TripPy API

User Guide and API Specification

Version 1.0.0

Table of Contents

1.0 Introduction

Travel information apps are very popular these days, as more and more travelers rely on their mobile devices or web applications to find information about places they visit. But where do you go for content? It can be cumbersome to have to deal with multiple APIs for weather, hotels, restaurants, events, and other sources for the information your app users need. Each source has its own interface, authentication scheme, terms of use, and so on.

That's where TripPy API can help. We've already done the work of integrating the most useful APIs for travelers. We then present them in a single, RESTful API, with no additional configuration needed. You just create a user account for authentication purposes, then consume the API endpoints for the requested information, such as:

- Weather, including current conditions and a five-day forecast
- Restaurants, including cuisine, price range, and rating
- Concerts or other events happening in the area
- Available hotels in the area, with an additional call to retrieve the address for a specific hotel

Calls to TripPy API's endpoints can pass in a single ZIP code as a parameter. If no parameter is used, the API attempts to determine the client's location via its IP address. At this time, only locations within the United States are supported. All responses from TripPy API endpoints, including error messages, are in JSON. Information on integrating TripPy API into other applications can be found in Appendix A.

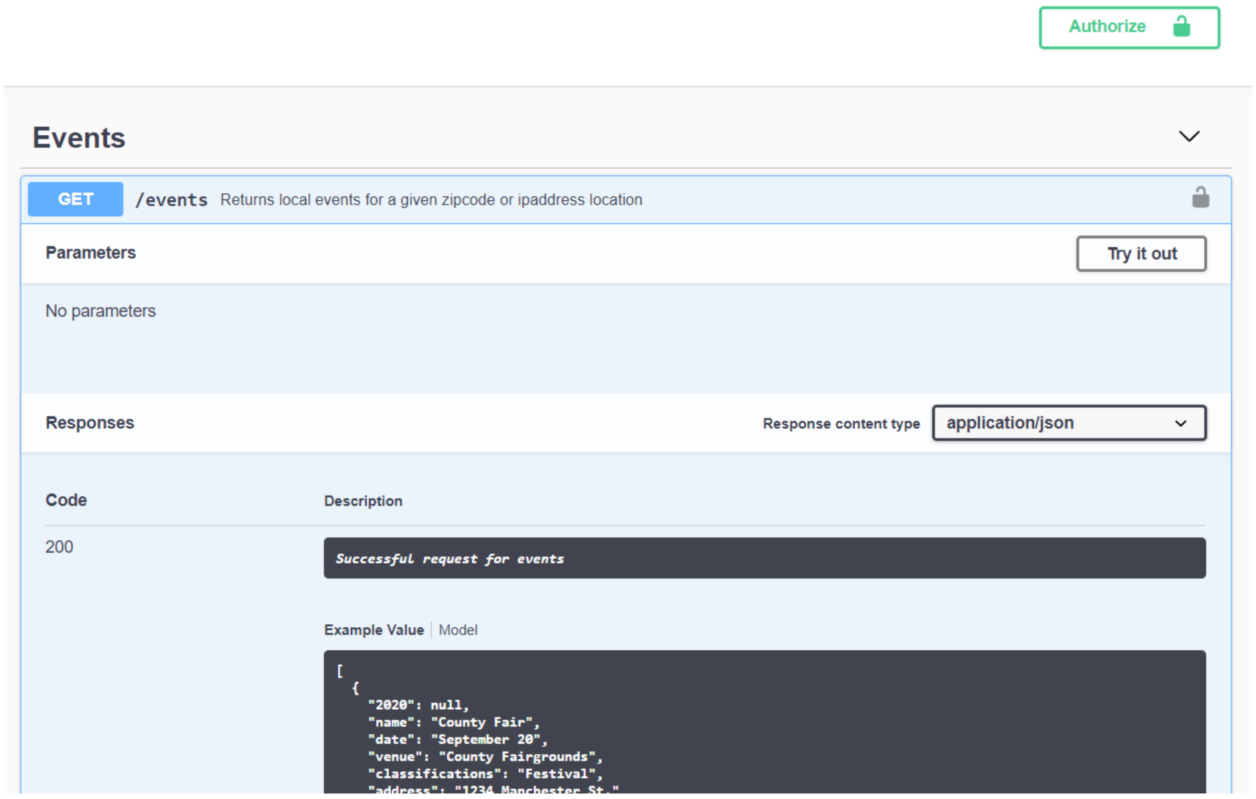
Ready? Let's travel!

2.0 Testing the API

If you wish to try out TripPy API's features without doing any programming, you can easily do so using several different methods.

2.1 Via Swagger

Perhaps the easiest way to test the API is through its Swagger page, which can be found at https://csc-478-travel-api.herokuapp.com/. Not only does Swagger provide concise documentation for the API, it allows you to try out the requests for each endpoint and see the results.



/events Returns local events for a given zipcode or ipaddress location

No parameters

200 Successful request for events

```
[
  {
    "2020": null,
    "name": "County Fair",
    "date": "September 20",
    "venue": "County Fairgrounds",
    "classifications": "Festival",
    "address": "1234 Manchester St."
```

Clicking "Try it out" will execute a sample request to the API for that endpoint. An access token is still required to make requests via Swagger; you can include one by clicking "Authorize" and entering an access token you have already received (see section 3.0 for more details on creating a user account and using access and refresh tokens).

2.2 Via the command line

You can use cURL, a command-line utility available on Windows 10, to make requests to the API endpoints. The requests are in a format similar to this:

```
curl -X GET https://csc-478-travel-api.herokuapp.com/weather?zipcode=80123 -H
"authorization: Bearer <access_token>"
```

where <access_token> is the authentication token provided by TripPy API when a user is logged in. Creating an account, logging in, and managing access and refresh tokens are described in section 3.0.

NOTE: Command line examples assume using cURL in the Windows command line processor (cmd.exe). If you use a different shell such as Git bash or PowerShell, the syntax may be slightly different. Also be aware that PowerShell redirects cURL to its internal cmdlet `Invoke-WebRequest`, which works like cURL but uses a different syntax.

2.3 Via an API testing service or application

If you use an API testing service such as Postman, or other services or software that can test RESTful APIs, they can also be used to test endpoints in TripPy API. Please refer to the documentation for the testing service or application you are using for details.

3.0 Logging in and using access and refresh tokens

To help secure the API and protect it against abuse, TripPy API uses a simple authentication scheme that requires a user to login and use an access token when making calls to the API endpoints.

3.1 Creating a user

To use TripPy API, you need a user account. To create one, use cURL or another method to send a request to the API's `/registration` endpoint (formatted for clarity):

```
curl -X POST https://csc-478-travel-api.herokuapp.com/registration
  -H "content-type: application/json"
  -d "{
        \"username\": \"apiuser\",
        \"password\": \"please_use_a_strong_password\"
     }"
```

NOTE: if using cURL in the Windows command line processer (cmd.exe), double quotation marks within the request text must be escaped with a backslash as shown above.

If successful, the API will respond with:

```
{
  "message": "User apiuser was created",
  "access_token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1NzU3NjMyODUsIm5iZiI6MTU3NTc
2MzI4NSwianRpIjoiNTI0ZmZiNDUtODM2NS00NGE4LWI5ZGItM2ZlZDRlZDQzZGYzIiwiZXhwIjox
NTc1NzY0MTg1LCJpZGVudGl0eSI6ImFwaXVzZXIiLCJmcmVzaCI6ZmFsc2UsInR5cGUiOiJhY2Nlc
3MifQ.evPSRm5kPE4b27uozTEhHgU6lxLpJZo6dXxOQgLzkU0",
  "refresh_token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1NzU3NjMyODUsIm5iZiI6MTU3NTc
2MzI4NSwianRpIjoiZWMxMTgzODAtZTg5Yi00OWNjLTliMDQtOTQ1NTdkNmMyMTY1IiwiZXhwIjox
NTc4MzU1Mjg1LCJpZGVudGl0eSI6ImFwaXVzZXIiLCJ0eXBlIjoicmVmcmVzaCJ9.D93AwqKrmgdc
ystLkUsx2sJYbmz14bIEWfnBKcfJGu0"
}
```

Note that you receive two tokens when successfully registering a user: an access token that can be used for making API requests, and a refresh token that can be used to receive a new access token when the old one expires.

3.2 Logging into the API

You do not have to login immediately after creating a new account, as you are already authenticated by the API. You can just use the access token that is provided to you. On subsequent sessions, however, you will need to login first:

```
curl -X POST https://csc-478-travel-api.herokuapp.com/login
    -H "content-type: application/json"
    -d "{
          \"username\":\"apiuser\",
          \"password\":\"You_used_a_strong_password_right?\"
        }"
```

You will receive a new access token and refresh token:

```
{
  "message": "Logged in as apiuser",
  "access_token":
```

```
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1NzU3NjQwMDIsIm5iZiI6MTU3NTc
2NDAwMiwianRpIjoiNzNkOWZjNGEtM2ZlNi00MzQyLWI3YjMtOTIwMjgyYmJmMzhlIiwiZXhwIjox
NTc1NzY0OTAyLCJpZGVudGl0eSI6ImFwaXVzZXIiLCJmcmVzaCI6ZmFsc2UsInR5cGUiOiJhY2Nlc
3MifQ.nSIy8j108kAjTIv2aPr99kySyxV0bC3vcQgBq3BSVA0",
  "refresh_token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1NzU3NjQwMDIsIm5iZiI6MTU3NTc
2NDAwMiwianRpIjoiZDVjNjc3MDYtNmVkMi00ZWM1LTkxODktNjg3ZDI2ZjNkOTIwIiwiZXhwIjox
NTc4MzU2MDAyLCJpZGVudGl0eSI6ImFwaXVzZXIiLCJ0eXBlIjoicmVmcmVzaCJ9.6L_R7uc4MR9K
IndG4EdxhiuqJ-fvVpYVhv1OzK901l8"
}
```

You can then send requests to the API.

3.3 Logging out

Logging out revokes your access token, so it can no longer be used for requests. To logout, send the following:

```
curl -X POST https://csc-478-travel-api.herokuapp.com/logout/access
   -H "Authorization: Bearer <access_token>"
```

Response:

```
{
  "message": "Access token has been revoked"
}
```

For optimal security, you should also revoke your refresh token; in this case, you would pass in your refresh token in the header:

```
curl -X POST https://csc-478-travel-api.herokuapp.com/logout/refresh
   -H "Authorization: Bearer <refresh_token>"
```

Response:

```
{
  "message": "Refresh token has been revoked"
}
```

3.4 Refreshing the access token

Access tokens for TripPy API expire after 15 minutes; after that, you'll need a new one. You can either login again, or you can request a new access token using your refresh token:

```
curl -X POST https://csc-478-travel-api.herokuapp.com/token/refresh
   -H "Authorization: Bearer <refresh_token>"
```

Response:

```
{
  "access_token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1NzU3NjQwMDIsIm5iZiI6MTU3NTc
2NDAwMiwianRpIjoiNzNkOWZjNGEtM2ZlNi00MzQyLWI3YjMtOTIwMjgyYmJmMzhlIiwiZXhwIjox
NTc1NzY0OTAyLCJpZGVudGl0eSI6ImFwaXVzZXIiLCJmcmVzaCI6ZmFsc2UsInR5cGUiOiJhY2Nlc
3MifQ.nSIy8j108kAjTIv2aPr99kySyxV0bC3vcQgBq3BSVA0"
}
```

4.0 The API endpoints

NOTE: for all request examples, replace `<access_token>` with the access token you receive from the API following a successful login.

4.1 /weather

The `/weather` endpoint returns current weather conditions for a given location.

```
curl -X GET https://csc-478-travel-api.herokuapp.com/weather?zipcode=80123 -H
"Authorization: Bearer <access_token>”
```

Parameter (optional):

`zipcode:` a valid five-digit ZIP code. If left blank, the API will attempt to determine the ZIP code via a geolocational lookup of the client’s current IP address. If this is unsuccessful, an error message will display: `unknown location for IP: <ip_address>`.

Response:

```
{
  "temperature" : 38.08,
  "city" : "Littleton",
  "date" : "2019-12-08 04:06:21",
  "description" : "clear sky"
}
```

If an error occurs (from an invalid ZIP code, for example):

```
{
  "error" : "No weather information found"
}
```

4.2 /fiveday

The `/fiveday` endpoint returns a five-day weather forecast for a given location.

```
curl -X GET https://csc-478-travel-api.herokuapp.com/fiveday?zipcode=80123 -H
"Authorization: Bearer <access_token>”
```

Parameter (optional):

`zipcode:` a valid five-digit ZIP code. If left blank, the API will attempt to determine the ZIP code via a geolocational lookup of the client’s current IP address. If this is unsuccessful, an error message will display: `unknown location for IP: <ip_address>`.

The response will be a list of weather forecasts at three-hour intervals for each of the next five days following the current date:

```
[
  {
    "description" : "broken clouds",
    "temperature" : 35.58,
    "time" : "2019-12-08 06:00:00",
    "city" : "Littleton"
  },
  {
    "description" : "overcast clouds",
    "temperature" : 34.32,
    "time" : "2019-12-08 09:00:00",
    "city" : "Littleton"
  },
  {
    "time" : "2019-12-08 12:00:00",
    "city" : "Littleton",
    "description" : "overcast clouds",
    "temperature" : 32.9
  }, . . .
]
```

If an error occurs (from an invalid ZIP code, for example):

```
{
  "error" : "No weather information found"
}
```

4.3 /restaurants

The `/restaurants` endpoint returns a list of nearby restaurants for a given location.

```
curl -X GET https://csc-478-travel-api.herokuapp.com/restaurants?zipcode=80123 -H
"Authorization: Bearer <access_token>"
```

Parameter (optional):

`zipcode:` a valid five-digit ZIP code. If left blank, the API will attempt to determine the ZIP code via a geolocational lookup of the client's current IP address. If this is unsuccessful, an error message will display: `unknown location for IP: <ip_address>`.

The response will be a list of restaurants, including their name, address, phone number, style of cuisine, price range, and rating:

```
[
   {
      "address" : "2707 West Main Street 80120",
      "name" : "The Melting Pot of Littleton",
      "rating" : "4.3",
      "price_scale" : 3,
      "phone" : "(303) 794-5666",
      "cuisine" : "American, French"
   },
   {
      "cuisine" : "Italian, Pizza",
      "phone" : "(303) 972-1011",
      "rating" : "4.0",
      "price_scale" : 2,
      "name" : "Virgilio's Pizzeria & Wine Bar",
      "address" : "10025 West San Juan Way, Littleton 80127"
   },
   {
      "name" : "Wild Ginger",
      "address" : "399 West Littleton Boulevard, Littleton 80120",
      "rating" : "4.7",
      "price_scale" : 2,
      "cuisine" : "Asian, Thai",
      "phone" : "(303) 794-1115"
   }, . . .
]
```

If an error occurs (from an invalid ZIP code, for example):

```
{
  "error" : "No restaurant information found"
}
```

4.4 /events

The `/events` endpoint returns a list of upcoming concerts and other events for a given location.

```
curl -X GET https://csc-478-travel-api.herokuapp.com/events?zipcode=98101 -H
"Authorization: Bearer <access_token>"
```

Parameter (optional):

`zipcode:` a valid five-digit ZIP code. If left blank, the API will attempt to determine the ZIP code via a geolocational lookup of the client's current IP address. If this is unsuccessful, an error message will display: `unknown location for IP: <ip_address>`.

The response will be a list of events, including their name, date, venue name and address, and a sub-list of "classifications" that include the type, genre, and sub-genre of the event:

```
[
   {
      "classifications" : [
         "Music",
         "Rock",
         "Alternative Rock"
      ],
      "name" : "Patti Smith",
      "address" : "911 Pine St",
      "date" : "2020-03-11",
      "venue" : "Paramount Theatre"
   },
   {
      "date" : "2020-03-25",
      "venue" : "Paramount Theatre",
      "classifications" : [
         "Music",
         "Rock",
         "Pop"
      ],
      "name" : "Melanie Martinez: The K-12 Tour",
      "address" : "911 Pine St"
   }, . . .
]
```

If an error occurs (from an invalid ZIP code, for example):

```
{
  "error" : "No event information found"
}
```

4.5 /hotels

The `/hotels` endpoint returns a list of hotels for a given location.

```
curl -X GET https://csc-478-travel-api.herokuapp.com/hotels?zipcode=80123 -H
"Authorization: Bearer <access_token>"
```

Parameter (optional):

`zipcode:` a valid five-digit ZIP code. If left blank, the API will attempt to determine the ZIP code via a geolocational lookup of the client's current IP address. If this is unsuccessful, an error message will display: `unknown location for IP: <ip_address>`.

The response will be a list of hotels, including their name, rating, and a reference id ("xid") that can be used to query for the hotel's address (see section 4.6):

```
[
   {
      "xid" : "W226297599",
      "rating" : 1,
      "name" : "Hampton Inn Denver-Southwest/Lakewood"
   },
   {
      "xid" : "W51454988",
      "name" : "Comfort Inn & Suites",
      "rating" : 1
   },
   {
      "xid" : "W213192951",
      "rating" : 2,
      "name" : "extended stay america - denver - lakewood south"
   }, . . .
]
```

If an error occurs (from an invalid ZIP code, for example):

```
{
  "error" : "No hotel information found"
}
```

4.6 /hotel

The `/hotel` endpoint returns the address for a specific hotel.

```
curl -X GET https://csc-478-travel-api.herokuapp.com/hotel?xid=W226297599 -H
"Authorization: Bearer <access_token>"
```

Parameter (required):

`xid:` the ID code of a specific hotel from the results returned by the `/hotels` endpoint (section 4.5). If left blank, the API will return an error: `{ "xid":"This field cannot be left blank" }`

The response will be the specified hotel's address:

```
{
   "city" : "Lakewood",
   "house_number" : "3605",
   "street" : "South Wadsworth Boulevard"
}
```

If an error occurs (from an invalid `xid`, for example):

```
{
  "error" : "No hotel information found"
}
```

5.0 API Specification

5.1 Introduction

5.1.1 Scope Statement

We, the members of The Missing Semicolon, propose to build the following software as our semester project: Location Content Services

An API that will provide several location-based content services, as a way for users to learn about the area they are currently in, or any other area they specify. It can determine location either by receiving a ZIP code as input, or from a user's IP address. It differs from location-based services for mobile devices in that it does not use GPS technology.

Users who may find such a service useful include tourists and people who have just moved to a new city or town in the United States. As an API, the service could also be consumed by other application developers who wish to incorporate location-based content into their own projects. Some examples of the content that may be provided by the service, pending formal requirements, include:

- Local weather information, including five-day forecast
- Local entertainment events
- Travel-related content (hotels, restaurants, etc.)

The API will be hosted on the Heroku cloud platform. As such, it has no particular system requirements to use it, other than an Internet connection. A simple command-line or GUI client can be provided to use as a client to access the services.

The API will be developed using Python. The GUI, if included, will also be built using Python and a framework such as Kivy. The source code will be hosted on GitHub, which will also be used for project management.

5.1.2 Definitions, acronyms, and abbreviations

Local: The area pertaining to a given zip code

API: Application program interface

5.1.3 References

weather API:

- https://openweathermap.org/api

event API:

- https://apilist.fun/api/ticketmaster

restaurant API:

- https://developers.zomato.com/api?ref=apilist.fun

hotel API:

- https://dev.opentripmap.com/

5.2 General Description

5.2.1 Product Perspective

The product will simplify end-user requests to various travel services for cities in the US by acting as the middle-man and reducing/removing the need for the user to register for multiple API tokens. By collecting and bundling related travel APIs, the end-user will have a single source to implement.

The product is also useful in that it will not rely on GPS data but will instead provide information based off of the IP address of the user, or by the user inputting a specific zip code.

5.2.2 Product Functions

The product is an aggregation of multiple travel services that will provide information relating to:

- weather
- restaurants
- events
- hotels

This information will be provided for a given location either through deriving the use's location via the IP address of the user, or by the user providing a US zip code.

5.2.3 User characteristics

As an API, the users are developers who would like to make free, low volume requests to acquire travel and event information for their own products and services without having to register for multiple API tokens.

As its own application, the users are individuals traveling in the US who are unfamiliar in their current city, or locals wishing to have more information about local points of interest and current events.

5.2.4 General constraints

An Internet connection is required to make requests to the API.

5.2.5 Assumptions and Dependencies

Assumes end-user has cURL installed.

5.3 Specific Requirements

5.3.1 User Needs

5.3.1.1 Location Look-Up

As a user, I want the API to find my location by IP address. I want to be able to specify a zip code as well. I want to be informed if this location is invalid.

5.3.1.2 Weather Information

As a user, I want information on the local weather. I want to be able to retrieve a five day forecast. I want the information to include the date, the temperature in Fahrenheit and a description of the weather. I want to be informed if no information is available.

5.3.1.3 Restaurant information

As a user, I want information on local restaurants. I want information to include the restaurant name, address, phone number, cuisine type(s), price scale, and average rating. I want to be informed if no information is available.

5.3.1.4 Event information

As a user, I want information on local, current events. I want to be able to input a date range. I want information to include the event name, type, address, phone number, and price. I want to be informed if no information is available.

5.3.1.5 Hotel information

As a user, I want information on local hotels. I want this information to include the hotel name, address, phone number, average rating, and amenities. I want to be informed if no information is available.

5.3.2 Functional Requirements

Note: Functional requirements retain the original numbering from their source document so they can be referenced against design documents, test cases, and source code documentation.

5.3.2.1 Location Look-up

1.0.0 Each API request shall derive the location from the user's IP address

1.1.0 The user shall be able to enter a zip code

1.2.0 The user should be informed if no information was found

5.3.2.2 Weather Retrieval

2.0.0 The API shall provide information pertaining to local weather

2.1.0 Information shall be returned for the current date

2.2.0 Information shall be available for a five day forecast

2.3.0 Information retrieved shall include temperature in Fahrenheit, a description of the weather, and the date

5.3.2.3 Restaurant Retrieval

3.0.0 The API shall provide information pertaining to local restaurants

3.1.0 Information retrieved should include the restaurant name, address, phone, price ratings, average review rating, and cuisine type(s)

5.3.2.4 Events Retrieval

4.0.0 The API shall provide information pertaining to local events

4.1.0 Information retrieved should include the event name, address, event type, and date

5.3.2.5 Hotels Retrieval

5.0.0 The API shall provide information pertaining to local hotels

5.1.0 Information retrieved should include the hotel name, address, and average review rating

5.3.3 Nonfunctional Requirements

5.3.3.1 User Registration

6.0.0 The API shall require the user to register

6.1.0 The user shall create a username

6.1.1 Usernames shall be unique

6.1.2 The user shall be informed if the username already exists

6.2.0 The user shall create a password

6.2.1 The password shall be encrypted

6.3.0 User data shall be stored in a database

5.3.3.2 User Login

7.0.0 The API shall require the user to login

7.1.0 The user shall be informed if login credentials are invalid

5.3.3.3 Access Token

8.0.0 The API shall generate an access token on user login

8.1.0 The access token will expire in 15 minutes

8.1.1 The user shall be informed when the access token expires

8.2.0 A refresh token shall be generated

8.2.1 The refresh token shall generate a new access token

8.3.0 The user shall be informed of missing or invalid access tokens

5.3.3.4 Data Representation

9.0.0 The data returned by the API shall be in JSON format