**Project 1 Run Book**

**System Overview**

**System Name: Project 1 - Microservices**

**Support Contacts**

|  |  |  |
| --- | --- | --- |
| **Level** | **Position** | **Contact Information** |
| Level 1 | Operations | Alexandria Wolfram aleewolfram@csu.fullerton.edu |
| Level 2 | SDET | Emily Pham tpham523@csu.fullerton.edu |
| Level 3 | Developer | 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

**Run Guide**

Installations required:

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

Creating instances:

1. Generating 2 instances of foreman in the terminal: foreman start -m post=2,vote=2
2. Open a separate terminal and run the following: ulimit -n 8192 && caddy
3. Open localhost:2015/posts or localhost:2015/votes on a browser.

API uses:

1. Create a sample post:

curl -i -X POST -H 'Content-Type:application/json' -d '{“title”:”Post Title”, “description”:”Post description!”, “community\_name”:”4”, “username”:”user”} <http://localhost:2015/posts/create>;

1. Create a sample vote:

curl -i -X POST -H “Content-Type: application/json” -d ‘{“vote\_id”:”0”}’ http://localhost:2015/votes/upvotes

1. Remove a post:

curl -i -X DELETE <http://localhost:2015/posts/delete?post_id=2>;

1. Add a downvote to a post:

curl -i -X POST -H “Content-Type: application/json” -d '{“vote\_id”:”0”}’ ‘http://localhost:2015/votes/downvotes’

1. Get a post:

curl -i <http://localhost:2015/posts/get?post_id=2>;

1. Get votes:

curl -i http://localhost:2015/votes/get?vote\_id=0;

1. 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=sample;

1. 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]

1. Install configobj:

pip3 install configobj

1. Run to test:

py.test

**Load Testing**

1. Install faker:

pip3 install faker

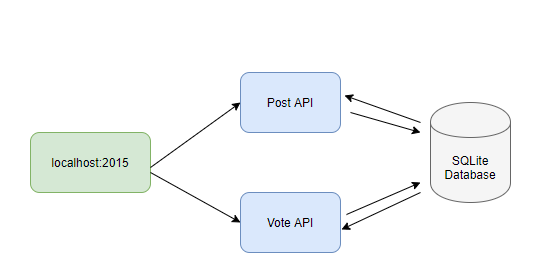
1. Install locustio:

pip3 install locustio

1. 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 | 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** |
| Post | 2015 | TCP - HTTP |
| Vote | 2015 | TCP - HTTP |
| Database | 2015 | TCP - SQLite |

**Directory Locations**

|  |  |  |  |
| --- | --- | --- | --- |
| **Service** | **Configuration** | **Logs** | **Data** |
| 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 |
| Database | data | SEV1 | Restart process |