# COMP9321 Cars Analytics API endpoints:

* **Get /cars/{budget}/{brand}** (Eg: Budget: 6500, Brand: bmw)

Give suggestions of cars that the user can buy, based on the brand he wants and the price he wishes to spend.

* **Get /loans**  (Eg: Principal: 15000, Term: 36, Interest: 7)

A loan calculator for user to calculate the amount he has to pay monthly in order to fulfill the payment of the car he wants to purchase.

* **Get /price**

This endpoint predicts price for the used car, based on the car details entered by the user. This endpoint needs to get brand, model, vehicleType, yearOfRegistration, gearbox, powerPS, kilometer, fuelType, notRepairedDamage. Price prediction is done using DecisionTreeRegressor

* **Get /reliability no params**

Get request, returns chart of brand reliability analysis.

* **Get /comparisons**

This endpoint lets the user compare the cars based on the reliability index and the average repair cost. Customer needs to enter the Brand.

* **Get /graphcomparisons**

This endpoint gives a graphical view of the previous endpoint by displaying a grouped bar chart of the two cars compared.

* **Get /session** params: token e.g. token=”extremely long” returns=”username”

Get session returns the end point providing information in which one of the user has the ownership of the session/token

* **Post /session** params:username, password e.g. username=morty, password=albanna, returns: “auth-token”: “extremely long token”

Post creates a user session, the token is required for authentication and stay signed in for the access of the other api endpoints for 100minutes

* **Get /user** params: username e.g. username=user

Get http request, sends a text param as username, looks up the user records and returns corresponding user’s information to the client

* **Post /user** params: username, password e.g. username=morty, password=albanna, returns: user created

If create successful, Post http creates a user record on request with username and password

* **Put /user** params: username e.g. username= user, returns: User Grant Admin Access

Raises the permission of the user to become an Admin, to access other endpoints, modifies resource hence put.

* **Delete /user**  params: username e.g. username= user, returns: user deleted

Deletes a user record (username and password) from the database, erases it from admin and user if exist on admin

* **Get /user/{user\_id}** params: user\_id e.g. user\_id=2, returns: [2,”user2”]

Input a user\_id to the end point, retrieves a username for the corresponding user\_id’s activities