Test Report

This report answers following questions:

- How did you test your code?
- How long does it take to process a single post (performance)?
- Does the size of the data submitted to the server impact the performance?
- How does the number of requests impact the performance of the server?
- How does the level of concurrency impact the performance of the server?

Test Method

To test the server, I used the `loadtest` package from npm.

since there is a need to testing under different parameters, I wrote a script to run the test multiple times.

```
#!/bin/bash

# prepare an empty directory for the test
rm -rf loadtestData
mkdir loadtestData

TEST_DIR=loadtestData
# use `npx loadtest` to test the performance of the server
npm install -g loadtest

# the post data is a JSON array, each element has the following format:
# {
    # "topic": "string",
    # "data": "string"
# }

# 1. How long does it take to process a single post (performance) of with a simple single request?
DIR_T1=$TEST_DIR/test_1
mkdir $DIR_T1
```

```
20
21  TYPE='application/x-www-form-urlencoded'
22  DATA_0='[{"topic":"test","data":"a"}]'
23  echo "single request" > $DIR_T1/result_0.txt
24  npx loadtest -n 1 -c 1 -m POST -T $TYPE --data $DATA_0
    http://localhost:8000/ >> $DIR_T1/result_0.txt
25  # extract the time from the result
26  echo "single request" > $DIR_T1/time_0.txt
27  cat $DIR_T1/result_0.txt | grep "Requests per second" >>
    $DIR_T1/time_0.txt
28  cat $DIR_T1/result_0.txt | grep "Mean latency" >>
    $DIR_T1/time_0.txt
29
30  # 2. Does the size of the data submitted to the server impact
    the performance?
31  MSG_2="size of data: "
32  DIR_T2=$TEST_DIR/test_2
33  mkdir $DIR_T2
34  # 2.1. 1KB
35  DATA_1='[{"topic":"test","data":"a"}]'
36  echo $MSG_2 "1KB" > $DIR_T2/result_1.txt
37  npx loadtest -n 1 -c 1 -m POST -T $TYPE --data $DATA_1
    http://localhost:8000/ >> $DIR_T2/result_1.txt
38  # extract the time from the result
39  echo $MSG_2 "1KB" > $DIR_T2/time_1.txt
40  cat $DIR_T2/result_1.txt | grep "Requests per second" >>
    $DIR_T2/time_1.txt
41  cat $DIR_T2/result_1.txt | grep "Mean latency" >>
    $DIR_T2/time_1.txt
42
43  # 2.2. 10KB
44  DATA_2='[{"topic":"test","data":"a'
45  for i in {1..10}
46  do
47      DATA_2=$DATA_2'a'
48  done
49  DATA_2=$DATA_2'"}]'
50
51  echo $MSG_2 "10KB" > $DIR_T2/result_2.txt
52  npx loadtest -n 1 -c 1 -m POST -T $TYPE --data $DATA_2
    http://localhost:8000/ >> $DIR_T2/result_2.txt
53  # extract the time from the result
```

```
echo $MSG_2 "10KB" > $DIR_T2/time_2.txt
cat $DIR_T2/result_2.txt | grep "Requests per second" >> $DIR_T2/time_2.txt
cat $DIR_T2/result_2.txt | grep "Mean latency" >> $DIR_T2/time_2.txt

# 2.3. 100KB
DATA_3='[{"topic":"test","data":"a'
for i in {1..100}
do
    DATA_3=$DATA_3'a'
done
DATA_3=$DATA_3'"}]'
echo $MSG_2 "100KB" > $DIR_T2/result_3.txt
npx loadtest -n 1 -c 1 -m POST -T $TYPE --data $DATA_3 http://localhost:8000/ >> $DIR_T2/result_3.txt
# extract the time from the result
echo $MSG_2 "100KB" > $DIR_T2/time_3.txt
cat $DIR_T2/result_3.txt | grep "Requests per second" >> $DIR_T2/time_3.txt
cat $DIR_T2/result_3.txt | grep "Mean latency" >> $DIR_T2/time_3.txt

# 2.4. 1MB
DATA_4='[{"topic":"test","data":"a'
for i in {1..1000}
do
    DATA_4=$DATA_4'a'
done
DATA_4=$DATA_4'"}]'
echo $MSG_2 "1MB" > $DIR_T2/result_4.txt
npx loadtest -n 1 -c 1 -m POST -T $TYPE --data $DATA_4 http://localhost:8000/ >> $DIR_T2/result_4.txt
# extract the time from the result
echo $MSG_2 "1MB" > $DIR_T2/time_4.txt
cat $DIR_T2/result_4.txt | grep "Requests per second" >> $DIR_T2/time_4.txt
cat $DIR_T2/result_4.txt | grep "Mean latency" >> $DIR_T2/time_4.txt

# 2.5. 10MB
DATA_5='[{"topic":"test","data":"a'
```

```
for i in {1..10000}
do
    DATA_5=$DATA_5'a'
done
DATA_5=$DATA_5'"}]'
echo $MSG_2 "10MB" > $DIR_T2/result_5.txt
npx loadtest -n 1 -c 1 -m POST -T $TYPE --data $DATA_5 http://localhost:8000/ >> $DIR_T2/result_5.txt
# extract the time from the result
echo $MSG_2 "10MB" > $DIR_T2/time_5.txt
cat $DIR_T2/result_5.txt | grep "Requests per second" >> $DIR_T2/time_5.txt
cat $DIR_T2/result_5.txt | grep "Mean latency" >> $DIR_T2/time_5.txt

# 3. How does the number of requests impact the performance of the server?
MSG_3="number of requests: "
DIR_T3=$TEST_DIR/test_3
mkdir $DIR_T3
# 3.1. 1
echo $MSG_3 "1" > $DIR_T3/result_1.txt
npx loadtest -n 1 -c 1 -m POST -T $TYPE --data $DATA_0 http://localhost:8000/ >> $DIR_T3/result_1.txt
# extract the time from the result
echo $MSG_3 "1" > $DIR_T3/time_1.txt
cat $DIR_T3/result_1.txt | grep "Requests per second" >> $DIR_T3/time_1.txt
cat $DIR_T3/result_1.txt | grep "Mean latency" >> $DIR_T3/time_1.txt

# 3.2. 10
echo $MSG_3 "10" > $DIR_T3/result_2.txt
npx loadtest -n 10 -c 10 -m POST -T $TYPE --data $DATA_0 http://localhost:8000/ >> $DIR_T3/result_2.txt
# extract the time from the result
echo $MSG_3 "10" > $DIR_T3/time_2.txt
cat $DIR_T3/result_2.txt | grep "Requests per second" >> $DIR_T3/time_2.txt
cat $DIR_T3/result_2.txt | grep "Mean latency" >> $DIR_T3/time_2.txt

```

```
# 3.3. 100
echo $MSG_3 "100" > $DIR_T3/result_3.txt
npx loadtest -n 100 -c 100 -m POST -T $TYPE --data $DATA_0 http://localhost:8000/ >> $DIR_T3/result_3.txt
# extract the time from the result
echo $MSG_3 "100" > $DIR_T3/time_3.txt
cat $DIR_T3/result_3.txt | grep "Requests per second" >> $DIR_T3/time_3.txt
cat $DIR_T3/result_3.txt | grep "Mean latency" >> $DIR_T3/time_3.txt

# 3.4. 1000
echo $MSG_3 "1000" > $DIR_T3/result_4.txt
npx loadtest -n 1000 -c 1000 -m POST -T $TYPE --data $DATA_0 http://localhost:8000/ >> $DIR_T3/result_4.txt
# extract the time from the result
echo $MSG_3 "1000" > $DIR_T3/time_4.txt
cat $DIR_T3/result_4.txt | grep "Requests per second" >> $DIR_T3/time_4.txt
cat $DIR_T3/result_4.txt | grep "Mean latency" >> $DIR_T3/time_4.txt

# 3.5. 10000
echo $MSG_3 "10000" > $DIR_T3/result_5.txt
npx loadtest -n 10000 -c 10000 -m POST -T $TYPE --data $DATA_0 http://localhost:8000/ >> $DIR_T3/result_5.txt
# extract the time from the result
echo $MSG_3 "10000" > $DIR_T3/time_5.txt
cat $DIR_T3/result_5.txt | grep "Requests per second" >> $DIR_T3/time_5.txt
cat $DIR_T3/result_5.txt | grep "Mean latency" >> $DIR_T3/time_5.txt

# 4. How does the number of concurrent requests impact the performance of the server?
MSG_4="with 5 requests, number of concurrent requests: "
DIR_T4=$TEST_DIR/test_4
mkdir $DIR_T4
# 4.1. 1
echo $MSG_4 "1" > $DIR_T4/result_1.txt
npx loadtest -n 5 -c 1 -m POST -T $TYPE --data $DATA_0 http://localhost:8000/ >> $DIR_T4/result_1.txt
```

```
# extract the time from the result
echo $MSG_4 "1" > $DIR_T4/time_1.txt
cat $DIR_T4/result_1.txt | grep "Requests per second" >> $DIR_T4/time_1.txt
cat $DIR_T4/result_1.txt | grep "Mean latency" >> $DIR_T4/time_1.txt

# 4.2. 10
echo $MSG_4 "10" > $DIR_T4/result_2.txt
npx loadtest -n 5 -c 10 -m POST -T $TYPE --data $DATA_0 http://localhost:8000/ >> $DIR_T4/result_2.txt
# extract the time from the result
echo $MSG_4 "10" > $DIR_T4/time_2.txt
cat $DIR_T4/result_2.txt | grep "Requests per second" >> $DIR_T4/time_2.txt
cat $DIR_T4/result_2.txt | grep "Mean latency" >> $DIR_T4/time_2.txt

# 4.3. 100
echo $MSG_4 "100" > $DIR_T4/result_3.txt
npx loadtest -n 5 -c 100 -m POST -T $TYPE --data $DATA_0 http://localhost:8000/ >> $DIR_T4/result_3.txt
# extract the time from the result
echo $MSG_4 "100" > $DIR_T4/time_3.txt
cat $DIR_T4/result_3.txt | grep "Requests per second" >> $DIR_T4/time_3.txt
cat $DIR_T4/result_3.txt | grep "Mean latency" >> $DIR_T4/time_3.txt

# 4.4. 1000
echo $MSG_4 "1000" > $DIR_T4/result_4.txt
npx loadtest -n 5 -c 1000 -m POST -T $TYPE --data $DATA_0 http://localhost:8000/ >> $DIR_T4/result_4.txt
# extract the time from the result
echo $MSG_4 "1000" > $DIR_T4/time_4.txt
cat $DIR_T4/result_4.txt | grep "Requests per second" >> $DIR_T4/time_4.txt
cat $DIR_T4/result_4.txt | grep "Mean latency" >> $DIR_T4/time_4.txt

# 4.5. 10000
echo $MSG_4 "10000" > $DIR_T4/result_5.txt
```

```
182 npx loadtest -n 5 -c 10000 -m POST -T $TYPE --data $DATA_0
    http://localhost:8000/ >> $DIR_T4/result_5.txt
183 # extract the time from the result
184 echo $MSG_4 "10000" > $DIR_T4/time_5.txt
185 cat $DIR_T4/result_5.txt | grep "Requests per second" >>
    $DIR_T4/time_5.txt
186 cat $DIR_T4/result_5.txt | grep "Mean latency" >>
    $DIR_T4/time_5.txt
```

While the docker container is running, entering bash of the container

```
1 docker exec -it <container_id> bash
```

Running the script inside the container's bash:

```
1 chmod +x loadtest.sh
2 ./loadtest.sh
```

The outputs are stored in `loadtestData` directory.

The analysis is mainly on two aspects:
- The number of requests per second
- The mean latency

Analysis

Single request

How long does it take to process a single post (performance)?

NUMBER OF REQUESTS	REQUESTS PER SECOND	MEAN LATENCY
1	25	37.8 ms

Size of the data sent

Does the size of the data submitted to the server impact the performance?

SIZE OF DATA	REQUESTS PER SECOND	MEAN LATENCY
1 kB	20	46.8 ms
10 kB	20	46.2 ms
100 kB	29	31.6 ms
1 MB	27	34.9 ms
10 MB	28	32.3 ms

Conclusion on size of data

It seems that the size of the data does not impact the performance of the server. The number of requests per second and the mean latency are almost the same for all the data sizes.

Number of requests

How does the number of requests impact the performance of the server?

with same small data size (1 kB) and concurrency level (1)

NUMBER OF REQUESTS	REQUESTS PER SECOND	MEAN LATENCY
1	20	46.5 ms
10	119	59.2 ms
100	360	234.7 ms
1000	214	16827.6 ms
10000	408	2231.1 ms

Conclusion on number of requests

It seems that, as the number of requests increases, the performance of the server also increase up to a certain point. No server error occured for the 5 tests.

I assume that this is because the backend daemon find an optimization to process the requests faster.

Number of concurrent requests

How does the level of concurrency impact the performance of the server?

Use the same small data size (1 kB) and number of requests (5)

NUMBER OF CONCURRENT REQUESTS	REQUESTS PER SECOND	MEAN LATENCY
1	110	8.2 ms
10	79	44.8 ms
100	106	27.5 ms
1000	71	51.3 ms

Conclusion on number of concurrent requests

It seems that as the number of concurrent requests increases, the performance of the server decreases. However, the affect is not severe and there are no server errors for all test cases. I suspect that, with a higher number of concurrent requests, the server will start to have errors.