Taxi Booking System

API Interface Specification

Created by: Tan Si Ling
Latest as of 28th December 2017

Table of Contents

Introduction

This document provides a list of APIs available for Taxi Booking System. These APIs are called using HTTP POST method. “Book a Taxi” API requires authorization with a testing id and secret provided in the request. Request and responses for these APIs are in JSON format.

API Services

The table below shows the API endpoints available for this Taxi Booking System.

S/No	Name of API Services	Description	HTTP Method	URL (Endpoint of API)
1	Book a Taxi	1. Picks a nearest available car to passenger’s location. 2. Returns total time for car to bring passenger to their destination.	POST	http://TaxiCustomerA:CustA123@localhost:5000/api/book
2	Increment Time Stamp	Increments the system time stamp by 1 time unit.	POST	http://localhost:5000/api/tick
3	Reset Taxi System	Reset all cars data back to initial state regardless of cars that are currently booked.	POST	http://localhost:5000/api/reset

Request Parameters | Book a Taxi

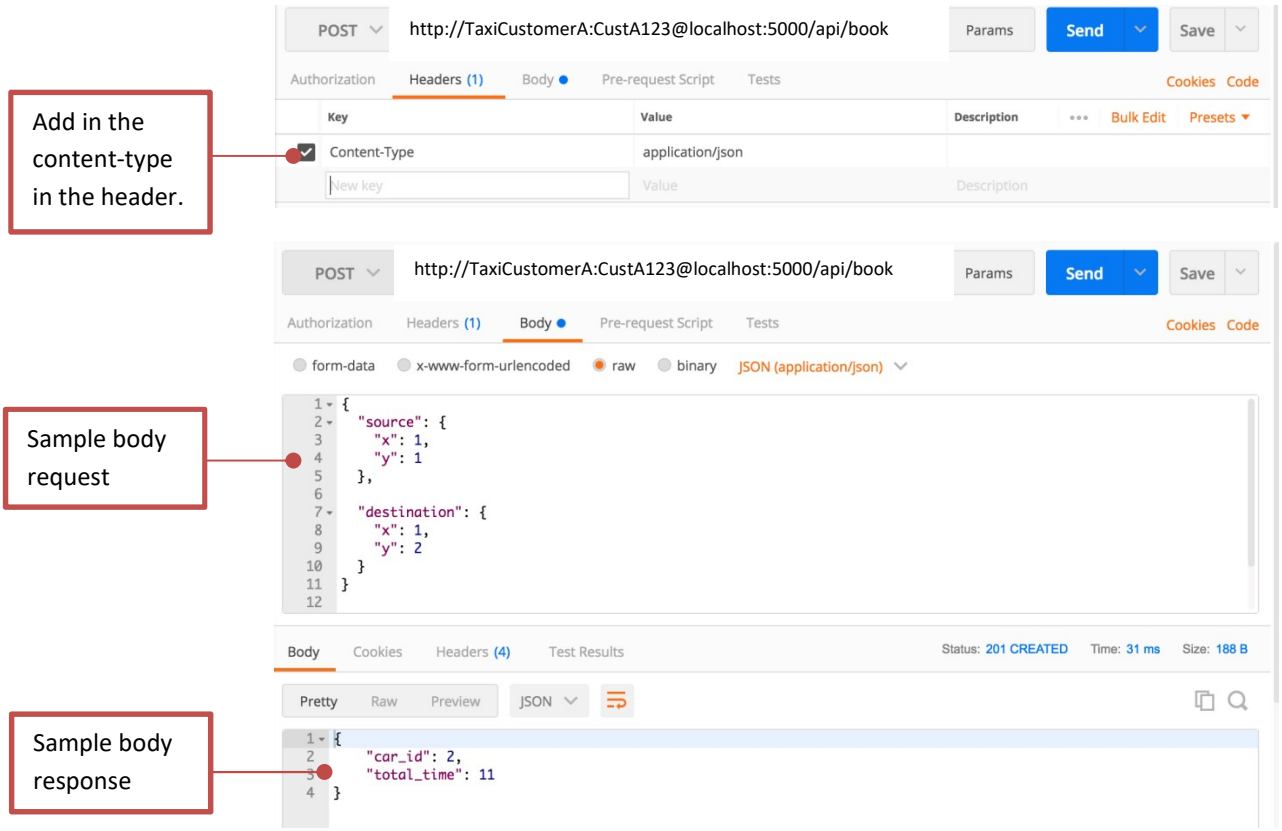
Parameter	Type	Description
source	**object**	
x	integer	Passenger's initial position x1 coordinate
y	integer	Passenger's initial position y1 coordinate
destination	**object**	
x	integer	Passenger's destination position x2 coordinate
y	integer	Passenger's destination position y2 coordinate

Response Parameters | Book a Taxi

HTTP Status Code: 200 indicates that the HTTP call succeeded.
HTTP Status Code: 4XX/5XX indicates that the HTTP call failed.

Parameter	Type	Description
car_id	integer	Car ID that would fetch passenger to destination.
total_time	integer	Total time taken for the car to reach passenger and pick the passenger up from source to destination.

Sample API Request and Response | Book a Taxi



Testing | Running Program Locally

This section provides detailed instructions on how to run the API locally.

Pre-requisities

1. Download Postman (https://www.getpostman.com/postman)
2. Unzip the source code (TaxiServiceApi.zip) and place it into your local path directory.
3. Install Python (my version 2.7.5)
4. Install virtualenv (https://pypi.python.org/pypi/virtualenv)

```
$ cd TaxiServiceApi

$ virtualenv flask

$ pip install flask
```

Run the Program

1. Navigate into your TaxiServiceApi folder and run app.py.

```
$ python app.py

 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

2. Launch your postman and type `http://TaxiCustomerA:CustA123@localhost:5000/api/book` to fire a request.

Note: For authorization of "Book a Taxi" API, please use the following credentials.

Client ID: TaxiCustomerA
Client Secret: CustA123

Run the Unit Test

1. Navigate into your TaxiServiceApi folder and modify app.py. **Remember to revert back after performing unit test.**

```
64  @app.route('/api/tick', methods=['POST'])
65  def incrementSystemTime():
66      global systemTime
67      systemTime += 1
68
69      #if the car has reached its destination, set it to be available
70      #At the next unit time, do we free up this car
71      for car in cars:
72          if(car['bookedTillTime'] + 1 == systemTime):
73              car['booked'] = False
74
75      #For unit test
76      return systemTime
77      #return jsonify({'tick': systemTime})
78
79
80  @app.route('/api/reset', methods=['POST'])
81  def resetTaxiSystem():
82      global systemTime
83      systemTime = 0
84
85      for car in cars:
86          car['x'] = 0
87          car['y'] = 0
88          car['booked'] = False
89          car['bookedTillTime'] = -1
90
91      #For unit test
92      return systemTime
93      #return jsonify({'tick': systemTime})
94
```

Comment off the line 77 and uncomment line 76

Comment off the line 93 and uncomment line 92

2. Run pytest

```
$ pytest -s tests/test_app.py

============================= test session starts ==============================

platform darwin -- Python 2.7.5, pytest-3.3.1, py-1.5.2, pluggy-0.6.0

rootdir: <<your root directory>>, inifile:

collected 4 items


tests/test_app.py ....                                                  [100%]

========================== 4 passed in 0.08 seconds ===========================
```