Project 2 Run Book

System Overview

System Name: Project 2 – Microservices (Project 1 Extension)

Support Contacts Project 2 roles

Level	Position	Contact Information
Level 1	SDET	Alexandria Wolfram aleewolfram@csu.fullerton.edu
Level 2	Developer	Emily Pham tpham523@csu.fullerton.edu
Level 3	Operations	Mohit Kumar mohit_kumar@csu.fullerton.edu

Overview

Application includes a pair of web microservices that provide functionality for a reddit-style application as well as two automation test suites for these specific services. These microservices allow for voting and posting back-end functionality on the site.

Github repository: https://github.com/tpham523/CPSC_449_Project_1
(Project 2 is Project 1 extension in same repository)

Run Guide

Installations required:

- pip3 install flask
- sudo apt install --yes gunicorn3

Creating instances:

1. Generating below instances of foreman in the terminal: `foreman start -m post=3,vote=3,user=3,msg=3`
2. Open a separate terminal and run the following: `ulimit -n 8192 && caddy`
3. Open **Project 2 URLs localhost:2015/users or localhost:2015/messages** on a browser.

   (Project 1 URLs localhost:2015/users or localhost:2015/)

API uses:

Project 2 API User and Message API

1. USER API

-Create a new user

```
curl -i -X POST -H 'Content-Type:application/json' -d '{"username":"axel","email":"axel@animalcrossing.com"}' http://localhost:2015/users/register
```

-Update user's email

```
curl -i -X PUT -H 'Content-Type:application/json' -d '{"username":"axel", "email":"newAxel@animalcrossing.com"}' http://localhost:2015/users/update_email
```

-Add karma to user

```
curl -i -X PUT -H 'Content-Type:application/json' -d '{"username":"axel"}' http://localhost:2015/users/add_karma;
```

-Remove karma from user

```
curl -i -X PUT -H 'Content-Type:application/json' -d '{"username":"axel"}' http://localhost:2015/users/remove_karma
```

-Delete a user

```
curl -i -X DELETE http://localhost:2015/users/delete?username=axel;
```

2. MESSAGE API

-Send a message

```
curl -i -X POST -H 'Content-Type:application/json' -d '{"user_from":"ilovedog", "user_to":"ilovecat", "msg_content":"I think dogs are better", "msg_flag":"facts"}' http://localhost:2015/messages/send;
```

-Delete a message

```
curl -i -X DELETE http://localhost:2015/messages/delete?msg_id=3;
```

-Favorite a message
curl -i -X POST -H 'Content-Type:application/json' http://localhost:2015/messages/favorite?msg_id=3;

(**Previous Project 1 APIS Post and Vote API**)

1. Create a sample post:
   curl –i –X POST –H 'Content–Type:application/json' –d '{"title":"Post Title", "description":"Post description!", "username":"user", "community_name":"cheesecake"}' http://localhost:2015/posts/create

2. Create a sample vote:
   curl –i –X POST –H "Content–Type: application/json" –d '{"vote_id":"0"}' http://localhost:2015/votes/upvotes

3. Remove a post:
   curl –i –X DELETE http://localhost:2015/posts/delete?post_id=2

4. Add a downvote to a post:
   curl –i –X POST –H "Content–Type: application/json" –d '{"vote_id":"0"}' http://localhost:2015/votes/downvotes

5. Get a post:
   curl –i http://localhost:2015/posts/get?post_id=2

6. Get votes:
   curl –i http://localhost:2015/votes/get?vote_id=0

7. Get most recent posts:
   - Overall:
     curl –i http://localhost:2015/posts/filter?n=2
   - By Community:
     curl –i
     http://localhost:2015/posts/filter?n=5&community_name=coronavirus
8. Get most popular post:
   - Overall:
     curl –i http://localhost:2015/votes/getTop?n=3
   - Sort posts by popularity:
     curl –i –X POST –H "Content–Type: application/json" –d '{"post_ids":["0","1", "2"]}' http:// localhost:2015/getList

Function Testing

1. Install tavern:
   `pip3 install tavern[pytest]`

2. Install configobj:
   `pip3 install configobj`

3. Run to test:
   `py.test`

(This will test Project 2 and project 1 API both)

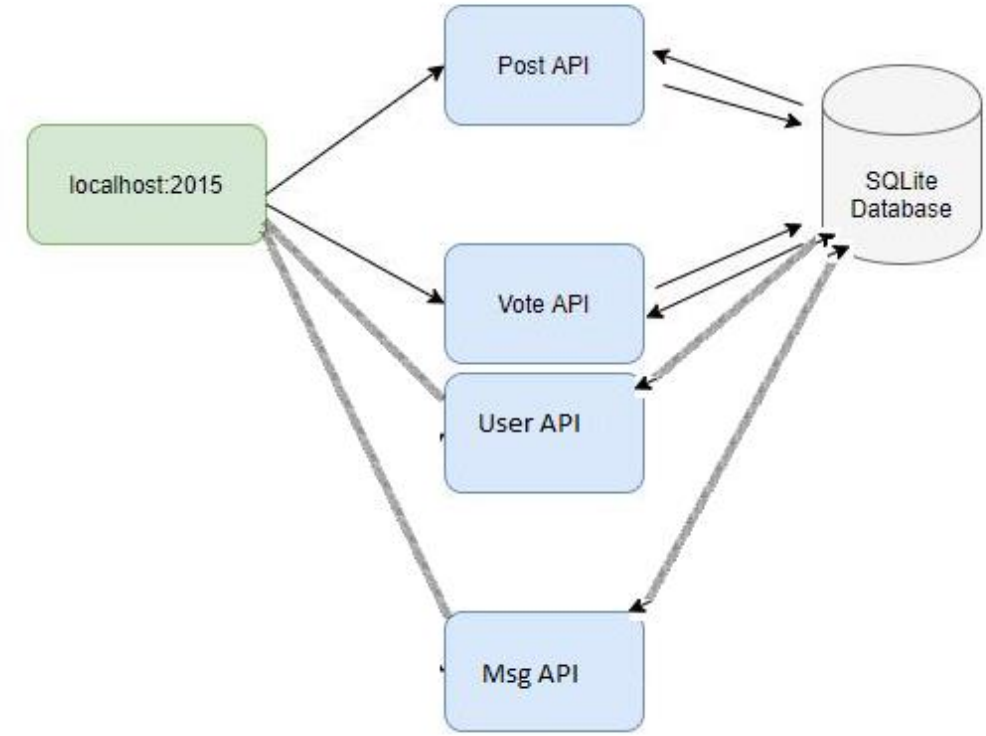
Load Testing

1. Install faker:
   `pip3 install faker`

2. Install locustio:
   `pip3 install locustio`

3. Run to demonstrate 500 simultaneous users, adding 20 users per second.
   `locust -f tests/loadTest.py --host=http://localhost:2015 --no-web -c 500 -r 20`

Architecture



Hosts

Env	Role	Hostname
Test	User	localhost:2015/users
	Message	localhost:2015/messages
	Database	localhost:2015
Production	User	localhost:2015/users
	Message	localhost:2015/messages
	Database	localhost:2015
Test	Post	localhost:2015/posts
	Vote	localhost:2015/votes
	Database	localhost:2015
Production	Post	localhost:2015/posts
	Vote	localhost:2015/votes
	Database	localhost:2015

Network

Service	Port	Protocol
User	2015	TCP – HTTP
Message	2015	TCP – HTTP
Post	2015	TCP – HTTP
Vote	2015	TCP – HTTP
Database	2015	TCP – SQLite

Directory Locations

Service	Configuration	Logs	Data

User	/users	n/a	n/a
Message	/messages	n/a	n/a
Post	/posts	n/a	n/a
Vote	/votes	n/a	n/a
Database	/data.sql	n/a	/data.db

Monitoring

Host	Item	Severity	Resolution
Application	post	SEV1	Restart process
	vote	SEV2	Restart process
	user	SEV1	Restart process
	messages	SEV2	Restart process
Database	data	SEV1	Restart process