Overview

The goal of this project is to create a Reddit clone using microservices. The REST (REpresentational State Transfer) architectural design for web APIs is used to make requests to the database in the form of a JSON blob. These web services are created using Python and Flask.

Teams

Name	Role	Contact Info
Javier Melendrez	Developer 1	javim1224@csu.fullerton.edu
Zexin Zhuang	Developer 2	zexinzhuang@csu.fullerton.edu
Nathaniel Richards	SDET	nathrich23@csu.fullerton.edu
Antonio Lopez	Operations	antonio_lopez@csu.fullerton.edu

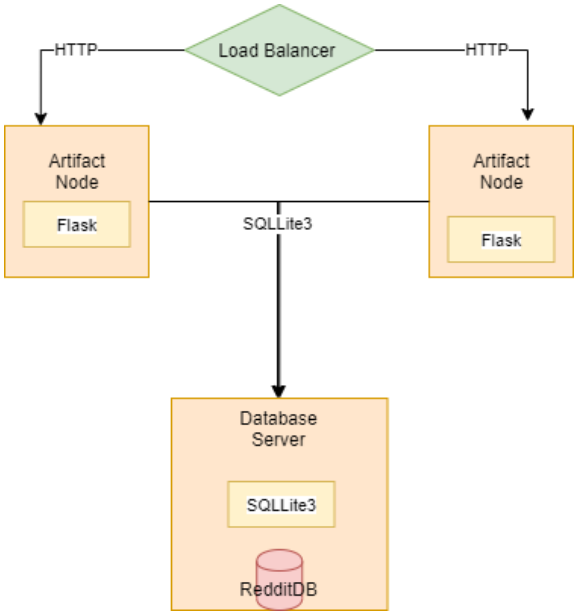
Production Stack

This project is written in Python 3 and the deployment environment is in Tuffix (Xubuntu). You can check the version of Python by running the command *python3 --version* If Python 3 is not installed you can get it using the command *sudo apt-get install python3* Once you have installed Python check the version of Pip you have by running the command *pip3 --version* Python includes Pip upon installation but if Pip cannot be found you can install it using the *sudo apt-get install pip3* command. The database is handled by the microservices using SQLite3 and is viewed using API requests, DB Browser for SQLite, or any database viewer of your choice. For this project we are using DB Browser for SQLite. CURL (Client URL Request Library) requests are created using Postman. Requests update/retrieve information to/from the database. The retrieved information is outputted in JSON format.

These are the libraries/dependencies that are required for the project to run and test it. A shell script is provided at the end of the SETUP section to easily install all of them.

- Flask
- Gunicorn 3
- Caddy
- Foreman

Architecture



Hosts

Environment	Microservice	Hostname
Test	Post	http://127.0.0.1:5000
Test	User	http://127.0.0.1:5000
Prod	Post	http://localhost:2015/posts
Prod	User	http://localhost:2015/users

Caddy configuration

Post Microservice Proxy (Production)	User Microservice Proxy (Production)
127.0.0.1:5000 127.0.0.1:5001 127.0.0.1:5002	127.0.0.1:5100 127.0.0.1:5101 127.0.0.1:5102

Setup

These instructions will only work for Linux based distributions, specifically Tuffix. These steps will help create a working environment while installing all libraries and dependences.

Open a terminal and create a directory:
$ mkdir reddit_clone

Change the current directory:
$ cd reddit_clone/

Clone repository:
$ git clone https://github.com/antonio-lopez/cspc-449-project-1.git

Change the current directory:
$ cd cspc-449-project-1/

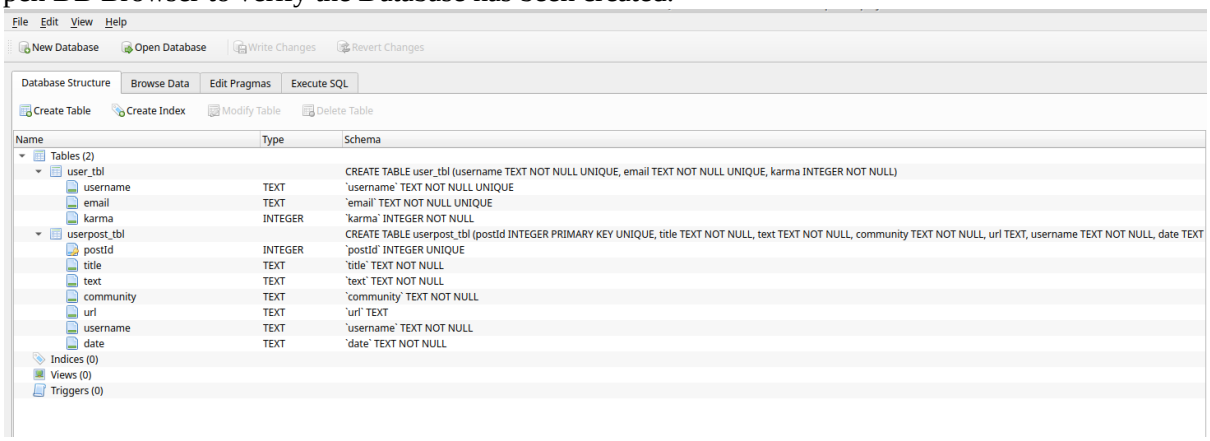
Create a virtual environment:
$ python3 -m venv ENV

Activate the virtual environment:
$ source ENV/bin/activate

Create the database by running the python file:
$ python3 create_db.py

Run shell script to download and install requirements:
$ bash requirements.sh

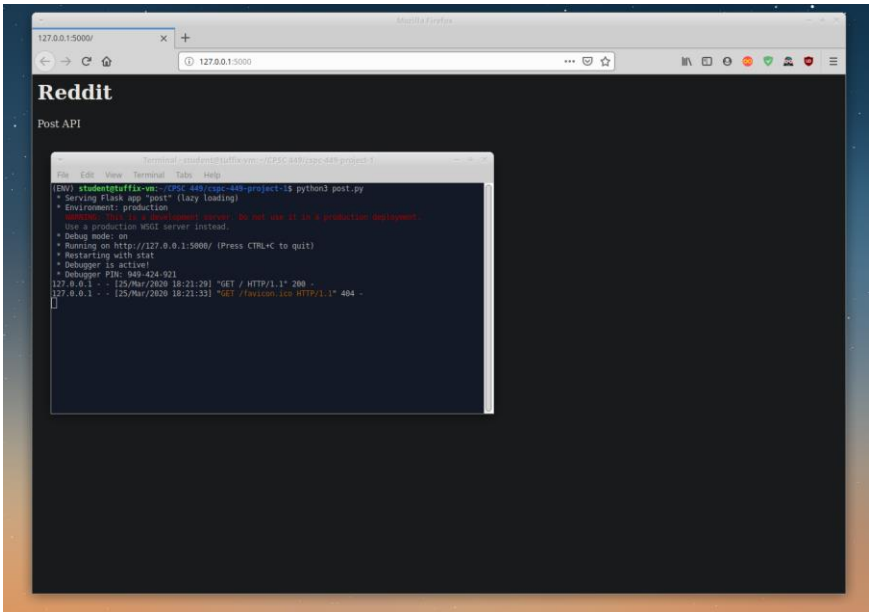
Open DB Browser to verify the Database has been created:

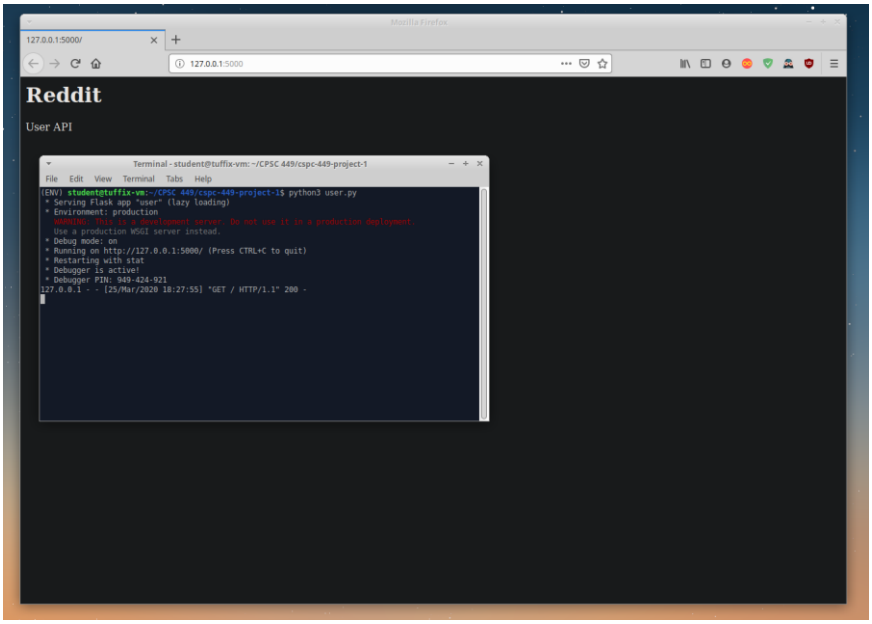


Run

To verify the setup is complete run either the Post (post.py) or User (user.py) microservice.
$ python3 post.py
Or
$ python3 user.py

Follow the link provided (ctrl + left click). Your browser will open with the main page greeting for the chosen microservice.





Testing

Testing can be done in either non-production or production. In the initial testing for either, make sure you are using a newly created database. The testing scripts create new users and posts. If you are following the instructions and have completed the SETUP and RUN sections continue with testing. If the database is populated, delete it and create a new one using the *$ python3 create_db.py* command.

Testing (Non-production):

Non-production testing is done using the Flask Development Server.

<u>Post Microservice Testing:</u>

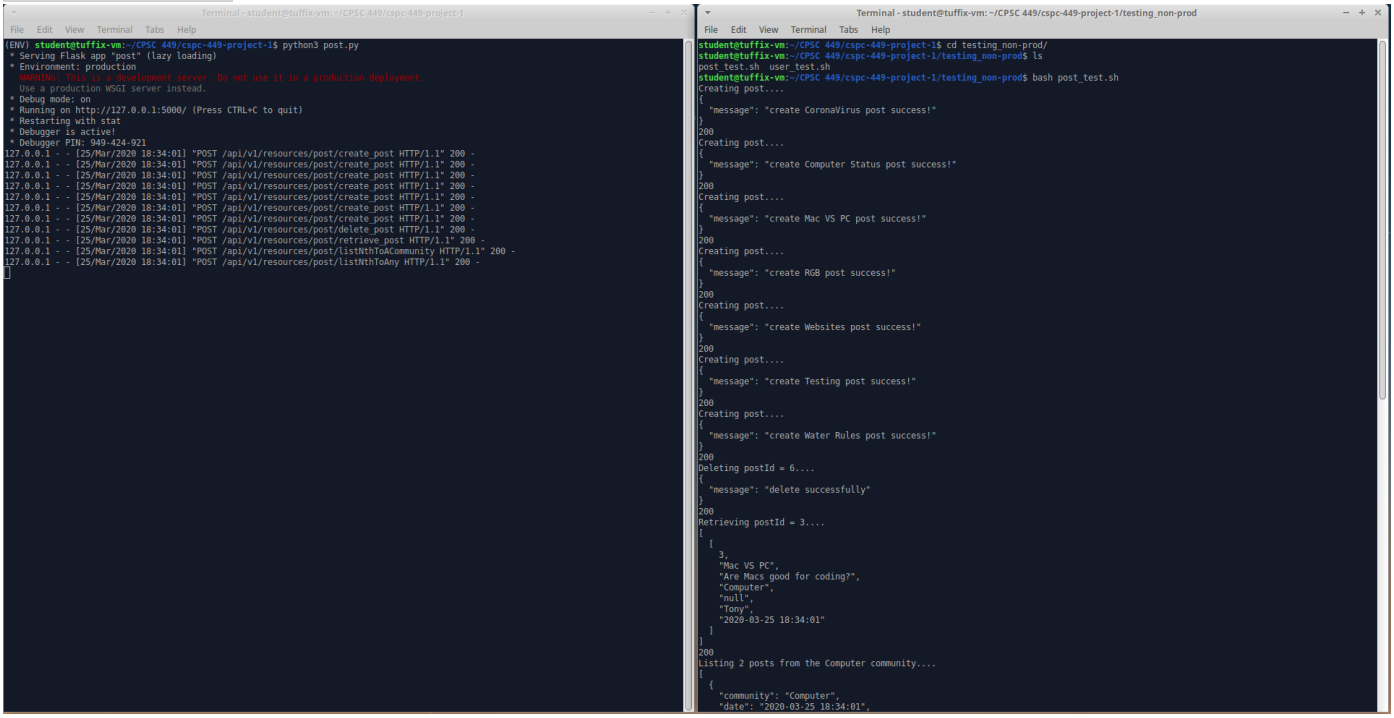
Run the Post microservice on a terminal:

$ python3 post.py

Open a new terminal, change to the testing directory, and run the Post testing shell script:

$ cd testing_non-prod/

$ bash post_test.sh



```
(ENV) student@tuffix-vm:~/CPSC 449/cspc-449-project-1$ python3 post.py
 * Serving Flask app "post" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 949-424-921
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/create_post HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/create_post HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/create_post HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/create_post HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/create_post HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/create_post HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/create_post HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/delete_post HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/retrieve_post HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/listNthToACommunity HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 18:34:01] "POST /api/v1/resources/post/listNthToAny HTTP/1.1" 200 -
```

```
student@tuffix-vm:~/CPSC 449/cspc-449-project-1$ cd testing_non-prod/
student@tuffix-vm:~/CPSC 449/cspc-449-project-1/testing_non-prod$ ls
post_test.sh  user_test.sh
student@tuffix-vm:~/CPSC 449/cspc-449-project-1/testing_non-prod$ bash post_test.sh
Creating post....
{
  "message": "create CoronaVirus post success!"
}
200
Creating post....
{
  "message": "create Computer Status post success!"
}
200
Creating post....
{
  "message": "create Mac VS PC post success!"
}
200
Creating post....
{
  "message": "create RGB post success!"
}
200
Creating post....
{
  "message": "create Websites post success!"
}
200
Creating post....
{
  "message": "create Testing post success!"
}
200
Creating post....
{
  "message": "create Water Rules post success!"
}
200
Deleting postId = 6....
{
  "message": "delete successfully"
}
200
Retrieving postId = 3....
[
  [
    3,
    "Mac VS PC",
    "Are Macs good for coding?",
    "Computer",
    "null",
    "Tony",
    "2020-03-25 18:34:01"
  ]
]
200
Listing 2 posts from the Computer community....
[
  {
    "community": "Computer",
    "date": "2020-03-25 18:34:01",
```

Running the script tests the following API requests:

- Create a new post
- Delete an existing post given a postId
- Retrieve an existing post given a postId
- List the n most recent posts to a particular community
- List the n most recent posts to any community

To verify if the posts were added to your database, open DB Browser and select the “Browse Data” tab and select the “userpost_tbl” table.

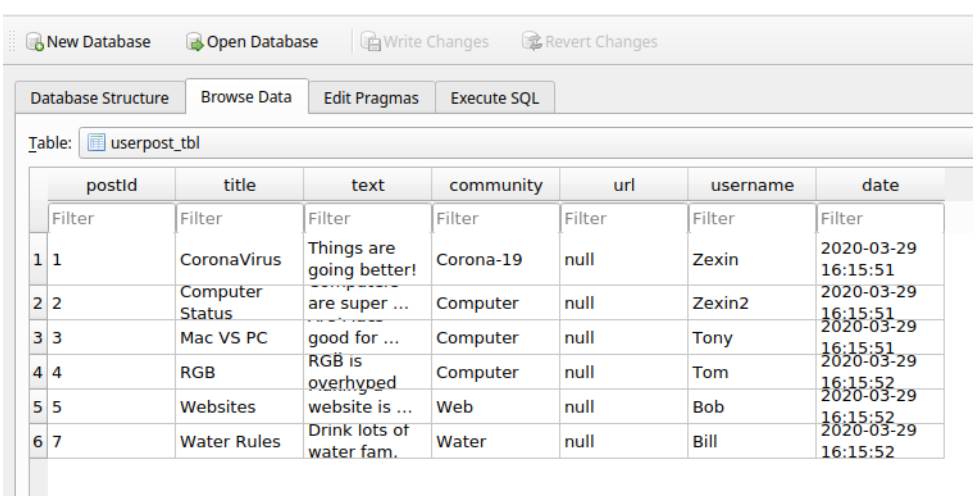


Table: userpost_tbl

	postId	title	text	community	url	username	date
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	CoronaVirus	Things are going better!	Corona-19	null	Zexin	2020-03-29 16:15:51
2	2	Computer Status	are super ...	Computer	null	Zexin2	2020-03-29 16:15:51
3	3	Mac VS PC	good for ...	Computer	null	Tony	2020-03-29 16:15:51
4	4	RGB	RGB is overhyped	Computer	null	Tom	2020-03-29 16:15:52
5	5	Websites	website is ...	Web	null	Bob	2020-03-29 16:15:52
6	7	Water Rules	Drink lots of water fam.	Water	null	Bill	2020-03-29 16:15:52

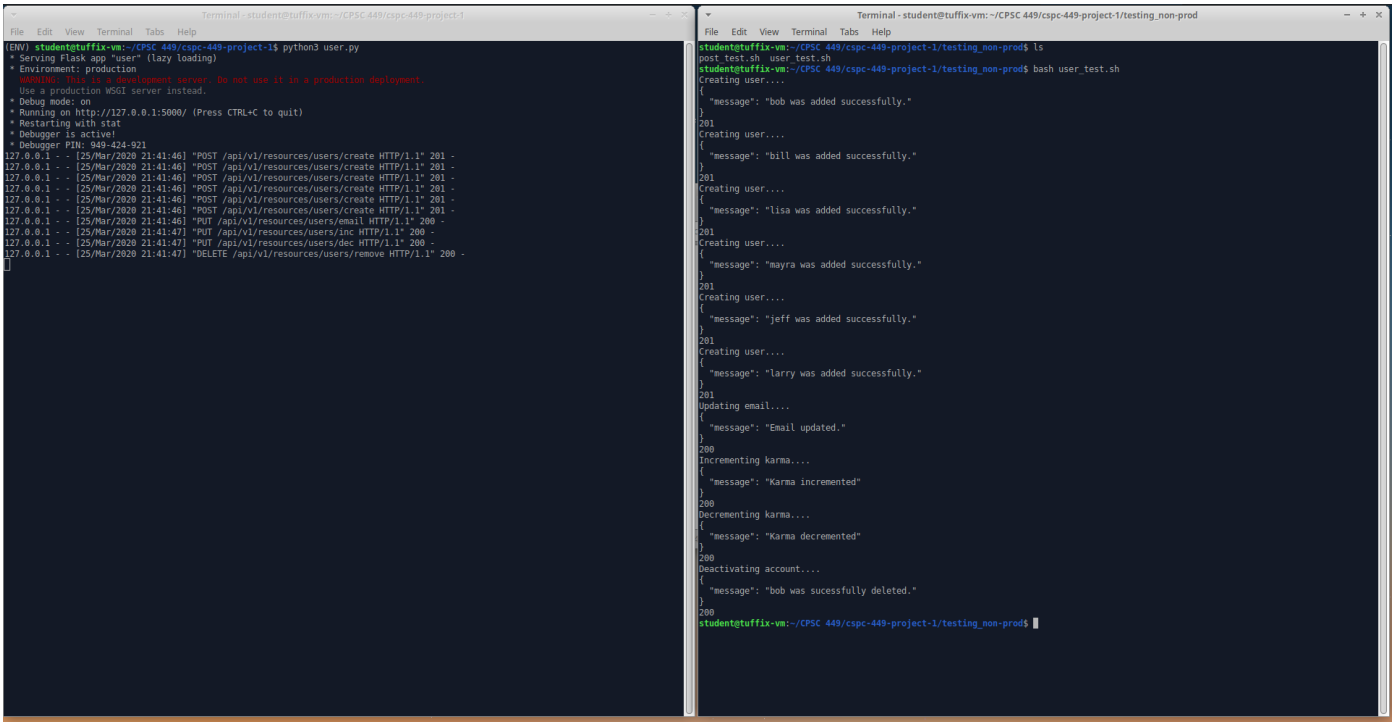
User Microservice Testing:

Cancel the Post microservice and run the User microservice:

```
$ python3 user.py
```

On the other terminal run the User testing shell script:

```
$ bash user_test.sh
```



```
(ENV) student@tuffix-vm:~/CPSC 449/cspc-449-project-1$ python3 user.py
 * Serving Flask app "user" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 949-424-921
127.0.0.1 - - [25/Mar/2020 21:41:46] "POST /api/v1/resources/users/create HTTP/1.1" 201 -
127.0.0.1 - - [25/Mar/2020 21:41:46] "POST /api/v1/resources/users/create HTTP/1.1" 201 -
127.0.0.1 - - [25/Mar/2020 21:41:46] "POST /api/v1/resources/users/create HTTP/1.1" 201 -
127.0.0.1 - - [25/Mar/2020 21:41:46] "POST /api/v1/resources/users/create HTTP/1.1" 201 -
127.0.0.1 - - [25/Mar/2020 21:41:46] "POST /api/v1/resources/users/create HTTP/1.1" 201 -
127.0.0.1 - - [25/Mar/2020 21:41:46] "POST /api/v1/resources/users/create HTTP/1.1" 201 -
127.0.0.1 - - [25/Mar/2020 21:41:46] "PUT /api/v1/resources/users/email HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 21:41:47] "PUT /api/v1/resources/users/inc HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 21:41:47] "PUT /api/v1/resources/users/dec HTTP/1.1" 200 -
127.0.0.1 - - [25/Mar/2020 21:41:47] "DELETE /api/v1/resources/users/remove HTTP/1.1" 200 -
```

```
student@tuffix-vm:~/CPSC 449/cspc-449-project-1/testing_non-prod$ ls
post_test.sh  user_test.sh
student@tuffix-vm:~/CPSC 449/cspc-449-project-1/testing_non-prod$ bash user_test.sh
Creating user....
{
  "message": "bob was added successfully."
}
201
Creating user....
{
  "message": "bill was added successfully."
}
201
Creating user....
{
  "message": "lisa was added successfully."
}
201
Creating user....
{
  "message": "mayra was added successfully."
}
201
Creating user....
{
  "message": "jeff was added successfully."
}
201
Creating user....
{
  "message": "larry was added successfully."
}
201
Updating email....
{
  "message": "Email updated."
}
200
Incrementing karma....
{
  "message": "Karma incremented"
}
200
Decrementing karma....
{
  "message": "Karma decremented"
}
200
Deactivating account....
{
  "message": "bob was sucessfully deleted."
}
200
student@tuffix-vm:~/CPSC 449/cspc-449-project-1/testing_non-prod$ 
```

Running the script tests the following API requests:

- Create user
- Update email
- Increment Karma
- Decrement Karma
- Deactivate account

Check DB Browser to verify if the data was inserted into the database by selecting the “user_tbl” table.

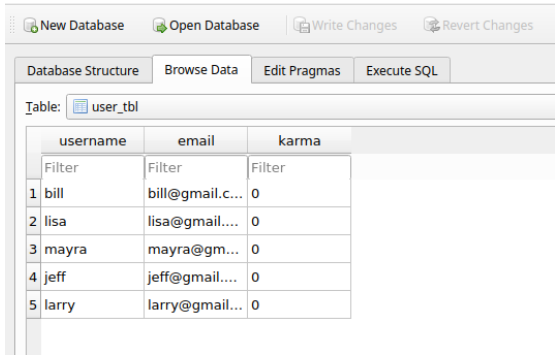


Table: user_tbl

	username	email	karma
	Filter	Filter	Filter
1	bill	bill@gmail.c...	0
2	lisa	lisa@gmail....	0
3	mayra	mayra@gm...	0
4	jeff	jeff@gmail....	0
5	larry	larry@gmail...	0

On the same terminal where you ran the User testing shell script, run the 100 Users Test shell script:

```
$ bash 100_users_test.sh
```

```
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
127.0.0.1 - - [29/Mar/2020 16:27:25] "POST /api/v1/resources/users/create
 HTTP/1.1" 201 -
```

```
}
201
{
  "message": "sample94 was added successfully."
}
201
{
  "message": "sample95 was added successfully."
}
201
{
  "message": "sample96 was added successfully."
}
201
{
  "message": "sample97 was added successfully."
}
201
{
  "message": "sample98 was added successfully."
}
201
{
  "message": "sample99 was added successfully."
}
201
{
  "message": "sample100 was added successfully."
}
201
(ENV) student@tuffix-vm:~/Videos/cspc-449-project-1/testing_non-prod$
```

Refresh DB Browser and verify the 100 users were added to the database:

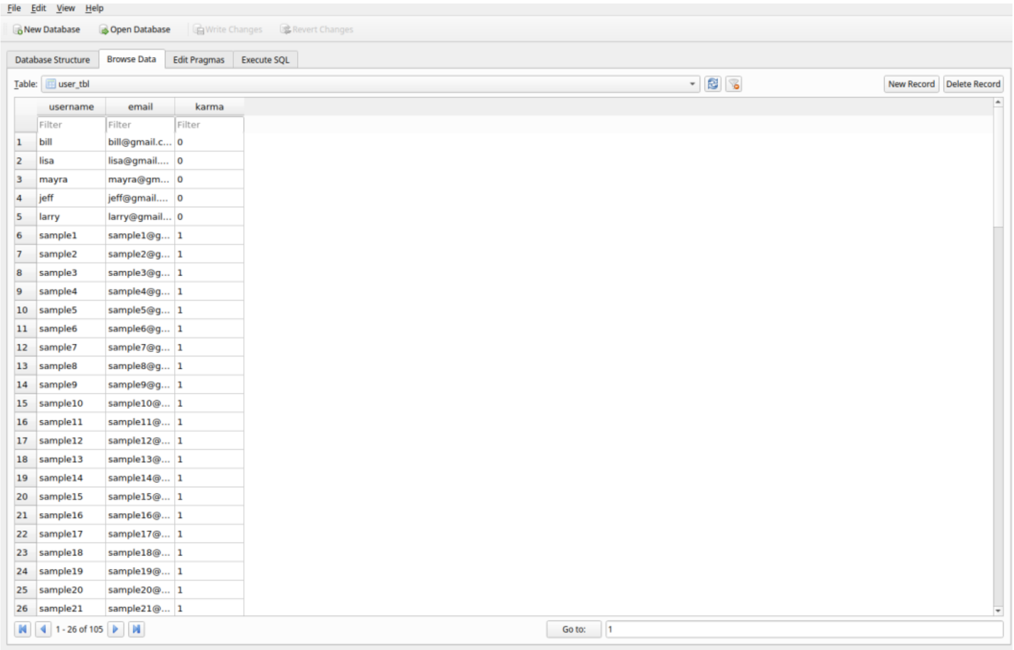


Table: user_tbl

	username	email	karma
	Filter	Filter	Filter
1	bill	bill@gmail.c...	0
2	lisa	lisa@gmail....	0
3	mayra	mayra@gm...	0
4	jeff	jeff@gmail....	0
5	larry	larry@gmail...	0
6	sample1	sample1@g...	1
7	sample2	sample2@g...	1
8	sample3	sample3@g...	1
9	sample4	sample4@g...	1
10	sample5	sample5@g...	1
11	sample6	sample6@g...	1
12	sample7	sample7@g...	1
13	sample8	sample8@g...	1
14	sample9	sample9@g...	1
15	sample10	sample10@...	1
16	sample11	sample11@...	1
17	sample12	sample12@...	1
18	sample13	sample13@...	1
19	sample14	sample14@...	1
20	sample15	sample15@...	1
21	sample16	sample16@...	1
22	sample17	sample17@...	1
23	sample18	sample18@...	1
24	sample19	sample19@...	1
25	sample20	sample20@...	1
26	sample21	sample21@...	1

1 - 26 of 105

Go to: 1

Testing (Production)

Production testing is done using a WSGI server called Gunicorn. Foreman is used to run a Procfile that manages the Gunicorn and Caddy processes for each microservice. 3 instances are created for each microservice. The Caddyfile proxies’ direct requests for “http://localhost:2015/posts” to the Posts microservice and requests for “http://localhost:2015/users” to the Users microservice. It also handles the load balancing method which in this case is round robin.

<u>Post Microservice Testing:</u>

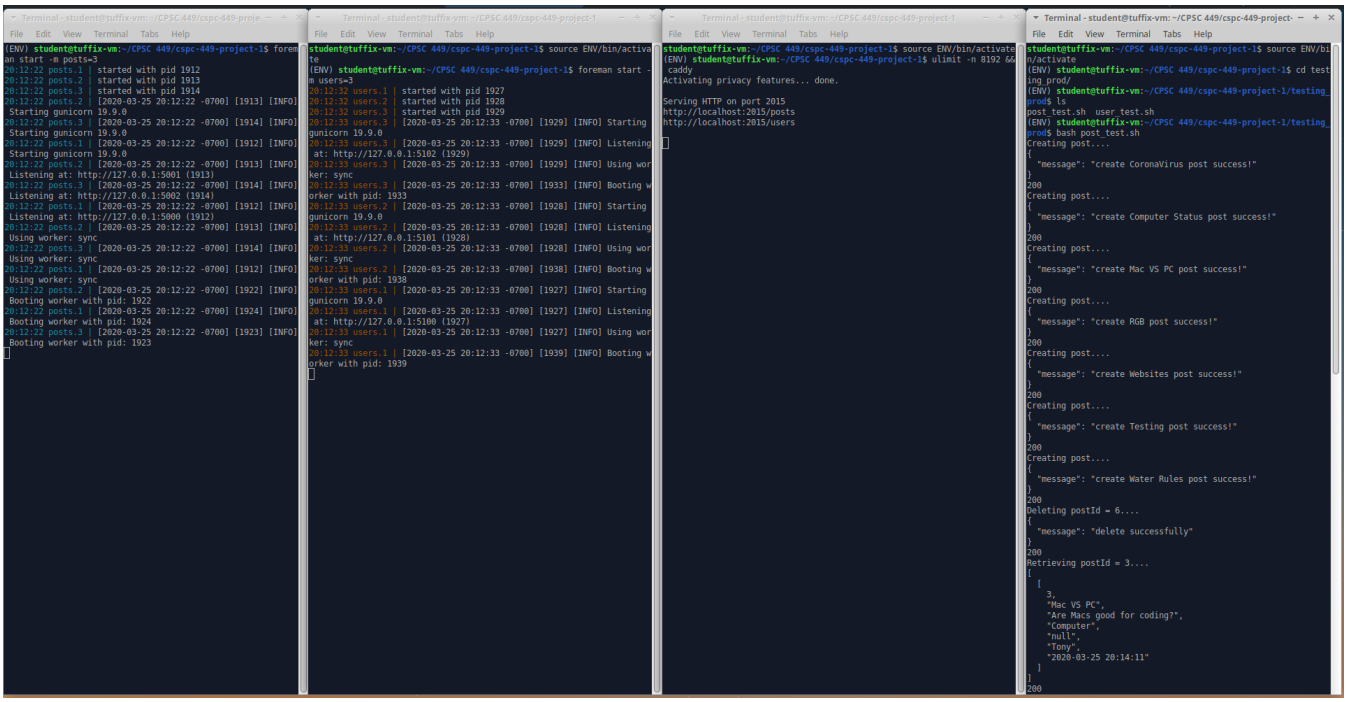
Open 4 terminals, be sure to activate the virtual environment for each terminal using the *$ source ENV/bin/activate* command.

On one terminal run Foreman for the Post microservice:
$ foreman start -m posts=3

On the second terminal run Foreman for the User microservice:
$ foreman start -m users=3

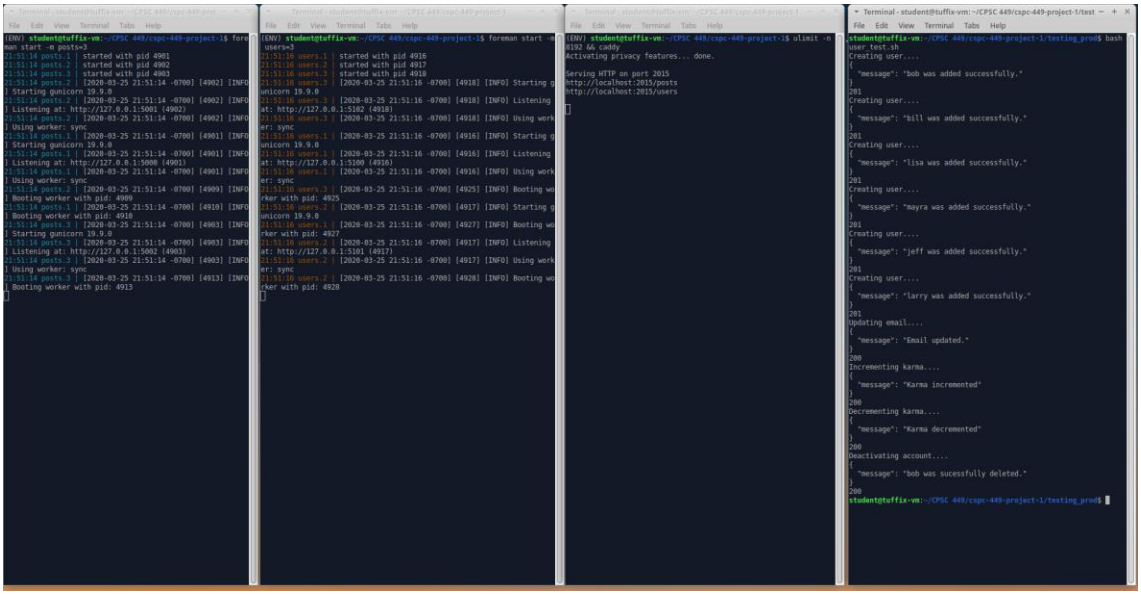
On the third terminal run the Caddyfile:
$ ulimit -n 8192 && caddy

On the fourth terminal change to the testing_prod directory
and run the Post testing shell script:
$ cd testing_prod/
$ bash post_test.sh



Check DB Browser to verify if the data was inserted into the database.

<u>User Microservice Testing:</u>
Cancel the Post testing shell script and run the User testing shell script:
$ bash user_test.sh



Check DB Browser to verify if the data was inserted into the database.

API Requests

CURL commands for User microservice (non-production)

- For production commands, change http://127.0.0.1:5000 → http://localhost:2015/users
- Ex: http://127.0.0.1:5000/api/v1/resources/users/create → http://localhost:2015/users/api/v1/resources/users/create

HTTP Method	CURL Command Example
POST	#Create user curl --location --request POST 'http://127.0.0.1:5000/api/v1/resources/users/create' \ --header 'Content-Type: application/json' \ --data-raw '{"email":"bob@gmail.com","username":"bob","karma":"0"}' \ --write-out '%{http_code}\n'
PUT	#Update email curl --location --request PUT 'http://127.0.0.1:5000/api/v1/resources/users/email' \ --header 'Content-Type: application/json' \ --data-raw '{"username":"bob","email":"b@gmail.com"}' \ --write-out '%{http_code}\n'
PUT	#Increment Karma curl --location --request PUT 'http://127.0.0.1:5000/api/v1/resources/users/inc' \ --header 'Content-Type: application/json' \ --data-raw '{"username":"bob"}' \ --write-out '%{http_code}\n'
PUT	#Decrement Karma curl --location --request PUT 'http://127.0.0.1:5000/api/v1/resources/users/dec' \ --header 'Content-Type: application/json' \

	--data-raw '{"username":"bob"}' \ --write-out '%{http_code}\n'
DELETE	#Deactivate account curl --location --request DELETE 'http://127.0.0.1:5000/api/v1/resources/users/remove' \ --header 'Content-Type: application/json' \ --data-raw '{"username":"bob"}' \ --write-out '%{http_code}\n'

CURL commands for Post microservice (non-production)

- For production commands, change http://127.0.0.1:5000 → http://localhost:2015/posts
- Ex: http://127.0.0.1:5000/api/v1/resources/post/create_post → http://localhost:2015/posts/api/v1/resources/post/create_post

HTTP Method	CURL Commands Example
POST	#Create a new post curl --location --request POST 'http://127.0.0.1:5000/api/v1/resources/post/create_post' \ --header 'Content-Type: application/json' \ --data-raw '{"community": "Corona-19", "text": "Things are going better!", "title": "CoronaVirus", "url": "null", "username": "Zexin"}' \ --write-out '%{http_code}\n'
DELETE	#Delete an existing post curl --location --request DELETE 'http://127.0.0.1:5000/api/v1/resources/post/delete_post' \ --header 'Content-Type: application/json' \ --data-raw '{"postId": "6"}' \ --write-out '%{http_code}\n'
GET	# Retrieves a post based on the id of a post curl --location --request GET 'http://127.0.0.1:5000/api/v1/resources/post/retrieve_post' \ --header 'Content-Type: application/json' \ --data-raw '{"postId": "3"}' \ --write-out '%{http_code}\n'
GET	#List the n most recent posts to a particular community curl --location --request GET 'http://127.0.0.1:5000/api/v1/resources/post/listNthToACommunity' \ --header 'Content-Type: application/json' \ --data-raw '{"nth": 2,"community": "Computer"}' \ --write-out '%{http_code}\n'
GET	#List the n most recent posts to any community curl --location --request GET 'http://127.0.0.1:5000/api/v1/resources/post/listNthToAny' \ --header 'Content-Type: application/json' \ --data-raw '{"nth": 5}' \ --write-out '%{http_code}\n'

Supplemental Issues

Virtual Machine Issues in Setup

Tuffix is known to have some errors when trying to download, update, or install many packages at a given time. If you are having locking issues when trying to install packages/dependences in the virtual environment, follow the instructions below.

See which processes are running:

$ ps aux | grep -I apt

```
student@tuffix-vm:~/Desktop/CPSC449$ ps aux | grep -i apt
root       525  0.0  0.0   4628   816 ?        Ss   21:23   0:00 /bin/sh /usr/lib/apt/apt.systemd.daily install
root       560  0.0  0.0   4628  1684 ?        S    21:23   0:00 /bin/sh /usr/lib/apt/apt.systemd.daily lock_is_held install
student   1656  0.0  0.0  22000  1148 pts/0    S+   21:29   0:00 grep --color=auto -i apt
```

Kill all root processes:

$ sudo kill <PID>

```
student@tuffix-vm:~/Desktop/CPSC449$ sudo kill 525
student@tuffix-vm:~/Desktop/CPSC449$ sudo kill 560
```

To kill the remaining process, we must identify the real PID of the process in the lock-frontend folder

$ sudo lsof /var/lib/dpkg/lock-frontend

```
student@tuffix-vm:~/Desktop/CPSC449$ sudo lsof /var/lib/dpkg/lock-frontend
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1000/gvfs
      Output information may be incomplete.
COMMAND    PID USER   FD   TYPE DEVICE SIZE/OFF   NODE NAME
unattende 1010 root    5uW  REG    8,1        0 264117 /var/lib/dpkg/lock-frontend
```

Kill all the remaining process with PID you discovered from the previous command

$ sudo kill <PID>

Everything should be good to go. Install virtual environment with this command

$ sudo apt-get install python3-venv