

Parking Toll Rest api

Parking toll application is a Spring Boot Rest application. The project can be launched like a simple Java application from any IDE like Eclipse/IntelliJ or with command line.

Once the application launched, we can start requesting it. Two main operations are supported:

1. Add Vehicle:

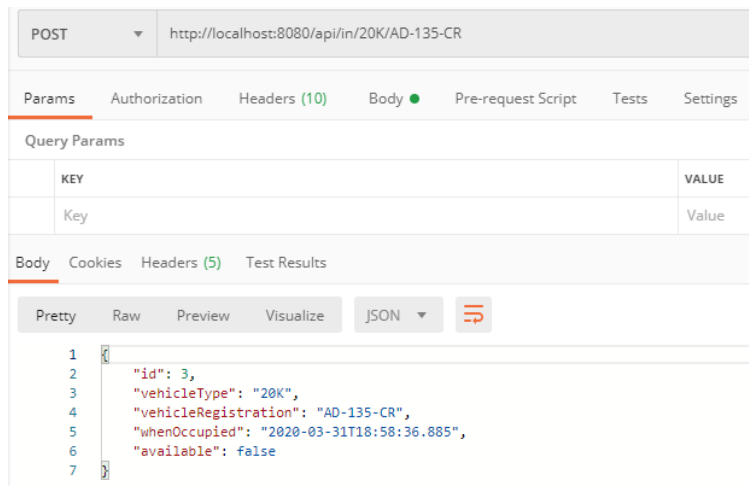
- Request

POST `/api/in/{vehicleType}/{vehicleRegistration}`

- Parameters:

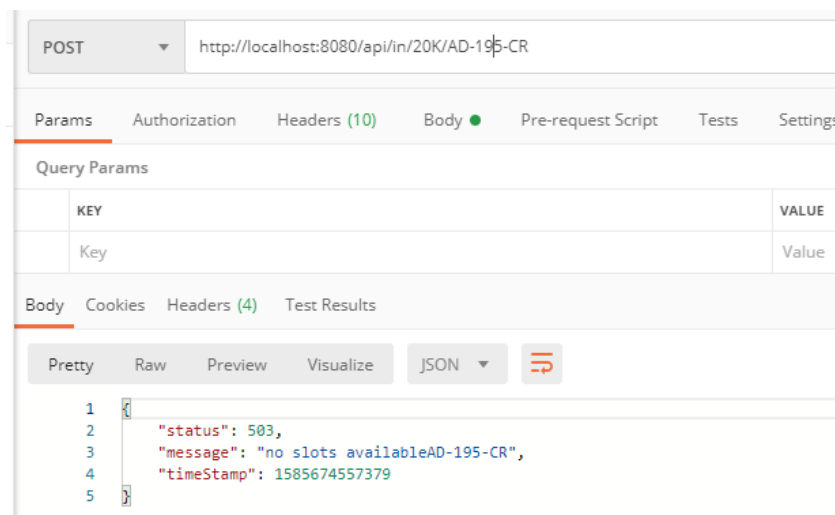
- vehicleType: e.g. GAS/20K/50K
- vehicleRegistration: e.g. AB-00-CD

1.1 Adding Vehicle Success



If the Vehicle is added successfully, the associated Slot will be returned as json response

1.2 Adding Vehicle, No available slots



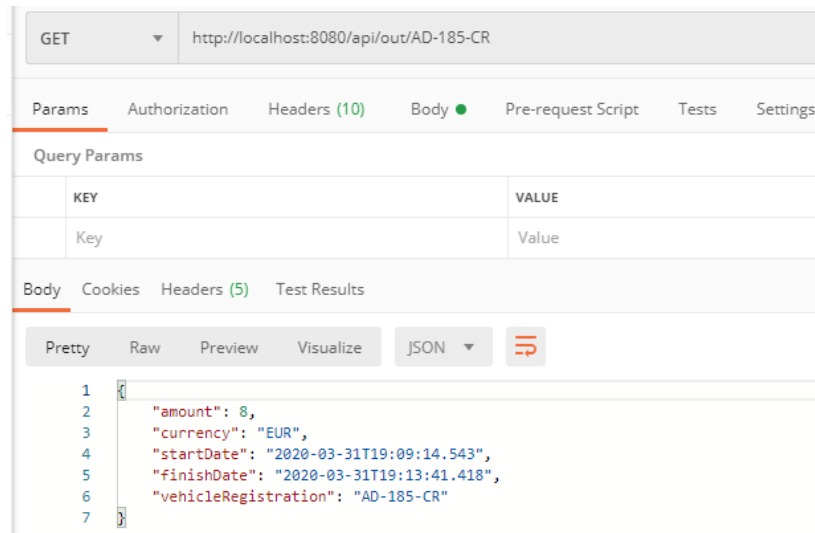
2. Remove Vehicle:

- Request

GET /api/out/{vehicleRegistration}

- Parameters:
 - vehicleRegistration: e.g. AB-00-CD

2.1 Remove Vehicle Success



GET http://localhost:8080/api/out/AD-185-CR

Params Authorization Headers (10) Body ● Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

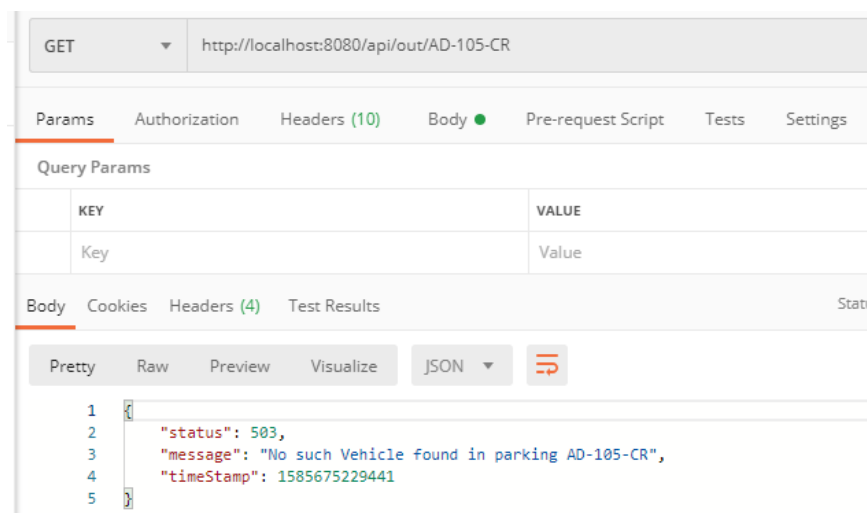
Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON ↗

```
1 {
2   "amount": 8,
3   "currency": "EUR",
4   "startDate": "2020-03-31T19:09:14.543",
5   "finishDate": "2020-03-31T19:13:41.418",
6   "vehicleRegistration": "AD-185-CR"
7 }
```

If the Vehicle is removed successfully, a bill will be issued and returned as Json response

2.2 Remove Vehicle Registration Number not valid



GET http://localhost:8080/api/out/AD-105-CR

Params Authorization Headers (10) Body ● Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (4) Test Results Status

Pretty Raw Preview Visualize JSON ↗

```
1 {
2   "status": 503,
3   "message": "No such Vehicle found in parking AD-105-CR",
4   "timeStamp": 1585675229441
5 }
```

3. Data base:

For The demo a H2 Data base is used. This will avoid any DB setup from user. By default, the DB is initialized with following data: 2 Slots of GAS, 2 Slots for 20K and 2 Slots for 50K.

When launching the application, we can visualise the DB console from:

<http://localhost:8080/h2-console>

English ▼ Preferences Tools Help

Login

Saved Settings: Generic H2 (Embedded) ▼

Setting Name: Generic H2 (Embedded) Save Remove

Driver Class: org.h2.Driver

JDBC URL: jdbc:h2:mem:testdb

User Name: sa

Password:

Connect Test Connection

We can then launch a select * to visualise the content: SELECT * FROM SLOT

localhost:8080/h2-console/login.do?jsessionId=f291cf07600ddf2a2ddaa80d8e0f3714

Auto commit Auto complete Off Auto select On

jdbc:h2:mem:testdb

- SLOT
 - ID
 - IS_AVAILABLE
 - VEHICLE_REGISTRATION
 - VEHICLE_TYPE
 - WHEN_OCCUPIED
- Indexes
- INFORMATION_SCHEMA
- Sequences
- Users

H2 1.4.200 (2019-10-14)

Run Run Selected Auto complete Clear SQL statement:

select * from slot

select * from slot:

ID	IS_AVAILABLE	VEHICLE_REGISTRATION	VEHICLE_TYPE	WHEN_OCCUPIED
1	TRUE	null	GAS	null
2	TRUE	null	GAS	null
3	TRUE	null	20K	null
4	TRUE	null	20K	null
5	TRUE	null	50K	null
6	TRUE	null	50K	null

(6 rows, 10 ms)

Edit

4. Swagger Documentation

Once the application is started swagger documentation could be reached at :

<http://localhost:8080/swagger-ui.html>

