**Introduction**

This is an implementation of a RESTful web service that performs CRUD operations (Create, Read, Update, and Delete) for a Vehicle entity.

The following table shows the available REST end-points and what they achieve:

|  |  |
| --- | --- |
| **API** | **Description** |
| **GET** vehicles/{*id*} | Returns a vehicle specified by its unique ID. |
| **GET**  vehicles?make={*make*}&model={*model*}&year={*year*} | Returns vehicles in the directory filtered by their properties. |
| **GET** vehicles | Returns all vehicles in the directory. |
| **POST** vehicles | Adds a vehicle to the directory |
| **PUT** vehicles | Updates a vehicle to the directory. |
| **DELETE** vehicles/{id} | Deletes a vehicle specified by its unique ID from the directory. |

**Implementation Details**

|  |  |
| --- | --- |
| Usage of either C# or Java. | Java. |
| Some form of automated testing. | JUnit. |
| Some form of in-memory persistence of created vehicle objects. | In-memory persistence by using ConcurrentHashMap. |
| Function properly with the provided test web client. | Tested extensively using Postman REST client. |

**Optional Implementation**

|  |
| --- |
| 1. Added validation to my web service.   • Vehicles must have a non-null / non-empty make and model specified, and the year must be between 1950 and 2050 inclusive. |
| 1. Added filtering to my web service.   • The GET vehicles route should support filtering vehicles based on one or more vehicle properties. (EX: retrieving all vehicles where the ‘Make’ is ‘Toyota’) |

**How to run application**

1. Visit <https://drive.google.com/drive/folders/1mPDrk7AKmDQVEd_84-fcrC0PBSLQD1Sm>. You’ll find the source code as well as the standalone deployable artifact **vehicle-restAPI-1.0.jar** which has embedded tomcat web server, so you will not need an application server to run this application**.**
2. Download **vehicle-restAPI-1.0.jar** from the visited URL into a folder of your choice.
3. Open Windows command prompt or Linux command shell depending on your OS.
4. Change path to the directory where you put the downloaded jar.
5. Run the command **java -jar vehicle-restAPI-1.0.jar.**

[Note: Make sure you have Java installed. If not, get it installed before you follow these steps.]

1. The application will run on port 8080 by default. To change the port number, run this command instead: **java -jar -Dserver.port=*port-number* vehicle-restAPI-1.0.jar**
2. Wait for the application to start. It won’t take more than a few seconds.
3. Test the RESTful end-points listed in the table above using a rest client of your choice. I used Postman to test my web service.

If you have any questions, please free to reach out to me via email: emfernandes49@gmail.com

Thank you!