# Exercise 2:

We will continue with module that we created in the first exercise. If you created two maven projects then you will now add a new one for the web application.

Since we do not want to waste time on the setup of tomcat and war creating, we will be using spring boot. Spring boot has an embedded tomcat inside and simplifies the whole creation and setup of a web application (see for more details ...).

I suggest you use the pom file from the git as a start-up for the spring boot application.

The idea behind this exercise is to create a spring rest api that will return data from our web server.

The functionality should be as follows:

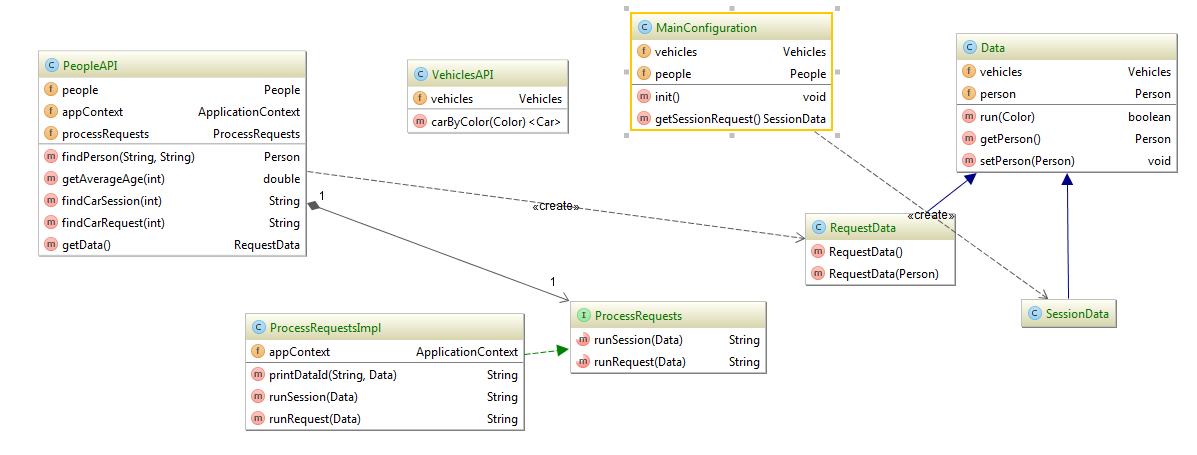
* Vehicles rest service:
  + Return a list of cars based on color as a parameter
* Person rest service:
  + Find person based on first and last name
  + Calculate average of all people above age x
  + Two methods that will simulate a process that will find cars for people. We will need another object for the processing.
    - One object will have scope of request
    - One object will have scope of session

What will be the difference between the two?

* MainConfiguration

This class should be a configuration class to create all data when server comes up. You should try to create a bean using this class as well.

## Domain classes



We want here to focus more on the spring aspect of the project and not the actual domain or java features.

In this project we want to have both xml configuration files and also classes with annotation.

## Xml file

*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd"**>  
  
 <**bean id="vehicles" class="com.tikal.course.java.service.impl.VehiclesImpl"**>  
  
 </**bean**>  
  
 <**bean id="people" class="com.tikal.course.java.service.impl.PeopleImpl"**>  
  
 </**bean**>  
  
  
</**beans**>

## Rest Requests

In this section there are sample rest examples that should be implemented in the server

http://localhost:8090/vehicles/byColor?color=red

http://localhost:8090/people/findPerson?firstName=jacob&lastName=israel

http://localhost:8090/people/getAverageAge?overAge=2