# Lab JPA: Spring Data

## Objectives

In this lab, you will use Spring Data to implement a basic entity type.

## Instructions

### Step 1 Create the Spring Boot project

1. Go to https://start.spring.io.
2. Graphical user interface, text, application

   Description automatically generatedWe will be using the Spring data started and use an in-memory database called H2

### Step 2: Ensure Maven is installed

1. Text

   Description automatically generatedCheck to see Maven is installed by opening a command window and executing “mvn -version. You should see the following

### Step 3: Create the project

1. Create a new eclipse workspace
2. Graphical user interface, text, application

   Description automatically generatedUse the file import option to import the unzipped director as a Maven project.
3. A picture containing timeline

   Description automatically generatedOne the project has been imported; Maven will build all the dependencies. This may take a while but eventually your project should look like this.
4. Run the DataApplication.java main() method as a Java Application to ensure that everything is working.

Text

Description automatically generated

### Step 3: Create the Customer entity class and fields

1. Ensure that you are in the Java Perspective.
2. The code so far does nothing because all we have is a hibernate framework and no actual data
3. The entity to be persisted is a Customer object that has an ID which will serve as the primary key in the underlying database; and first and last name string fields.
4. Graphical user interface, text, application, letter

   Description automatically generatedImplement a Customer class as follows:
5. The @Entity annotation indicates that Customer is a persistent entity
6. If the table that Customer is mapped to is not named Customer, there would be an additional @Table annotation that would specify the table name
7. The @Id Annotation on the id field indicates it is the primary key and that it will be auto-generated.

### Step 4: Create the constructors

1. JPA needs a constructor of no args which is protected.
2. The other constructor creates a Customer object populated with data
3. The toString() method give you a nice way to print out a customer

Text, letter

Description automatically generated

### Sept 5: Create the getters

1. Text, letter

   Description automatically generatedFinally, add the get methods for the data fields

### Step 6: Create a Repository Interface

1. Create a new Interface that extends the standard CRUD repository Interface
2. This new interface adds two new methods to the repository interface
3. The first looks up a Customer by id. It returns a single Customer object since the id is a primary key.
4. The second returns a list of all customers with the same last name.
5. The Spring JAP will then write the implementation of these methods using Hibernate when the application is run.

Text

Description automatically generated

### Step 7: Adding to the Runner class

1. The CommandLineRunner class takes a repository as an argument.
2. It returns a series of commands to be executed by the Spring Boot application.
3. Text

   Description automatically generatedThe first set of methods create a few Customer objects and writes them into the H2 database
4. Text

   Description automatically generatedThe findAll() lists the contents of the Customer table. This is a method inherited from the CRUD repository interface.
5. Text

   Description automatically generatedAnd the methods defined in the CustomerRepository are also usable.

The output is shown on the next page.

Text

Description automatically generated