**Testing and Documentation – ADMIN API**

**Endpoint URL -** https://rjohnston80.webhosting3.eeecs.qub.ac.uk/adminapi/

**Security –** APIis protected using an API Key using parameter name (“API-KEY”) in the request, it is a random 50 character length string generated upon user registration and applies both to admins and users, admin API Keys provide access to the admin API on top of the standard APIs (parkingapi and registerapi). Customers/users registered are unable to use their API Key to access the admin API as it is a low level of access. This adds protection for the admins and system overall.

**Default Parameters** – The default parameters added to admin api requests are the **adminusername** parameter which holds the username of the admin account accessing the API and also the **API-KEY** parameter which holds the admin’s API Key.

In total there are 19 different functions associated with the admin API. They are documented and tested below.

**API-Key Testing**

Graphical user interface, text, application, email

Description automatically generatedWhen a correct API Key is passed in combination with its associated username, we see the correct response below.

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generatedHowever, when an incorrect API Key is passed with an unmatching username we see the response instead informing us of a bad request.

|  |  |  |  |
| --- | --- | --- | --- |
| **API Function** | **Request Type** | **Parameter** | **Pass** |
| Get all cars registered on database | GET | &function=cars | **Yes** |
| Get all parking logs from database for location | GET | function = locationlogs | **Yes** |
| Get logs for all current parked cars at location | GET | function = parkedlogs | **Yes** |
| Get all parking logs from database | GET | function = allparking | **Yes** |
| Get all currently parked guest logs from database | GET | function = allguestparking | **Yes** |
| Get all CO2 Rates and their tiers from database | GET | function = co2rates | **Yes** |
| Get all weekly Rates and their tiers from database | GET | function = weeklyrates | **Yes** |
| Get logs for all guest parking from database | GET | function = guestparking | **Yes** |
| Get for all users from database | GET | function = getusers | **Yes** |
| Get for all parking locations from database | GET | function = locations | **Yes** |
| Delete a specific location | DELETE | NA | **Yes** |
| Update parking rate for specific location | PUT | function = ratechange | **Yes** |
| Update CO2 rate for specific tier | PUT | function = co2ratechange | **Yes** |
| Update the CO2 level at which a specific tier ends at | PUT | function = co2endtier | **Yes** |
| Update the CO2 level at which a specific tier starts at | PUT | function = co2starttier | **Yes** |
| Update the weekly tally at which a specific tier ends at | PUT | function = weeklyendtier | **Yes** |
| Update the weekly tally at which a specific tier starts at | PUT | function = weeklystarttier | **Yes** |
| Update weekly rate for a specific tier | PUT | function = weeklyratechange | **Yes** |
| Update capacity for a specific location | PUT | function = capacitychange | **Yes** |

**Function 1 –** Get all cars registered on database.

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function=cars |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

Text

Description automatically generated

A JSON schema returning all cars registered on the system with the fields:

id - string

username - string

isParked – int (1)

make - string

colour - string

co2Class - string

timeParked - int

totalDue - string

**Function 2 –** Get all parking logs from database for location

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function=locationlogs |
| location=<LOCATION> |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

A picture containing text

Description automatically generated

A JSON schema returning all cars parked to have parked at a particular location in history

parkingID - string

location - string

inTime – int

outTime - int

userName - string

id - string

cost - string

exitID - string

**Function 3 –** Get all parking logs from database for location

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function= parkedlogs |
| location=<LOCATION> |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

A JSON schema returning all cars currently parked at a location specified.

parkingID - string

location - string

inTime – int

userName - string

id - string

cost - string

exitID - string

**Function 4 –** Get all parking logs from database

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function= allparking |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

Text

Description automatically generated

A JSON schema returning all cars currently parked at a location specified.

parkingID - string

location - string

inTime – int

outTime - int

userName - string

id - string

cost - string

exitID - string

**Function 5 –** Get all parking logs from database

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function= guestparking |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

Text

Description automatically generated

A JSON schema returning all cars currently parked at a location specified.

parkingID - string

location - string

inTime – int

outTime - int

userName - string

id - string

cost - string

exitID - string

**Function 6 –** Get all co2 rates from database

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function= co2rates |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

Text

Description automatically generated

A JSON schema returning the CO2 Rates of Ecopark

charge – float

tier – int

tierStart – int

tierEnd - int

**Function 7 –** Get all weekly rates from database

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function= weeklyrates |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

Text

Description automatically generated with low confidence

A JSON schema returning the weekly Rates of Ecopark

charge – float

tier – int

tierStart – int

tierEnd - int

**Function 8 –** Get all registered guests from the database

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function= guestparking |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

A JSON schema returning the registered guest cars of Ecopark

id – string

hours – int

co2Class – int

isParked – int

userName – string

location - string

Text

Description automatically generated

**Function 9 –** Get all registered users from the database

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function= getusers |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

Text

Description automatically generated

A JSON schema returning the registered users of Ecopark

userName – string

email – string

fname – string

lname – string

postcode – string

verified – int

grandTotal - string

**Function 10 –** Get all registered locations from the database.

|  |  |
| --- | --- |
| Parameters | adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function= locations |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

Graphical user interface, website, timeline

Description automatically generated

A JSON schema returning the registered locations of Ecopark

location – string

**Function 11 –** Delete a specific location from the database.

|  |  |
| --- | --- |
| Parameters | Request Method = DELETE |
| adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| location= <NAME OF LOCATION TO DELETE> (string) |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |



A response indicating the successful deletion of the location from database if match found.

**Function 12 –** Update parking rate for specific location

|  |  |
| --- | --- |
| Parameters | Request Method = PUT |
| adminusername=<ADMIN USERNAME> |
| API-KEY=<YOUR API KEY> |
| function= ratechange |
| location=<LOCATION TO CHANGE> (string) |
| rate=<NEW RATE> (int) |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |

A response indicating the successful change of the specific rate for that location



**Function 13 –** Update CO2 rate for a specific tier

|  |  |
| --- | --- |
| Parameters | Request Method = PUT |
| adminusername= <ADMIN USERNAME> |
| API-KEY= <YOUR API KEY> |
| function= co2ratechange |
| tier = <CO2 Tier TO CHANGE> (int) |
| rate= <NEW RATE> (int) |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |



A response indicating the successful change of the CO2 rate for that tier.

**Function 14 –** Update CO2 start tier for a specific tier

|  |  |
| --- | --- |
| Parameters | Request Method = PUT |
| adminusername= <ADMIN USERNAME> |
| API-KEY= <YOUR API KEY> |
| function= co2starttier |
| tier = <CO2 Tier TO CHANGE> (int) |
| tierstart= <NEW START TIER> (int) |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |



A response indicating the successful change of the CO2 start of tier value for that tier.

**Function 15 –** Update CO2 end tier for a specific tier

|  |  |
| --- | --- |
| Parameters | Request Method = PUT |
| adminusername= <ADMIN USERNAME> |
| API-KEY= <YOUR API KEY> |
| function= co2endtier |
| tier= <CO2 Tier TO CHANGE> (int) |
| tierend= <NEW TIER END> (int) |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |



A response indicating the successful change of the CO2 end of tier value for that tier.

**Function 16 –** Update weekly end tier for a specific tier

|  |  |
| --- | --- |
| Parameters | Request Method = PUT |
| adminusername= <ADMIN USERNAME> |
| API-KEY= <YOUR API KEY> |
| function= weeklyendtier |
| tier = <WEEKLY TIER TO CHANGE> (int) |
| tierend= <NEW TIER END> (int) |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |



A response indicating the successful change of the weekly end of tier value for that tier.

**Function 17 –** Update weekly start tier for a specific tier

|  |  |
| --- | --- |
| Parameters | Request Method = PUT |
| adminusername= <ADMIN USERNAME> |
| API-KEY= <YOUR API KEY> |
| function= weeklystarttier |
| tier = <WEEKLY TIER TO CHANGE> (int) |
| tierstart= <NEW TIER START> (int) |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |



A response indicating the successful change of the weekly start of tier value for that tier.

**Function 18 –** Update weekly rates for a specific tier

|  |  |
| --- | --- |
| Parameters | Request Method = PUT |
| adminusername= <ADMIN USERNAME> |
| API-KEY= <YOUR API KEY> |
| function= weeklyratechange |
| tier = <WEEKLY TIER TO CHANGE> (int) |
| charge= <NEW RATE FOR TIER> (int) |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |



A response indicating the successful change of the weekly start of tier value for that tier.

**Function 19 –** Update capacity of a specific location

|  |  |
| --- | --- |
| Parameters | Request Method = PUT |
| adminusername= <ADMIN USERNAME> |
| API-KEY= <YOUR API KEY> |
| function= capacitychange |
| location = <NAME OF LOCATION TO CHANGE> (string) |
| Capacity = <NEW CAPACITY FOR LOCATION> (int) |

|  |  |
| --- | --- |
| **Expected Response** | **Actual Response** |



A response indicating the successful change of the capacity for that location.

**Error Scenarios**

In certain cases when incorrect functions are passed to the API the API will detect this and inform the user for debugging purposes.

**Incorrect function name will return this response from the API**



In other cases where no admin username or API Key are passed at all the API will also detect this and inform the user.



**Postman Tests**

We can also use postman and JS functions to verify our responses from the API in order to further troubleshoot issues using the Tests tab provided.

**Graphical user interface, text, application, chat or text message

Description automatically generated**Graphical user interface, text, application, email

Description automatically generated**Validating data types**

**Graphical user interface, text, application, email

Description automatically generatedValidating Error Responses**

**Graphical user interface, text, application, chat or text message

Description automatically generated**