**US State Geolocator**

A Python API (web service) that retrieves primarily the State in United States for the Street address that is passed as an input.

**Getting Started**

The code base for this API is pushed into GitHub and is available in the following link.

https://github.com/shyamnk/clutch-code

**Prerequisites/Assumptions:**

1. This API was developed on Windows 10 Home Edition with installations of [Python](https://www.python.org/downloads/) (v3.7.2) and [PostGreSQL](https://www.postgresql.org/download/windows/) 12.
2. Instructions and scripts were created based on the assumption that executions or test runs will be performed in a similar setup in Windows Operating System.
3. [PostGIS](https://postgis.net/source/)3.0 windows based installation to be performed to utilize the Spatial and Geographic objects for PostgreSQL.
4. A valid Google API Key is available to perform this execution or test run. This API uses Google Geocoding API to perform reverse lookup to retrieve latitude and longitude of the street address.
5. All necessary firewall, ports to/from the machine(s) (database, processing server etc) are open to allow back and forth traffic.
6. PostgreSQL database runs on port 5432.
7. All necessary path variables are setup in the environment before execution of this API.

For ex. psql path: C:\Program Files\Postgresql\12\bin as part of the PATH variable

shp2pgsql path: C:\Program Files\Postgresql\12\postgis\bin as part of PATH variable

PYTHONPATH C:\Program Files\Python\bin or as part of PATH variable

1. All necessary python libraries are installed to run the API. This API specifically uses the flask API for python. All the necessary libraries/modules are listed below or in the requirements.txt file.

* logging
* psycogp2
* sqlalchemy
* requests
* flask
* flask\_restful

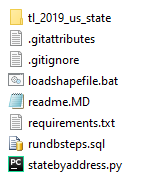
1. Postman (or similar REST API client) installed and available for execution/test runs.

**Software requirements:**

* Python version 3.x
* PostgreSQL 11 or later
* PostGIS 3.x
* [Postman](https://www.getpostman.com/downloads/) (or similar REST API client for testing)

**Code Deployment Instructions:**

Clone or download the repository into a Windows Directory (local path, for ex. D:\codebase). Your repository should have the following files/folders:



**Database Setup:**

The following files are used to perform database setup required for this API. The scripts are to be run in the same order as mentioned below.

1. rundbsteps.sql
2. loadshapefile.bat

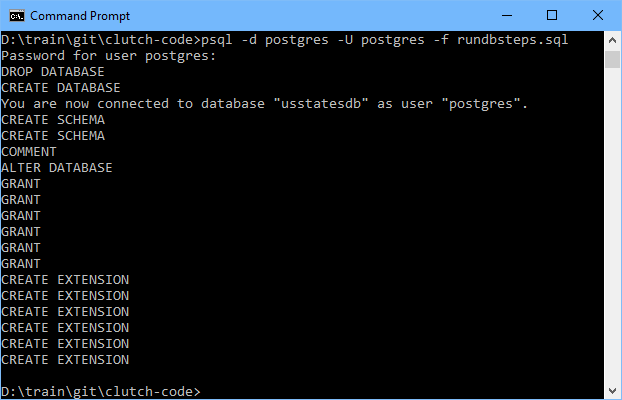
**rundbsteps.sql**

This script creates a spatial database (postgis) called 'usstatesdb' in PostgresSQL. It also creates necessary schemas and provides grants on objects to import data from the shape file (done as part of our next step). Run this script from your Windows Command prompt.

Command to run in PSQL prompt is:

psql -d postgres -U postgres -f rundbsteps.sql

You will be prompted to type the password, if PGPASSWORD environment variable is not set. Please use appropriate password for default user (postgres) to run this script.



Parameters:

**-d** --> default PostgreSQL database (most database installations use postgres)

**-U** --> default PostgreSQL user (most database installations use postgres)

**-f** --> location of the file (if you are in the same directory as the file just the file name is enough, if not please supply absolute file path along with the name)

Also for psql, if the path is not set in PATH environment variable make sure to supply the full path to the .exe.

**Note:** You will see some errors from the script when you run the first time, there are drop statements if the database/object exists. You may ignore the errors.

**loadshapefile.bat**

This script runs the shp2pgsql command to load the shape file into the database table. This script needs to be executed from Command Prompt (please see below).

The usage of the script is as follows:

loadshapefile.bat hostname username password shapefilepath

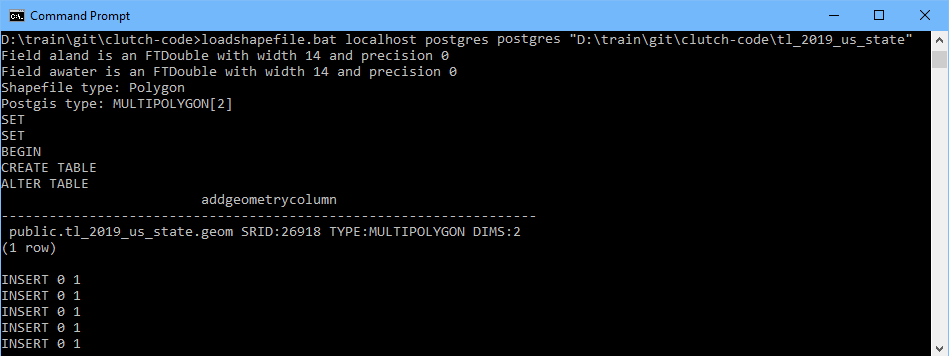
As mentioned above the script has 4 parameters required to load the data from the shapefile.

**hostname** - Hostname is the PostgreSQL database server host

**username** - Username is the PostgreSQL default username (postgres)

**password** - Password is the password for the username above

**shapefilepath** - Extraction path of .shp files along with the other files (tl\_2019\_us\_state folder) pulled from the git repository.

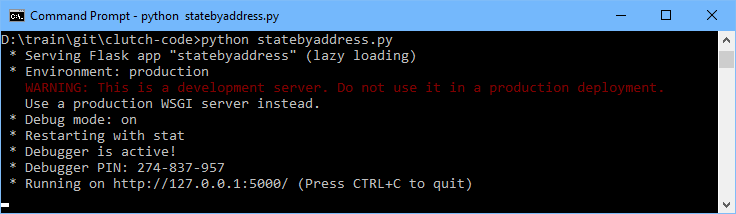


**Note:** Please ensure the path for shp2pgsql is set before you execute this script. This script does not include the absolute path of shp2pgsql.

**Python Setup:**

Python setup is pretty straight forward after the installation for this API as long as the prerequisites are met. statebyaddress.py is the script that will be executed. Again from Windows Command Prompt you may execute this script as follows:

python statebyaddress.py



Again, ensure PYTHONPATH is set and you are running this from the same path as where the python script resides, if not make sure to provide absolute path for statebyaddress.py script.

Once you get to the above prompt you are ready to test your API. Please note the flask API is now running on http://127.0.0.1:5000 as mentioned in the screen capture above.

**Running the tests**

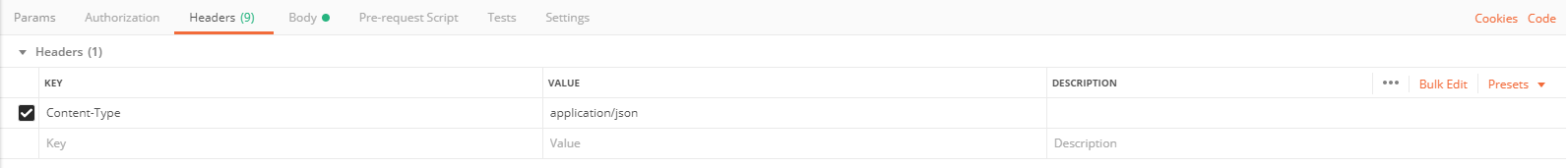
You may use Postman (REST API client or similar clients) to test the API. Open your REST client application (in my case Postman) on Windows.

Issue a GET request with the URL: http://127.0.0.1:5000/statebyaddress (please see below)

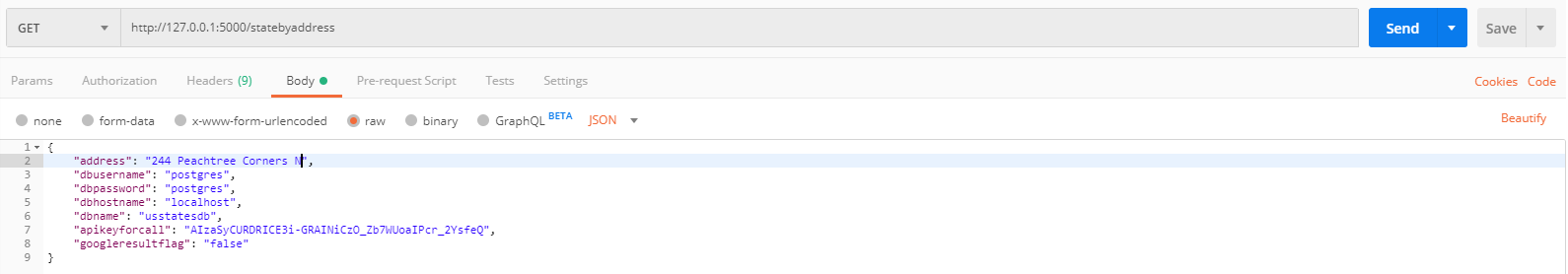


The request and response for this API is JSON.

In the Header section, please mention the content type as "application/json".



In the Body section, use raw type and use pass the input as follows:



In the request above, the keys are as follows:

**address** - Street address for which we need the state.

**dbusername** - Database User to the postgis/postgresql database.

**dbpassword** - Password for the postgis/postgresql database user.

**dbhostname** - Host server of postgis/postgresql database.

**dbname** - Name of the postgis/postgresql database.

**apikeyforcall** - Google API key to be used to pull the latitude/longitude.

**googleresultflag** - set to **true** if you need the Geocoding API response from google, else set **false**.

**Shorter response (when googleresultflag is set to false):**

{

  "accuracy": "ROOFTOP",

  "formatted\_address": "2010 Village Center Dr, Tarentum, PA 15084, USA",

  "google\_place\_id": "ChIJd9TiZUGUNIgRgCxM1x8SUd8",

  "google\_response": null,

  "input\_string": "2010 Village Center Dr",

  "latitude": 40.5635848,

  "longitude": -79.80380199999999,

  "postcode": "15084",

  "status\_msg": null,

  "type": "premise",

  "us\_state": "Pennsylvania"

}

**Note:** google\_response key has null as its value.

**Detailed response (when googleresultflag is set to true):**

{

  "accuracy": "ROOFTOP",

  "formatted\_address": "2010 Village Center Dr, Tarentum, PA 15084, USA",

  "google\_place\_id": "ChIJd9TiZUGUNIgRgCxM1x8SUd8",

  "google\_response": {

    "results": [

      {

        "address\_components": [

          {

            "long\_name": "2010",

            "short\_name": "2010",

            "types": [

              "street\_number"

            ]

          },

          {

            "long\_name": "Village Center Drive",

            "short\_name": "Village Center Dr",

            "types": [

              "route"

            ]

          },

          {

            "long\_name": "Tarentum",

            "short\_name": "Tarentum",

            "types": [

              "locality",

              "political"

            ]

          },

          {

            "long\_name": "Frazer Township",

            "short\_name": "Frazer Township",

            "types": [

              "administrative\_area\_level\_3",

              "political"

            ]

          },

          {

            "long\_name": "Allegheny County",

            "short\_name": "Allegheny County",

            "types": [

              "administrative\_area\_level\_2",

              "political"

            ]

          },

          {

            "long\_name": "Pennsylvania",

            "short\_name": "PA",

            "types": [

              "administrative\_area\_level\_1",

              "political"

            ]

          },

          {

            "long\_name": "United States",

            "short\_name": "US",

            "types": [

              "country",

              "political"

            ]

          },

          {

            "long\_name": "15084",

            "short\_name": "15084",

            "types": [

              "postal\_code"

            ]

          },

          {

            "long\_name": "3850",

            "short\_name": "3850",

            "types": [

              "postal\_code\_suffix"

            ]

          }

        ],

        "formatted\_address": "2010 Village Center Dr, Tarentum, PA 15084, USA",

        "geometry": {

          "bounds": {

            "northeast": {

              "lat": 40.5650146,

              "lng": -79.8033267

            },

            "southwest": {

              "lat": 40.5631793,

              "lng": -79.80492939999999

            }

          },

          "location": {

            "lat": 40.5635848,

            "lng": -79.80380199999999

          },

          "location\_type": "ROOFTOP",

          "viewport": {

            "northeast": {

              "lat": 40.5654459302915,

              "lng": -79.80277906970849

            },

            "southwest": {

              "lat": 40.5627479697085,

              "lng": -79.8054770302915

            }

          }

        },

        "place\_id": "ChIJd9TiZUGUNIgRgCxM1x8SUd8",

        "types": [

          "premise"

        ]

      }

    ],

    "status": "OK"

  },

  "input\_string": "2010 Village Center Dr",

  "latitude": 40.5635848,

  "longitude": -79.80380199999999,

  "postcode": "15084",

  "status\_msg": null,

  "type": "premise",

  "us\_state": "Pennsylvania"

}

**Versioning**

Repository is maintained in Github. All the versions pertaining to this API will remain in the same repository.

**Authors**

* Shyam Kasthuri

**Acknowledgments**

* Alex Urquhart - https://alexurquhart.com/post/set-up-postgis-with-docker/Inspiration
* PostgreSQL documentation - https://www.postgresql.org/docs/