# Web Application Proof of Concept

The proof of concept for the web application is a [Spring Boot](https://spring.io/) project written in Java.

## Goals

Team Delta set out to achieve the following goals in the Spring Boot project to prove that the Web Application was feasible:

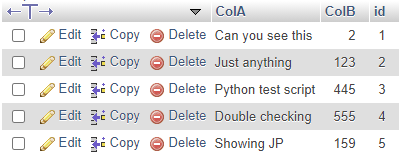
1. Calling data from a database from the web application.
2. Displaying the retrieved data in JSON format.
3. Adding data to the database.

## Running Instructions

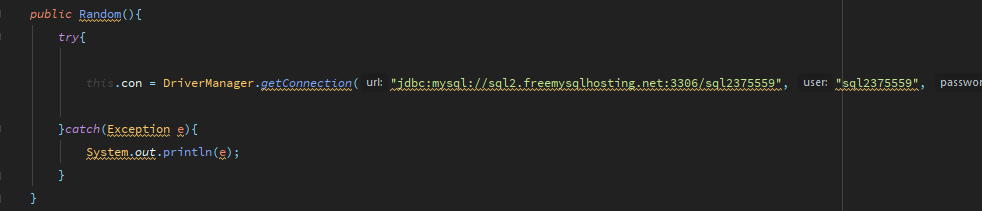
1. Please ensure that you have installed Java and [JDK](https://www.oracle.com/uk/java/technologies/javase-downloads.html) on your system and the environment variables are set.
2. Open your terminal and cd into the 'SpringAPI' folder and then cd in to the 'demo' folder.
3. Run the command mvnw spring-boot:run if you are using Windows or ./mvnw spring-boot:run on MacOS/Linux then wait for the web application to run.
4. Open your web browser and head to localhost:8080

## Goal 1 & Goal 2: Calling Data from the Database and Displaying as JSON

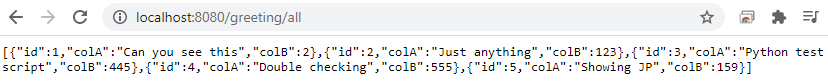
As stated in the Technical Report, the database is externally hosted by [FreeMySQLHosting](https://www.freemysqlhosting.net). The database only has one table called 'random' and this table has three columns called: ColA (string), ColB (int) and ID (primary key).

  
*Database table ‘random’ shown in PhpMyAdmin*

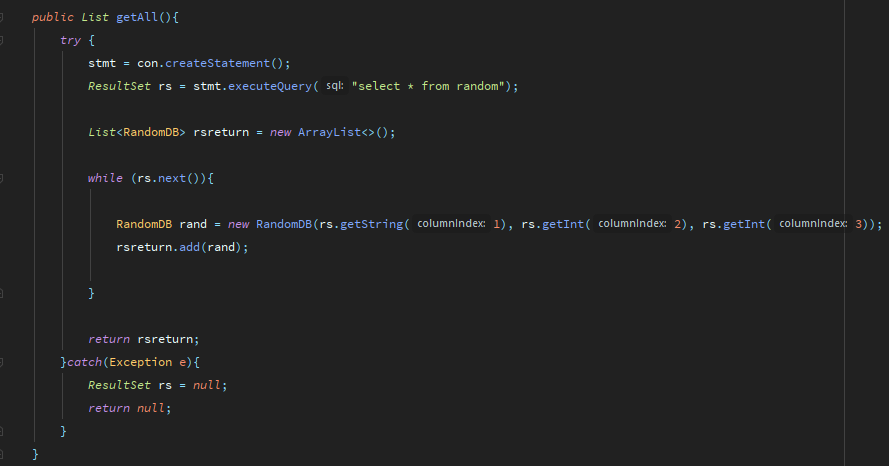
To call data from the database the application needs to connect to it first. This is achieved using JDBC.



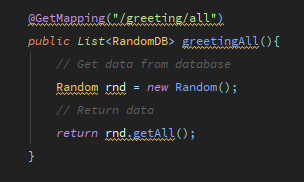
You can call all the data in the database by clicking the ‘View Table’ button or by typing in localhost:8080/greeting/all into the address bar which should return the below results.



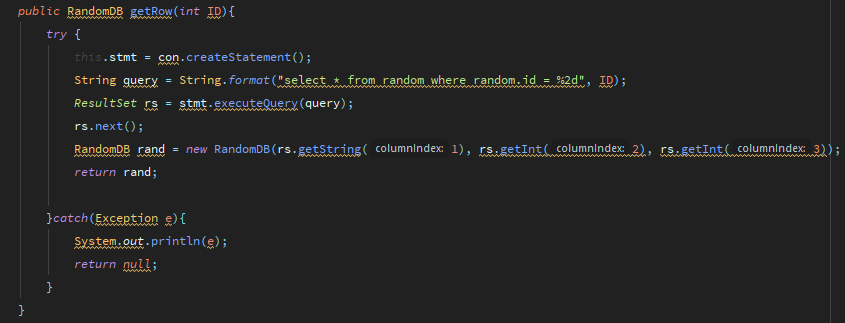
Below is how this data is retrieved and displayed:

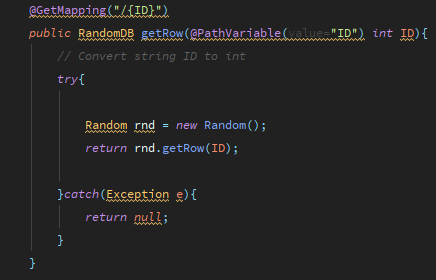


The method getAll() in the ‘Random’ model executes a query on the ‘random’ table in the database that selects all of content in the table and stores the result of the executed query as a ResultSet. The ResultSet is iterated through to store each of its items as a custom class, RandomDB and these items are put into a list that gets returned to the main program.



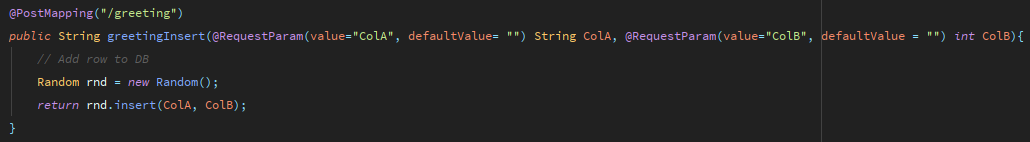
The URL /greeting/all maps to the greetingAll method in the GreetingController class that calls and returns the getAll() method in the ‘Random’ model. The GreetingController class is declared as Spring Boot’s @RestController which effectively means that it generates whatever gets returned to it as JSON.



The getRow(int ID) is a method in the ‘Random’ model that works similarly to get the getAll() method but just retrieves a single row by ID instead of all rows.

## Goal 3: Adding Data to the Database

Data can be added to the database through the web application using a Python script. The script, requestTester.py, sends a POST request to the locally hosted web application which is then handled by the greetingInsert method in the GreetingController class as shown below:



This method calls the insert(ColA, ColB) method from the ‘Random’ model:

