


Installation & Configuration

1. [Download](#) and install IntelliJ IDEA Ultimate edition; It **must** be version Ultimate 2020.2.1;
2. Install the Wildfly 20.0.1 Application Server ([download](#) and unzip it to a directory where you want to put the server, preferably to a short path in your system and without any special characters); It **must** be version 20.0.1;
3. Define Wildfly as an Application Server in IntelliJ IDEA:
 - Open the Settings/Preferences dialog;
 - In the left-hand pane, in the Build, Execution, Deployment category, select Application Servers;
 - On the Application Servers page that opens in the right-hand part of the dialog, click +;
 - Select the server you are going to use (JBoss Server, *Wildfly* for friends);
 - In the dialog that opens, specify the server home, that is, the server installation directory.
4. Create a new project in IntelliJ IDEA:
 - Click “Create New Project”;
 - Choose Java Enterprise and click “Next”;
 - In the next dialog, check the “Web Profile” checkbox in the “Specifications” folder, and check the “EclipseLink” checkbox in the “Implementations” folder and click “Next”;
 - Project Name: AcademicManagement; ○ Finish.
5. Now, before continuing with the project, you need to configure the EclipseLink module on WildFly:
 - Download the `eclipselink.jar` library file from moodle and copy it to the following folder:
 `../wildfly20.0.1.Final/modules/system/layers/base/org/eclipse/persistence/main/` ○
 - Open the `module.xml` file in that folder and add the following element to the `<resources>` element:

```
<resource-root path="eclipselink.jar">
  <filter>
    <exclude path="javax/**" />
  </filter>
</resource-root>
```
 - (right below the `<resource-root path="jipijapa-eclipselink-20.0.1.Final.jar"/>` element);
 - Save and close the file.
 -
9. To run the application, we should first create a Run Configuration to start up the Wildfly Server from IntelliJ and deploy our application. For that, click “Add Configuration...” on the top right bar of IntelliJ, and click “+” on the top left of the dialog. Choose “JBoss Server  Local”. Uncheck the “After launch” option, and click the “Fix” button below, for the “Warning: No artifacts marked for deployment”. Choose the “AcademicManagement:war” artifact. Click “OK”;
10. Run the application and pay attention to the Wildfly’s server log (you’ll be looking into it most of the time in this course). Be sure that the application is deployed, output from the `populateDB()` method;

14. You must use a database in order to persist all data from your enterprise application. We will use the database engine shipped with Wildfly (H2), and create our “dae” database:

- Go to `../wildfly-20.0.1.Final/modules/system/layers/base/com/h2database/h2/main/` and run the `h2-1.4.197.jar` Java Application. You’ll be redirected to a web console for the H2 database engine;
- To create a dae database, simply write:
 - ✦ JDBC URL: `jdbc:h2:tcp://localhost/~/projectdaedb`
 - ✦ Username: `sa`
 - ✦ Password: `sa`
 - ✦ Optionally, you can save these settings as `dae`; ○ Click “Test Connection”, and verify that the “Test successful” message appears below.

15. You now need to create a datasource in the Wildfly application server, to connect to this database. Do the following steps:

- Open a command line console;
- Go to directory `../wildfly-20.0.1.Final/bin/` ○ Run command `add-user`:
 - ✦ Choose option (a) (Management User);
 - ✦ Username: `admin`;
 - ✦ Choose option (a) (Update the existing user password and roles);
 - ✦ Enter and reenter password (e.g.:`PORJECT20_`); ✦ Press Enter to leave blank for none;
 - ✦ Answer “no” for the last question.
- Open the Server Administration Console (make sure that the Wildfly server is still running, and point your browser to `http://localhost:9990`);
- On the Configuration section, click “Start” next to “Create a Datasource”; ○ Choose Subsystems -> Datasources & Drivers -> Datasources -> click on ‘+’ / add Datasource:
 - ✦ Choose Template step: Choose H2, Next
 - ✦ Attributes step: Name: “dae”; JNDI: “`java:jboss/datasources/projectdaedb`”; Next;
 - ✦ JDBC Driver step: just click Next;
 - ✦ Connection step: Connection url: “`jdbc:h2:tcp://localhost/~/projectdaedb`”; User Name: “`sa`”; Password: “`sa`”; Security Domain: Leave empty. Click Next;
 - ✦ Test connection; Finish;

16. Go back to your IntelliJ project, and in the `persistence.xml` file

(`src/main/resources/META-INF` folder), replace the `<persistence-unit>` element for the following one:

```
<persistence-unit name="NewPersistenceUnit">
  <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>
```

```
<jta-data-source>java:jboss/datasources/dae</jta-data-source>
<properties>
  <property name="javax.persistence.schema-generation.database.action" value="dropand-
create"/>
</properties> </persistence-unit>
```

17.Run the application (you might want to first execute the command

`./jboss-cli.sh --connect command=:shutdown` on the console/terminal window, if you get a conflict with an URL address already taken in IntelliJ).

18.In order to see the generated data table, do the following: ○ Go back to the web console for the H2 database engine, select your database, fill in the username and password, and click “Connect”;

19.Notice that you can also see the database contents by configuring your datasource in IntelliJ:

○ On the right side of the main editor, click on the “Database” tab to expand it; ○

Click on + -> Data Source -> H2; ○ On the dialog box that opens:

✦ Name: `dae`;

✦ User: `sa`;

✦ Password: `sa`;

✦ URL: `jdbc:h2:tcp://localhost/~/projectdaedb`;

✦ Leave the other fields as they are; ✦ Click “Test Connection”;

✦ Click “Ok”.