[API DOCUMENTATION FOR LILYPOND TICKETING SOLUTION]

BY: DAGOGO CLINTON URANTA

Contents

1. Overview: 2

2. Methods 2

2.1. For processing the trucks ENTERING the park 2

2.2. For processing the trucks EXITING the park 5

1. Overview:

This API documentation provides access to the ticketing system (for programmers) to enable generation of entry and exit tickets for use at Lilypond park facility. It is also to serve as a source of data, for management and recording purposes.

2. Methods

The Documentation consists of two APIs, one to generate an entry ticket and one to generate an exit ticket, the instructions below are a guide on how to use them.

2.1. For processing the trucks ENTERING the park

1. **Type of Request : POST**

URL link : https://flacscarpark.herokuapp.com/api/users/parkenter

Usage:

The request accepts an object with the properties

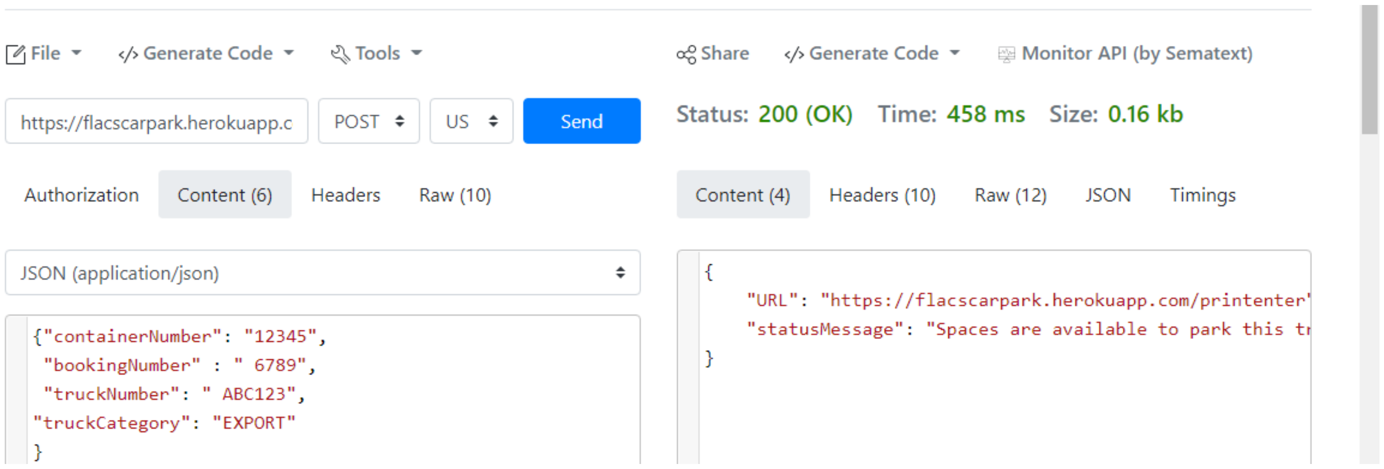
containerNumber: *//this is to be a string data type ,*

bookingNumber: *//this is to be a string data type ,*

truckNumber: *//this is to be a string data type ,*

truckCategory: *//this is to be a string data type ,*

As shown below (using the Postman API)



Status: 200 (OK) Time: 458 ms Size: 0.16 kb

```
{"containerNumber": "12345",
 "bookingNumber" : " 6789",
 "truckNumber": " ABC123",
"truckCategory": "EXPORT"
}
```

```
{
    "URL": "https://flacscarpark.herokuapp.com/printenter"
    "statusMessage": "Spaces are available to park this tr
}
```

N.B   the key “**truckCategory”,** can only accept the following values to enable allocation of parking zones and slot

“EXPORT”
“FLAT BED ENL/EKO”
“FLAT BED APMT“
“EXCEPTIONS”

EXAMPLE USAGE:

(USING JAVASCRIPT PROGRAMMING LANGUAGE AND FETCH PROGRAMMING LIBRARY):

PLEASE NOTE, THE WORDS AFTER THE DOUBLE FORWARD SLASH “//”, ARE COMMENTS TO ASSIST IN READING THE CODE

```
//assigning an object  <in the frontend>

const truckInformation = {

            containerNumber:  5678 ,

             bookingNumber:  1234 ,

            truckNumber:  1000 ,

           truckCategory:  EXPORT ,  // as this is sensitive to spaces, please be sure to type this in exactly

}


//sending the object as an argument in a POST request

//async-await syntax is used below  (Javascript ES6)

  const async function(truckInformation) {

  const options =  {

  method :’POST’,

  body:JSON.stringify(truckInformation),


headers:{

“content-type”: “application/json”

}
```

}

const url = https://flacscarpark.herokuapp.com/api/users/parkenter

//the API link (above)

const response = await fetch(url, options) *//using the url and options variables we just made*

return response *// end of the function*

}

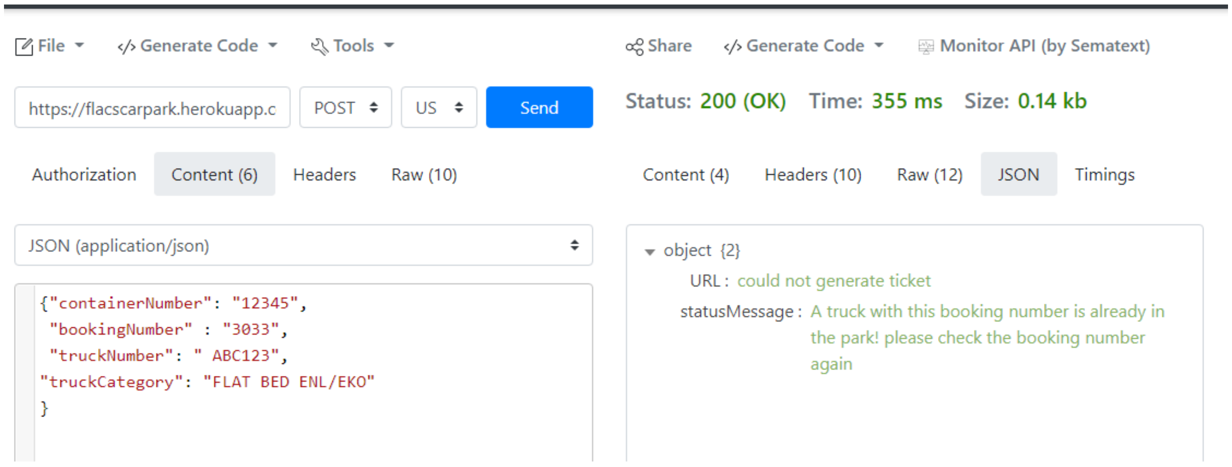
// the response returns an object with a URL property for locating the generated ticket and a status message property (which are both strings), indicating the availability of spaces in the park such as in the example below:

{URL: 'could not generate a URL'

statusMessage: "the truck with the booking number you entered is already in the park "

}

As shown below



Status: 200 (OK) Time: 355 ms Size: 0.14 kb

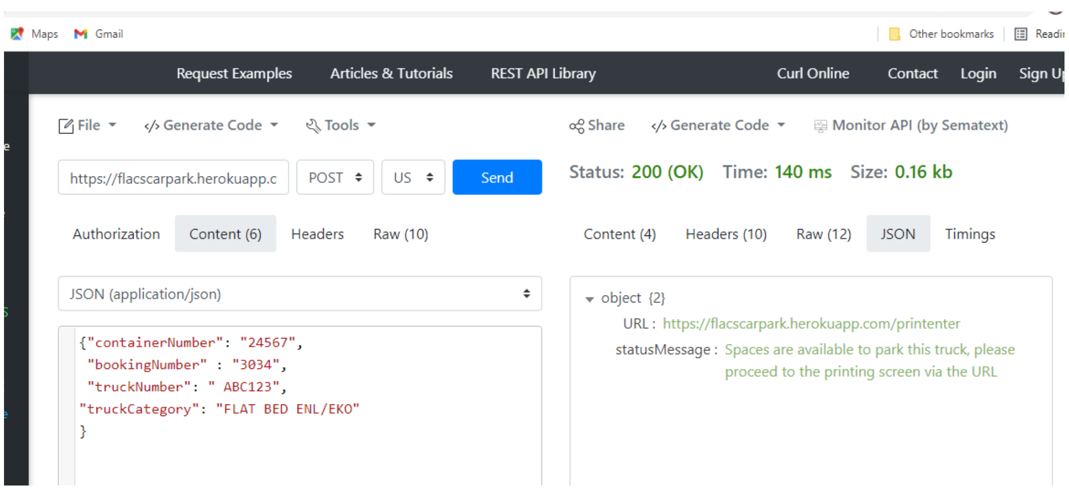
```
{"containerNumber": "12345",
 "bookingNumber" : "3033",
 "truckNumber": " ABC123",
"truckCategory": "FLAT BED ENL/EKO"
}
```

```
object {2}
    URL : could not generate ticket
statusMessage : A truck with this booking number is already in
                the park! please check the booking number
                again
```

Or

*{URL: '*https://flacscarpark.herokuapp.com/ printenter'

statusMessage: "this truck is cleared for entrance, please proceed to the generated URL to print your ticket" }



https://flacscarpark.herokuapp.c POST US

Status: 200 (OK) Time: 140 ms Size: 0.16 kb

JSON (application/json)

```
{"containerNumber": "24567",
 "bookingNumber" : "3034",
 "truckNumber": " ABC123",
"truckCategory": "FLAT BED ENL/EKO"
}
```

```
object {2}
  URL : https://flacscarpark.herokuapp.com/printenter
  statusMessage : Spaces are available to park this truck, please
                  proceed to the printing screen via the URL
```

NOTE: UPON APPROVAL , THE APIs WILL BE ALTERED TO ONLY ACCEPT REQUESTS FROM YOUR WEBSITE, USING CORS (*Cross Origin Resource Sharing***) SETTINGS.**

2.2. For processing the trucks EXITING the park

Type of Request : POST

URL link : https://flacscarpark.herokuapp.com/api/users/parkexit

Usage:

The request accepts an object with the property :

bookingNumber: *//this is to be a string data type ,*

EXAMPLE USAGE, USING JAVASCRIPT PROGRAMMING LANGUAGE AND FETCH PROGRAMMING LIBRARY:

//assigning an object (in the frontend)

const truckInformantion = {

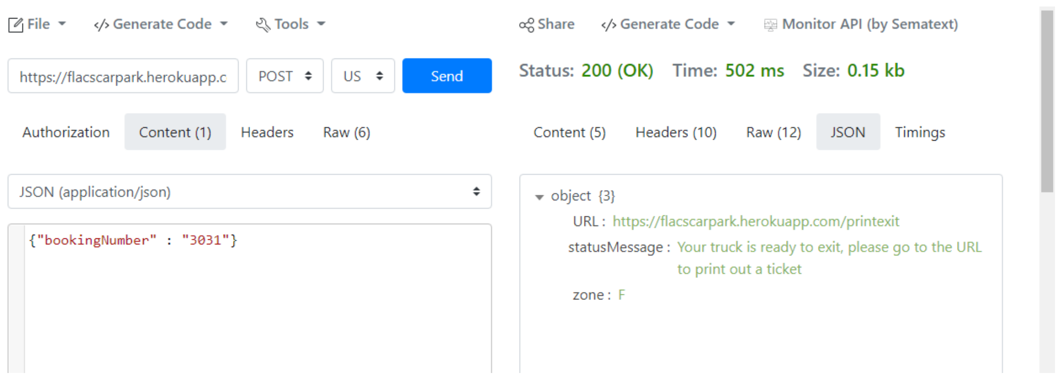
bookingNumber: ‘ 1234’ *//must be a string*

}

```
//sending the object as an argument in a POST request

//async-await syntax is used below  (Javascript ES6)

const async function(truckInformation) {

 const options =  {

  method :’POST’,

body:JSON.stringify(truckInformation),



headers:{

“content-type”: “application/json”

}

   }

const url = https://flacscarpark.herokuapp.com/api/users/parkexit

//the API link (above)

const response = await fetch(url, options)   //using the url and options variables we just made


return response    // end  of the function

}
```

// the response returns an object with a URL property, for locating the generated ticket and a status message property relating to the truck in question such as in the example below:

```
{URL: ‘could not generate a URL’

 statusMessage:  “the truck with the booking number you entered is not currently in the park ”

}
```



https://flacscarpark.herokuapp.c POST US Send

JSON (application/json)

```
{"bookingNumber" : "3031"}
```

Status: 200 (OK) Time: 502 ms Size: 0.15 kb

```
object {3}
  URL : https://flacscarpark.herokuapp.com/printexit
  statusMessage : Your truck is ready to exit, please go to the URL to print out a ticket
  zone : F
```

Or

*{URL: '*https://flacscarpark.herokuapp.com/ printexit'

statusMessage: "this truck has been processed and is ready to exit the park "

}

NOTE: UPON APPROVAL , THE APIs WILL BE ALTERED TO ONLY ACCEPT REQUESTS FROM YOUR WEBSITE, USING CORS (*Cross Origin Resource Sharing***) SETTINGS.**