# Build and Compile by Eclipse Maven

<http://roufid.com/no-compiler-is-provided-in-this-environment/>

**3 ways to solve the Maven error : No compiler is provided in this environment.**

## **Problem**

You may face the below Maven error while running a Maven build :

**No compiler is provided in this environment. Perhaps you are running on a JRE rather than a JDK?**

## **Solution**

Your **JAVA\_HOME** environment variable or your Eclipse IDE are pointing to a JRE rather than a JDK. You must set a correct compiler in your environment and below 3 ways to perform it:

1. [Refer the JAVA\_HOME to a JDK](http://roufid.com/no-compiler-is-provided-in-this-environment/#java_home)
2. [Maven on Eclipse : The correct configuration](http://roufid.com/no-compiler-is-provided-in-this-environment/#maven_eclipse)
3. [Fix the problem in the pom.xml](http://roufid.com/no-compiler-is-provided-in-this-environment/#pom_xml)

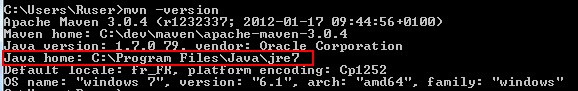
### **1- Refer the JAVA\_HOME to a JDK**

If you are running Maven on command line, it is very likely that your Maven environment is not configured correctly. In fact, Maven relies on the **JAVA\_HOME** environment variable to use the right compiler. **JAVA\_HOME** must refer to a **JDK** (JAVA Development Kit) and not a **JRE** (Java Runtime Environment). [See the difference between a JDK and a JRE](https://www.java.com/en/download/faq/techinfo.xml)

To check what your Maven uses, open a command line and type:

|  |  |
| --- | --- |
| 1 | mvn –version |

**Output:**

[](http://roufid.com/wp-content/uploads/2016/06/1_java_home_2.png)

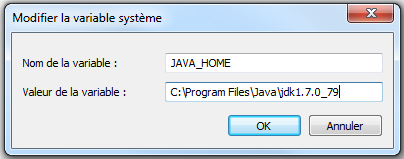
*JAVA\_HOME environment variable*

Verify that **JAVA\_HOME** refers to a JDK home and not a JRE. It refers to a JRE in the previous example.

Below how to update the value of JAVA\_HOME:

**On Windows:**

* Go to ***System properties*** -> ***Advanced system settings*** -> ***Advanced*** -> ***environment variable*** and on the System variables section select the **JAVA\_HOME** variable and click on ***Edit***
* Fill the form with the following  
  Variable name: **JAVA\_HOME**  
  Variable value: **<ABSOLUTE\_PATH\_TO\_JDK\_HOME>**

[](http://roufid.com/wp-content/uploads/2016/06/1_java_home.png)

*Setting the environment variable JAVA\_HOME*

If you are using command line you can set the variable before running the build as following:

|  |  |
| --- | --- |
| 1 | set JAVA\_HOME=<ABSOLUTE\_PATH\_TO\_JDK> |

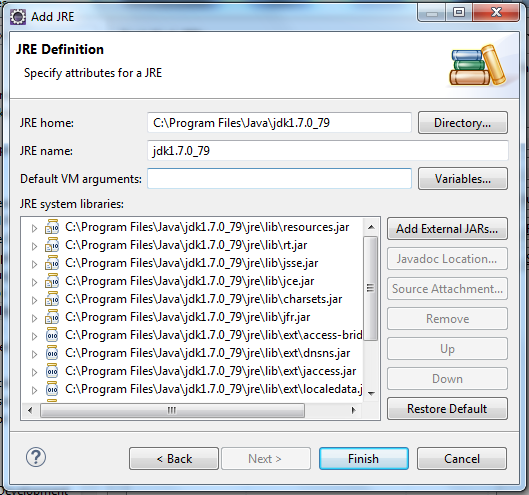
**On Unix:**

|  |  |
| --- | --- |
| 1 | export JAVA\_HOME=”<ABSOLUTE\_PATH\_TO\_JDK>” |

### **2- Running Maven in Eclipse**

If you are running Maven from your IDE you must check that your environment is using a JDK rather than a JRE. Below how to perform the check:

* Open your Eclipse, click on **Windows** -> **Preferences** -> **Java** -> **Installed JREs**
* Verify that the checked JRE refers to a JDK : Select the checked JRE and click **Edit…** and change the path to the JDK home.

[](http://roufid.com/wp-content/uploads/2016/06/2016-06-07-15_30_52-Preferences.png)

*Add JDK to Eclipse IDE*

* Click ***OK***.

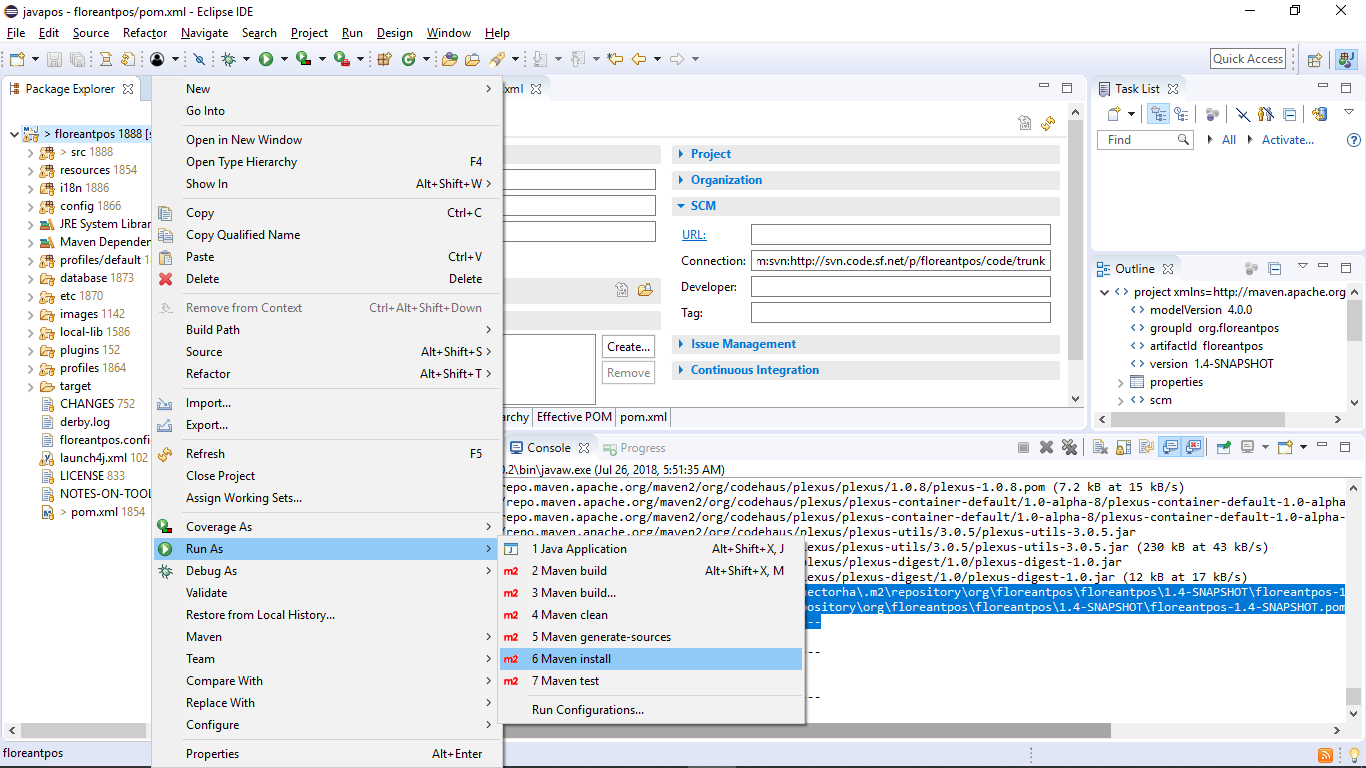
The ***“***No compiler is provided in this environment. Perhaps you are running on a JRE rather than a JDK?***”*** Maven error will disappear.

### **3- Adding the configuration in the pom.xml**

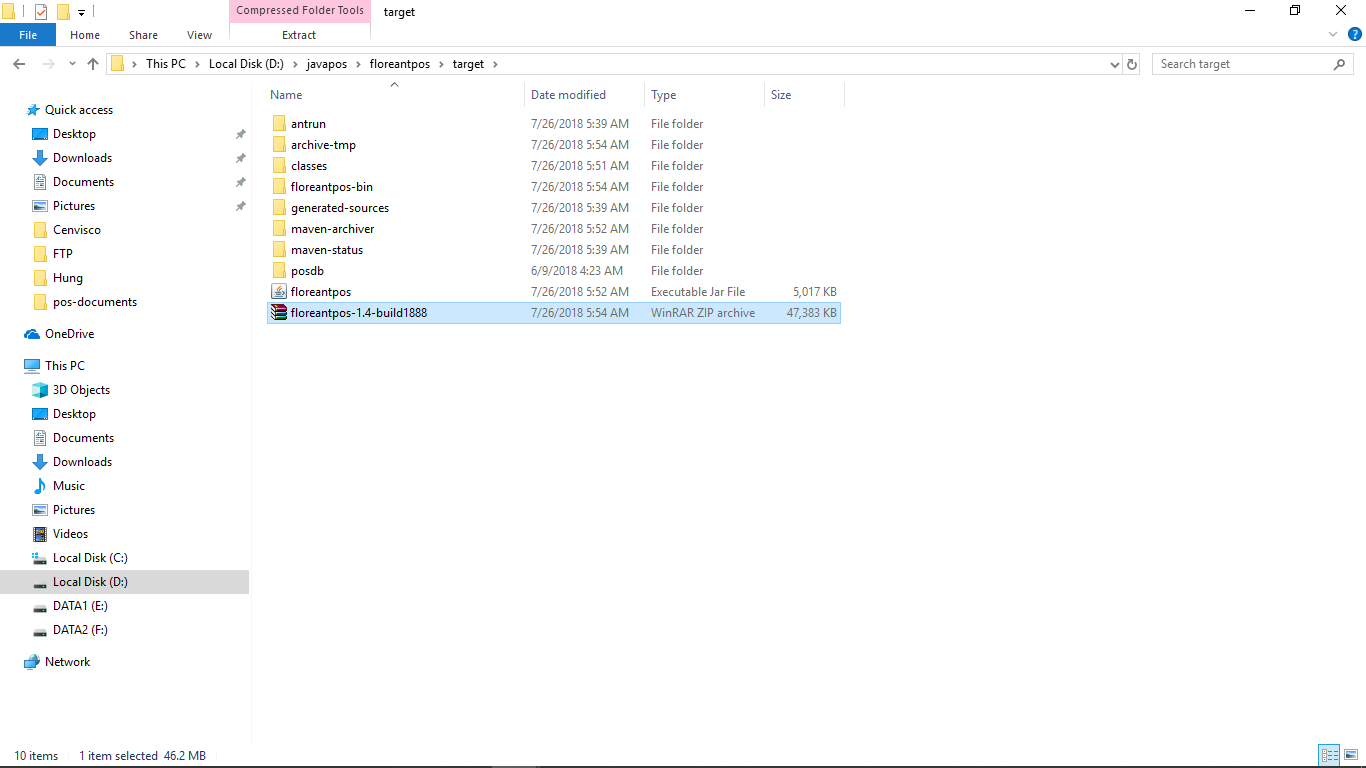
Another solution is to fork the[***maven-compiler-plugin***](https://maven.apache.org/plugins/maven-compiler-plugin/) and set the full path to the correct Java compiler as blow :

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | <project ...>  ...  <build>  ...  <plugins>  <plugin>  <artifactId>maven-compiler-plugin</artifactId>  <version>3.1</version>  <configuration>  <source>1.7</source>  <target>1.7</target>  <fork>true</fork>  <executable>C:\Program Files\Java\jdk1.7.0\_79\bin\javac</executable>  </configuration>  </plugin>  </plugins>  </build>  ...  </project> |

After config Maven with JDK, make build jar file as below:



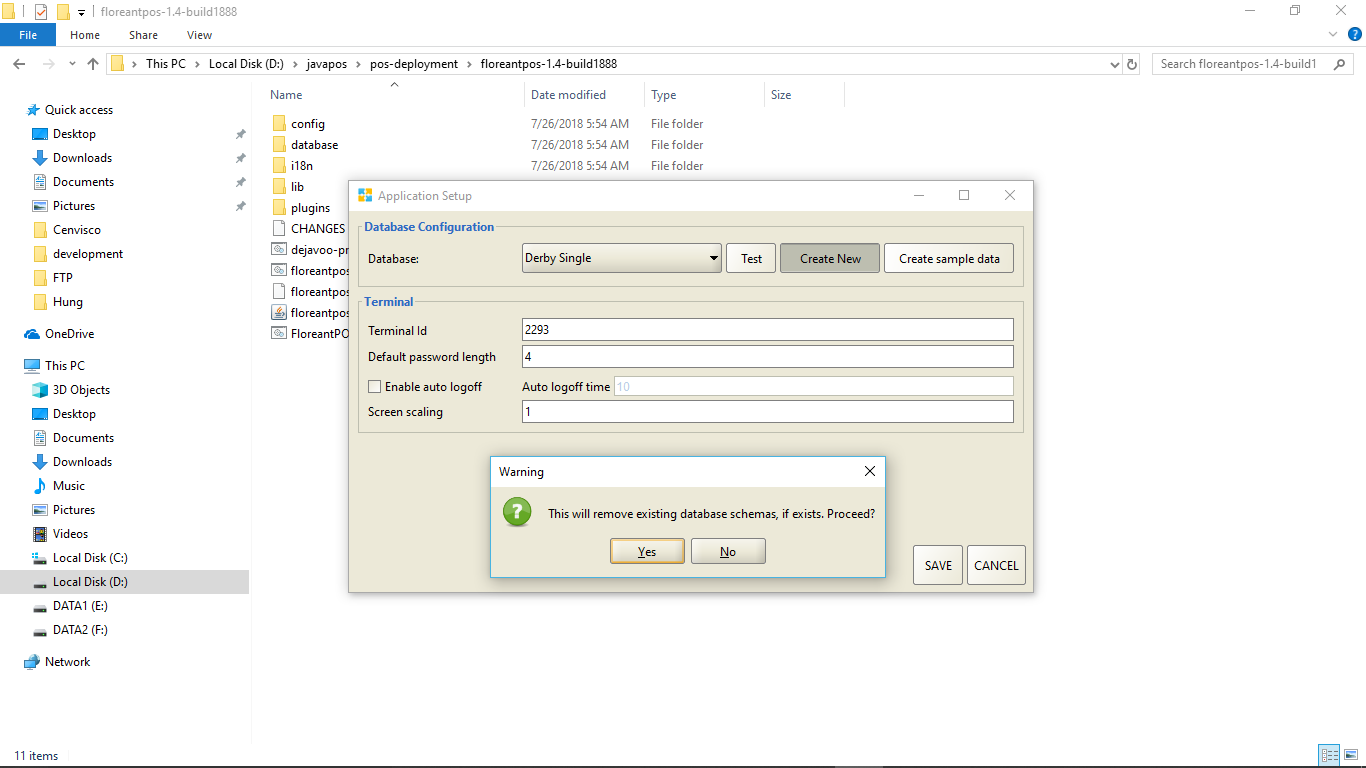
Result will have:

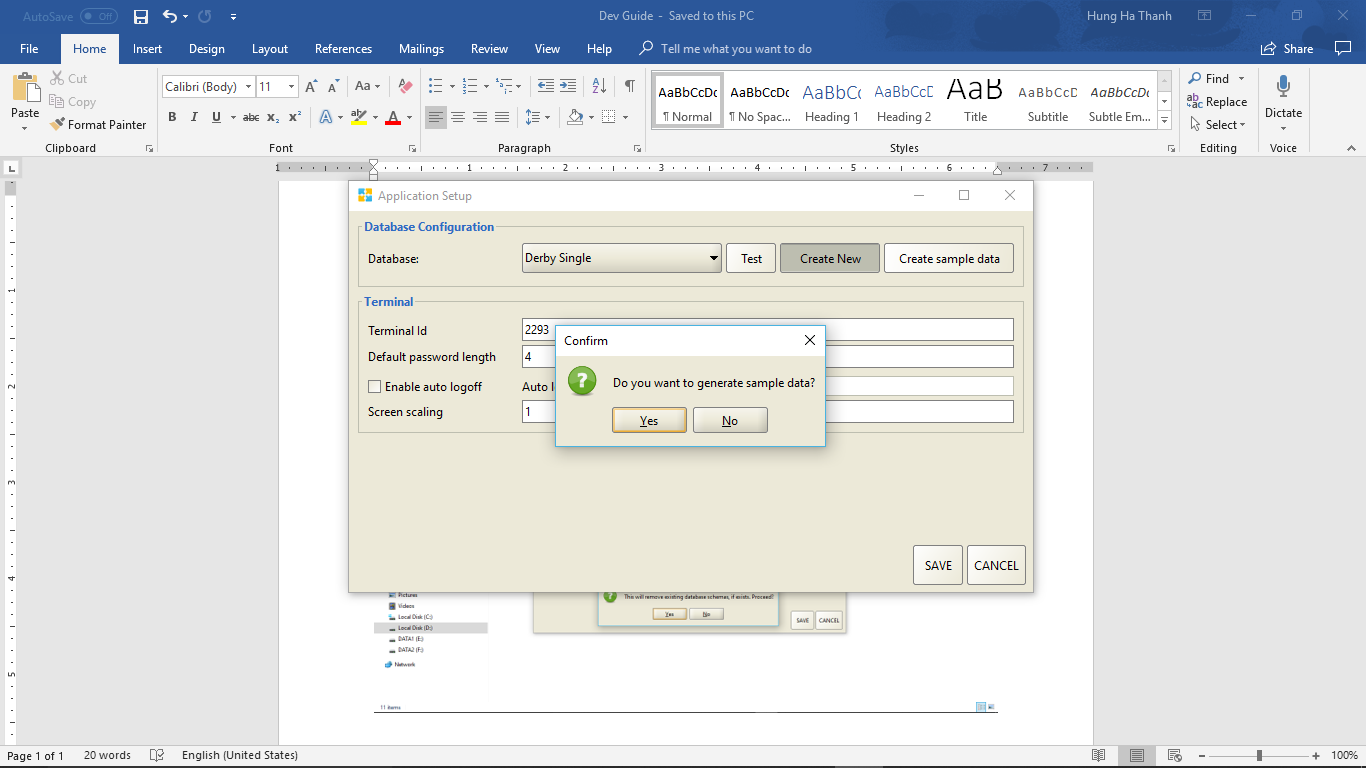


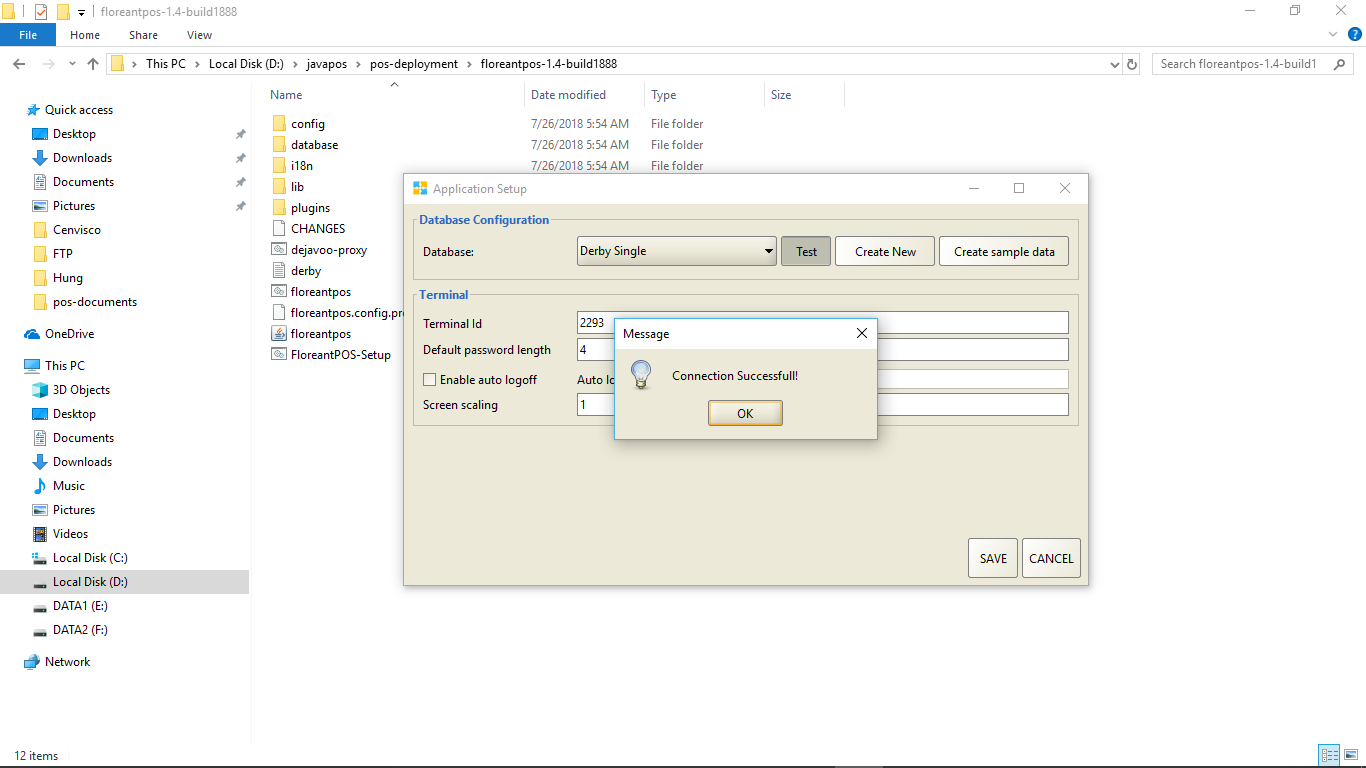
Copy the zip file to release and install on the Computer

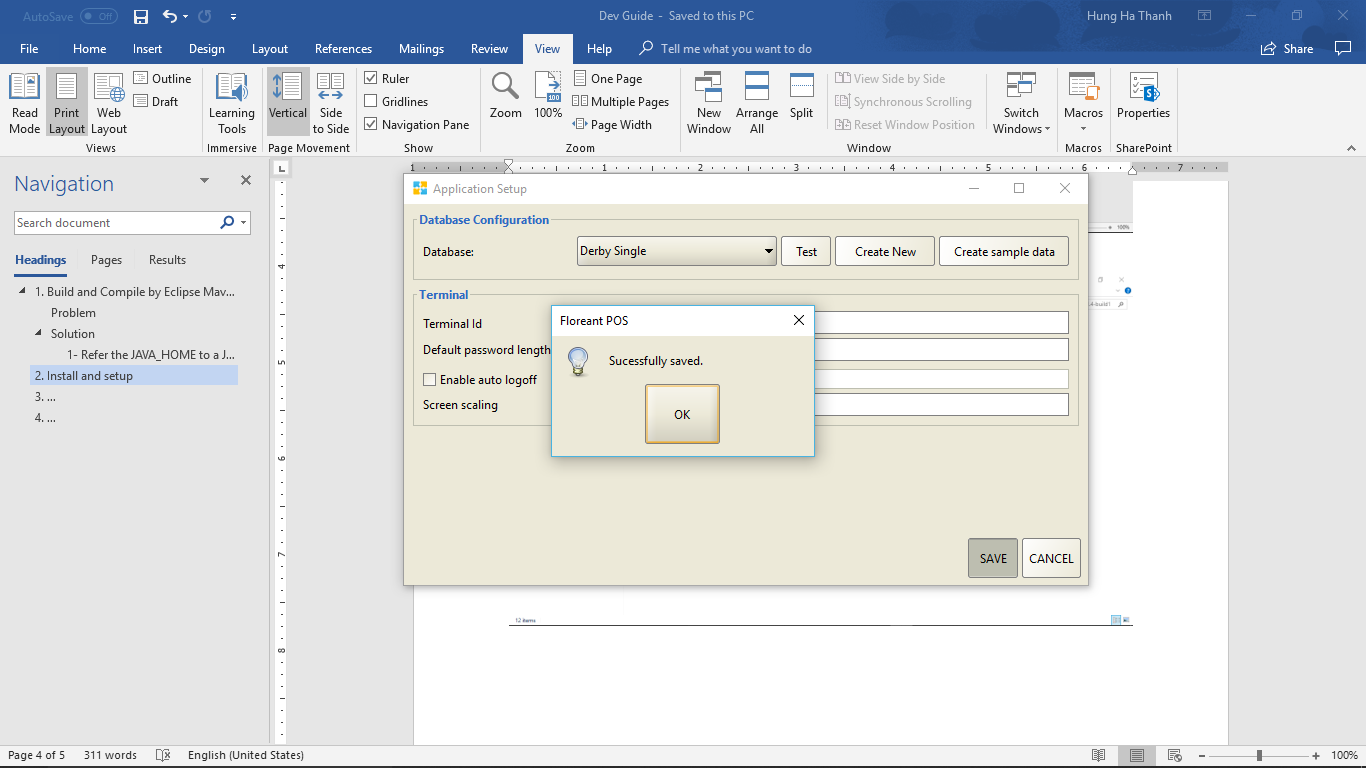
# Install and setup

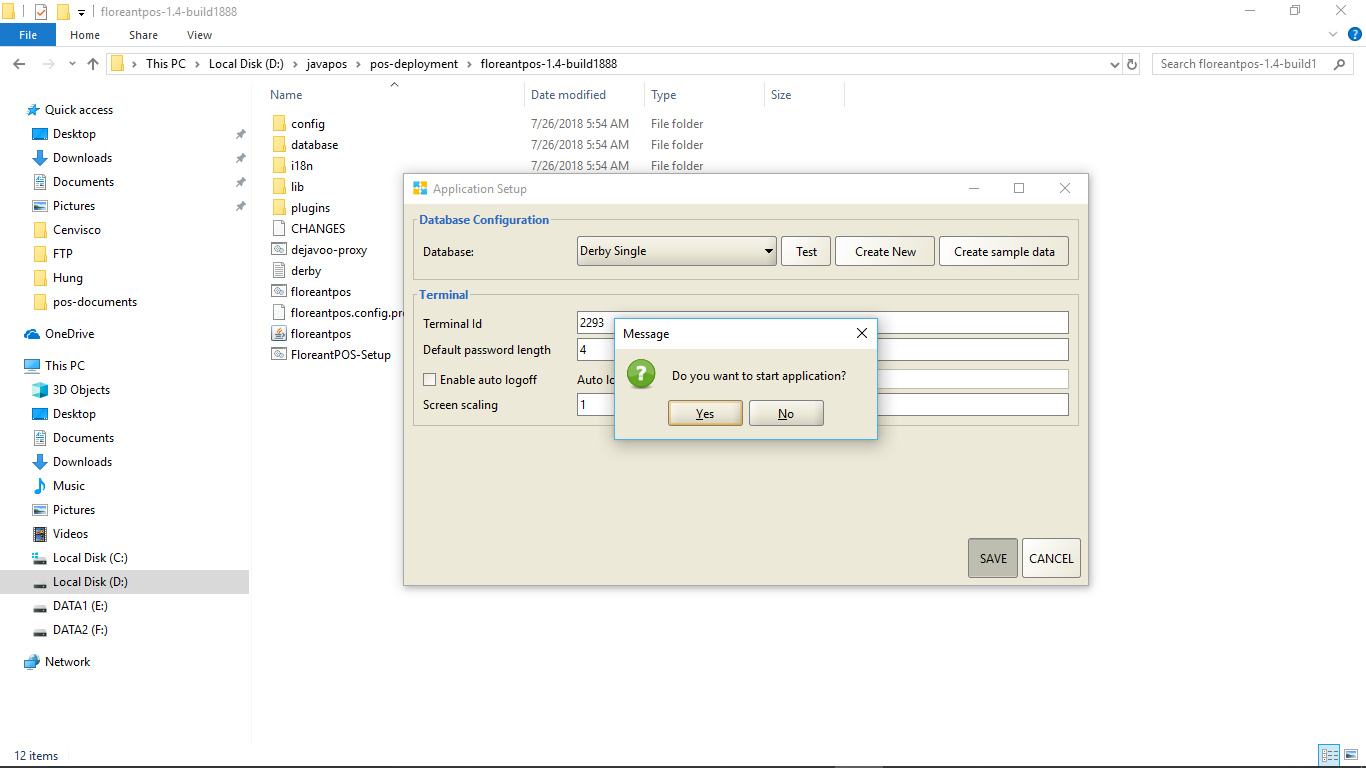
Run step from batch file





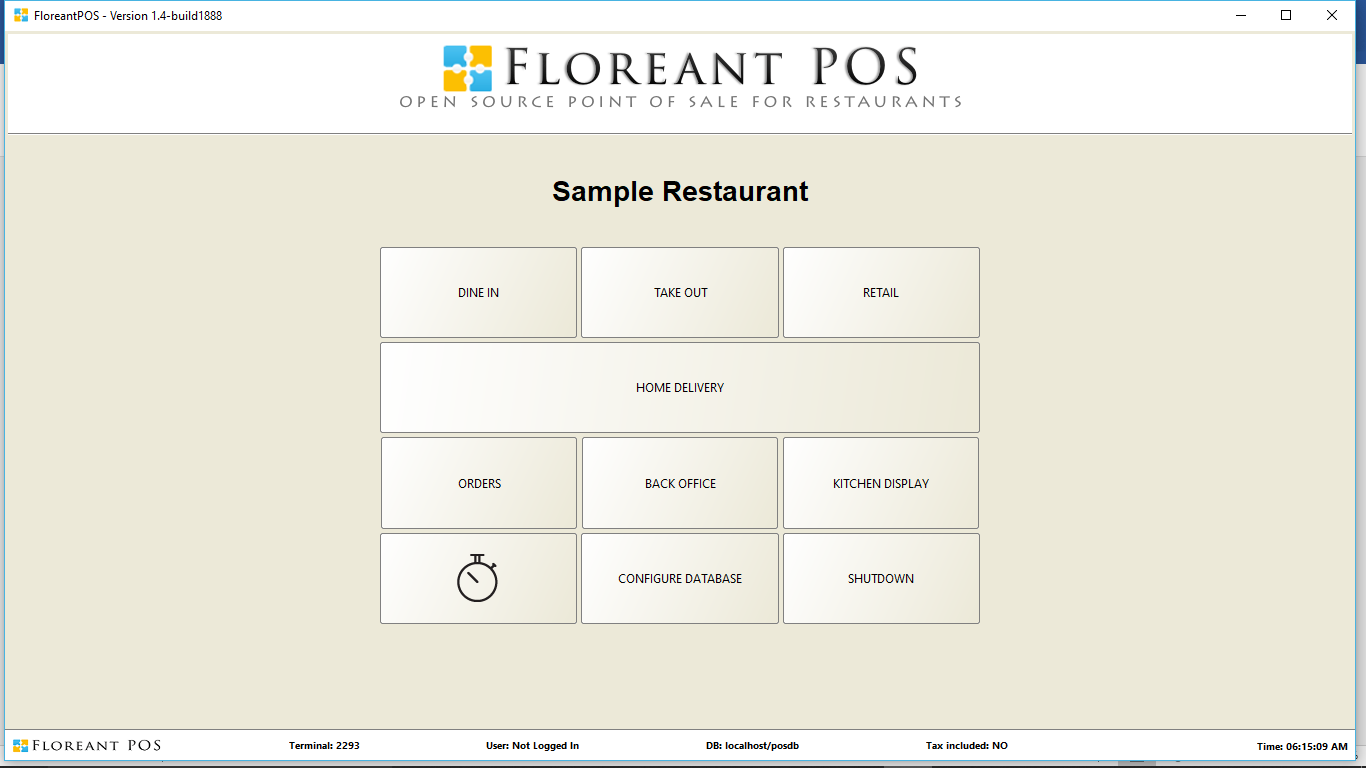




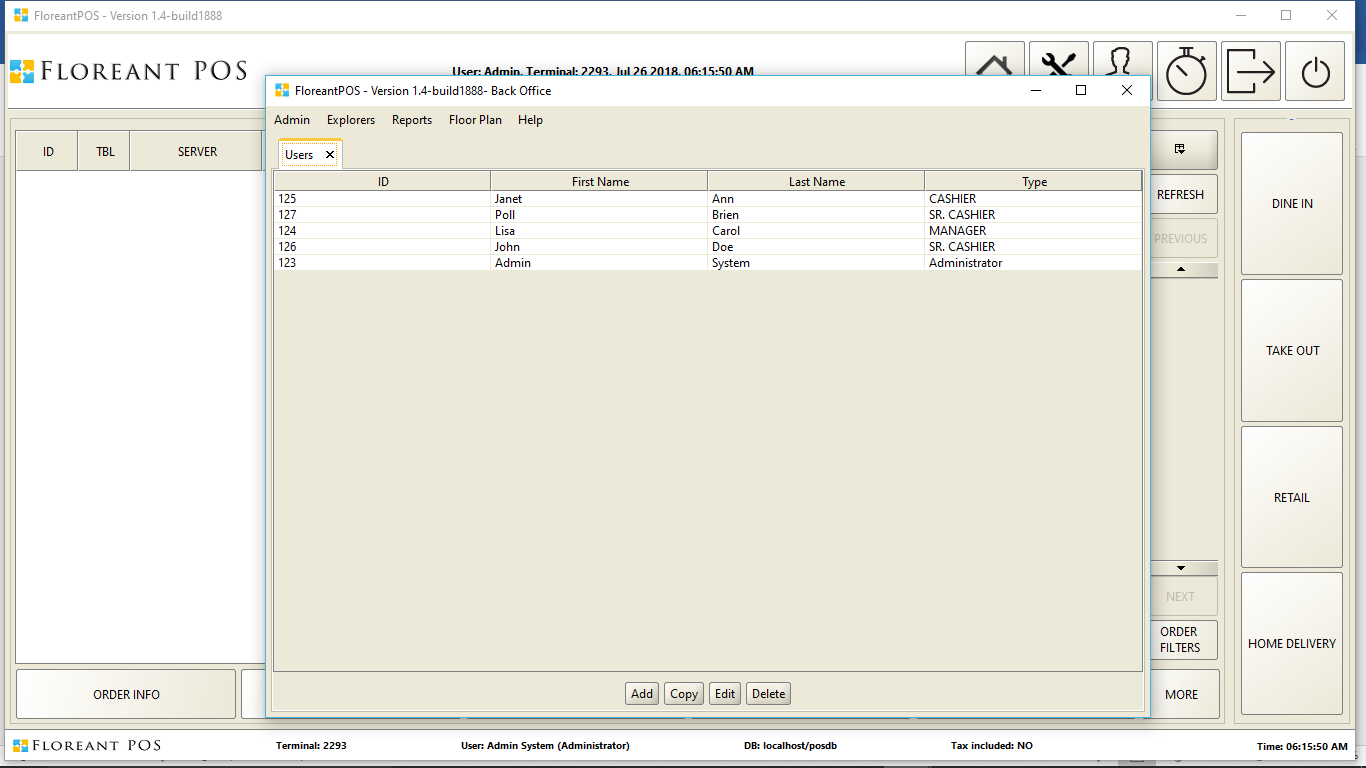


# Run the POS application

Run batch file [D:\javapos\pos-deployment\floreantpos-1.4-build1888\floreantpos.bat](sss)

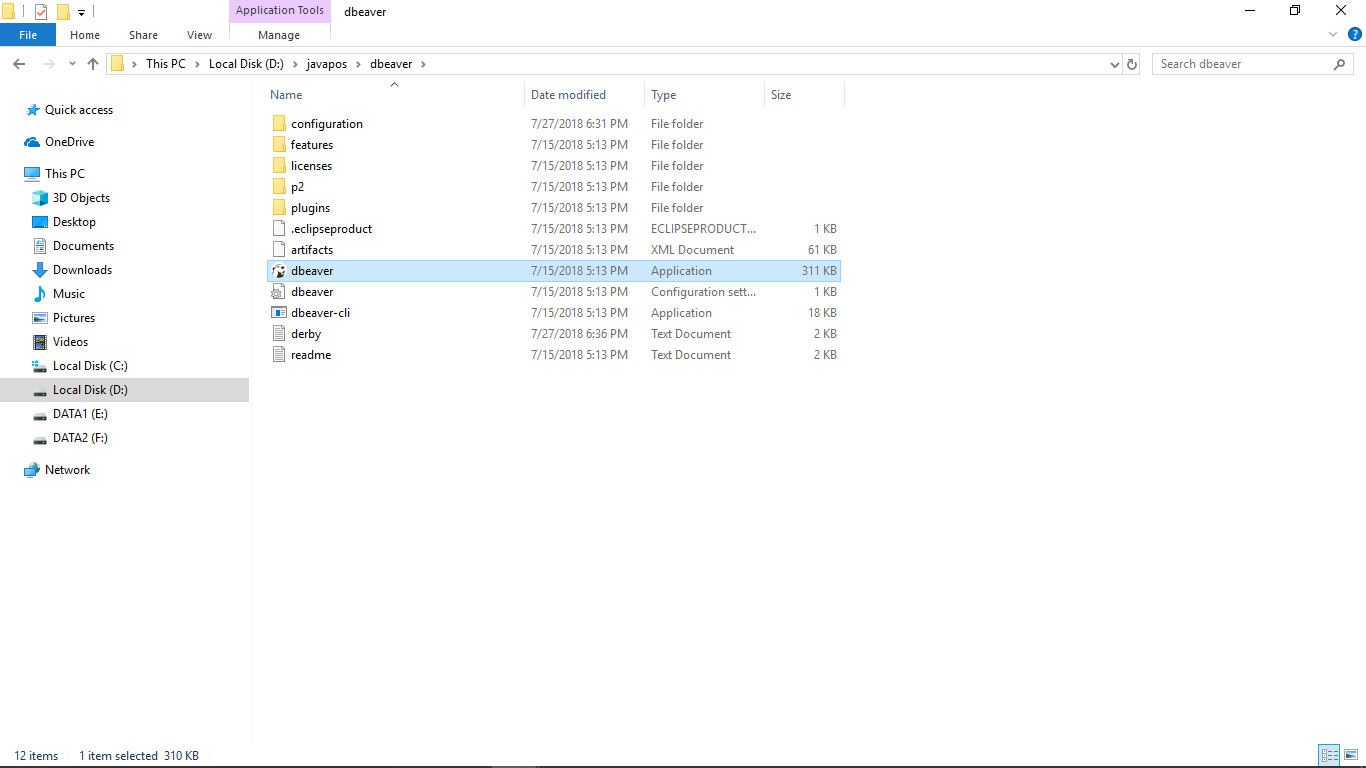


Login admin with default password: 1111



# **Database Access**

If you are power user you can directly access database with [DBeaver SQL client.](https://dbeaver.jkiss.org/docs/features/)



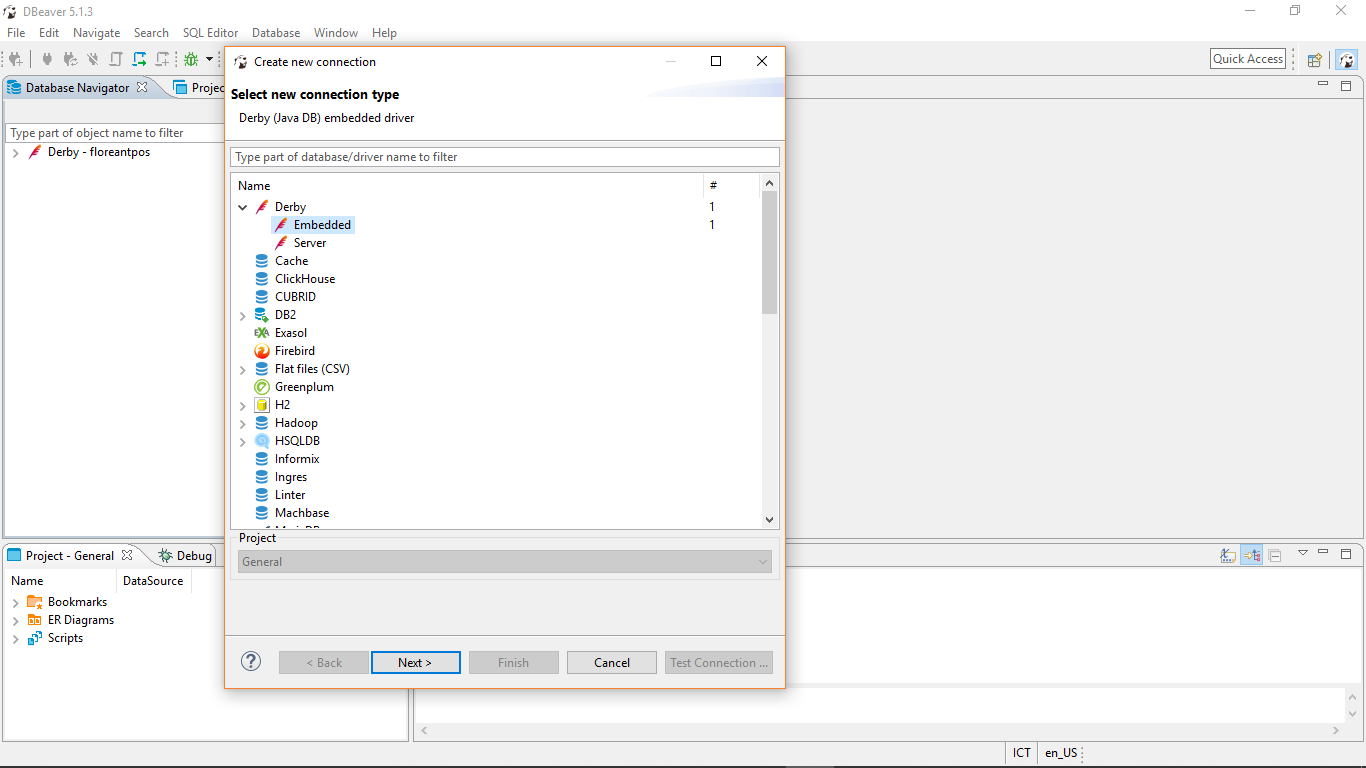
As our system uses hibernate, we would NOT recommend  inserting or updating directly to the database.

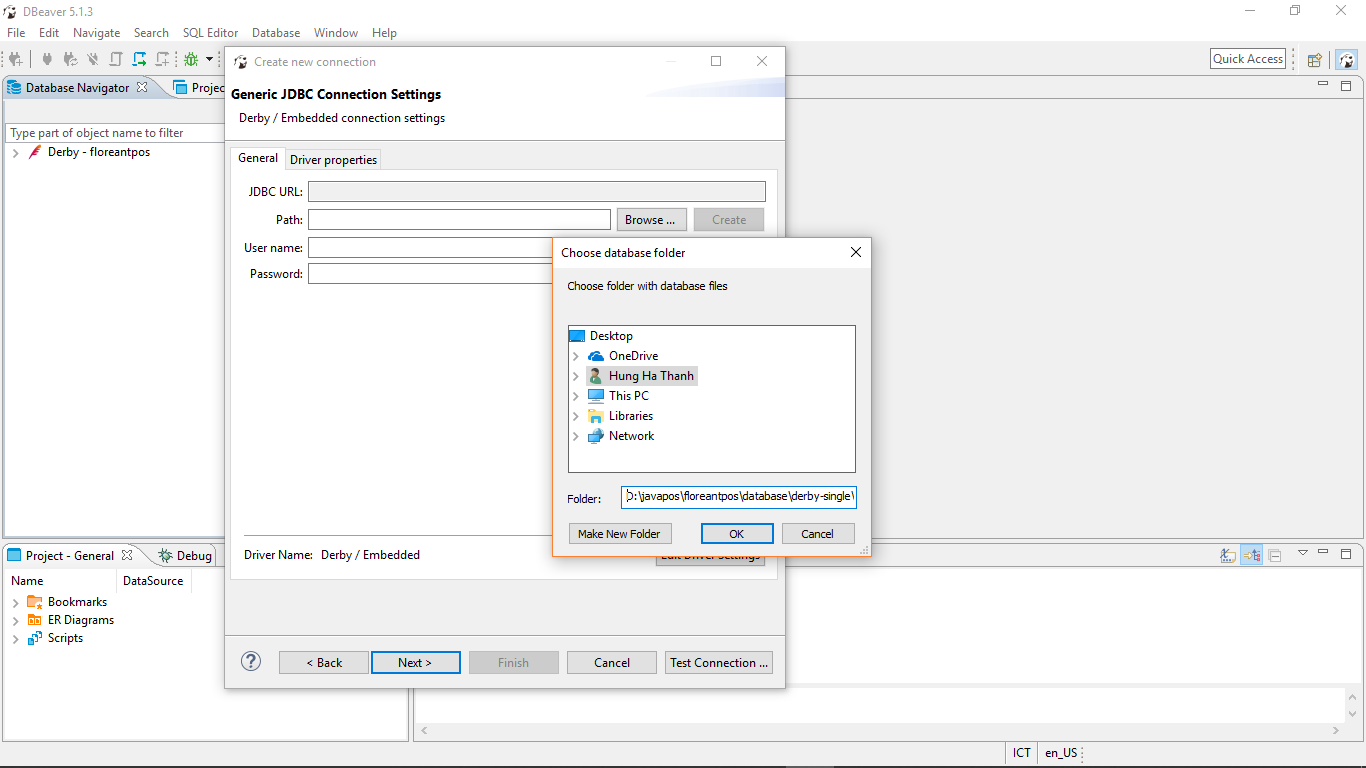
#### Create a Connection to Access Data

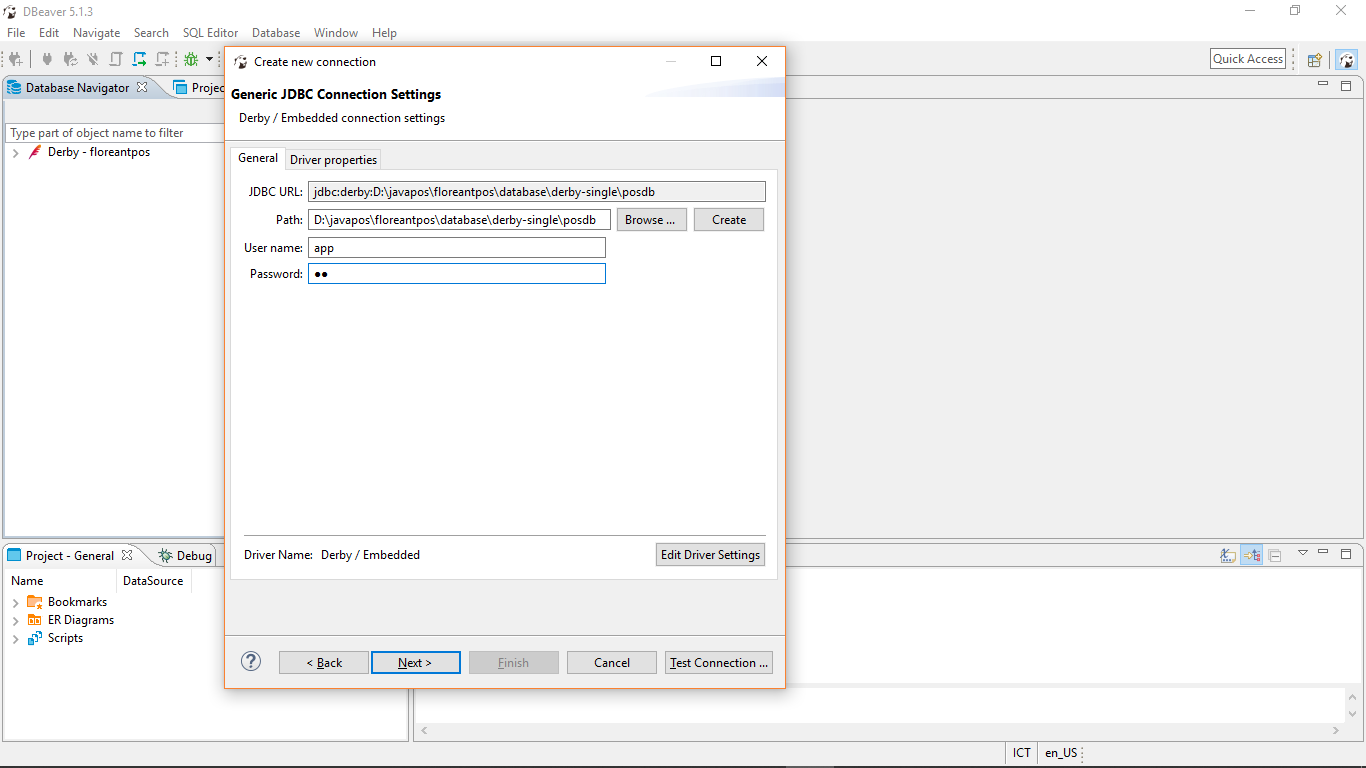
Follow the steps below to add credentials and other required connection properties.

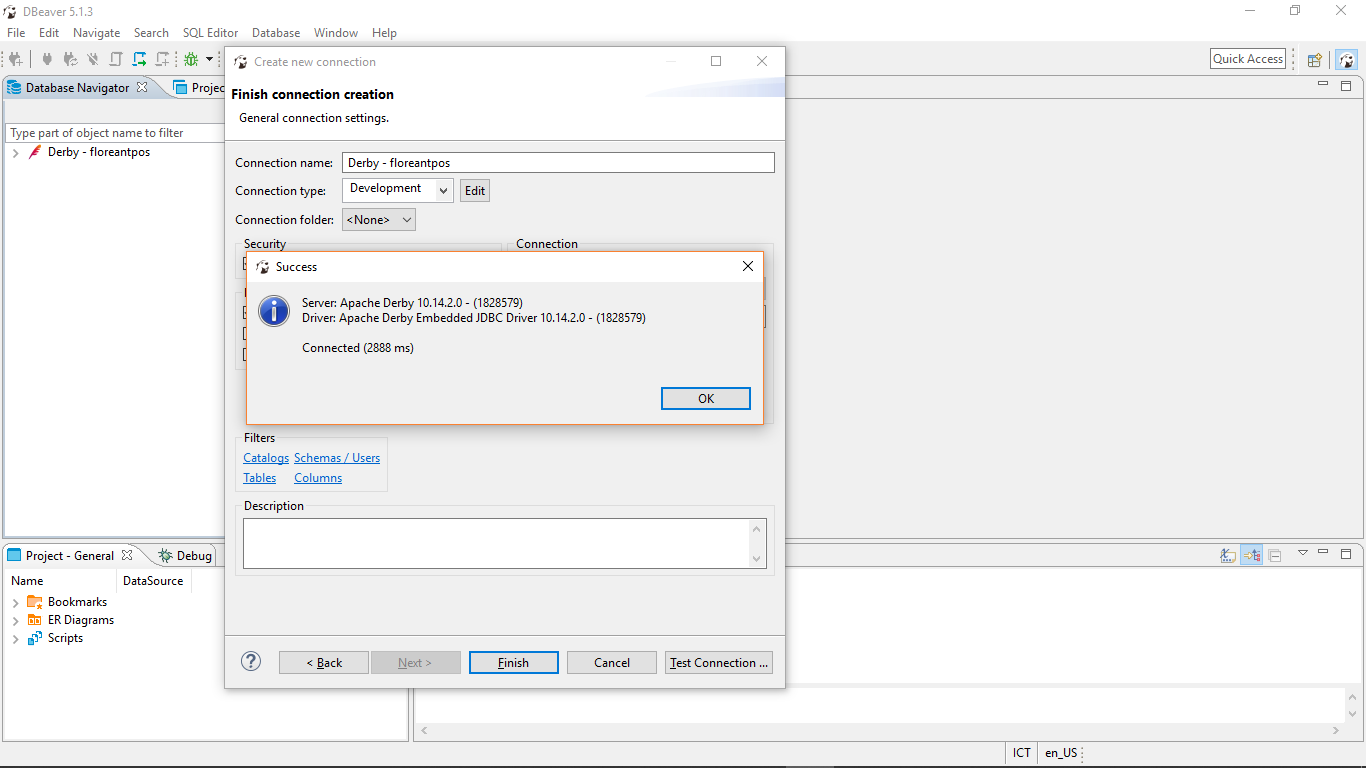
1. In the Databases menu, click New Connection.
2. In the Create new connection wizard that results, select the driver.
3. On the next page of the wizard, click the driver properties tab.
4. Enter values for authentication credentials

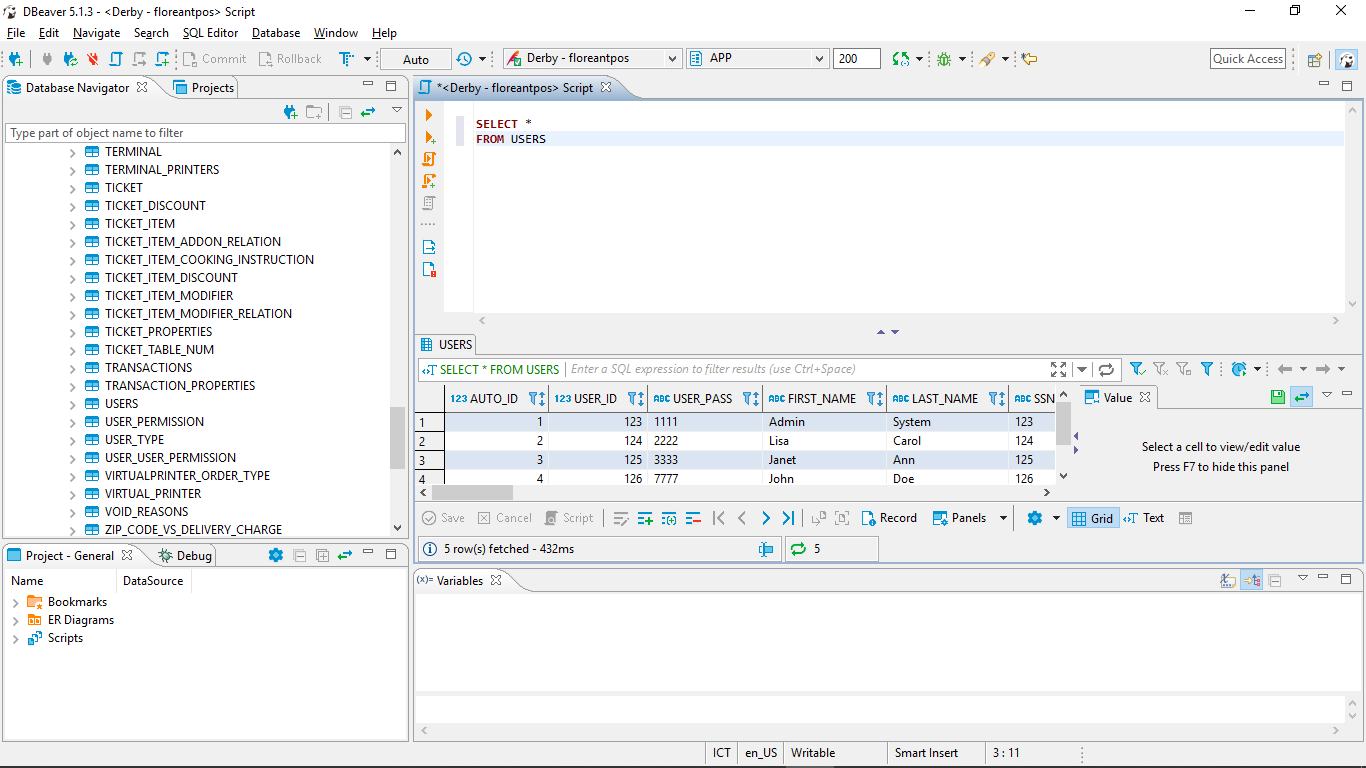
From Menu click on Database and select the new connection





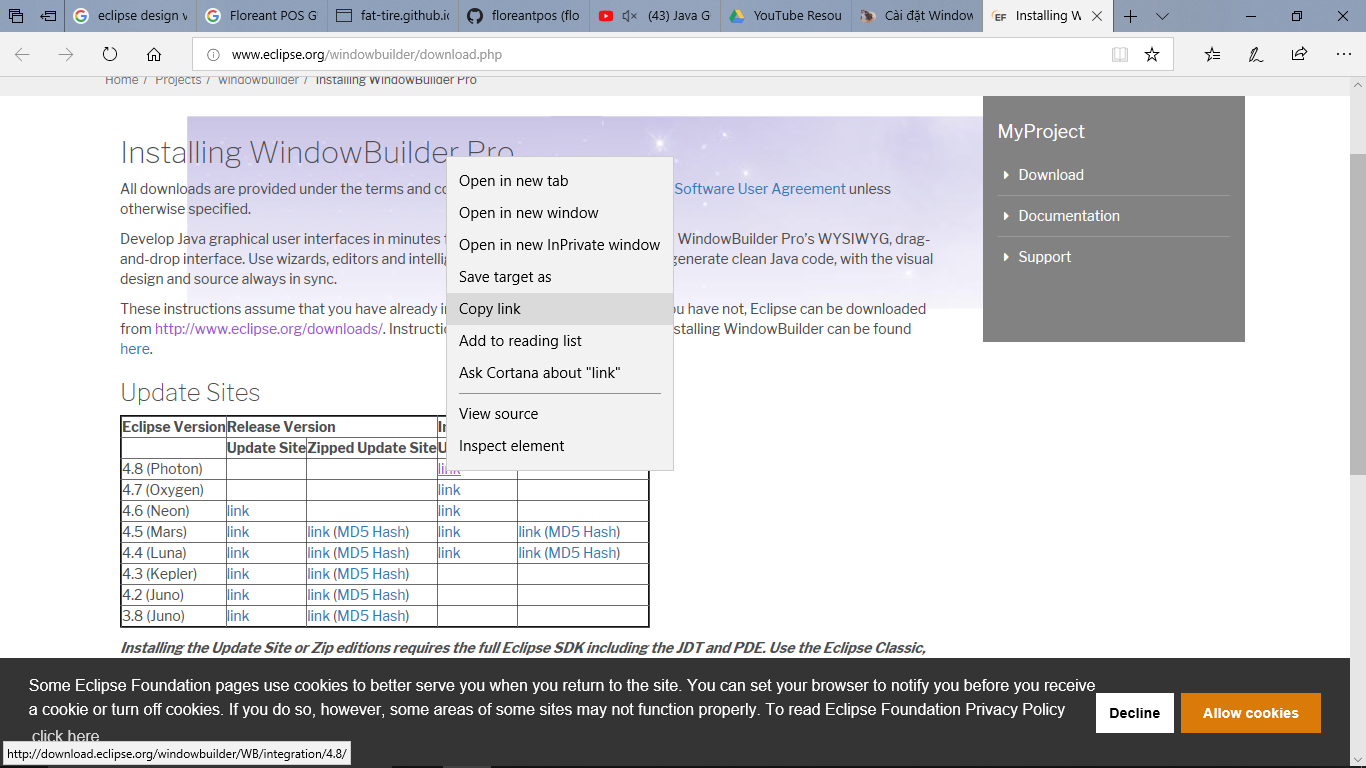


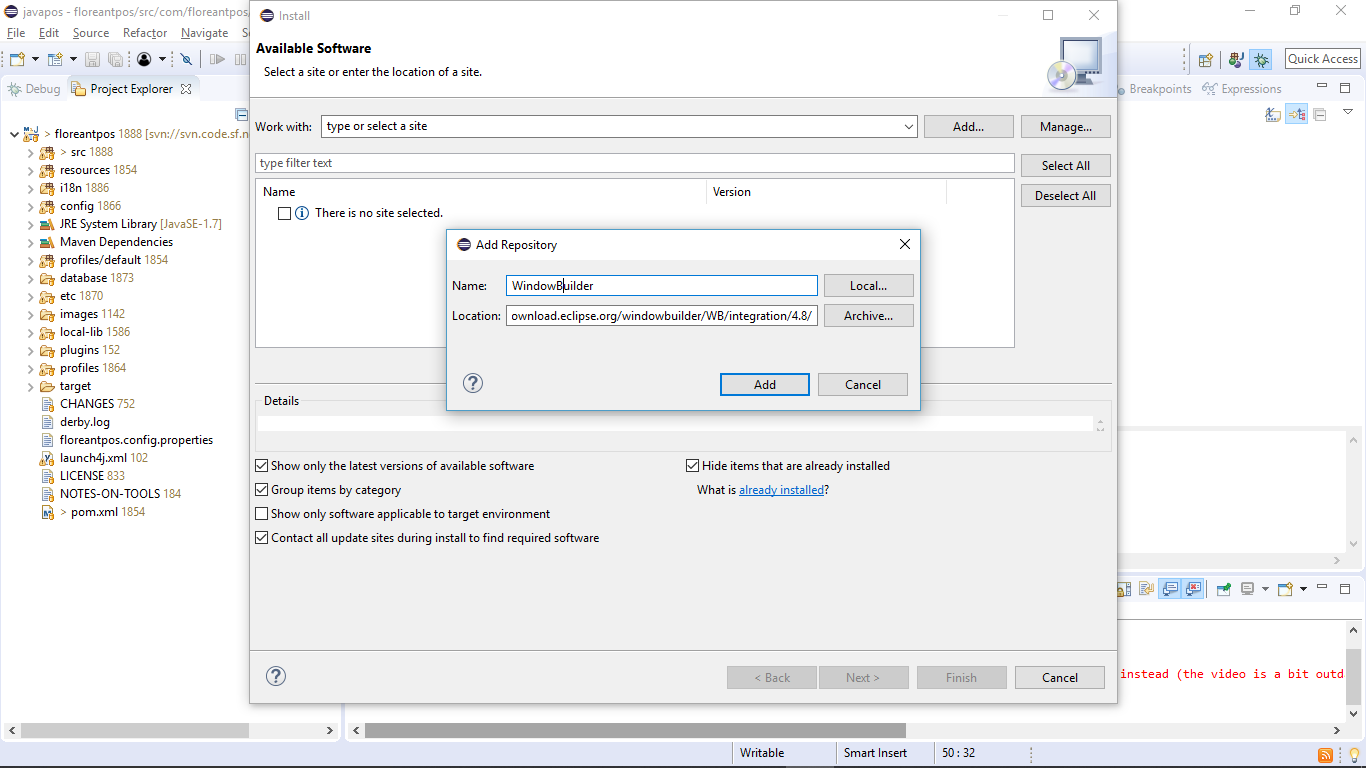


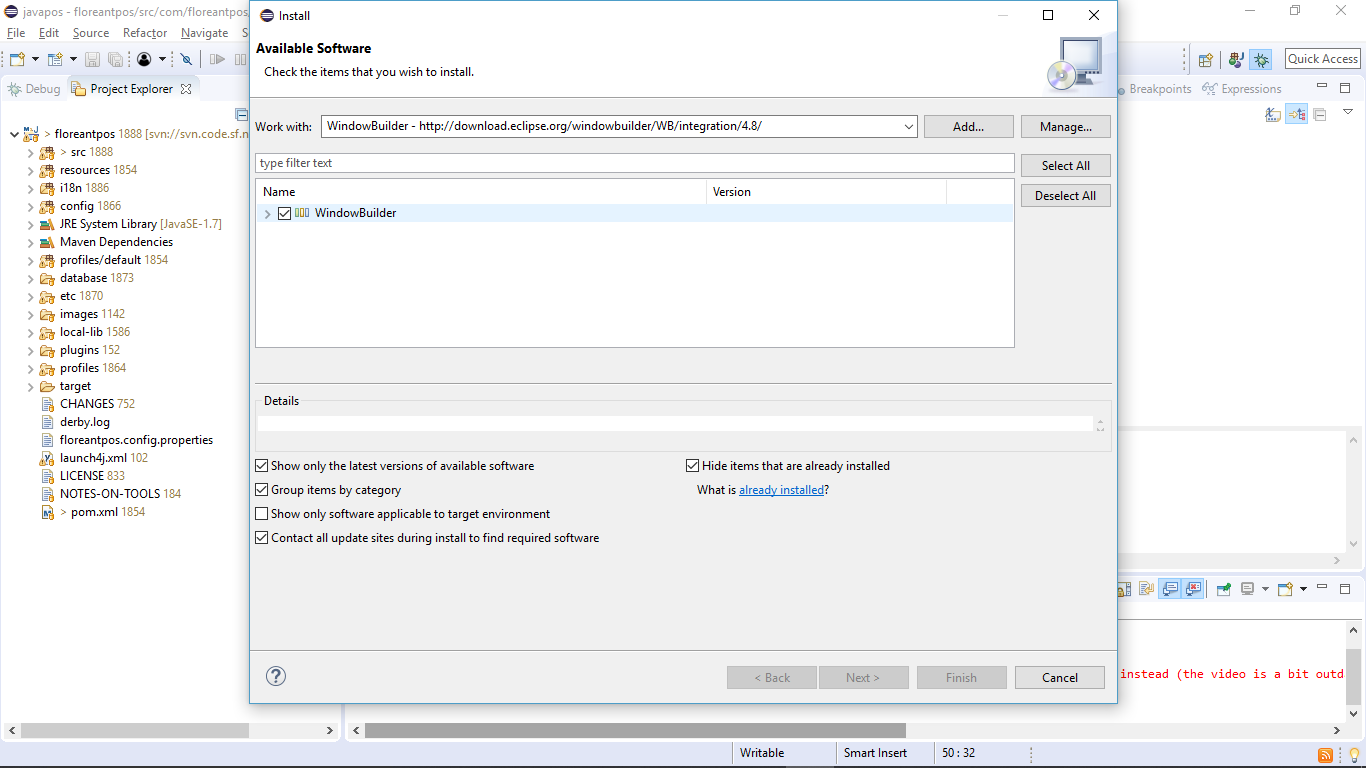


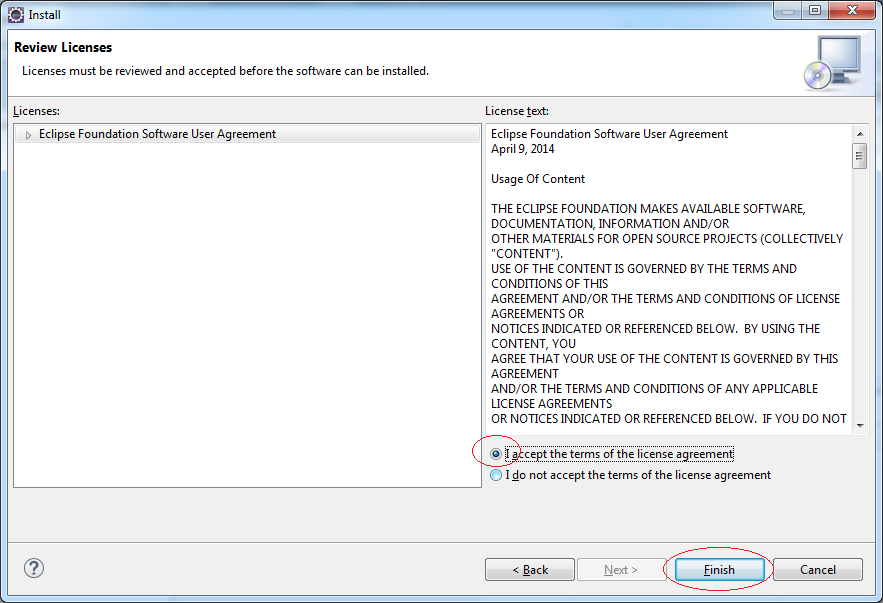
# **Floreant POS Gui Design with WindowBuilder Designer**

## **5.1 Installing WindowBuilder in Eclipse (if necessary) using update sites**



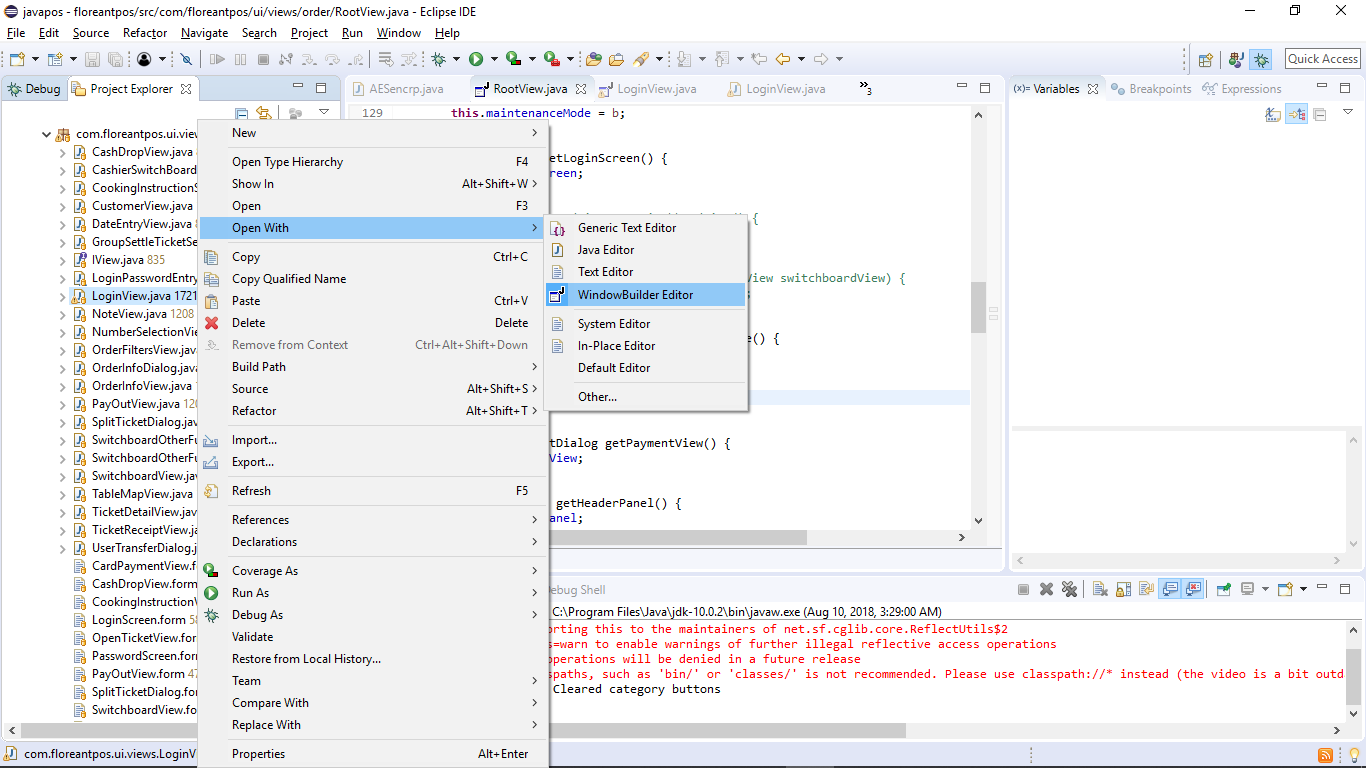


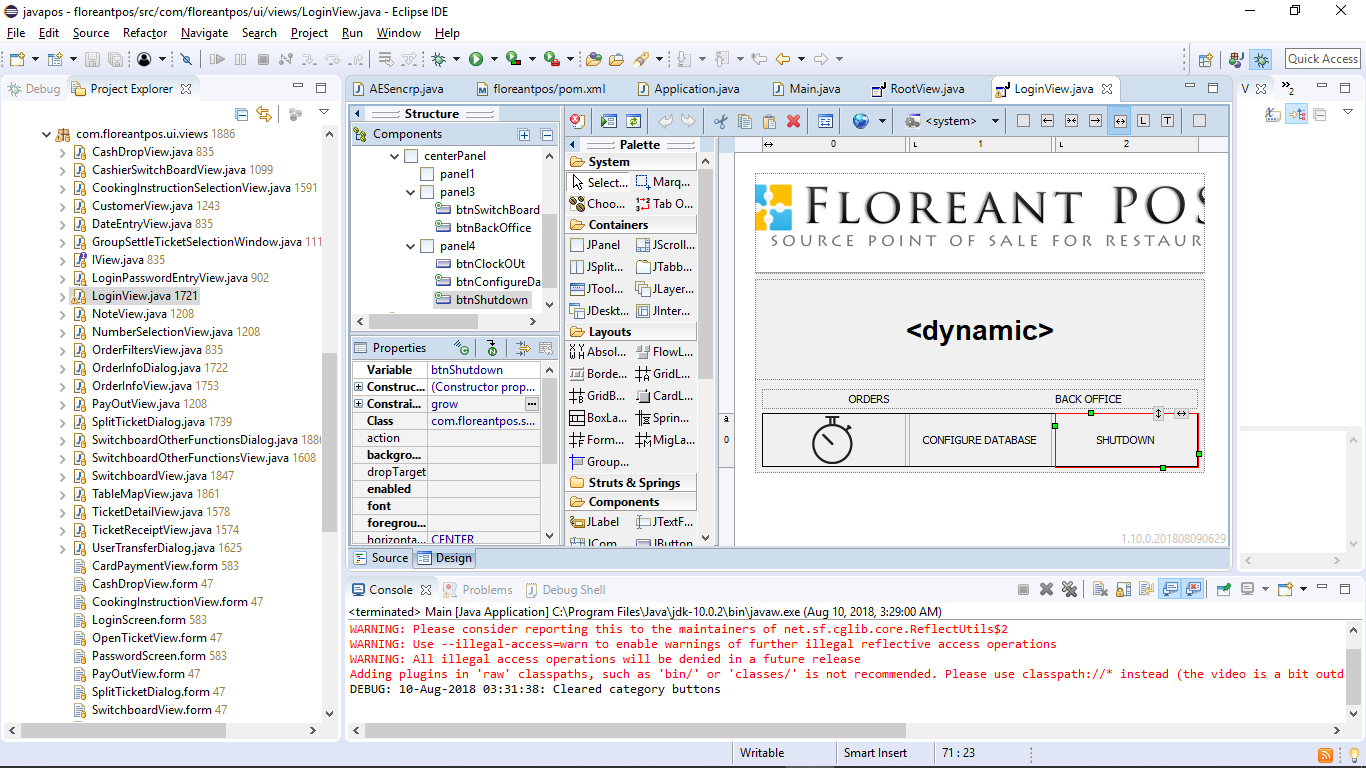




Restart Eclipse to enable Windows Builder plugin

## **5.2 Open View in Design Mode**





## **5.3 Start debug the application**

