UberEats Clone LAB 2 Report

Himaja Chandaluri | SJSU ID - 015247323

YouTube Link: https://youtu.be/yYFi2TmFZ5o

Goal: The goal of this project is to build new features on top of the already developed prototype of UberEats application. This prototype is developed using React, Node, Redux, Kafka and MongoDB.

Purpose: The purpose of the project is to get hands-on experience on the latest technologies such as React, Node, and Redux. Learn how to establish connections to database. Learn how a client and server architecture is built. How to pass data to client browsers through middleware and populate the data on a single page application.

System Design:

Client Side: I have designed the web application's user interface using React and React Bootstrap for styling and customization. Npm package axios is used to make calls to the backend. Additionally, supporting library, Redux is used in order to maintain a global store.

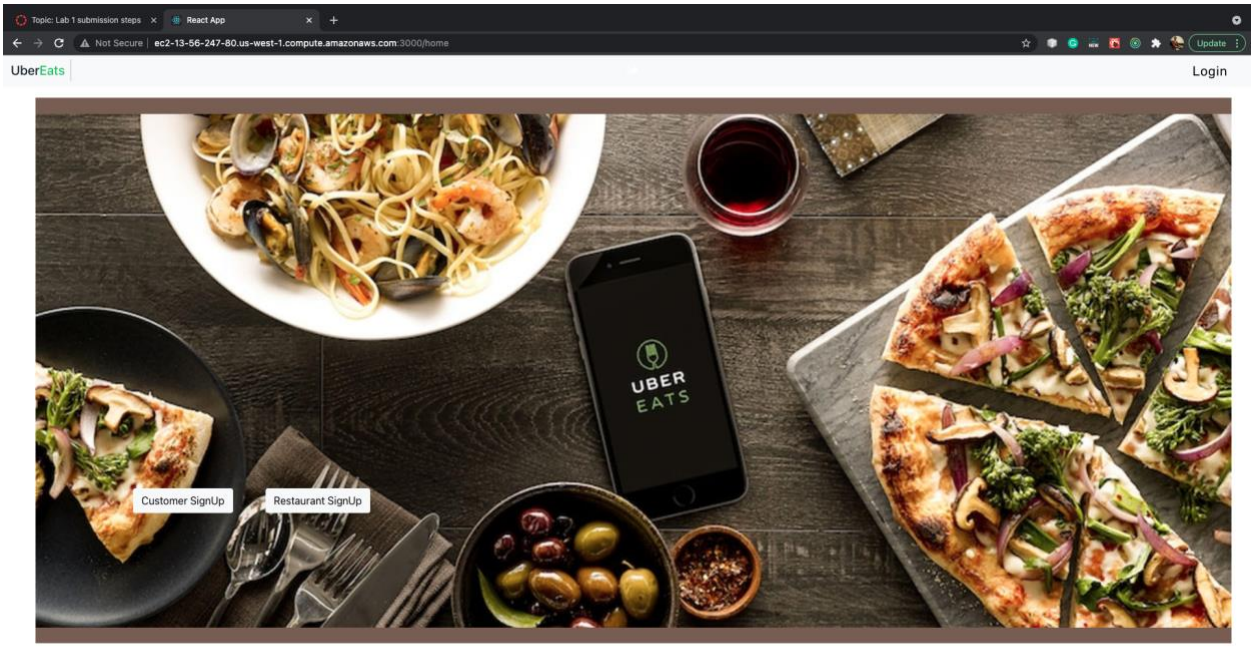
Server Side: I have used NodeJS, ExpressJS to handle incoming requests from the client end. The server will communicate with the database through kafka message queues. Used MongoDB as the database.

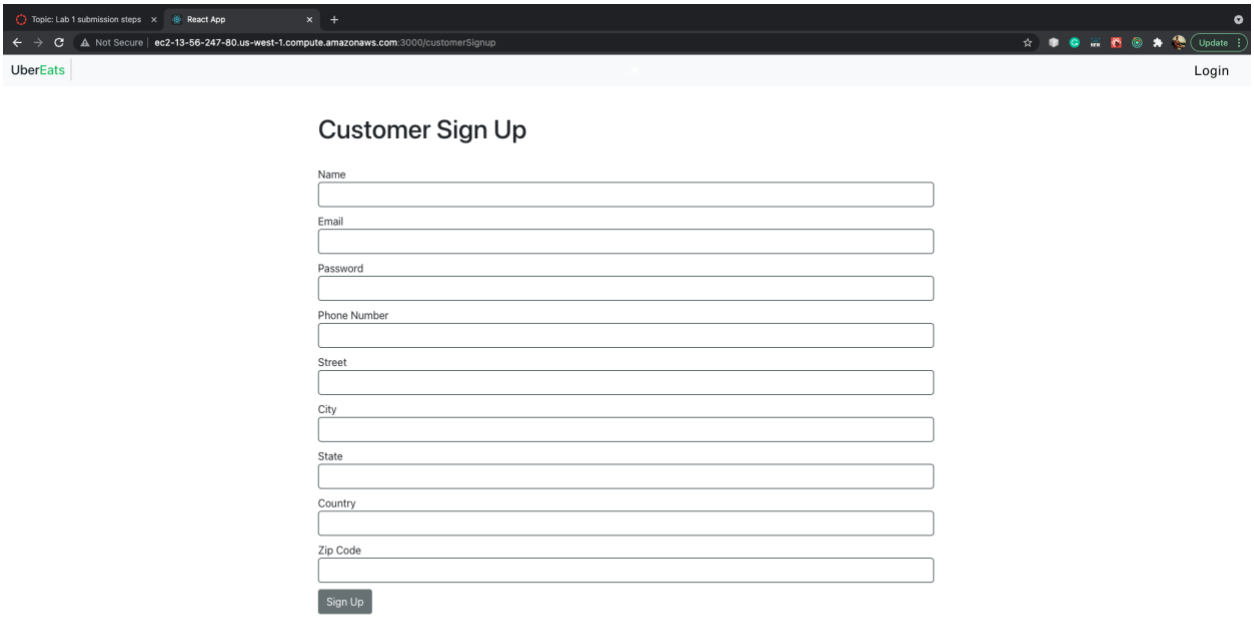
Database: I am using MongoDB database.

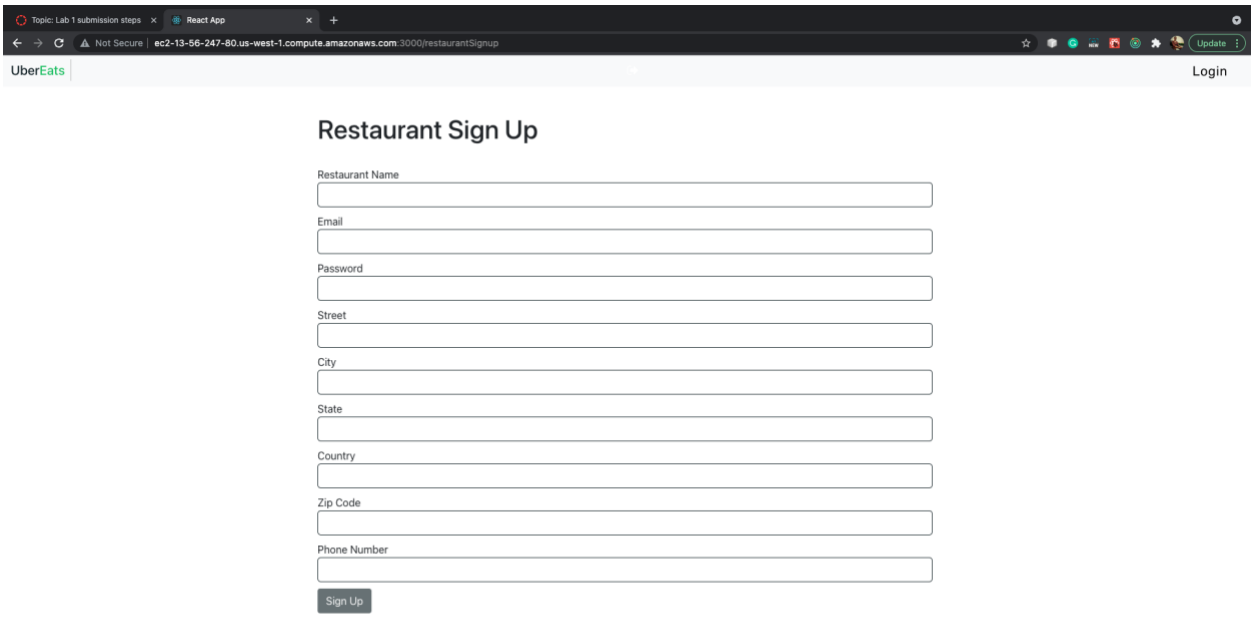
Additionally, used bcrypt to encrypt sensitive information like passwords. Used Passport-JWT for authentication and authorization. Application deployed on EC2 instance.

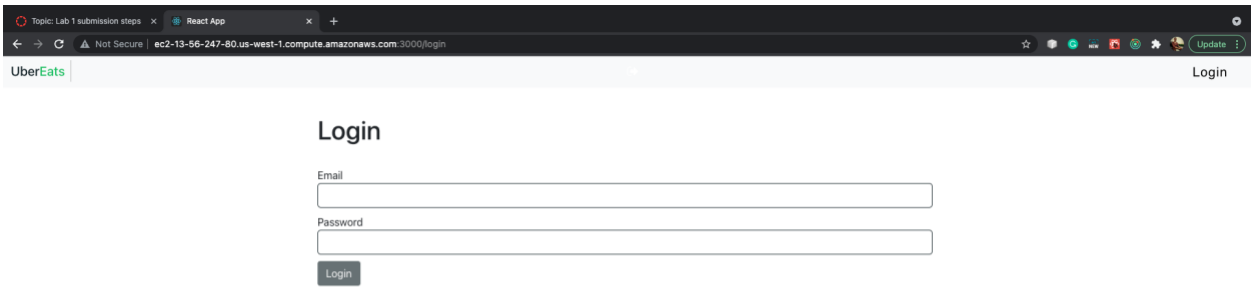
UI Screenshots:

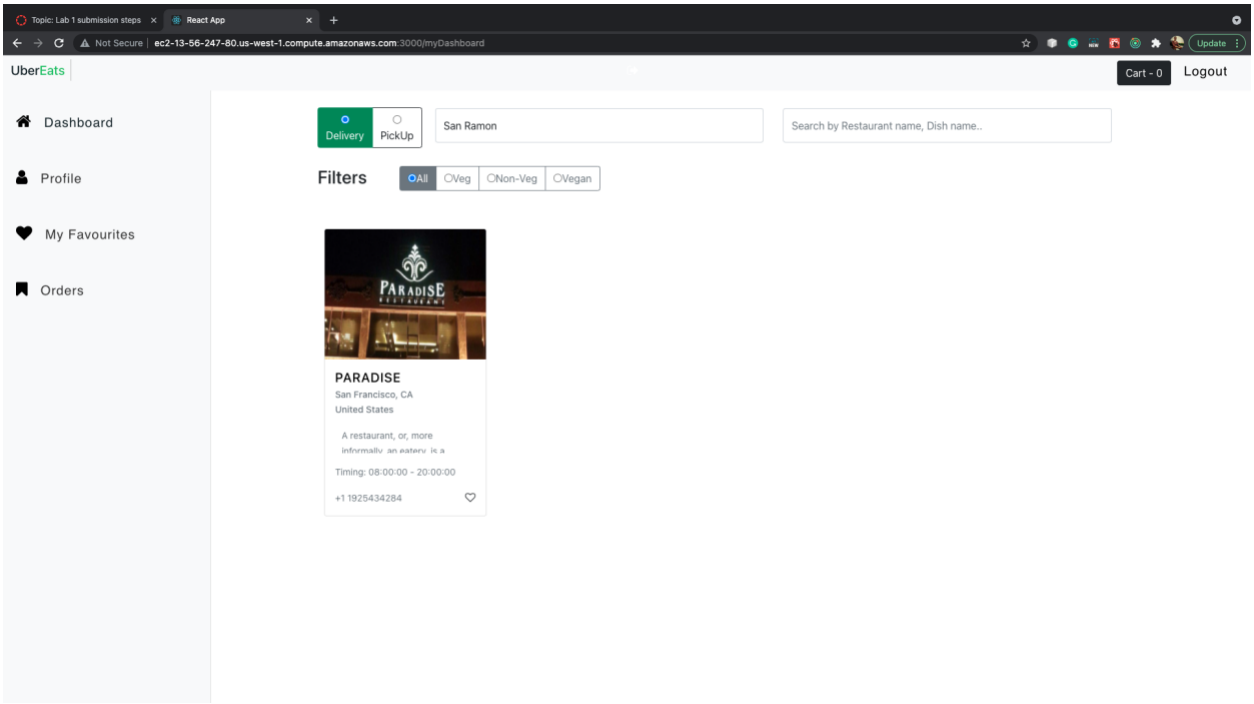
New features added were pagination in the customer's orders page. Accepting note while placing order.

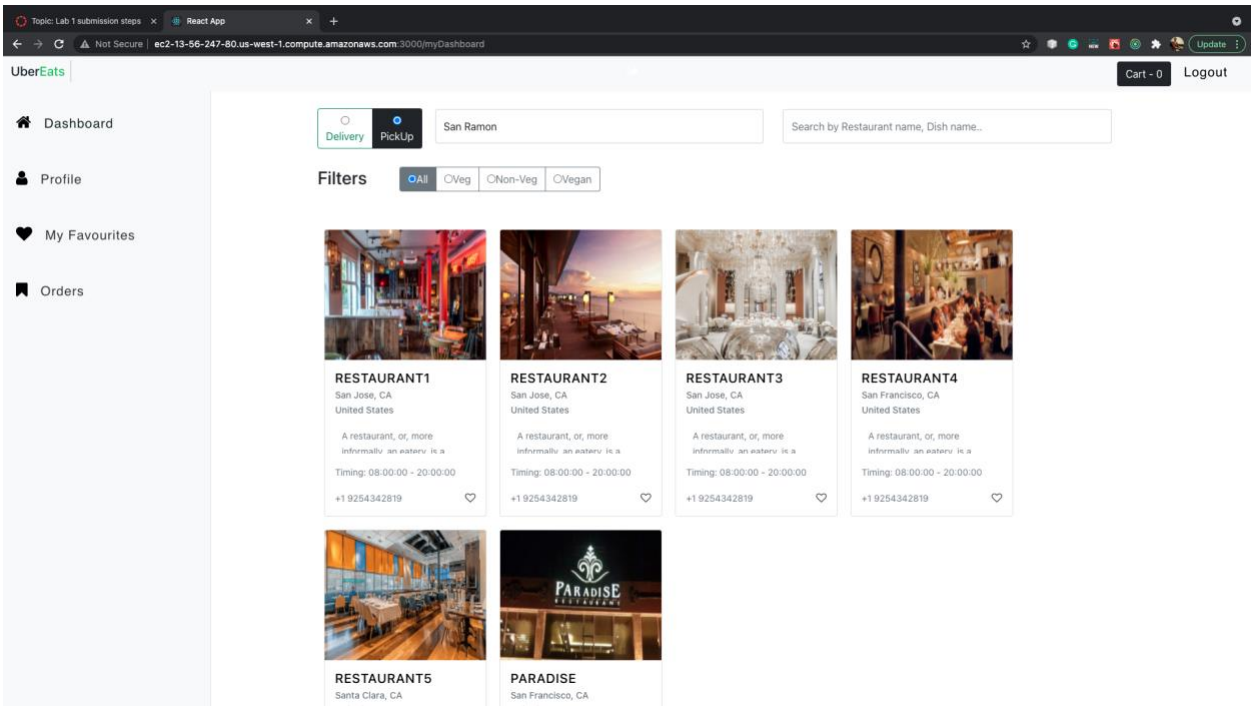


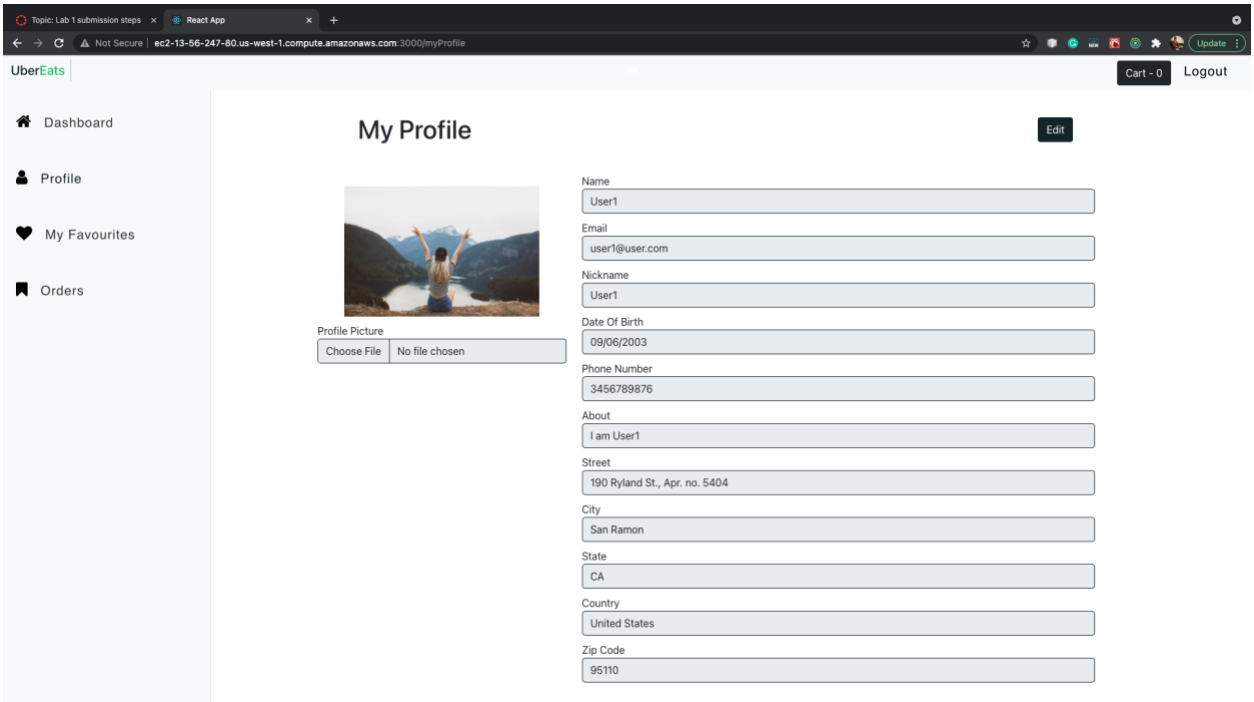

Topic: Lab 1 submission steps
React App
Not Secure
ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/customerSignup
Update
UberEats
Login
Customer Sign Up
Name
Email
Password
Phone Number
Street
City
State
Country
Zip Code
Sign Up

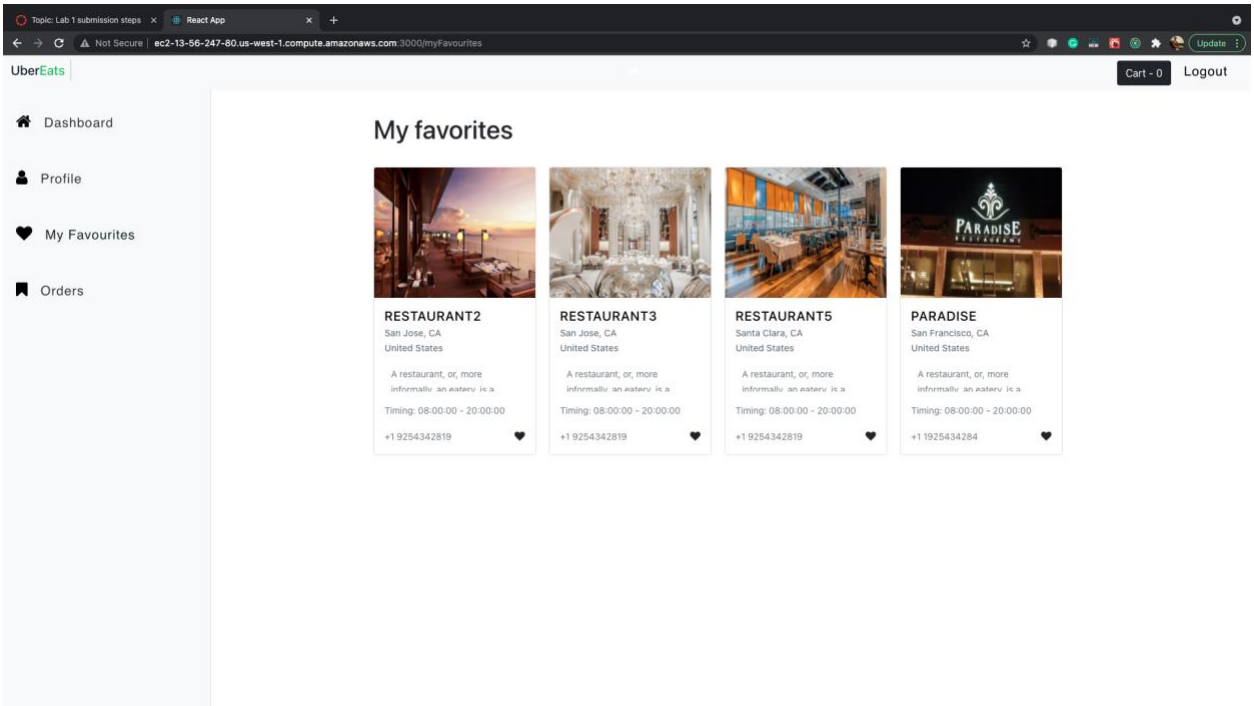

Topic: Lab 1 submission steps
React App
Not Secure
ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/restaurantSignup
Update
UberEats
Login
Restaurant Sign Up
Restaurant Name
Email
Password
Street
City
State
Country
Zip Code
Phone Number
Sign Up

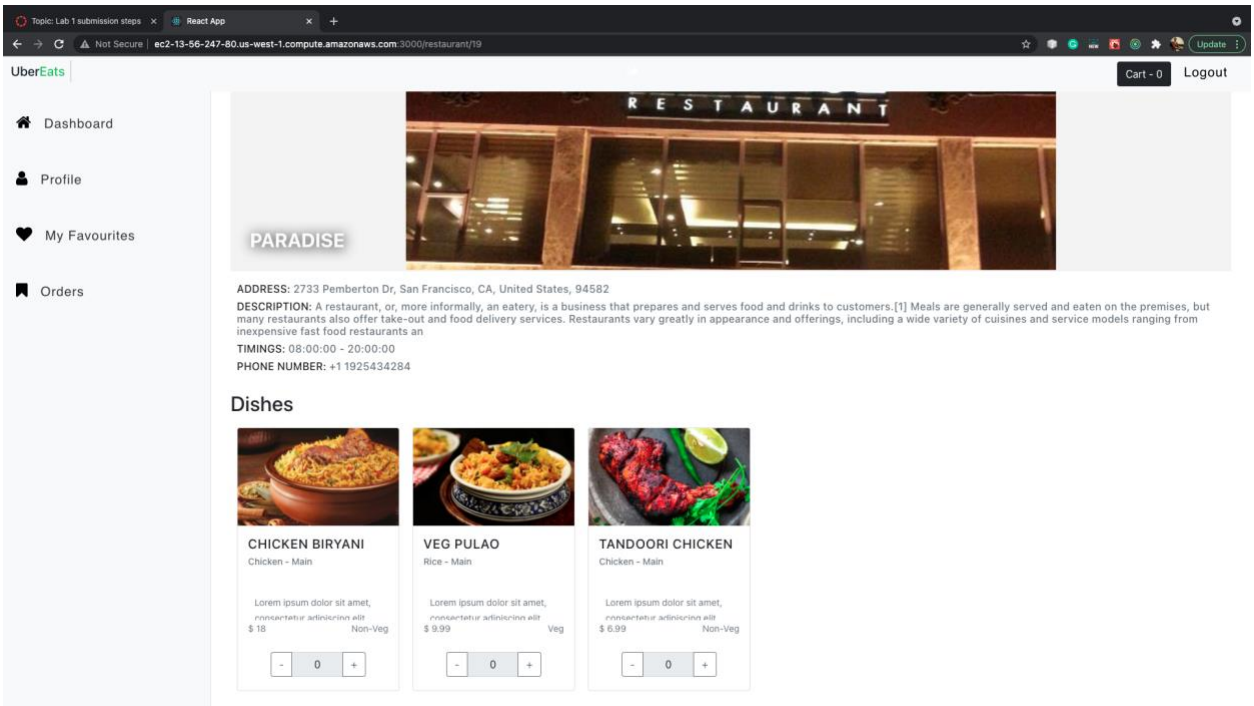

Topic: Lab 1 submission steps
React App
Not Secure
ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/login
Update
UberEats
Login
Login
Email
Password
Login

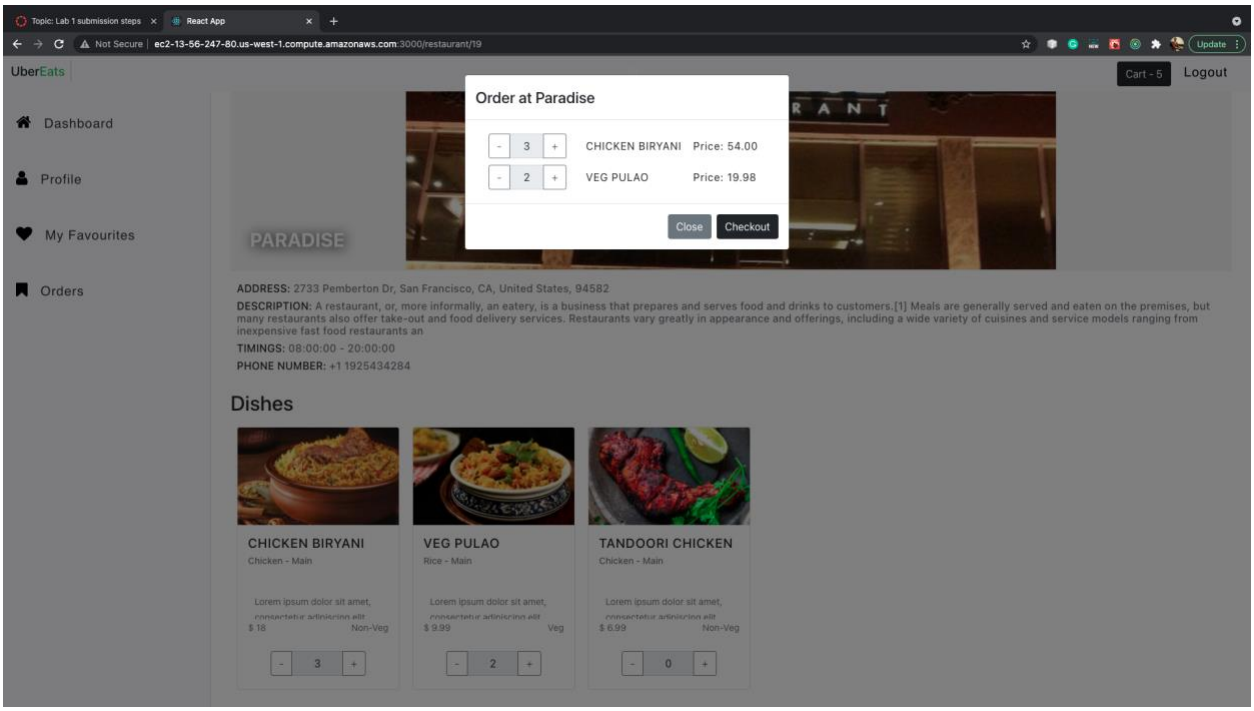

Topic: Lab 1 submission steps
React App
Not Secure
ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/myDashboard
Update
UberEats
Cart - 0
Logout
Dashboard
Profile
My Favourites
Orders
Delivery
PickUp
San Ramon
Search by Restaurant name, Dish name..
Filters
All
Veg
Non-Veg
Vegan
PARADISE
PARADISE
San Francisco, CA
United States
A restaurant, or, more
Timing: 08:00:00 - 20:00:00
+1 1925434284

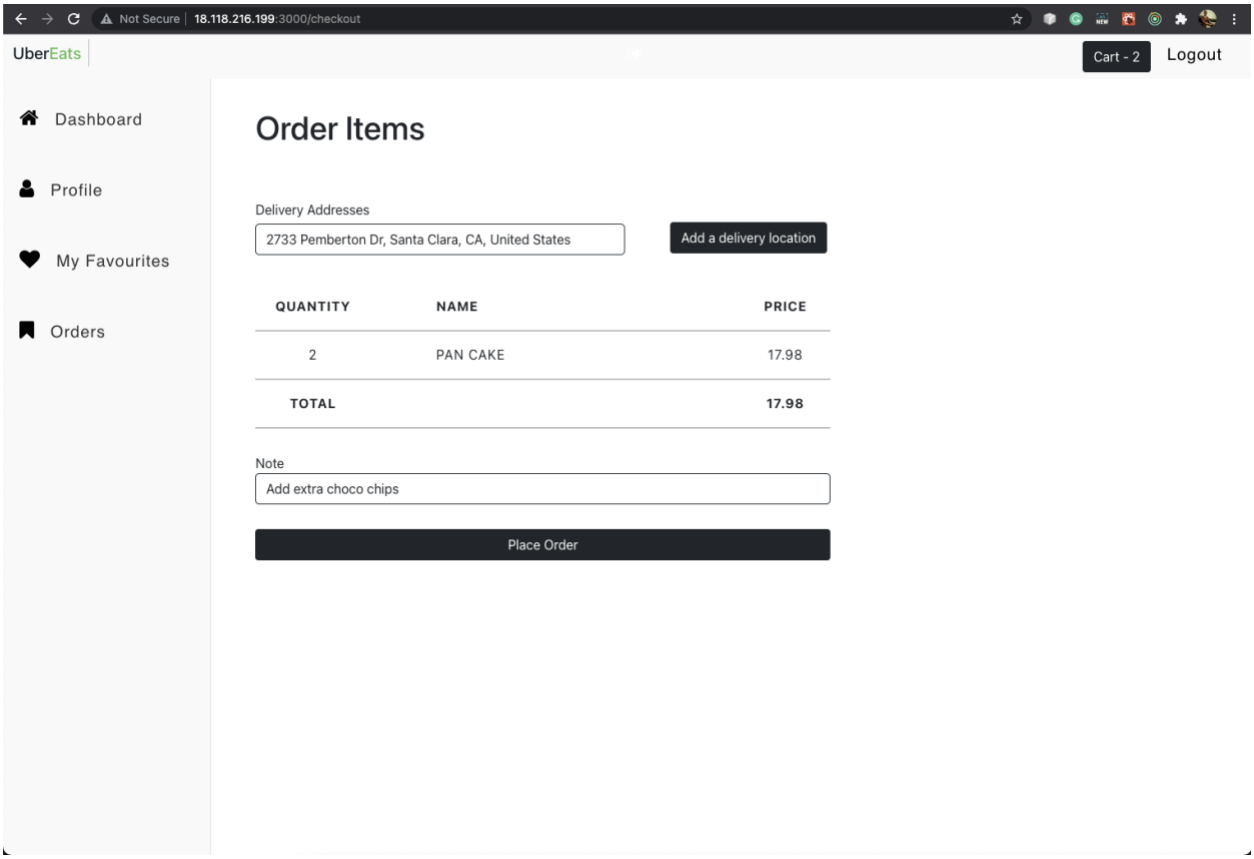

Topic: Lab 1 submission steps
React App
Not Secure
ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/myDashboard
Update
UberEats
Cart - 0
Logout
Dashboard
Profile
My Favourites
Orders
Delivery
PickUp
San Ramon
Search by Restaurant name, Dish name..
Filters
All
Veg
Non-Veg
Vegan
RESTAURANT1
San Jose, CA
United States
A restaurant, or, more
Timing: 08:00:00 - 20:00:00
+1 9254342819
RESTAURANT2
San Jose, CA
United States
A restaurant, or, more
Timing: 08:00:00 - 20:00:00
+1 9254342819
RESTAURANT3
San Jose, CA
United States
A restaurant, or, more
Timing: 08:00:00 - 20:00:00
+1 9254342819
RESTAURANT4
San Francisco, CA
United States
A restaurant, or, more
Timing: 08:00:00 - 20:00:00
+1 9254342819
PARADISE
RESTAURANT5
Santa Clara, CA
PARADISE
San Francisco, CA

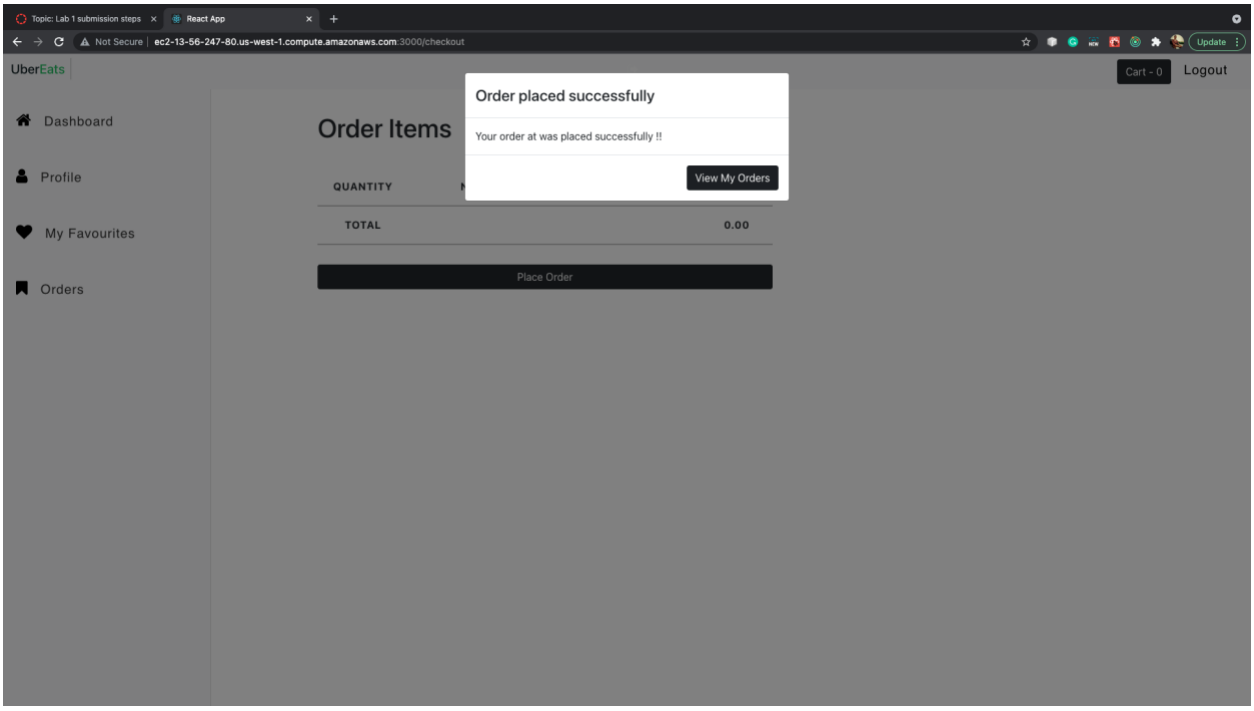

Topic: Lab 1 submission steps
React App
Not Secure
ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/myProfile
Update
UberEats
Cart - 0
Logout
Dashboard
Profile
My Favourites
Orders
My Profile
Edit
Profile Picture
Choose File
No file chosen
Name
User1
Email
user1@user.com
Nickname
User1
Date Of Birth
09/06/2003
Phone Number
3456789876
About
I am User1
Street
190 Ryland St., Apr. no. 5404
City
San Ramon
State
CA
Country
United States
Zip Code
95110

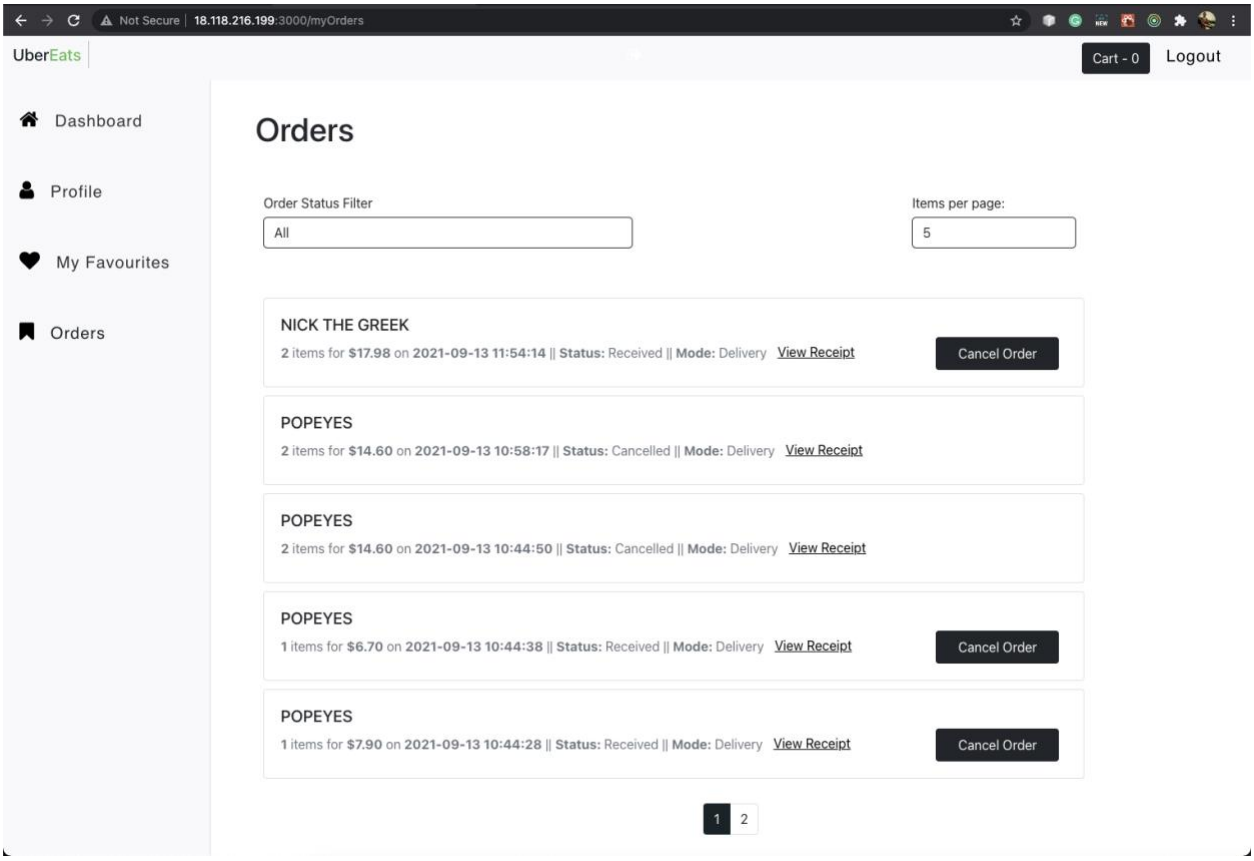

Topic: Lab 1 submission steps
React App
Not Secure | ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/myFavourites
Update
UberEats
Cart - 0
Logout
Dashboard
Profile
My Favourites
Orders
My favorites
RESTAURANT2
San Jose, CA
United States
A restaurant, or, more
Timing: 08:00:00 - 20:00:00
+1 9254342819
RESTAURANT3
San Jose, CA
United States
A restaurant, or, more
Timing: 08:00:00 - 20:00:00
+1 9254342819
RESTAURANT5
Santa Clara, CA
United States
A restaurant, or, more
Timing: 08:00:00 - 20:00:00
+1 9254342819
PARADISE
San Francisco, CA
United States
A restaurant, or, more
Timing: 08:00:00 - 20:00:00
+1 1925434284

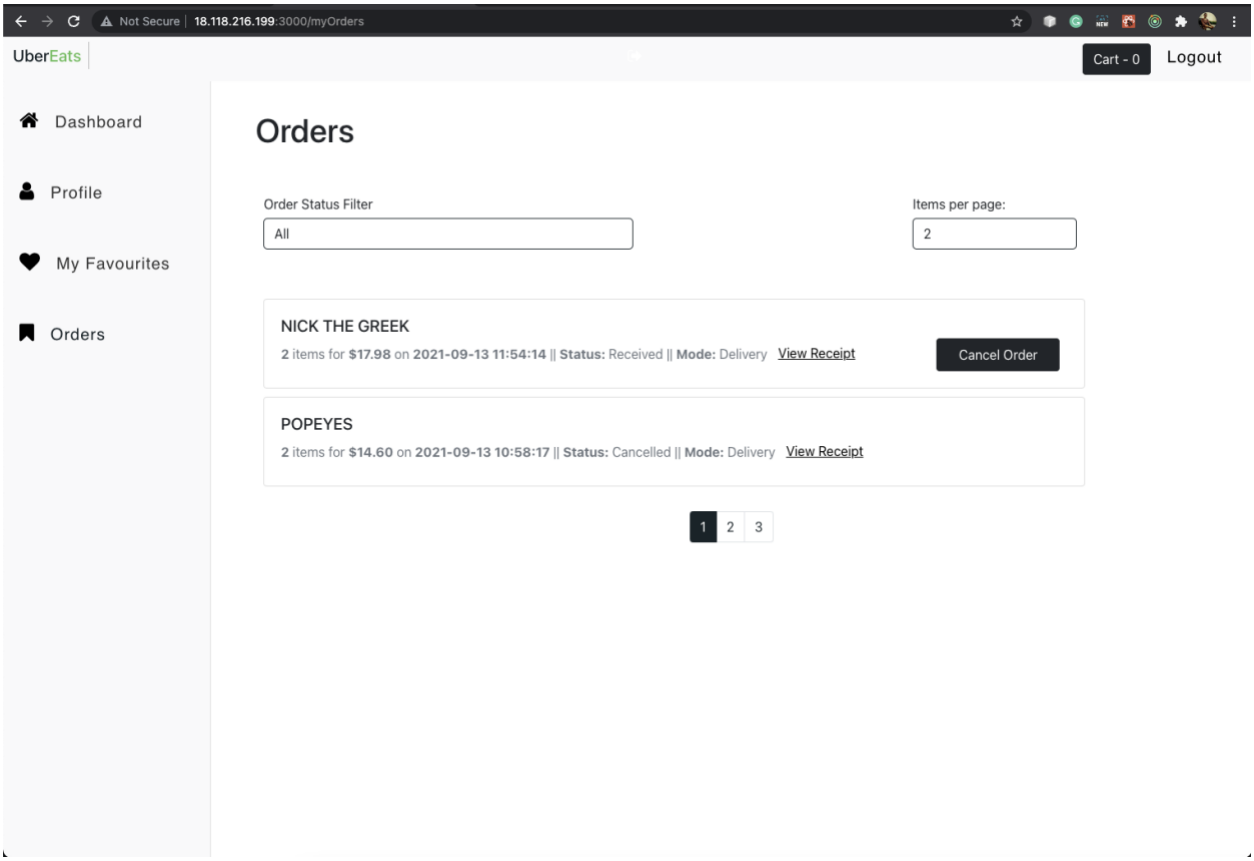

Topic: Lab 1 submission steps
React App
Not Secure | ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/restaurant/19
Update
UberEats
Cart - 0
Logout
Dashboard
Profile
My Favourites
Orders
RESTAURANT
PARADISE
ADDRESS: 2733 Pemberton Dr, San Francisco, CA, United States, 94582
DESCRIPTION: A restaurant, or, more informally, an eatery, is a business that prepares and serves food and drinks to customers.[1] Meals are generally served and eaten on the premises, but many restaurants also offer take-out and food delivery services. Restaurants vary greatly in appearance and offerings, including a wide variety of cuisines and service models ranging from inexpensive fast food restaurants an
TIMINGS: 08:00:00 - 20:00:00
PHONE NUMBER: +1 1925434284
Dishes
CHICKEN BIRYANI
Chicken - Main
Lorem ipsum dolor sit amet,
$ 18
Non-Veg
VEG PULAO
Rice - Main
Lorem ipsum dolor sit amet,
$ 9.99
Veg
TANDOORI CHICKEN
Chicken - Main
Lorem ipsum dolor sit amet,
$ 6.99
Non-Veg

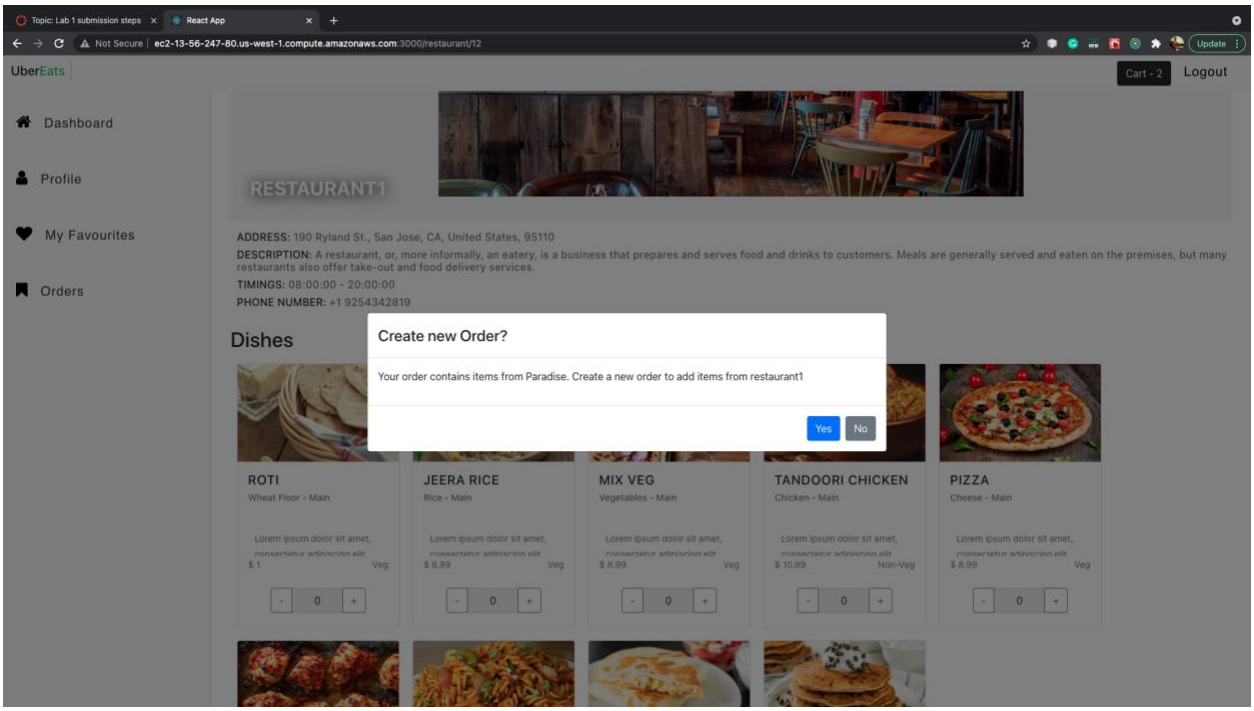

Topic: Lab 1 submission steps
React App
Not Secure | ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/restaurant/19
Update
UberEats
Cart - 5
Logout
Dashboard
Profile
My Favourites
Orders
Order at Paradise
3
CHICKEN BIRYANI
Price: 54.00
2
VEG PULAO
Price: 19.98
Close
Checkout
PARADISE
ADDRESS: 2733 Pemberton Dr, San Francisco, CA, United States, 94582
DESCRIPTION: A restaurant, or, more informally, an eatery, is a business that prepares and serves food and drinks to customers.[1] Meals are generally served and eaten on the premises, but many restaurants also offer take-out and food delivery services. Restaurants vary greatly in appearance and offerings, including a wide variety of cuisines and service models ranging from inexpensive fast food restaurants an
TIMINGS: 08:00:00 - 20:00:00
PHONE NUMBER: +1 1925434284
Dishes
CHICKEN BIRYANI
Chicken - Main
$ 18
Non-Veg
3
VEG PULAO
Rice - Main
$ 9.99
Veg
2
TANDOORI CHICKEN
Chicken - Main
$ 6.99
Non-Veg
0

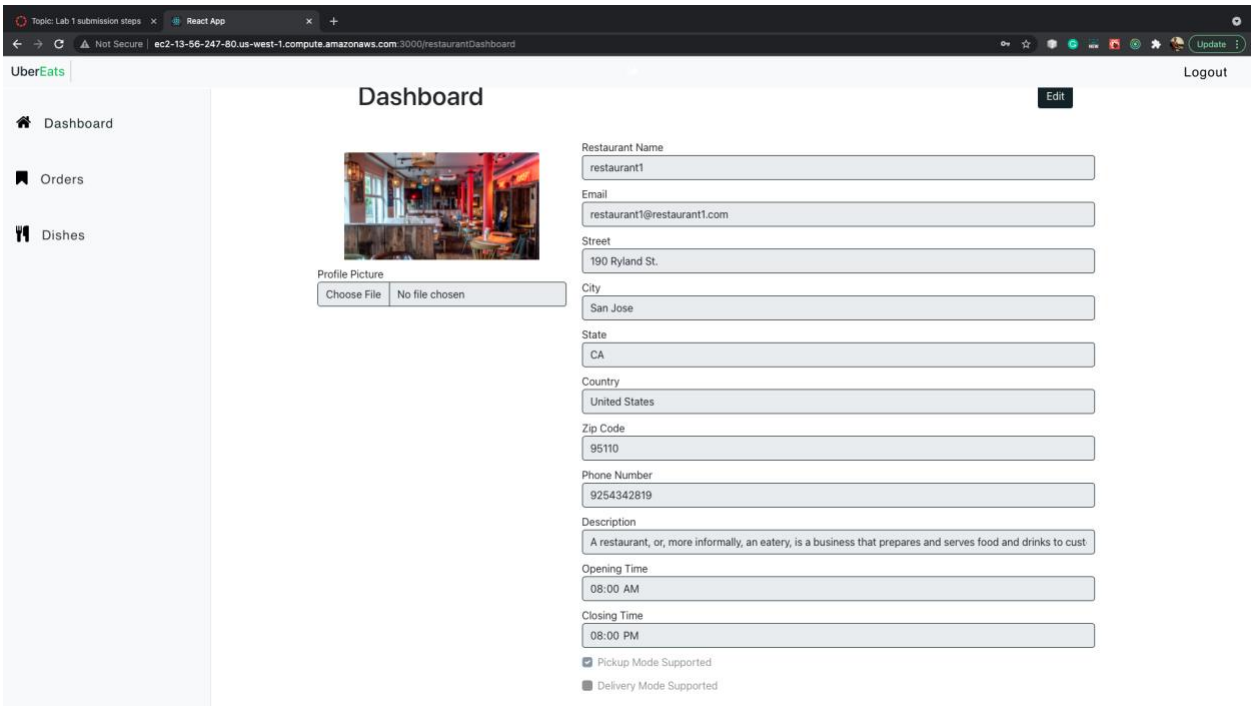

Not Secure | 18.118.216.199:3000/checkout
UberEats
Cart - 2
Logout
Dashboard
Profile
My Favourites
Orders
Order Items
Delivery Addresses
2733 Pemberton Dr, Santa Clara, CA, United States
Add a delivery location
QUANTITY
NAME
PRICE
2
PAN CAKE
17.98
TOTAL
17.98
Note
Add extra choco chips
Place Order

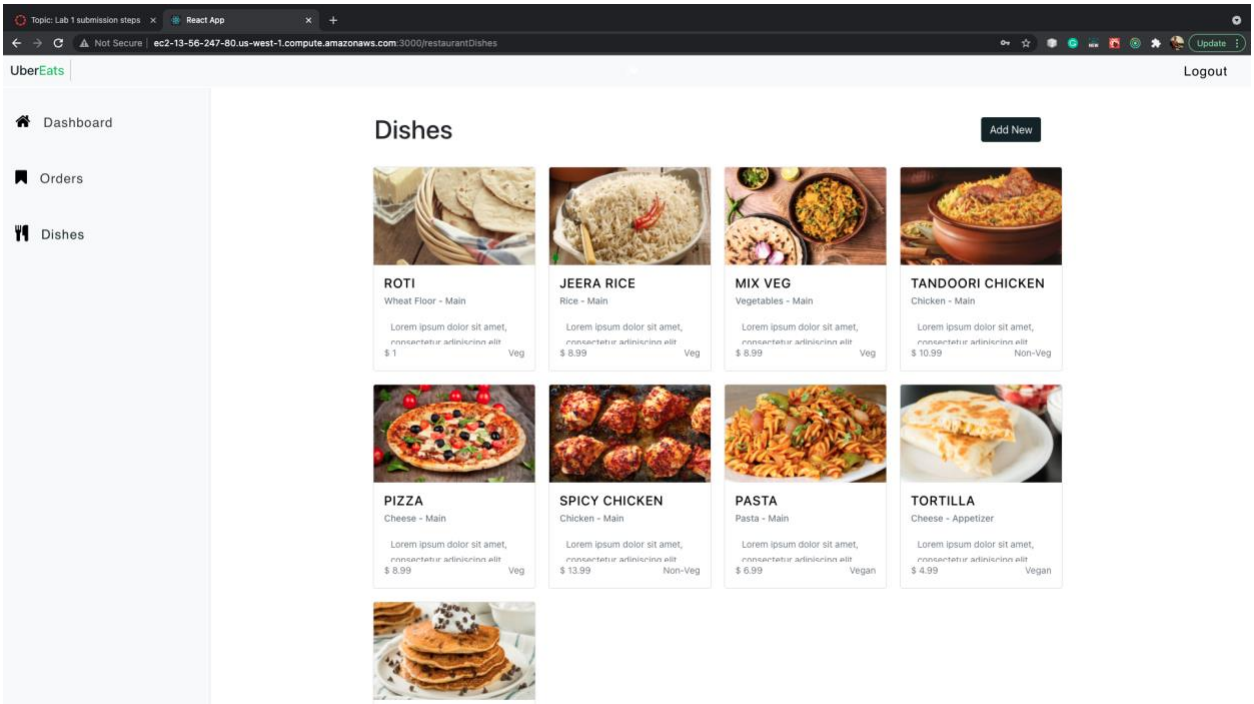

Topic: Lab 1 submission steps
React App
Not Secure | ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/checkout
Update
UberEats
Cart - 0
Logout
Dashboard
Profile
My Favourites
Orders
Order Items
QUANTITY
TOTAL
0.00
Place Order
Order placed successfully
Your order at was placed successfully !!
View My Orders

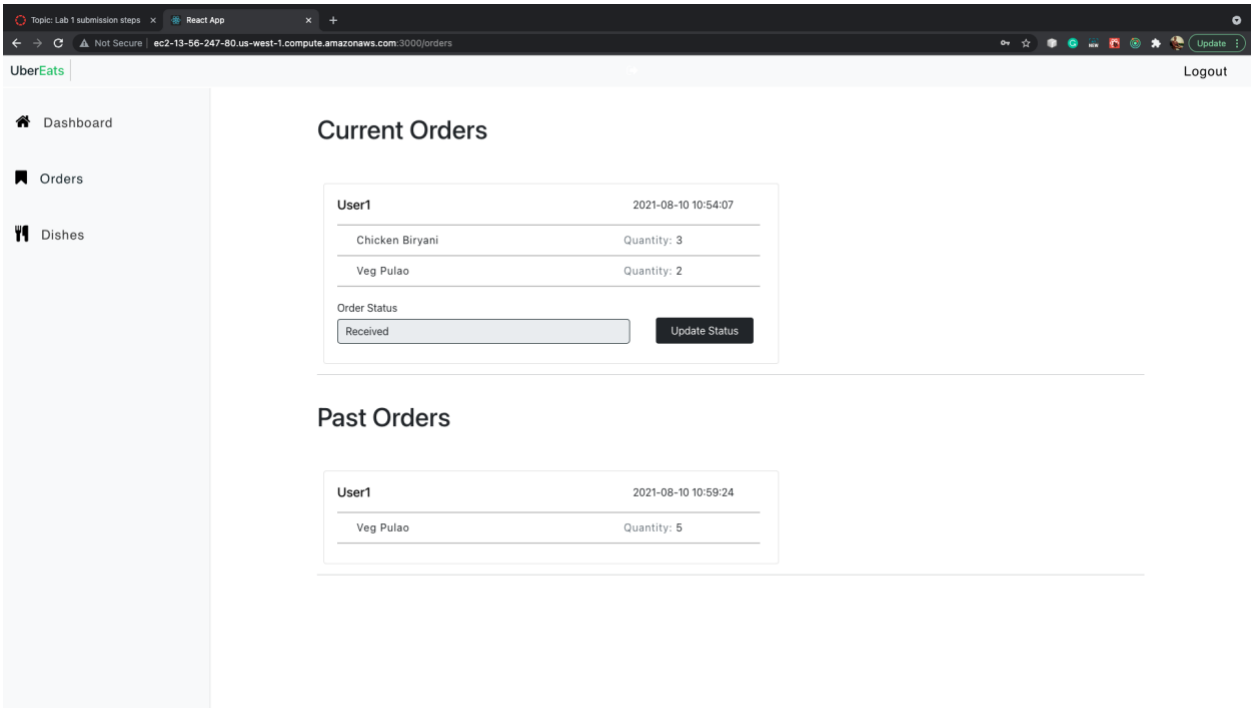

Not Secure | 18.118.216.199:3000/myOrders
UberEats
Cart - 0
Logout
Dashboard
Profile
My Favourites
Orders
Orders
Order Status Filter
All
Items per page:
5
NICK THE GREEK
2 items for $17.98 on 2021-09-13 11:54:14 || Status: Received || Mode: Delivery View Receipt
Cancel Order
POPEYES
2 items for $14.60 on 2021-09-13 10:58:17 || Status: Cancelled || Mode: Delivery View Receipt
POPEYES
2 items for $14.60 on 2021-09-13 10:44:50 || Status: Cancelled || Mode: Delivery View Receipt
POPEYES
1 items for $6.70 on 2021-09-13 10:44:38 || Status: Received || Mode: Delivery View Receipt
Cancel Order
POPEYES
1 items for $7.90 on 2021-09-13 10:44:28 || Status: Received || Mode: Delivery View Receipt
Cancel Order
1
2


Not Secure | 18.118.216.199:3000/myOrders
UberEats
Cart - 0
Logout
Dashboard
Profile
My Favourites
Orders
Orders
Order Status Filter
All
Items per page:
2
NICK THE GREEK
2 items for $17.98 on 2021-09-13 11:54:14 || Status: Received || Mode: Delivery View Receipt
Cancel Order
POPEYES
2 items for $14.60 on 2021-09-13 10:58:17 || Status: Cancelled || Mode: Delivery View Receipt
1 2 3


Topic: Lab 1 submission steps
React App
Not Secure | ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/restaurant/12
Update
UberEats
Cart - 2
Logout
Dashboard
Profile
My Favourites
Orders
RESTAURANT1
ADDRESS: 190 Ryland St., San Jose, CA, United States, 95110
DESCRIPTION: A restaurant, or, more informally, an eatery, is a business that prepares and serves food and drinks to customers. Meals are generally served and eaten on the premises, but many restaurants also offer take-out and food delivery services.
TIMINGS: 08:00:00 - 20:00:00
PHONE NUMBER: +1 9254342819
Dishes
Create new Order?
Your order contains items from Paradise. Create a new order to add items from restaurant1
Yes
No
ROTI
Wheat Floor - Main
JEERA RICE
Rice - Main
MIX VEG
Vegetables - Main
TANDOORI CHICKEN
Chicken - Main
PIZZA
Cheese - Main


Topic: Lab 1 submission steps
React App
Not Secure
ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/restaurantDashboard
Update
UberEats
Logout
Dashboard
Orders
Dishes
Dashboard
Edit
Profile Picture
Choose File
No file chosen
Restaurant Name
restaurant1
Email
restaurant1@restaurant1.com
Street
190 Ryland St.
City
San Jose
State
CA
Country
United States
Zip Code
95110
Phone Number
9254342819
Description
A restaurant, or, more informally, an eatery, is a business that prepares and serves food and drinks to cust
Opening Time
08:00 AM
Closing Time
08:00 PM
Pickup Mode Supported
Delivery Mode Supported


Topic: Lab 1 submission steps
React App
Not Secure
ec2-13-56-247-80.us-west-1.compute.amazonaws.com:3000/restaurantDishes
Update
UberEats
Logout
Dashboard
Orders
Dishes
Dishes
Add New
ROTI
Wheat Floor - Main
$ 1
Veg
JEERA RICE
Rice - Main
$ 8.99
Veg
MIX VEG
Vegetables - Main
$ 8.99
Veg
TANDOORI CHICKEN
Chicken - Main
$ 10.99
Non-Veg
PIZZA
Cheese - Main
$ 8.99
Veg
SPICY CHICKEN
Chicken - Main
$ 13.99
Non-Veg
PASTA
Pasta - Main
$ 6.99
Vegan
TORTILLA
Cheese - Appetizer
$ 4.99
Vegan

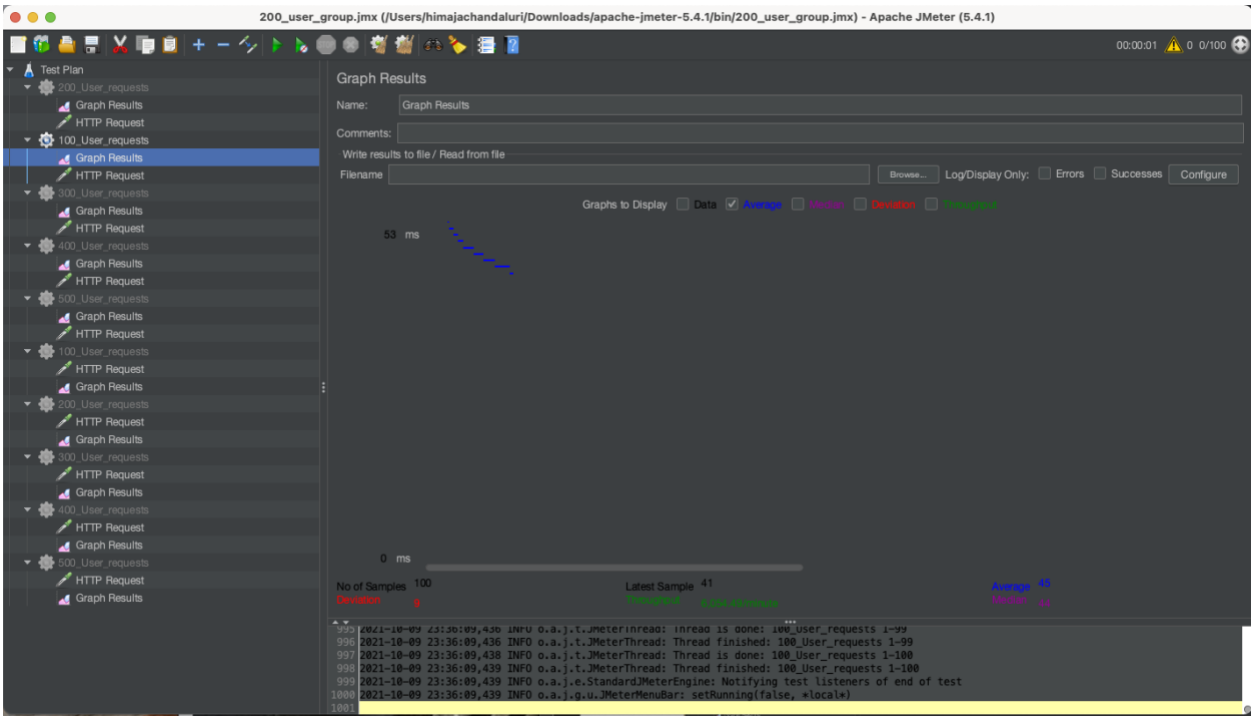


JMeter Testing:

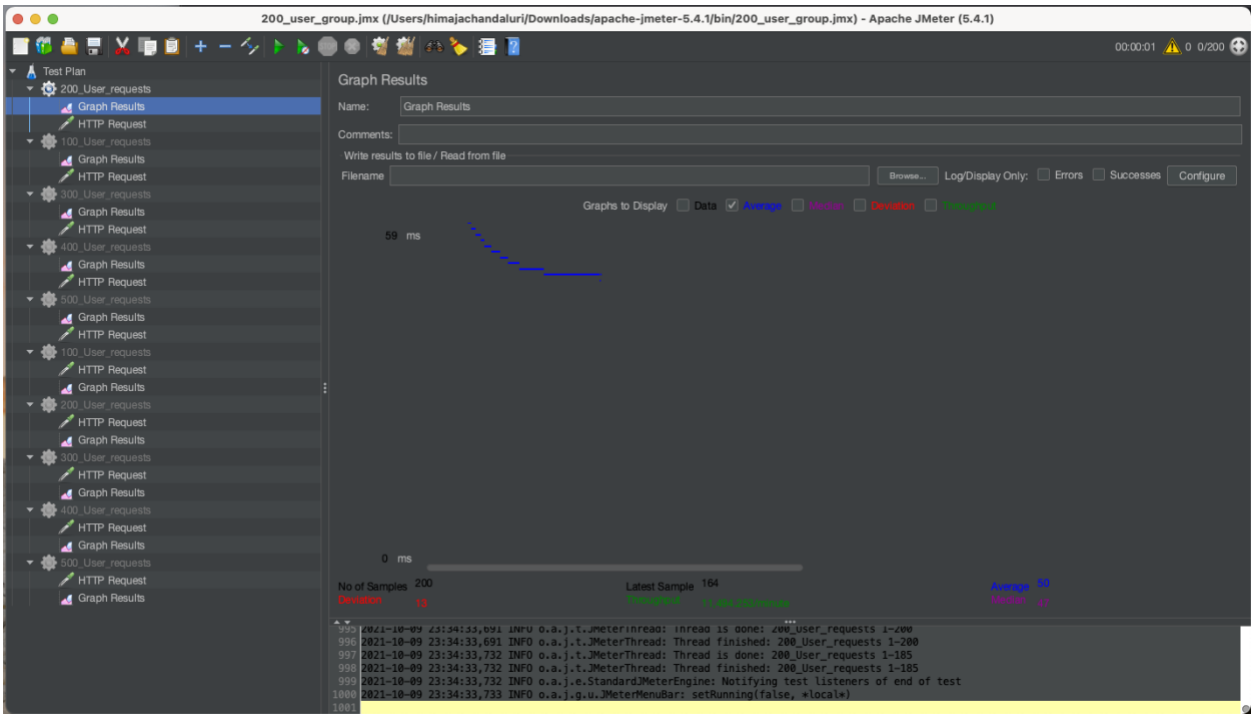
The application was tested with JMeter using 100, 200, 300, 400 and 500 threads. The throughput averaged around 6000 per minute with connection pooling and around 3000 without connection pooling.

With Connection Pooling – (mongodb connection pool - 20)

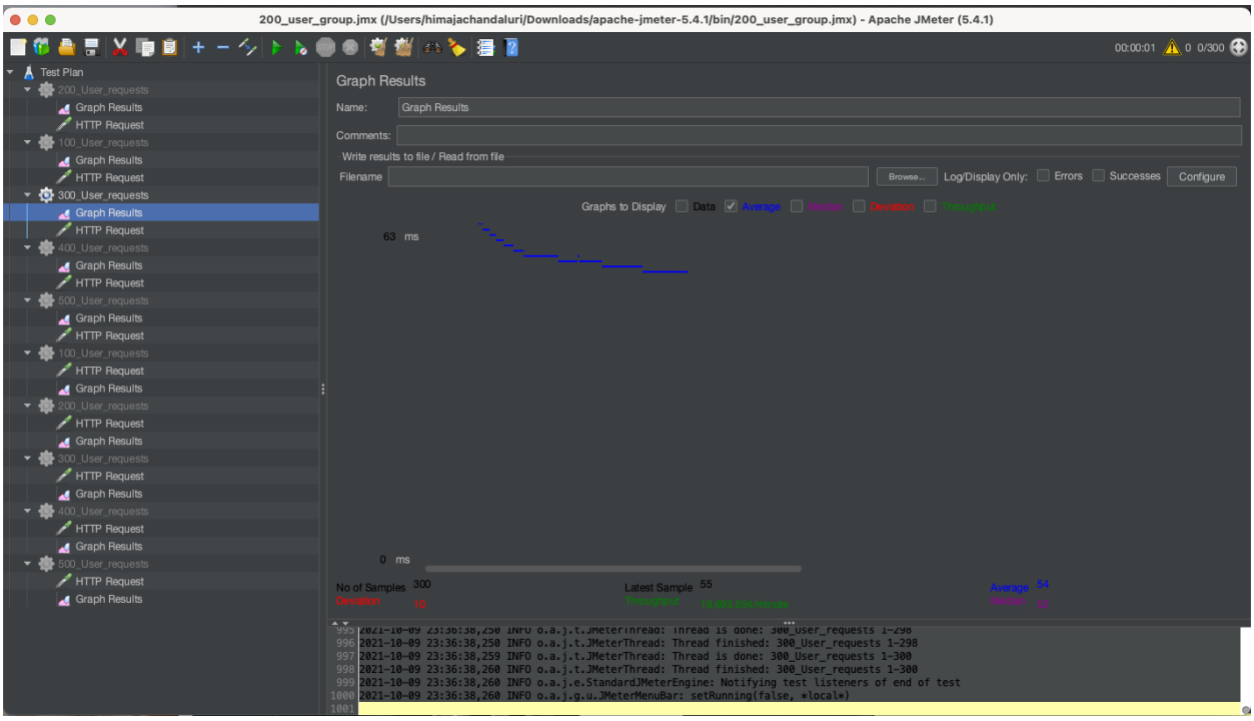
100 –


200_user_group.jmx (/Users/himajachandaluri/Downloads/apache-jmeter-5.4.1/bin/200_user_group.jmx) - Apache JMeter (5.4.1)
00:00:01 0 0/100
Test Plan
200_User_requests
Graph Results
HTTP Request
100_User_requests
300_User_requests
400_User_requests
500_User_requests
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Graphs to Display
Data
Average
Median
Deviation
Throughput
53 ms
0 ms
No of Samples 100
Latest Sample 41
Deviation 9
Average 48
Median 44
2021-10-09 23:36:09,436 INFO o.a.j.t.JMeterThread: Thread finished: 100_User_requests 1-99
2021-10-09 23:36:09,438 INFO o.a.j.t.JMeterThread: Thread is done: 100_User_requests 1-100
2021-10-09 23:36:09,439 INFO o.a.j.t.JMeterThread: Thread finished: 100_User_requests 1-100
2021-10-09 23:36:09,439 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
2021-10-09 23:36:09,439 INFO o.a.j.g.u.JMeterMenuBar: setRunning(false, *local*)

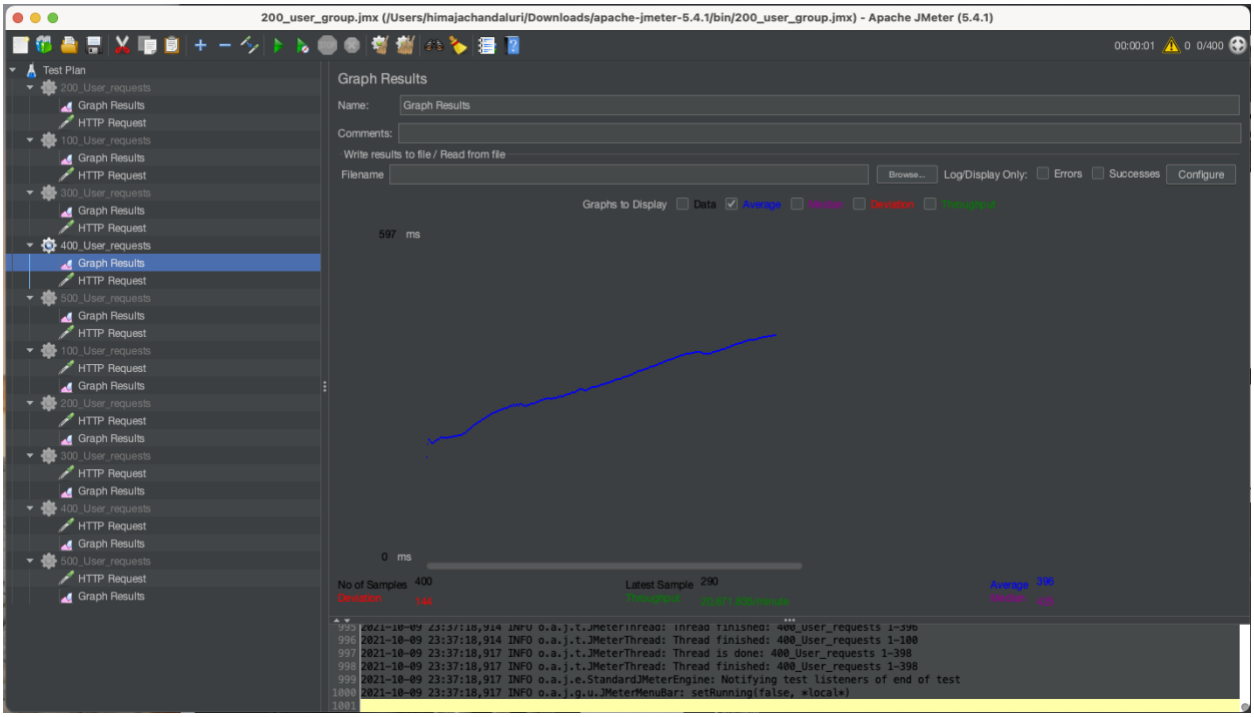
200 –


200_user_group.jmx (/Users/himajachandaluri/Downloads/apache-jmeter-5.4.1/bin/200_user_group.jmx) - Apache JMeter (5.4.1)
00:00:01 0 0/200
Test Plan
200_User_requests
Graph Results
HTTP Request
100_User_requests
300_User_requests
400_User_requests
500_User_requests
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Graphs to Display
Data
Average
Median
Deviation
Throughput
59 ms
0 ms
No of Samples 200
Latest Sample 164
Deviation 13
Average 50
Median 47
2021-10-09 23:34:33,691 INFO o.a.j.t.JMeterThread: Thread finished: 200_User_requests 1-200
2021-10-09 23:34:33,732 INFO o.a.j.t.JMeterThread: Thread is done: 200_User_requests 1-185
2021-10-09 23:34:33,732 INFO o.a.j.t.JMeterThread: Thread finished: 200_User_requests 1-185
2021-10-09 23:34:33,732 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
2021-10-09 23:34:33,733 INFO o.a.j.g.u.JMeterMenuBar: setRunning(false, *local*)

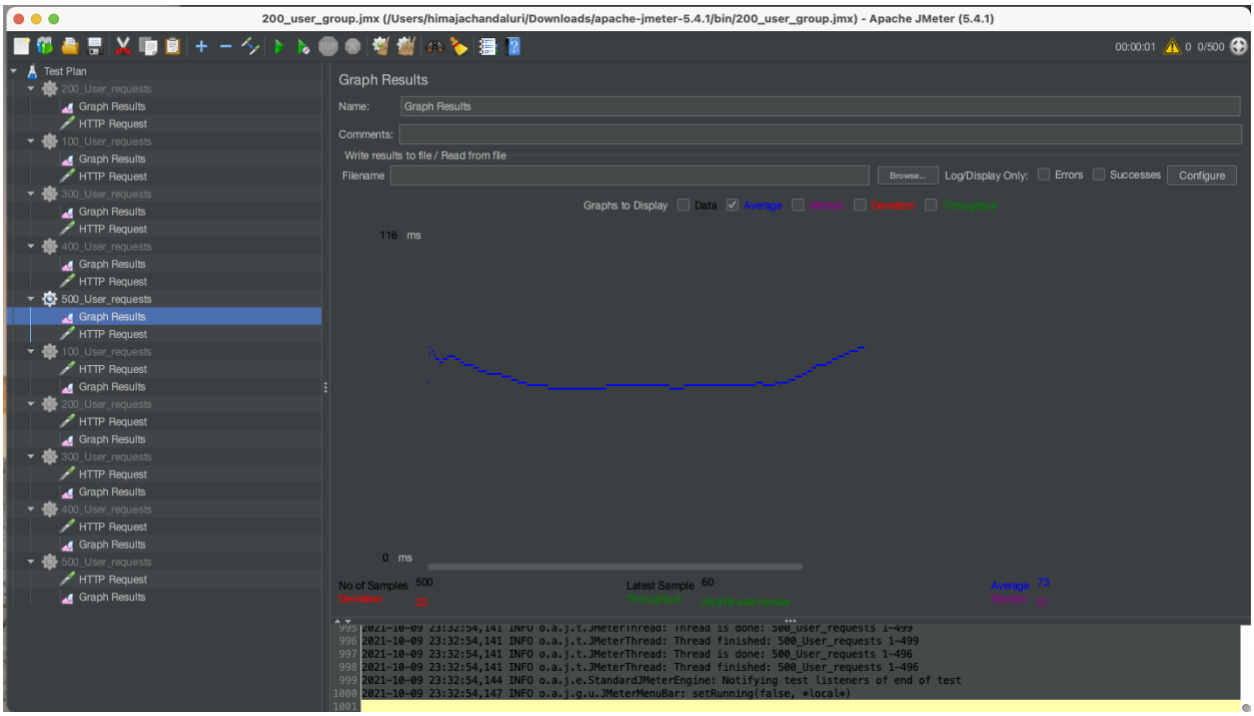
300 –


200_user_group.jmx (/Users/himajachandaluri/Downloads/apache-jmeter-5.4.1/bin/200_user_group.jmx) - Apache JMeter (5.4.1)
00:00:01 0 0/300
Test Plan
200_User_requests
Graph Results
HTTP Request
100_User_requests
300_User_requests
400_User_requests
500_User_requests
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes Configure
Graphs to Display Data Average Median Deviation Throughput
63 ms
0 ms
No of Samples 300
Latest Sample 55
Deviation 10
Average 54
Median 52
2021-10-09 23:36:38,250 INFO o.a.j.t.JMeterThread: Thread finished: 300_User_requests 1-298
2021-10-09 23:36:38,259 INFO o.a.j.t.JMeterThread: Thread is done: 300_User_requests 1-300
2021-10-09 23:36:38,260 INFO o.a.j.t.JMeterThread: Thread finished: 300_User_requests 1-300
2021-10-09 23:36:38,260 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
2021-10-09 23:36:38,260 INFO o.a.j.g.u.JMeterMenuBar: setRunning(false, *local*)

400 -

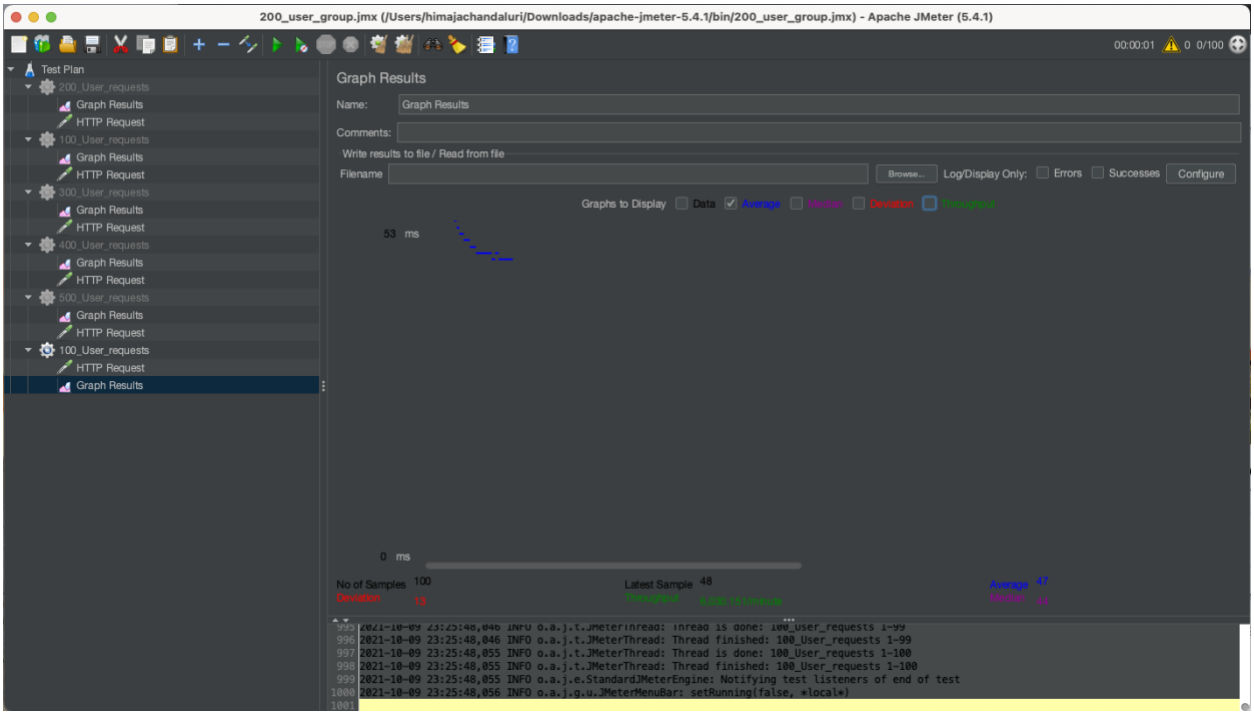

200_user_group.jmx (/Users/himajachandaluri/Downloads/apache-jmeter-5.4.1/bin/200_user_group.jmx) - Apache JMeter (5.4.1)
00:00:01 0 0/400
Test Plan
200_User_requests
Graph Results
HTTP Request
100_User_requests
300_User_requests
400_User_requests
500_User_requests
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes Configure
Graphs to Display Data Average Median Deviation Throughput
597 ms
0 ms
No of Samples 400
Latest Sample 290
Deviation 144
Average 396
Median 425
2021-10-09 23:37:18,914 INFO o.a.j.t.JMeterThread: Thread finished: 400_User_requests 1-100
2021-10-09 23:37:18,917 INFO o.a.j.t.JMeterThread: Thread is done: 400_User_requests 1-398
2021-10-09 23:37:18,917 INFO o.a.j.t.JMeterThread: Thread finished: 400_User_requests 1-398
2021-10-09 23:37:18,917 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
2021-10-09 23:37:18,917 INFO o.a.j.g.u.JMeterMenuBar: setRunning(false, *local*)

500 –

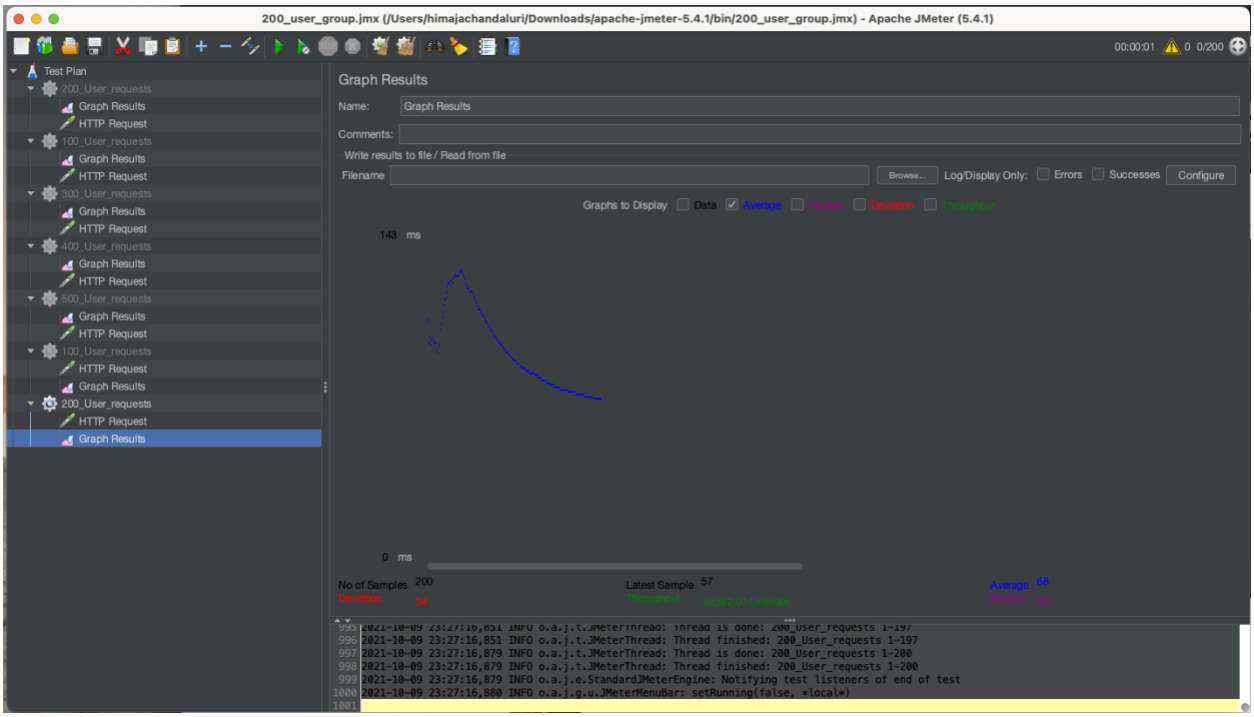


Without Connection Pooling – (MongoDB connection pool default – 5)

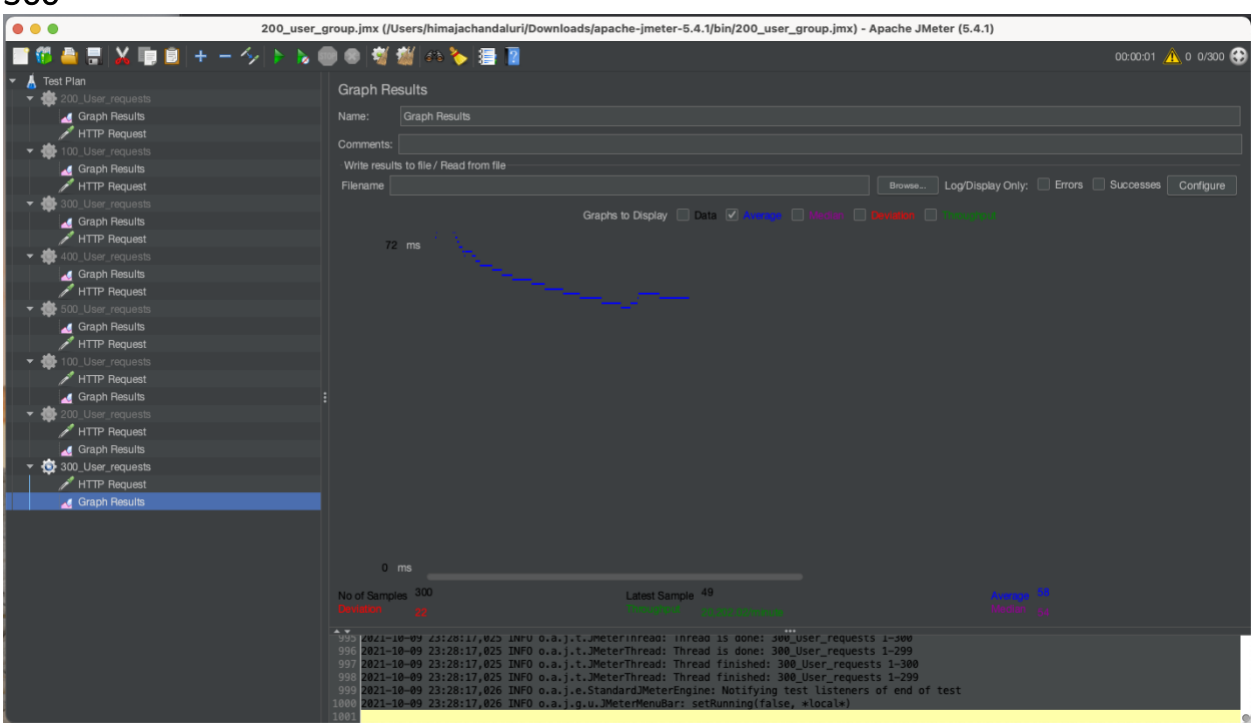
100 –



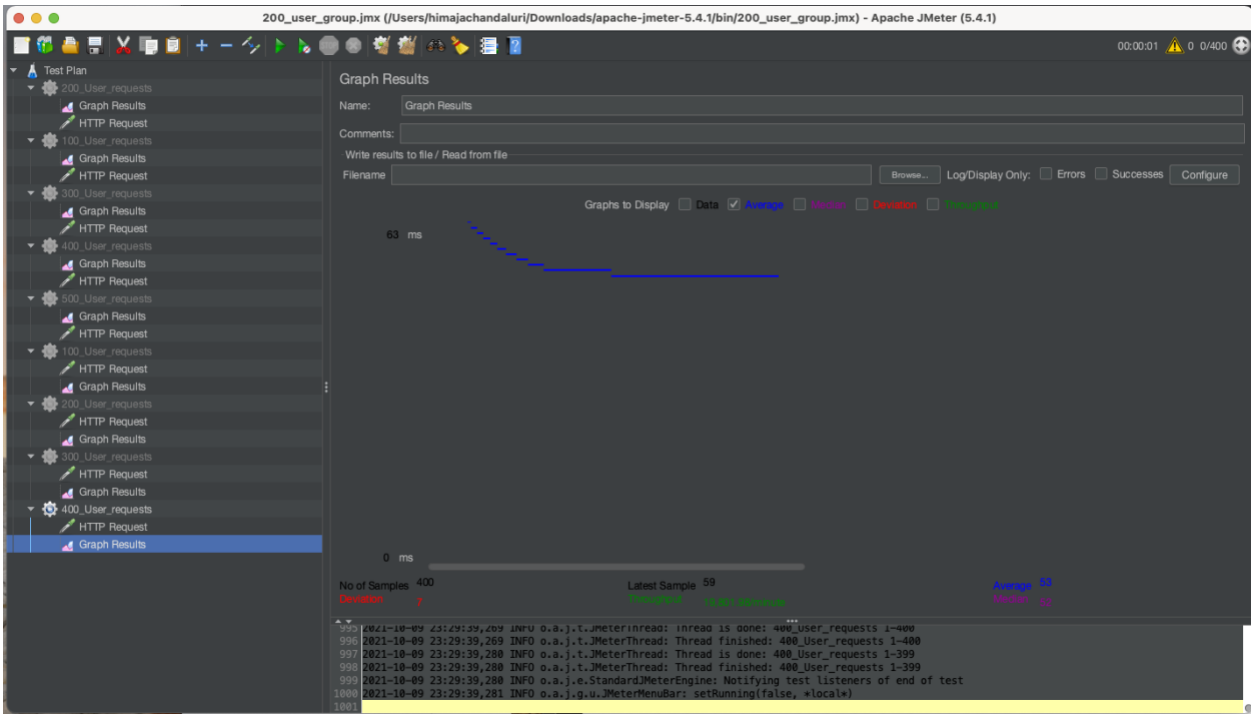
200 –


200_user_group.jmx (/Users/himajachandaluri/Downloads/apache-jmeter-5.4.1/bin/200_user_group.jmx) - Apache JMeter (5.4.1)
00:00:01 0 0/200
Test Plan
200_User_requests
Graph Results
HTTP Request
100_User_requests
Graph Results
HTTP Request
300_User_requests
Graph Results
HTTP Request
400_User_requests
Graph Results
HTTP Request
500_User_requests
Graph Results
HTTP Request
100_User_requests
HTTP Request
Graph Results
200_User_requests
HTTP Request
Graph Results
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes Configure
Graphs to Display Data Average Median Deviation Throughput
143 ms
0 ms
No of Samples 200
Latest Sample 57
Average 68
Deviation 34
Median 63
995 2021-10-09 23:27:16,851 INFO o.a.j.t.JMeterThread: Thread is done: 200_User_requests 1-197
996 2021-10-09 23:27:16,851 INFO o.a.j.t.JMeterThread: Thread finished: 200_User_requests 1-197
997 2021-10-09 23:27:16,879 INFO o.a.j.t.JMeterThread: Thread is done: 200_User_requests 1-200
998 2021-10-09 23:27:16,879 INFO o.a.j.t.JMeterThread: Thread finished: 200_User_requests 1-200
999 2021-10-09 23:27:16,879 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
1000 2021-10-09 23:27:16,880 INFO o.a.j.g.u.JMeterMenuBar: setRunning(false, *local*)
1001

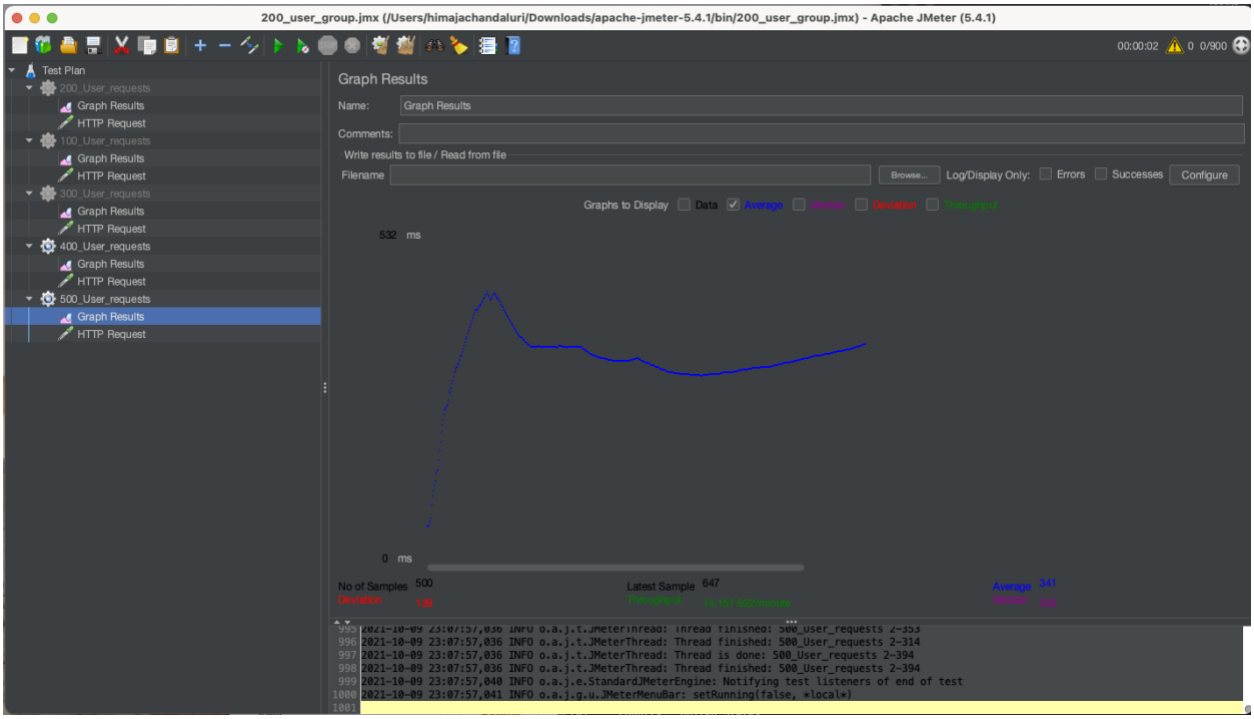
300 –


200_user_group.jmx (/Users/himajachandaluri/Downloads/apache-jmeter-5.4.1/bin/200_user_group.jmx) - Apache JMeter (5.4.1)
00:00:01 0 0/300
Test Plan
200_User_requests
Graph Results
HTTP Request
100_User_requests
Graph Results
HTTP Request
300_User_requests
Graph Results
HTTP Request
400_User_requests
Graph Results
HTTP Request
500_User_requests
Graph Results
HTTP Request
100_User_requests
HTTP Request
Graph Results
200_User_requests
HTTP Request
Graph Results
300_User_requests
HTTP Request
Graph Results
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes Configure
Graphs to Display Data Average Median Deviation Throughput
72 ms
0 ms
No of Samples 300
Latest Sample 49
Average 58
Deviation 22
Median 54
995 2021-10-09 23:28:17,025 INFO o.a.j.t.JMeterThread: Thread is done: 300_User_requests 1-300
996 2021-10-09 23:28:17,025 INFO o.a.j.t.JMeterThread: Thread is done: 300_User_requests 1-299
997 2021-10-09 23:28:17,025 INFO o.a.j.t.JMeterThread: Thread finished: 300_User_requests 1-300
998 2021-10-09 23:28:17,025 INFO o.a.j.t.JMeterThread: Thread finished: 300_User_requests 1-299
999 2021-10-09 23:28:17,026 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
1000 2021-10-09 23:28:17,026 INFO o.a.j.g.u.JMeterMenuBar: setRunning(false, *local*)
1001

400 –


200_user_group.jmx (/Users/himajachandaluri/Downloads/apache-jmeter-5.4.1/bin/200_user_group.jmx) - Apache JMeter (5.4.1)
00:00:01 0 0/400
Test Plan
200_User_requests
Graph Results
HTTP Request
100_User_requests
300_User_requests
400_User_requests
500_User_requests
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes Configure
Graphs to Display Data Average Median Deviation Throughput
63 ms
0 ms
No of Samples 400
Latest Sample 59
Average 53
Deviation 7
Median 52
2021-10-09 23:29:39,269 INFO o.a.j.t.JMeterThread: Thread finished: 400_User_requests 1-400
2021-10-09 23:29:39,280 INFO o.a.j.t.JMeterThread: Thread is done: 400_User_requests 1-399
2021-10-09 23:29:39,280 INFO o.a.j.t.JMeterThread: Thread finished: 400_User_requests 1-399
2021-10-09 23:29:39,280 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
2021-10-09 23:29:39,281 INFO o.a.j.g.u.JMeterMenuBar: setRunning(false, *local*)

500 –


200_user_group.jmx (/Users/himajachandaluri/Downloads/apache-jmeter-5.4.1/bin/200_user_group.jmx) - Apache JMeter (5.4.1)
00:00:02 0 0/900
Test Plan
200_User_requests
Graph Results
HTTP Request
100_User_requests
300_User_requests
400_User_requests
500_User_requests
Graph Results
Name: Graph Results
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes Configure
Graphs to Display Data Average Median Deviation Throughput
532 ms
0 ms
No of Samples 500
Latest Sample 647
Average 341
Deviation 139
Median 333
2021-10-09 23:07:57,036 INFO o.a.j.t.JMeterThread: Thread finished: 500_User_requests 2-314
2021-10-09 23:07:57,036 INFO o.a.j.t.JMeterThread: Thread is done: 500_User_requests 2-394
2021-10-09 23:07:57,036 INFO o.a.j.t.JMeterThread: Thread finished: 500_User_requests 2-394
2021-10-09 23:07:57,040 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
2021-10-09 23:07:57,041 INFO o.a.j.g.u.JMeterMenuBar: setRunning(false, *local*)

Mocha Testing

```
const axios = require("axios");

const assert = require("assert");

const apiUrl = "http://localhost:3900/api";

const customerJwt =
  "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOjIyLCJlbWFpbCI6InN1amFAbi5jb20iLCJp

describe("POST - get token during login (customer/restaurant)", () => {
  it("/auth", (done) => {
    axios
      .post(apiUrl + "/auth", { email: "user1@user.com", password: "123456" })
      .then((response) => {
        console.log(response.data);
        assert.equal(response.status, 200);
        done();
      })
      .catch((err) => {
        done(err);
      });
  });
});
```

```
25
26  describe("GET - get customer details by authID", () => {
27    axios.defaults.headers.common["x-auth-token"] = customerJwt;
28    it("/:id", (done) => {
29      axios
30        .get(apiUrl + "/customer/22")
31        .then((response) => {
32          console.log(response.data);
33          assert.equal(response.status, 200);
34          done();
35        })
36        .catch((err) => {
37          done(err);
38        });
39    });
40  });
41
42  describe("GET - customer delivery addresses by custID", () => {
43    axios.defaults.headers.common["x-auth-token"] = customerJwt;
44    it("/deliveryAddresses/:id", (done) => {
45      axios
46        .get(apiUrl + "/deliveryAddresses/8")
47        .then((response) => {
48          console.log(response.data);
49          assert.equal(response.status, 200);
50          done();
51        })
52        .catch((err) => {
53          done(err);
54        });
55    });
56  });
```

```
57
58  describe("GET - dish details by dishID", () => {
59    it("/dish/:id", (done) => {
60      axios
61        .get(apiUrl + "/dish/1")
62        .then((response) => {
63          console.log(response.data);
64          assert.equal(response.status, 200);
65          done();
66        })
67        .catch((err) => {
68          done(err);
69        });
70    });
71  });
72
73  describe("POST - toggle customer favorite option", () => {
74    it("/like", (done) => {
75      axios
76        .post(apiUrl + "/like", { _custId: 8, _restaurantId: 4 })
77        .then((response) => {
78          console.log(response.data);
79          assert.equal(response.status, 200);
80          done();
81        })
82        .catch((err) => {
83          done(err);
84        });
85    });
86  });
87
```

Output:

```
~/SJSU/2-Fall-2021/CMPE 273/CMPE273Lab1UberEats/backend npx mocha test/test.js


  POST - get token during login (customer/restaurant)
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJfaWQiOjU5LCJlbWFpbCI6InVzZXIxQHVzZXIuY29tIiwiaXNSZXN0YXVyYW50IjowLCJpYXQiO
    ✔ /auth (530ms)

  GET - get customer details by authID
{
  _id: 28,
  _authId: 59,
  nickname: 'User1',
  name: 'User1',
  dateOfBirth: '2003-10-07T07:00:00.000Z',
  profilePic: 'https://ubereatsimages273.s3.amazonaws.com/photo-1504194104404-433180773017.jpeg',
  phoneNumber: '3456789876',
  about: 'I am User1'
}
    ✔ /:id (200ms)

  GET - customer delivery addresses by custID
[
  {
    _id: 25,
    _custId: 28,
    city: 'San Ramon',
    state: 'CA',
    country: 'United States',
    zipCode: '95110',
    street: '190 Ryland St., Apr. no. 5404'
  }
]
    ✔ /deliveryAddresses/:id (186ms)

  GET - dish details by dishID
{
  _id: 20,
  _restaurantId: 12,
  name: 'Roti',
  mainIngrediant: 'Wheat Floor',
  image: 'https://ubereatsimages273.s3.amazonaws.com/34567833.jpeg',
  price: 1,
  description: 'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labor
ation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate
  category: 'Main',
  type: 'Veg'
}
    ✔ /dish/:id (186ms)

  POST - toggle customer favorite option
Succesfully disliked
    ✔ /like (362ms)


  5 passing (1s)

~/SJSU/2-Fall-2021/CMPE 273/CMPE273Lab1UberEats/backend
```

React Testing:

frontend > src > tests > customerDashboard.test.js > ...

```
import React from "react";
import { render, screen } from "@testing-library/react";
import CustomerDashboard from "../components/customerDashboard";
import { BrowserRouter } from "react-router-dom";
import { Provider } from "react-redux";
import store from "../redux/store";

test("renders Customer Dashboard", () => {
  render(
    <React.StrictMode>
      <BrowserRouter>
        <Provider store={store}>
          <CustomerDashboard />
        </Provider>
      </BrowserRouter>
    </React.StrictMode>
  );
  const linkElement = screen.getByText(/Filters/);
  expect(linkElement).toBeInTheDocument();
});
```

frontend > src > tests > customerSignUp.test.js > ...

```
import React from "react";
import { render, screen } from "@testing-library/react";
import CustomerSignUp from "../components/customerSignUp";
import { BrowserRouter } from "react-router-dom";
import { Provider } from "react-redux";
import store from "../redux/store";

test("renders Customer SignUp", () => {
  render(
    <React.StrictMode>
      <BrowserRouter>
        <Provider store={store}>
          <CustomerSignUp />
        </Provider>
      </BrowserRouter>
    </React.StrictMode>
  );
  const linkElement = screen.getByText(/Sign Up/, { selector: "button" });
  expect(linkElement).toBeInTheDocument();
});
```

frontend > src > tests > home.test.js > ...

```
import React from "react";
import { render, screen } from "@testing-library/react";
import Home from "../components/home";
import { BrowserRouter } from "react-router-dom";
import { Provider } from "react-redux";
import store from "../redux/store";

test("renders Home", () => {
  render(
    <React.StrictMode>
      <BrowserRouter>
        <Provider store={store}>
          <Home />
        </Provider>
      </BrowserRouter>
    </React.StrictMode>
  );
  const linkElement = screen.getByText(/Customer SignUp/);
  expect(linkElement).toBeInTheDocument();
});
```

frontend > src > tests > login.test.js > ...

```
import React from "react";
import { render, screen } from "@testing-library/react";
import Login from "../components/login";
import { BrowserRouter } from "react-router-dom";
import { Provider } from "react-redux";
import store from "../redux/store";

test("renders Login", () => {
  render(
    <React.StrictMode>
      <BrowserRouter>
        <Provider store={store}>
          <Login />
        </Provider>
      </BrowserRouter>
    </React.StrictMode>
  );
  const linkElement = screen.getByText(/Login/, { selector: "button" });
  expect(linkElement).toBeInTheDocument();
});
```

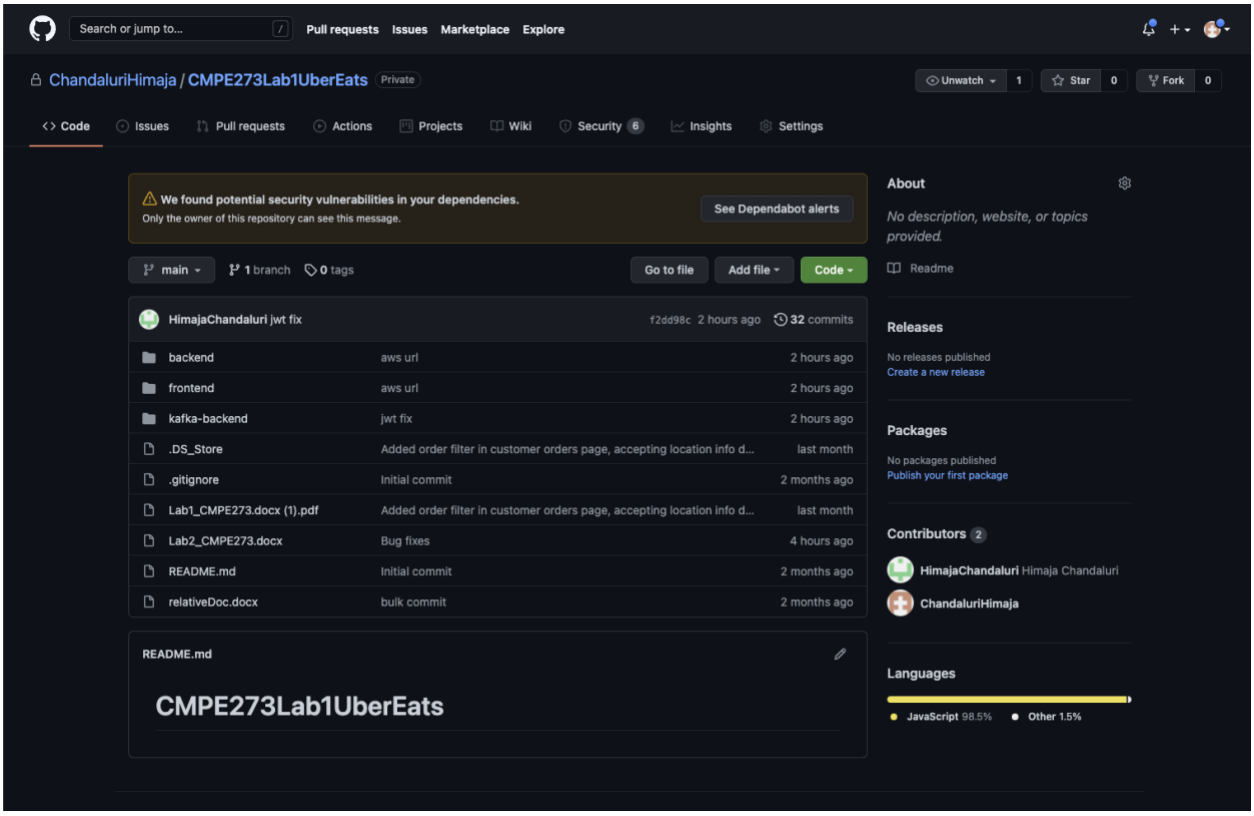
frontend > src > tests > restaurantSignUp.test.js > ...

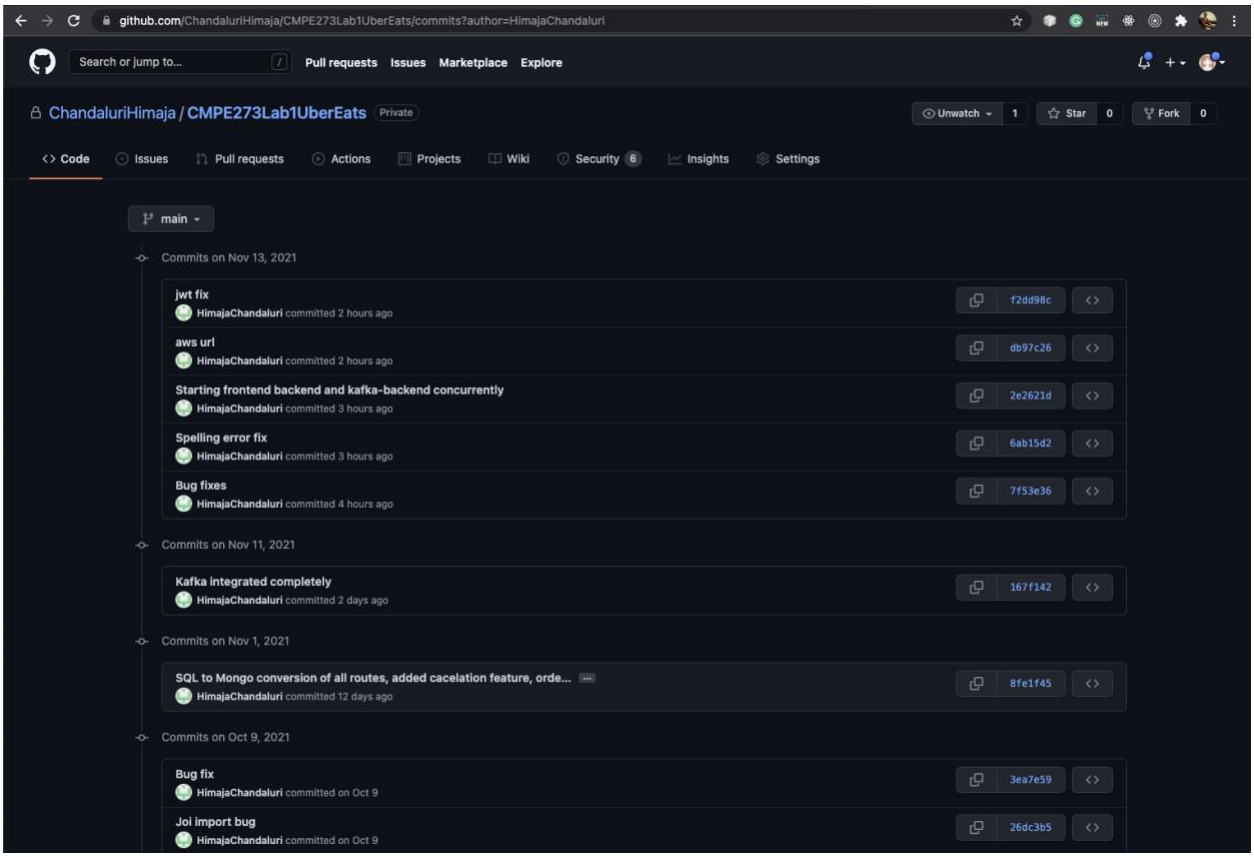
```
import React from "react";
import { render, screen } from "@testing-library/react";
import RestaurantSignUp from "../components/restaurantSignUp";
import { BrowserRouter } from "react-router-dom";
import { Provider } from "react-redux";
import store from "../redux/store";

test("renders Restaurant SignUp", () => {
  render(
    <React.StrictMode>
      <BrowserRouter>
        <Provider store={store}>
          <RestaurantSignUp />
        </Provider>
      </BrowserRouter>
    </React.StrictMode>
  );
  const linkElement = screen.getByText(/Sign Up/, { selector: "button" });
  expect(linkElement).toBeInTheDocument();
});
```

Output:

```
PASS  src/tests/home.test.js
  ● Console

    console.log
      Got user data again in HOME:  {}

      at Home.render (src/components/home.js:9:13)

PASS  src/tests/login.test.js
PASS  src/tests/customerSignUp.test.js
PASS  src/tests/restaurantSignUp.test.js
PASS  src/tests/customerDashboard.test.js
  ● Console

    console.log
      LOCATION FILTER:  []

      at CustomerDashboard.getFilteredRestaurants (src/components/customerDashboard.js:117:13)

    console.log
      PROPS:  { marginLeft: '20px', marginRight: '20px' }

      at SearchBox (src/components/common/searchBox.js:4:11)

    console.log
      PROPS:  { marginLeft: '10px', marginRight: '50px' }

      at SearchBox (src/components/common/searchBox.js:4:11)

    console.log
      getAllRestaurant call:  http://13.56.247.80:3900/api/restaurant

      at getAllRestaurantsData (src/redux/allRestaurants/allRestaurantsActions.js:26:11)

    console.log
      ENDPOINT:  http://13.56.247.80:3900/api/restaurant

      at src/redux/allRestaurants/allRestaurantsActions.js:29:15


Test Suites: 5 passed, 5 total
Tests:       5 passed, 5 total
Snapshots:   0 total
Time:        2.953 s
Ran all test suites.

Watch Usage: Press w to show more.
```

Git repository:




github.com/ChandaluriHimaja/CMPE273Lab1UberEats/commits?author=HimajaChandaluri
Search or jump to...
Pull requests
Issues
Marketplace
Explore
ChandaluriHimaja / CMPE273Lab1UberEats
Private
Unwatch 1
Star 0
Fork 0
Code
Issues
Pull requests
Actions
Projects
Wiki
Security 6
Insights
Settings
main
Commits on Nov 13, 2021
jwt fix
HimajaChandaluri committed 2 hours ago
f2dd98c
aws url
HimajaChandaluri committed 2 hours ago
db97c26
Starting frontend backend and kafka-backend concurrently
HimajaChandaluri committed 3 hours ago
2e2621d
Spelling error fix
HimajaChandaluri committed 3 hours ago
6ab15d2
Bug fixes
HimajaChandaluri committed 4 hours ago
7f53e36
Commits on Nov 11, 2021
Kafka integrated completely
HimajaChandaluri committed 2 days ago
167f142
Commits on Nov 1, 2021
SQL to Mongo conversion of all routes, added cacelation feature, orde...
HimajaChandaluri committed 12 days ago
8fe1f45
Commits on Oct 9, 2021
Bug fix
HimajaChandaluri committed on Oct 9
3ea7e59
Joi import bug
HimajaChandaluri committed on Oct 9
26dc3b5

Questions:

1. **Comparison of passport authentication process Vs the authentication process used in Lab1.**

   In Lab1 I implemented authentication using JWT tokens by using the jsonwebtoken library directly, without any middleware. In Lab2 I used the Passport-JWT strategy from the Passport middleware.

2. **Performance comparison with and without Kafka**

   Kafka makes sure that a request is not lost when the backend is overloaded with requests. It does this by placing a request into a message queue. Whenever the previous request execution is completed a new request from the queue is taken and processed. But Kafka does not increase the performance of the system. With Kafka in place an API call has to travel more in the network to reach the database layer and fetch the data and respond back to the frontend. Thus, using Kafka reduces the performance of the system but increases the latency of the system.

3. **If given an option to implement MySQL and MongoDB both in your application, specify which part of the data of the application you will store in MongoDB and MySQL respectively.**

   If given an option I might store User login credentials in MySQL database and the rest of the data in the MongoDB database. Because login credentials are read multiple times but updated rarely MySQL is a good option. The rest of my data has fields with array of values which makes the data schema unstructured. MongoDB being an unstructured database best fits this requirement.