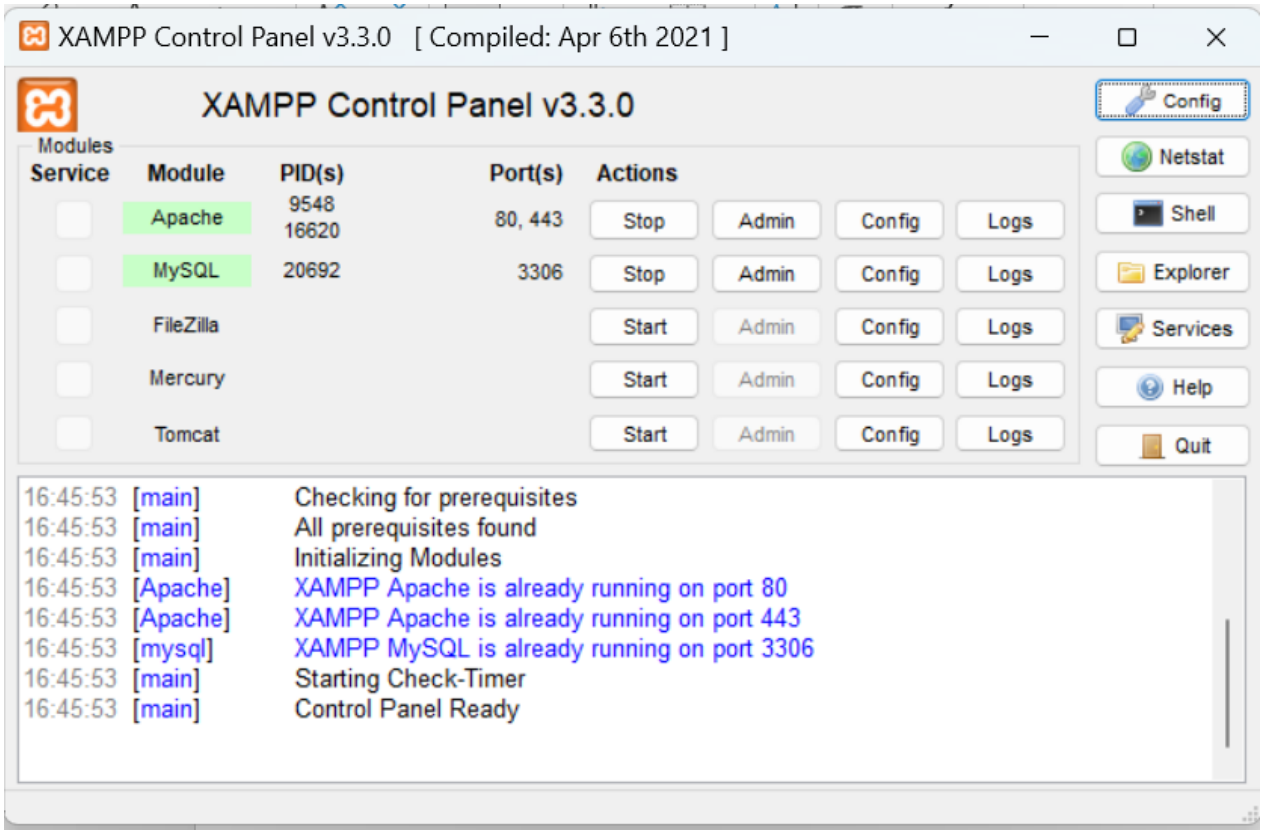
IntelliJ IDEA crée le package **com.example.helloworld** et la classe **HelloWorld**



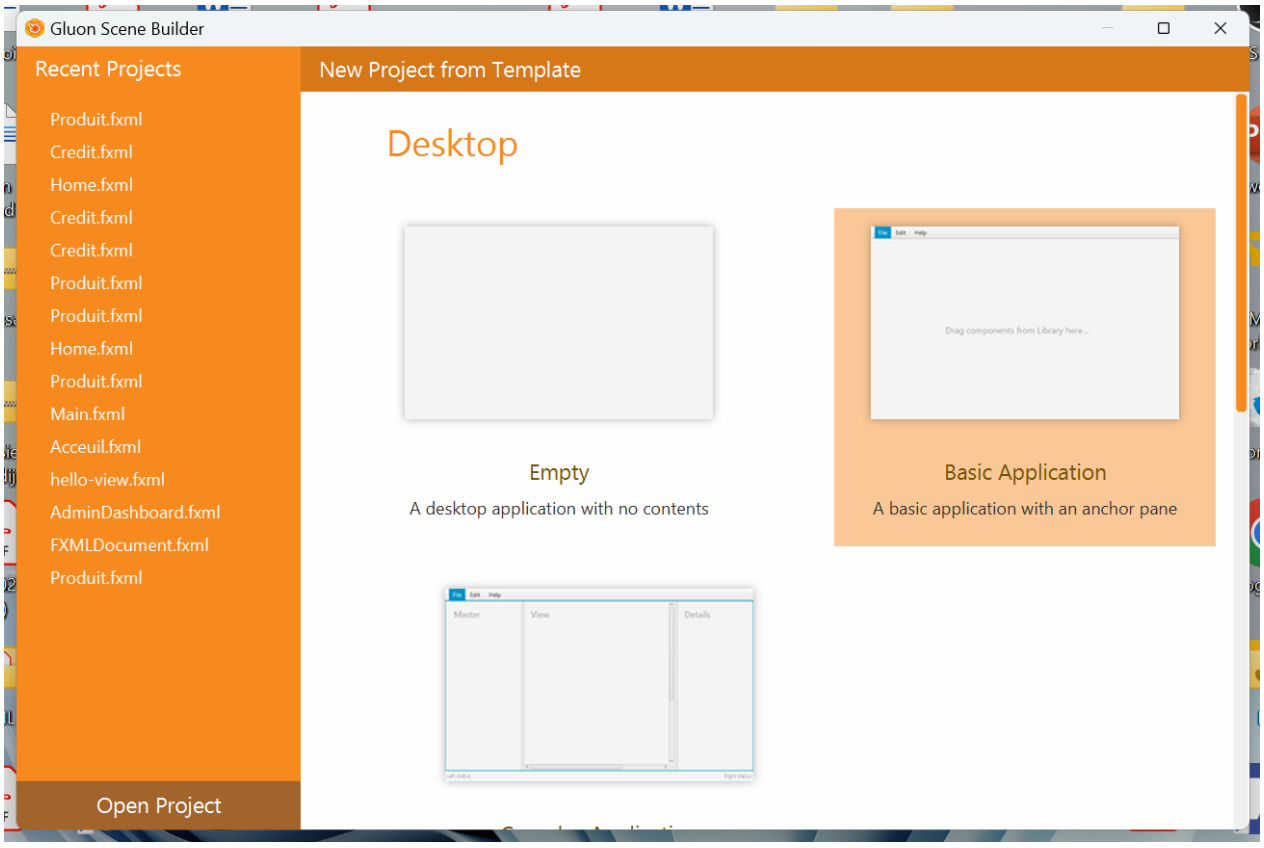
Et voici mes classes et mes interfaces :



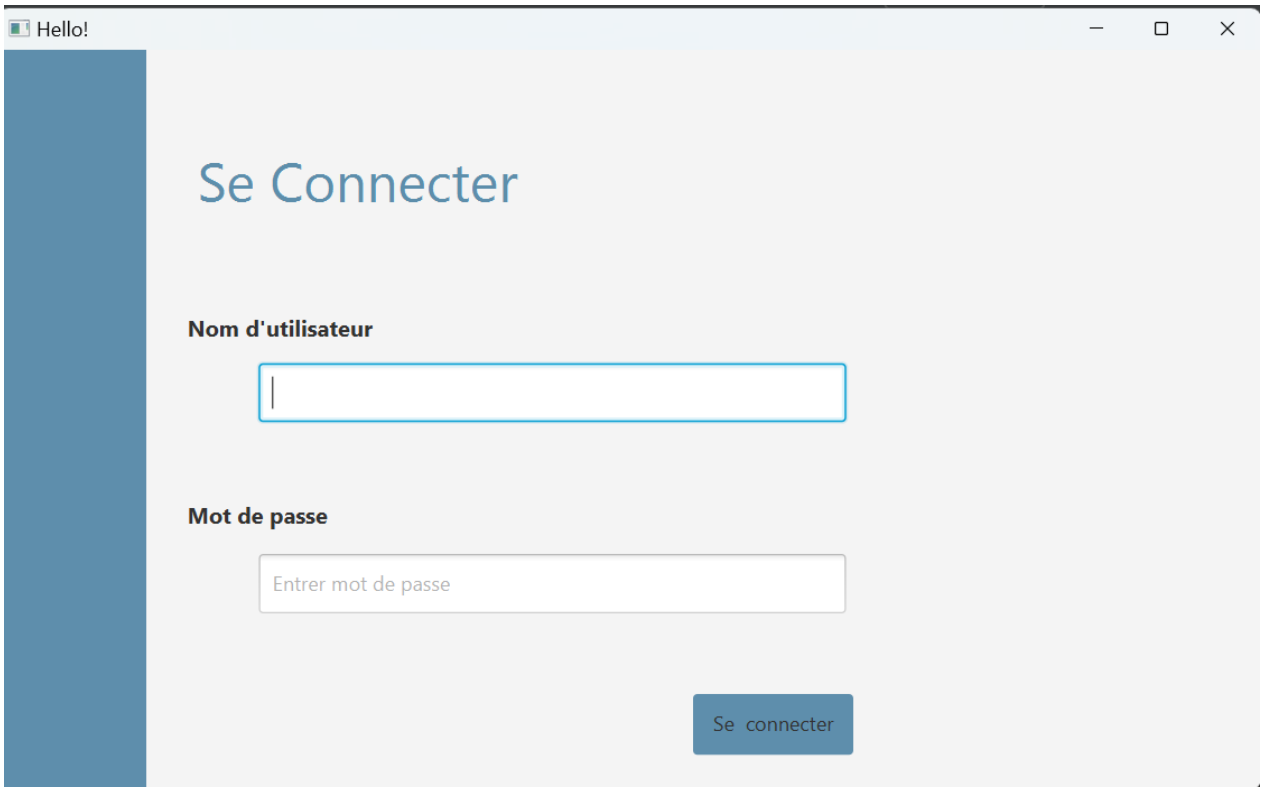
Et J'ai utiliser XAMPP pour la base de donnees :



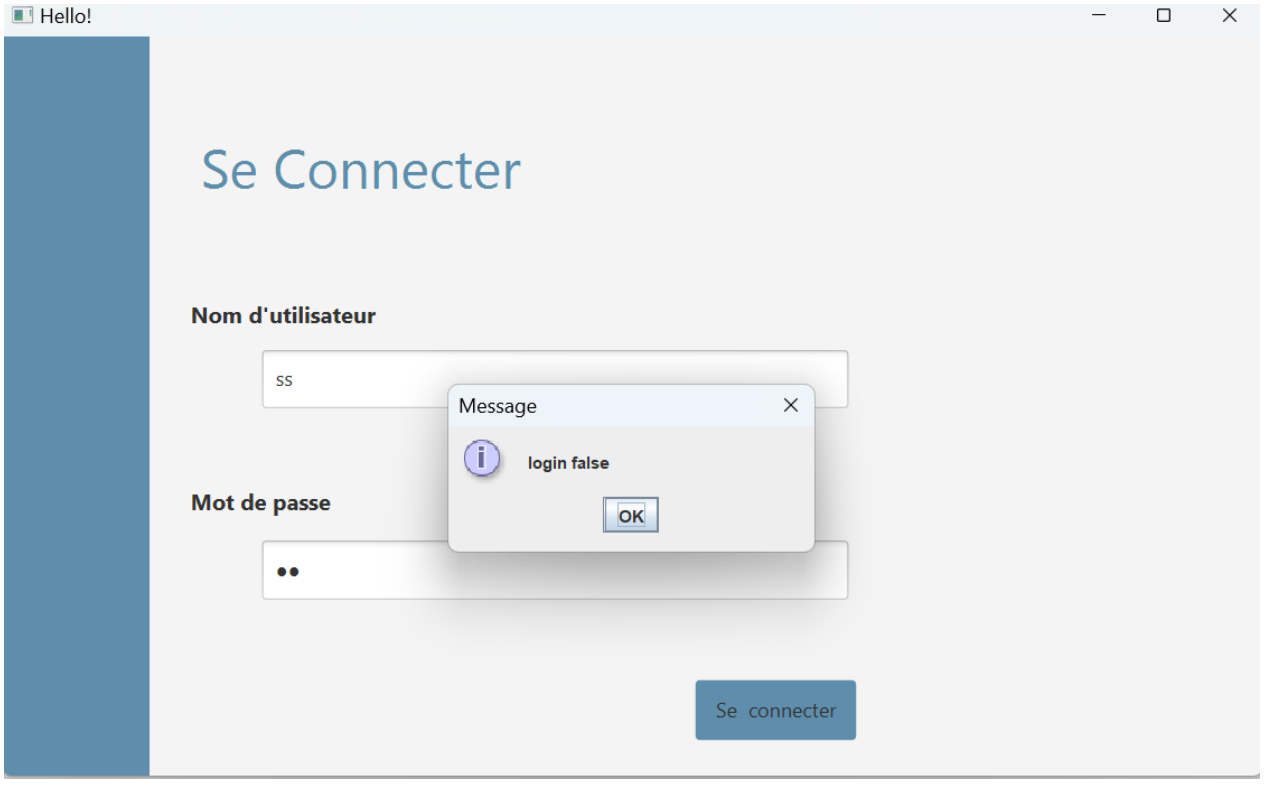
Et l’application SCENE BUILDER pour la creation des interfaces :



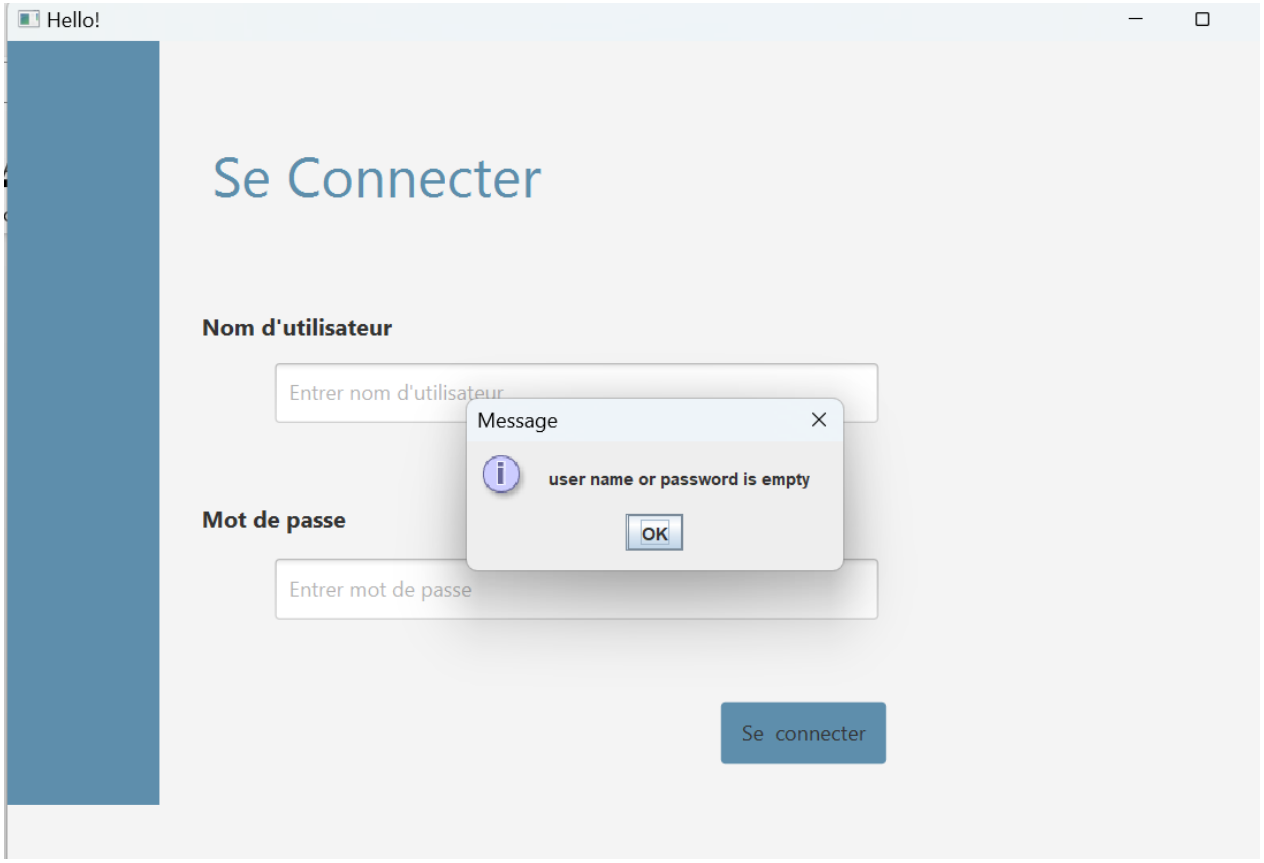
J’ai créer l’interface hello-view.fxml pour se connecter pour les employée :



Et si on a ecrit un nom d'utilisateur ou un mot de passe invalide :



Si on a lisser le texte vide on nous affiche cette message :



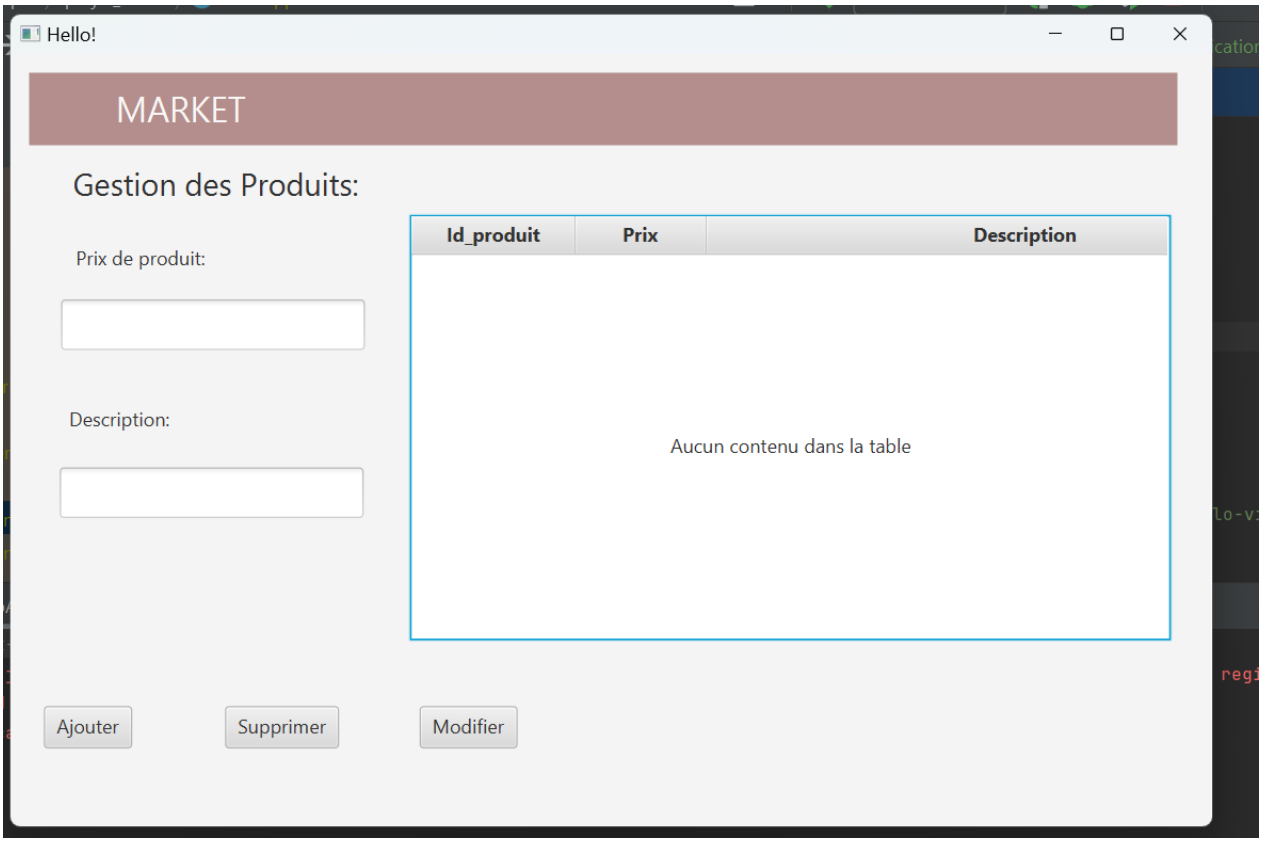
J’ai utiliser la fonction login() pour la connectivite :

```
    void login(ActionEvent event) {
        String uname = txtuname.getText();
        String pass = txtpass.getText();
        if (uname.equals("") && pass.equals("")) {


            JOptionPane.showMessageDialog( parentComponent: null,  message: "user name or password is empty");

        } else {
            try{

                Class.forName( className: "com.mysql.jdbc.Driver");
                con = DriverManager.getConnection( url: "jdbc:mysql://127.0.0.1:3306/esalaf",  user: "root",  password: "");
                pst = con.prepareStatement( sql: "select * from users where email=? and password=?");
                pst.setString( parameterIndex: 1, uname);
                pst.setString( parameterIndex: 2, pass);

                rs = pst.executeQuery();
                if (rs.next()) {

    FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource( name: "Home.fxml"));
    Parent root = (Parent) fxmlLoader.load();


Stage stage = (Stage) btnok.getScene().getWindow();
```

```
Stage stage = (Stage) btnok.getScene().getWindow();
stage.setScene(new Scene(root,  v: 901,  v1: 579));

                } else {
                 JOptionPane.showMessageDialog( parentComponent: null,  message: "login false");
                 txtuname.setText("");
                 txtpass.setText("");
                 txtuname.requestFocus();
                }
            } catch (ClassNotFoundException | SQLException ex) {
                throw new RuntimeException(ex);
            } catch (IOException e) {
                throw new RuntimeException(e);
            }


        }
    }

    new *
    @Override
    public void initialize(URL url, ResourceBundle resourceBundle) {
```

Et si on a ecrit des donnees exist dans la base de donnes(tableau users) on m’affiche la 2em interface de menu (Home.fxml)

Si on a choisit l’interface PRODUIT :

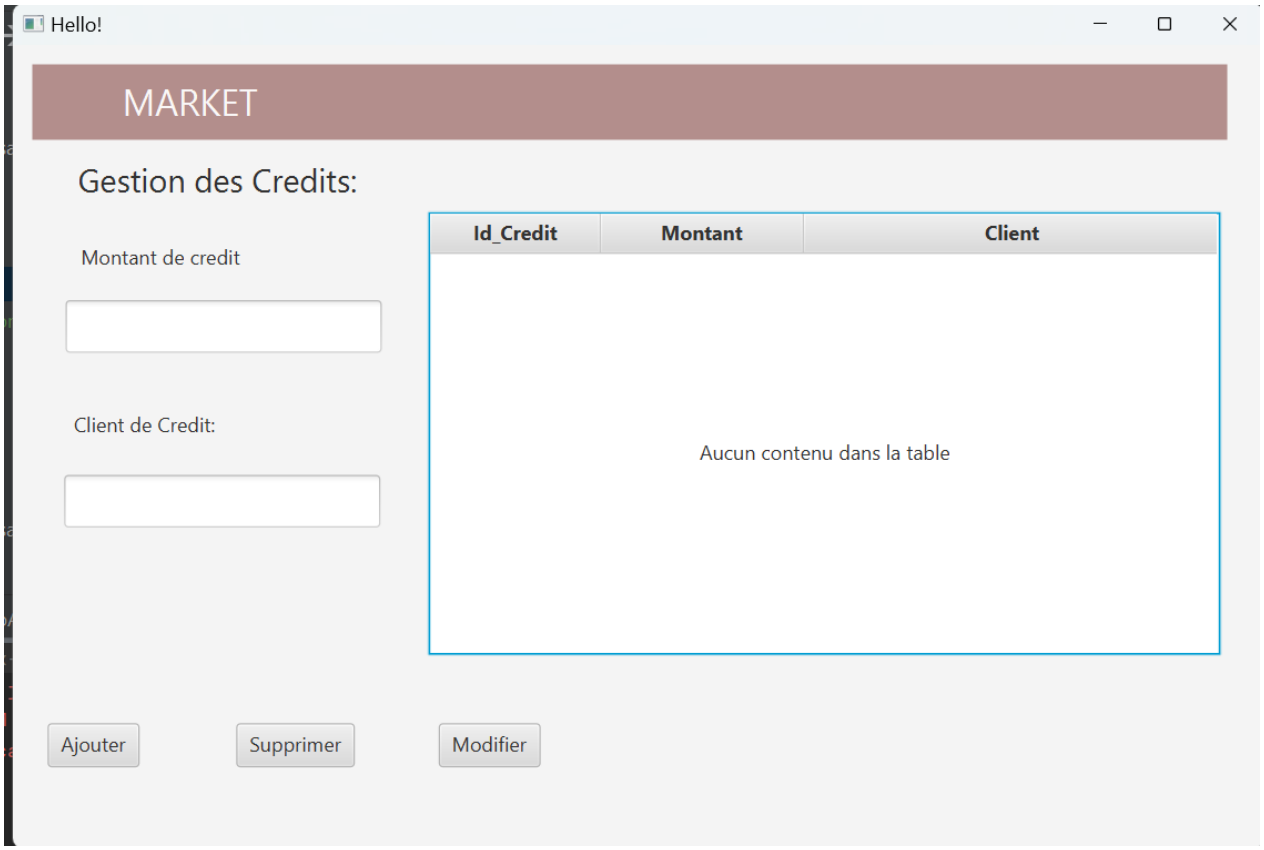


L’accer au l’interface se fait par la fonction produit() dans le class HomeController :

```
2 usages
@FXML
private Button produit;


1 usage   new *
@FXML
void produit(ActionEvent event) {
    try {
        FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource( name: "Produit.fxml"));
        Parent root = (Parent) fxmlLoader.load();
        Stage stage = (Stage) produit.getScene().getWindow();
        stage.setScene(new Scene(root,  v: 901,  v1: 579));


    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}
```

J’ai fait les buttons CRUD avec ses fonctions

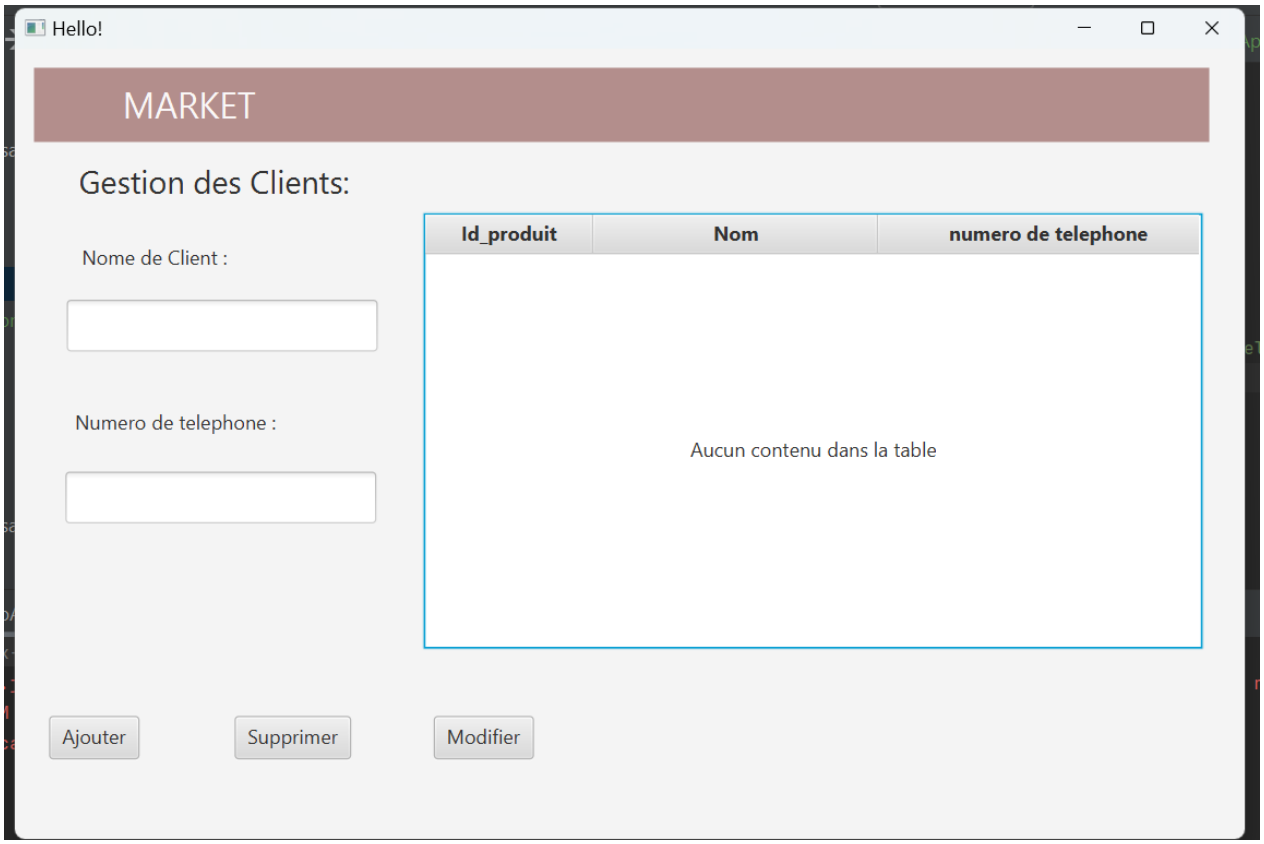
Pour linterface Credit.fxml :



Pour l’accer j’ai fait la fonction credit() :

```
1 usage   new *
@FXML
void credit(ActionEvent event) {
    try {
        FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource( name: "Credit.fxml"));
        Parent root = (Parent) fxmlLoader.load();
        Stage stage = (Stage) credit.getScene().getWindow();
        stage.setScene(new Scene(root, v: 901, v1: 579));

    }
    catch (IOException e) {
        throw new RuntimeException(e);
    }
}
```

Pour linterface Client.fxml avec les buttons CRUD:



Et la fonction Client() :

```
        @FXML
        void client(ActionEvent event){
            try {
                FXMLLoader fxmlLoader = new FXMLLoader(getClass().getResource( name: "Client.fxml"));
                Parent root = (Parent) fxmlLoader.load();
                Stage stage = (Stage) client.getScene().getWindow();
                stage.setScene(new Scene(root,  v: 901,  v1: 579));

            }
catch (IOException e) {
                    throw new RuntimeException(e);
                }
        }
```

Et merci

fin