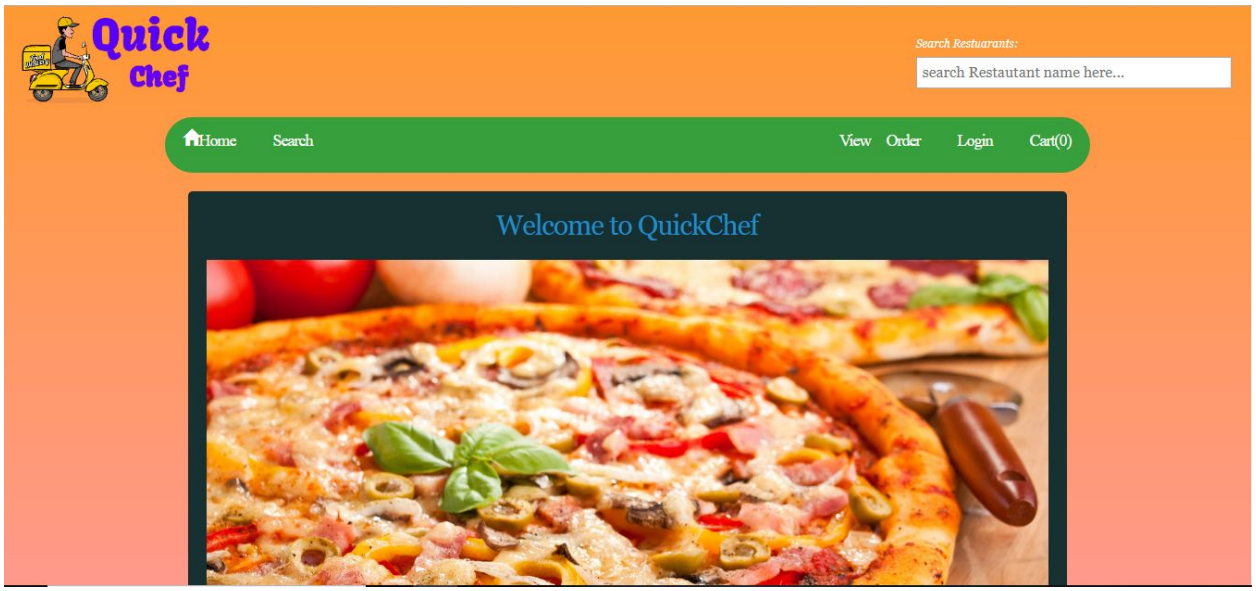
Srinivasa, Jaikeerthi - jsrinivasa@hawk.iit.edu
Vijay, Suraj - svijay1@hawk.iit.edu

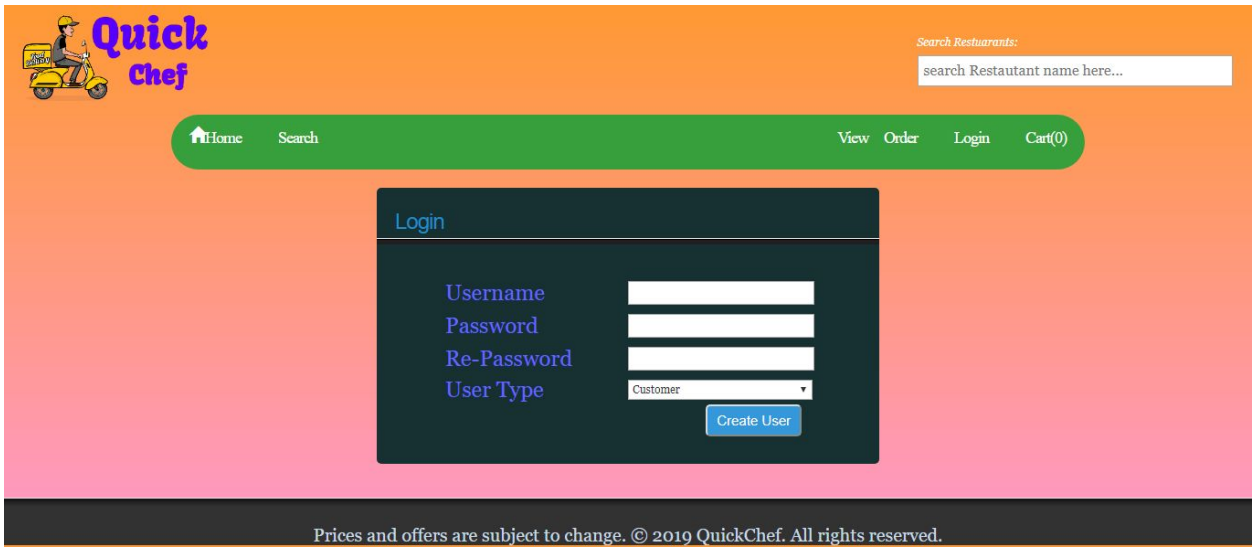
Enterprise Web Applications

Team 20 - QuickChef

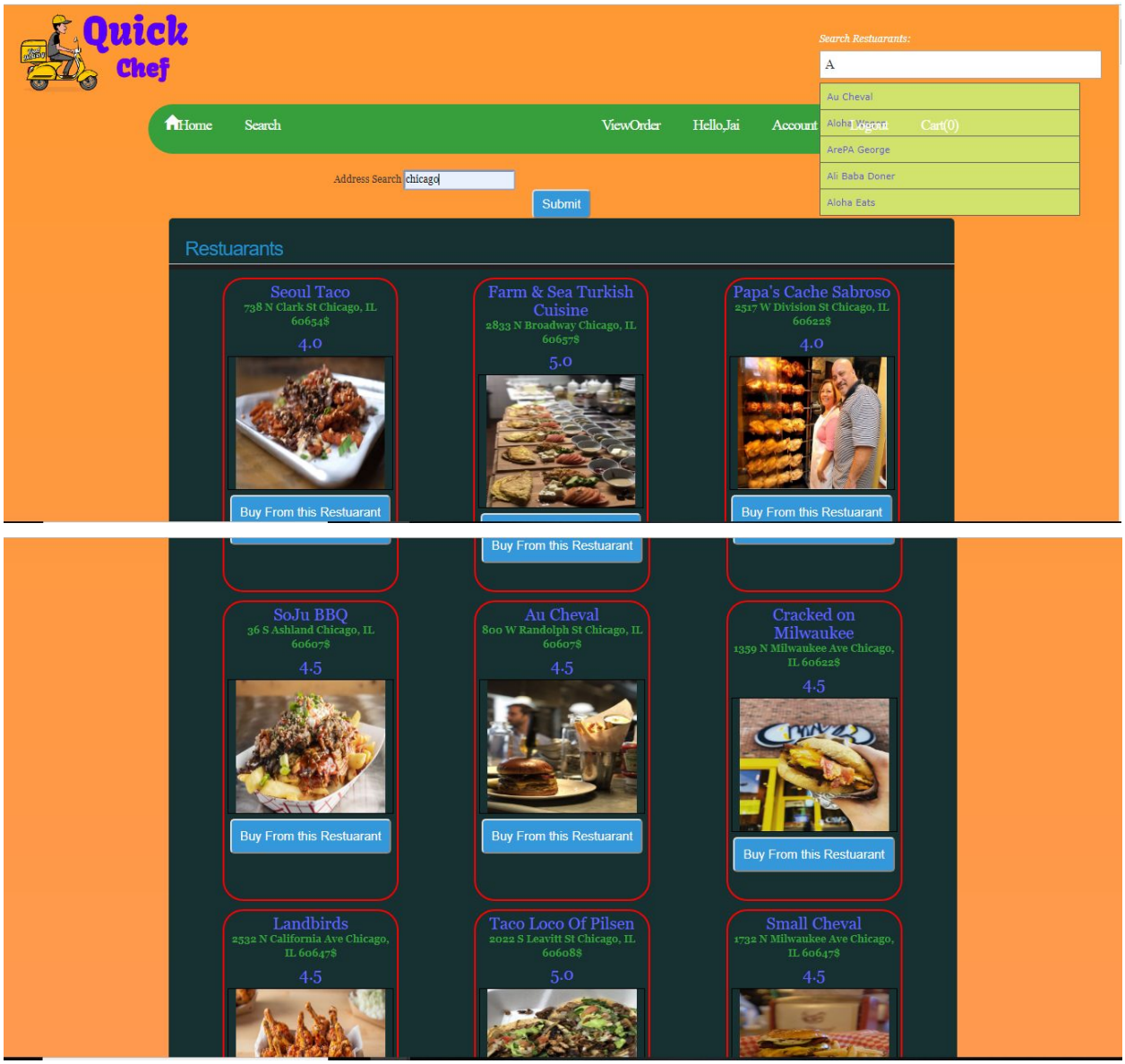
Final Project Snapshots



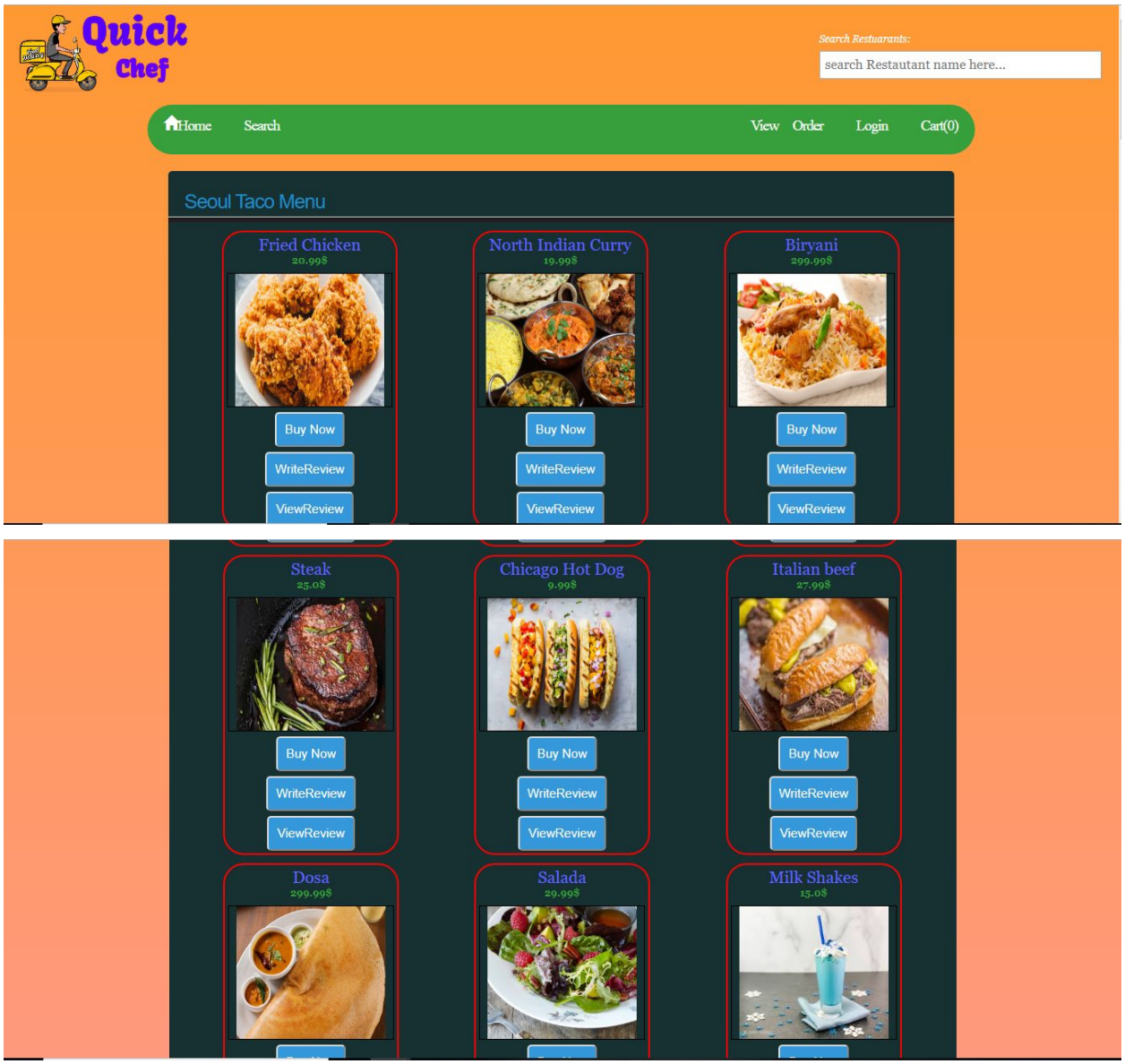
Home Page with a search tab(with autocomplete feature) with which we can search restaurants by its name. Clicking the search button on the menu tab, it redirects to the list of restaurants. View button is to view the orders for a particular user. Cart displays the items that the user has added for ordering.



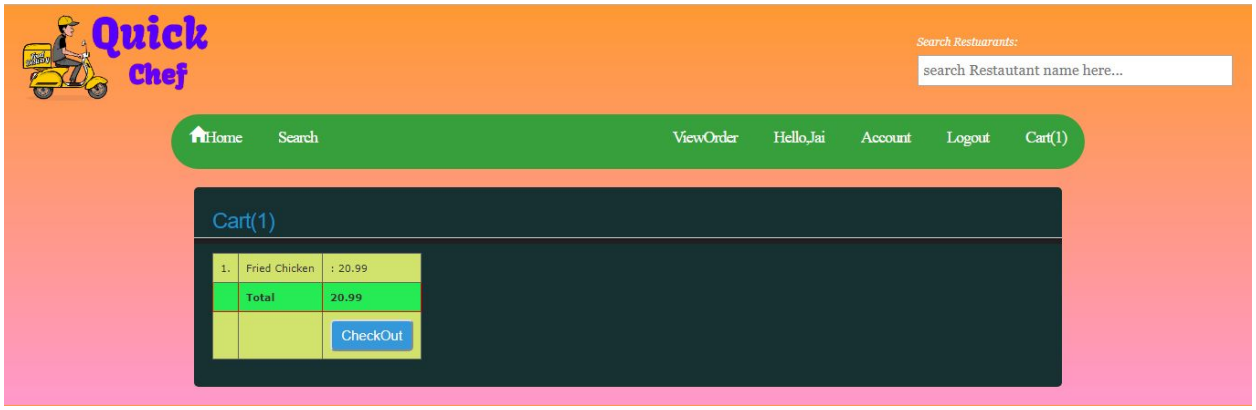
Login and Registration Screen. Existing users can login to the application and new user can register using this portal.



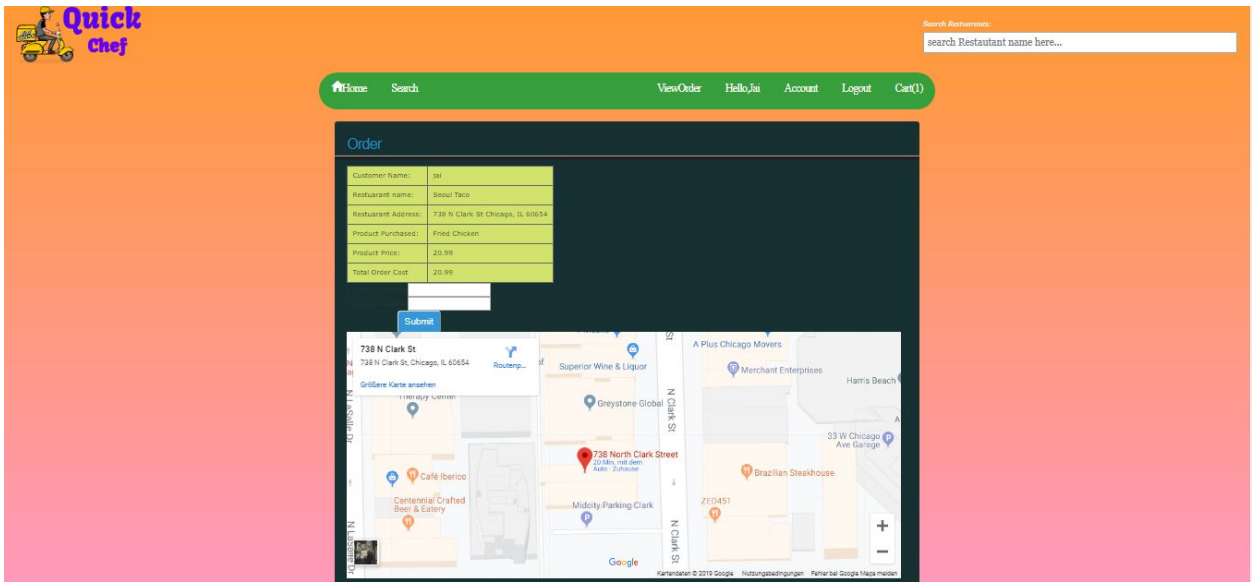
Search Page. On click of the search button, this page is displayed which shows the list of the restaurants.There is also another search bar where user can search the restaurants based on the address of the restaurant. All the restaurants, their images, names, addresses and their ratings are being displayed. Users have to select the restaurant that they need to order by clicking on 'Buy From this Restaurant' button.



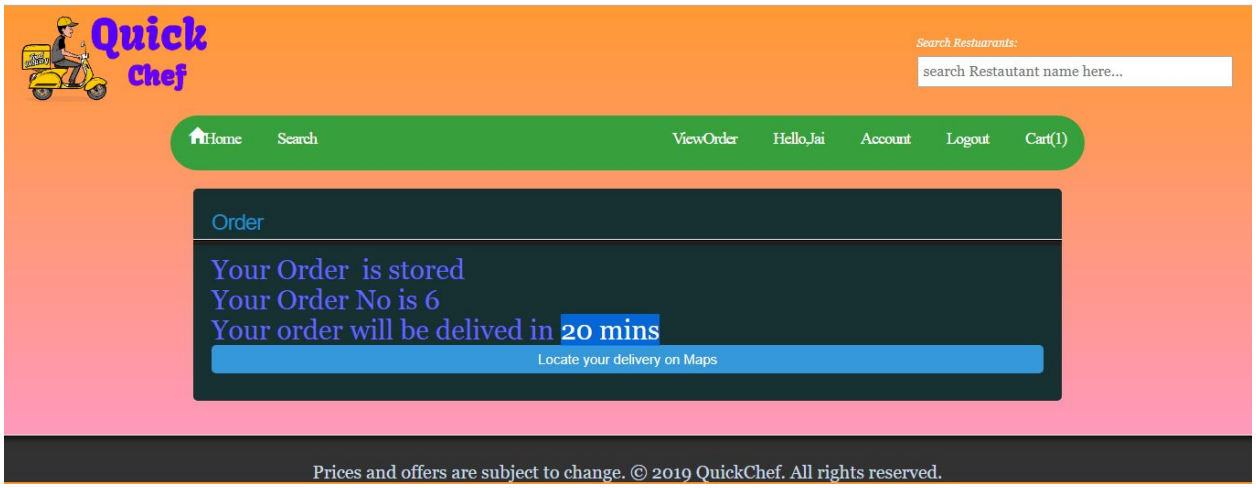
<u>Menu Page.</u> On click of 'Buy From this Restaurant' button, it is redirected to the menu page where in it displays all the menu available in that restaurant. Users can add the food from the list and can proceed to check out. Users can write reviews and view the reviews for a particular dish.



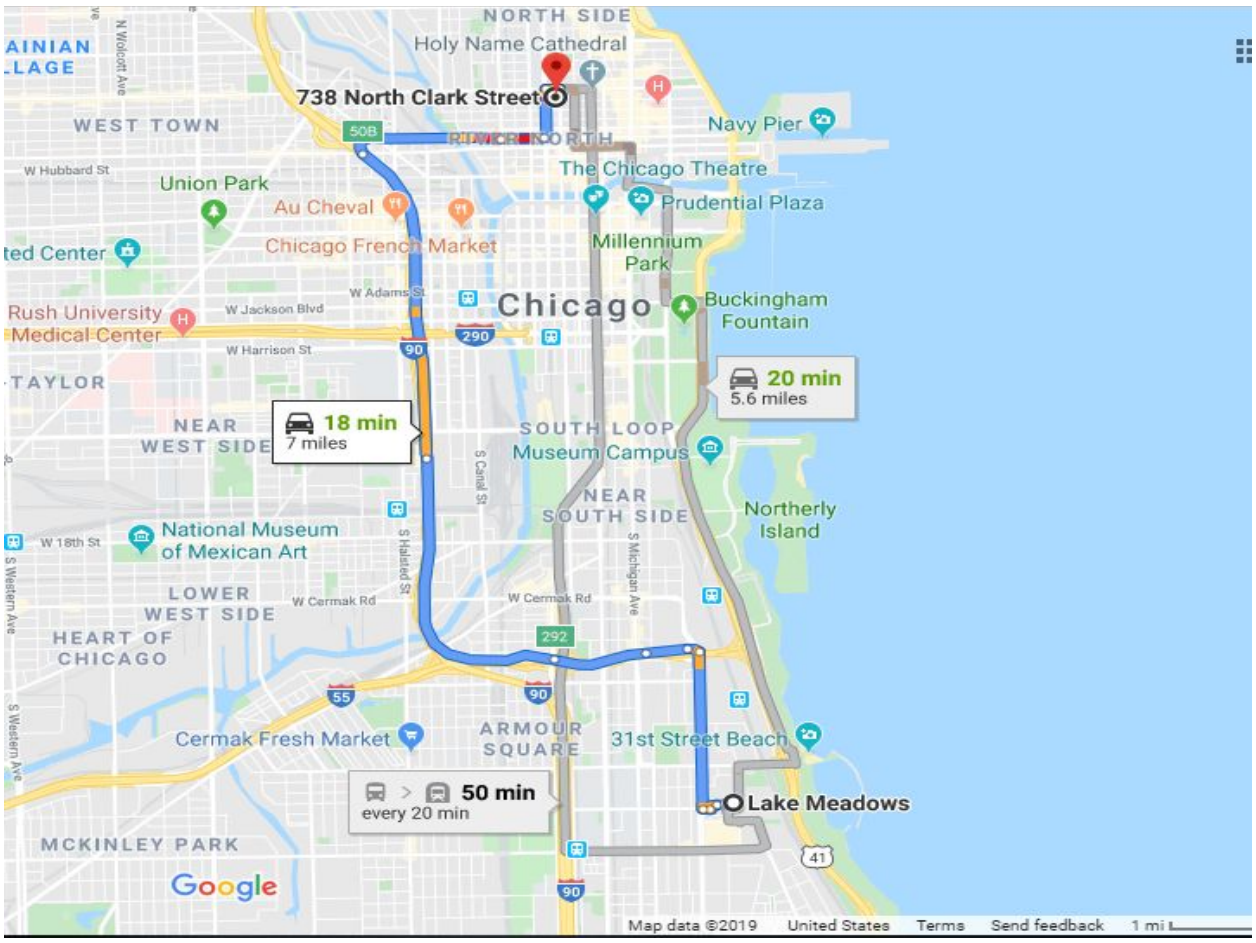
Cart displays all the items that are added by the user along with their total price.



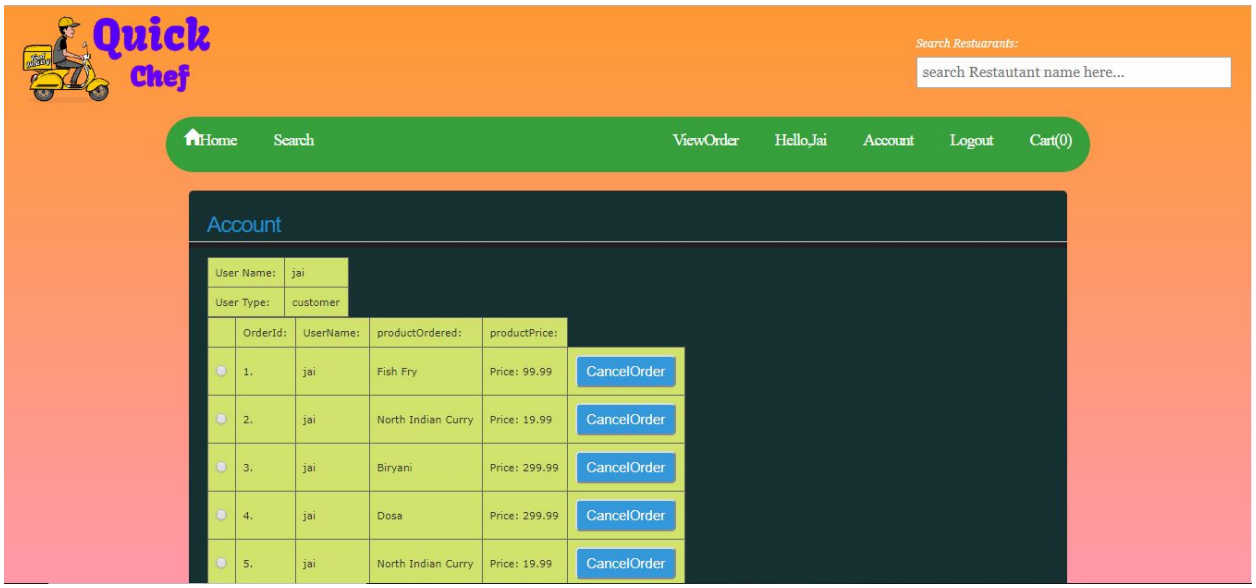
CheckOut.Order Summary Page displays the order details and along with it, it displays the restaurant's location on the map it use the *google maps API*. Users need to enter their valid card details and address to place an order.



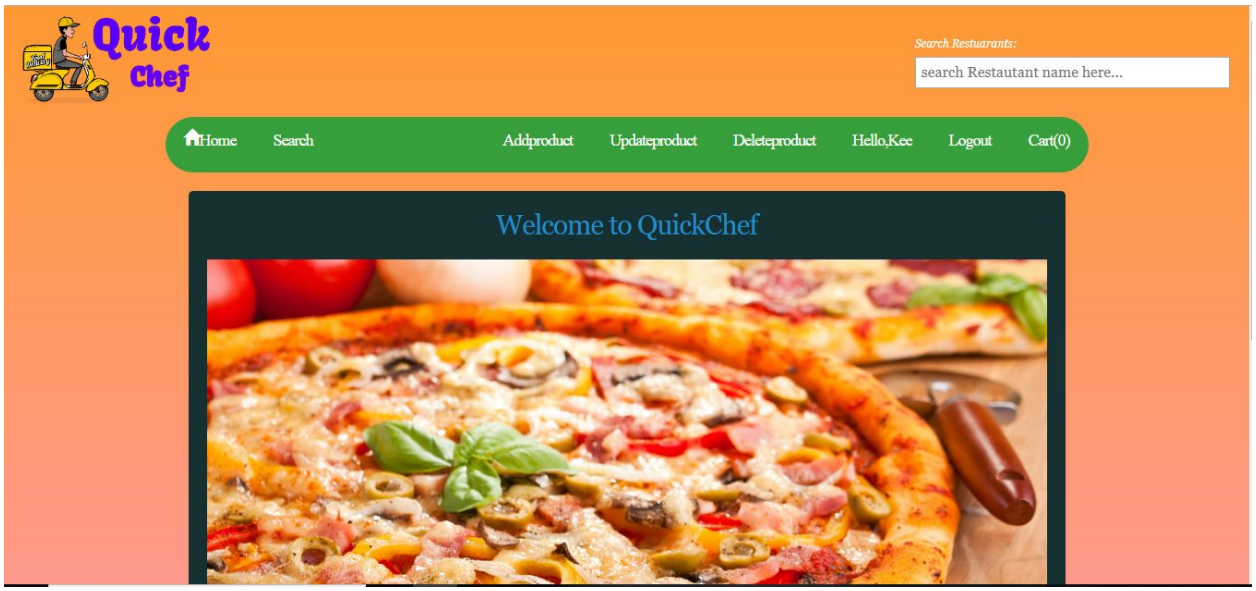
Users will be displayed with their order number and along with that, users can know the approximate delivery time of their order in this window. It uses ***google distance matrix API*** *for prediction*. Users can locate the distance and approximate time of arrival in a map view by clicking on 'Locate your delivery on Maps' button.



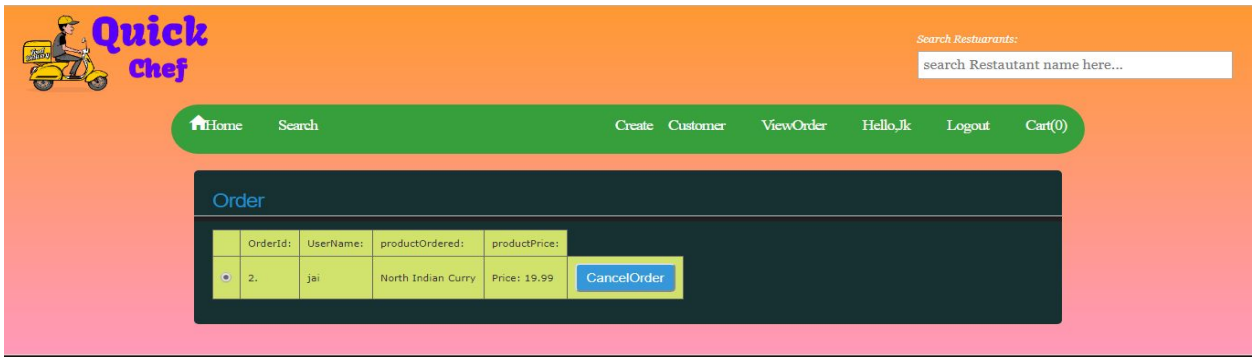
A map view of the direction, distance and approximate time of arrival from the restaurant to the user's location.



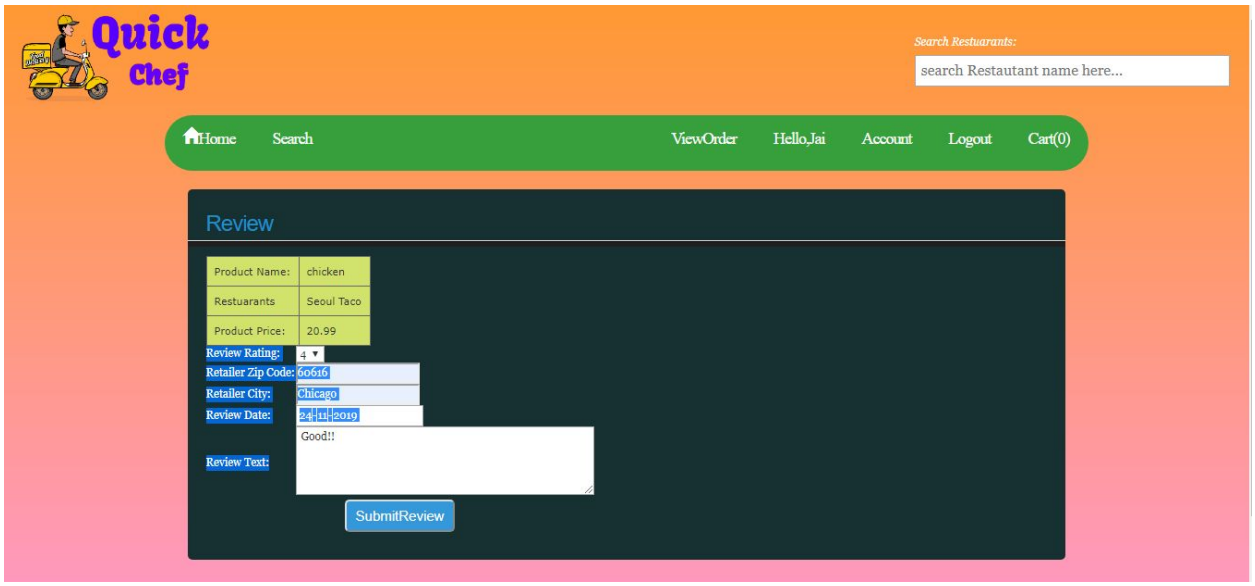
Users can **cancel the order** if they wish to, and can place a new order.



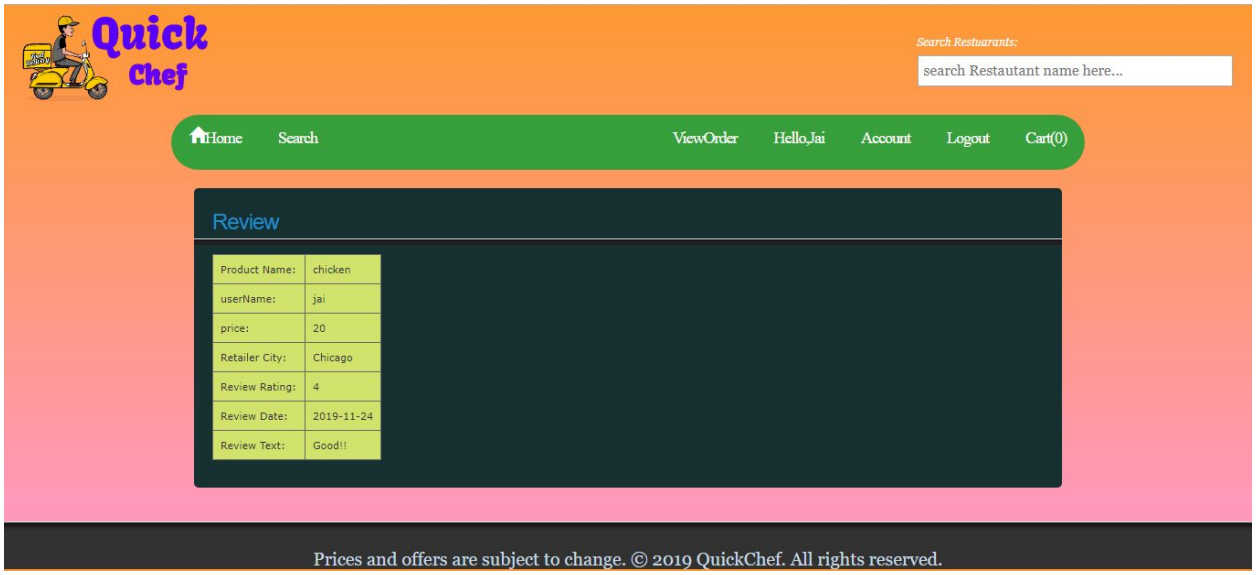
This window is for the **store manager**. He can add, delete, and update order upon user's request



Admin can cancel the order placed by the user.



Users can rate the food in that restaurant and **write reviews** about it .

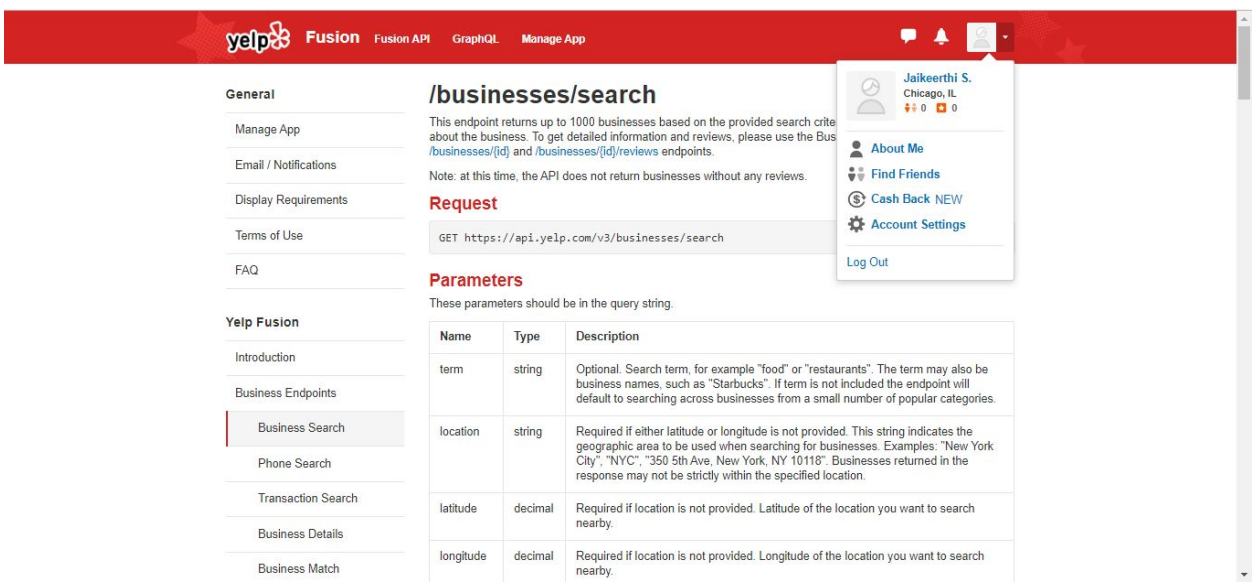


Review written by the user is stored in MongoDB(Nosql database)

```
{ "_id" : ObjectId("5dd88289b8c19f28080a1669"), "title" : "myReviews", "userName" : "jk", "retailername" : "Jai", "retailerstate" : "IL", "productsale" : "Yes", "manu
cturerebate" : "Yes", "userid" : "3454634", "userage" : "26", "usergender" : "Male", "useroccupation" : "Master", "productName" : "iPhone7", "productType" : "games",
roductMaker" : "apple", "reviewRating" : 5, "reviewDate" : "2019-11-24", "reviewText" : "Very good!", "retailerpin" : "60616", "retailercity" : "Chicago", "price" : 4
 }
{ "_id" : ObjectId("5dd8832cb8c19f28080a166c"), "title" : "myReviews", "userName" : "kj", "retailername" : "bhata", "retailerstate" : "IL", "productsale" : "Yes", "ma
facturerebate" : "Yes", "userid" : "1234", "userage" : "24", "usergender" : "Male", "useroccupation" : "Pro", "productName" : "LG2", "productType" : "televisions", "p
ductMaker" : "lg", "reviewRating" : 5, "reviewDate" : "2019-11-21", "reviewText" : " GOOD", "retailerpin" : "60616", "retailercity" : "Chicago", "price" : 139 }
{ "_id" : ObjectId("5dd89da7b8c19f28080a1670"), "title" : "myReviews", "userName" : "jai", "retailername" : "Jaikeerthi", "retailerstate" : "IL", "productsale" : "Yes
 "manufacturerebate" : "Yes", "userid" : "13898", "userage" : "29", "usergender" : "Male", "useroccupation" : "Govt Employee", "productName" : "att", "productType" :
irelessplans", "productMaker" : "Basic", "reviewRating" : 3, "reviewDate" : "2019-11-22", "reviewText" : " Good", "retailerpin" : "60616", "retailercity" : "Chicago",
price" : 50 }
{ "_id" : ObjectId("5dd8a094b8c19f28080a1674"), "title" : "myReviews", "userName" : "jai", "retailername" : "Jaikeerthi", "retailerstate" : "IL", "productsale" : "Yes
 "manufacturerebate" : "Yes", "userid" : "1234", "userage" : "23", "usergender" : "Male", "useroccupation" : "Bussiness Man", "productName" : "Microsoft", "productTyp
 : "soundsystems", "productMaker" : "Microsoft", "reviewRating" : 5, "reviewDate" : "2019-11-22", "reviewText" : " Good", "retailerpin" : "60616", "retailercity" : "C
cago", "price" : 238 }
{ "_id" : ObjectId("5dd8a395b8c19f28080a1679"), "title" : "myReviews", "userName" : "jai", "retailername" : "Jaikeerthi", "retailerstate" : "IL", "productsale" : "Yes
 "manufacturerebate" : "Yes", "userid" : "3454634", "userage" : "23", "usergender" : "Male", "useroccupation" : "Bussiness Man", "productName" : "att", "productType"
"wirelessplans", "productMaker" : "Basic", "reviewRating" : 2, "reviewDate" : "2019-11-13", "reviewText" : " Good", "retailerpin" : "60616", "retailercity" : "Chicago
 "price" : 50 }
{ "_id" : ObjectId("5dd8a485b8c19f28080a167d"), "title" : "myReviews", "userName" : "john", "retailername" : "Jaikeerthi", "retailerstate" : "IL", "productsale" : "Ye
, "manufacturerebate" : "yes", "userid" : "23432543", "userage" : "22", "usergender" : "Male", "useroccupation" : "Bussines", "productName" : "Sharp3", "productType"
"televisions", "productMaker" : "sharp", "reviewRating" : 4, "reviewDate" : "2019-11-06", "reviewText" : " Hi", "retailerpin" : "60616", "retailercity" : "Chicago",
ice" : 169 }
{ "_id" : ObjectId("5dd8a4a0b8c19f28080a1680"), "title" : "myReviews", "userName" : "john", "retailername" : "Jaikeerthi", "retailerstate" : "IL", "productsale" : "Ye
, "manufacturerebate" : "Yes", "userid" : "1234", "userage" : "23", "usergender" : "Male", "useroccupation" : "Bussiness Man", "productName" : "HuaweiP20", "productTy
" : "games", "productMaker" : "huawei", "reviewRating" : 4, "reviewDate" : "2019-11-22", "reviewText" : " Hi", "retailerpin" : "60616", "retailercity" : "Chicago", "p
ce" : 50 }
{ "_id" : ObjectId("5dd9edb6b8c19f0cb4c4e4b8"), "title" : "myReviews", "userName" : "jai", "productName" : "northindiancurry1", "productType" : "tablets", "productMak
" : null, "reviewRating" : 2, "reviewDate" : "2019-11-16", "reviewText" : " Good!", "retailerpin" : "60616", "retailercity" : "Chicago", "price" : 19 }
{ "_id" : ObjectId("5ddb1dc4b8c19f04189003f1"), "title" : "myReviews", "userName" : "jai", "productName" : "chicken", "productType" : "tablets", "productMaker" : null
"reviewRating" : 4, "reviewDate" : "2019-11-24", "reviewText" : " Good!!", "retailerpin" : "60616", "retailercity" : "Chicago", "price" : 20 }
>
```


MongoDB console which is used for storing reviews.

API's Used:



Yelp API developers account and Yelp business search API.

```python
import requests
import json
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="root",
  passwd="root",
  database="Quickchef"
)
mycursor = mydb.cursor()

business_id = "Hy_H5ebrXaxCBdYuAd0asQ"

API_KEY = "q4RM_cQckxk-B9W9k7a6LRr9Ze30IS_LNQx5cMJFcmINjpFGFqgUMSoVeaBBd0pWe7RL616efHNaX1HcEJPf5Rx7K5WV9nesISyQ35qIfOjqX90UIHN4GYaWNjEXXYx"
ENDPOINT = 'https://api.yelp.com/v3/businesses/search'
HEADERS = {'Authorization': 'bearer %s' % API_KEY}

PARAMETERS = {'term': 'food',
              'limit': 20,
          'location': 'USA'}

response = requests.get(url = ENDPOINT,
                        params = PARAMETERS,
                        headers = HEADERS)

business_data = response.json()
# print(business_data)
for i in business_data['businesses']:
    address =' '.join(i['location']['display_address'])
    sql = "INSERT INTO restaurants (id,name, image_url,rating,latitude,longitude,address) VALUES (%s, %s,%s, %s,%s, %s,%s)"
    val = (i['id'],i['name'], i['image_url'],i['rating'],i['coordinates']['latitude'],i['coordinates']['longitude'],address)
    mycursor.execute(sql, val)
    mydb.commit()
    print(mycursor.rowcount, "record inserted.")


    # print ("Name:", i['name'])
    # print ("Image URL:", i['image_url'])
    # print ("Rating:", i['rating'])
    # print("coordinates-latitude",i['coordinates']['latitude'])
```

Python script that fires the Yelp API to fetch the restaurant details and store it into the MySQL database

Query 1 | SQL File 3* | SQL File 4*

Limit to 1000 rows

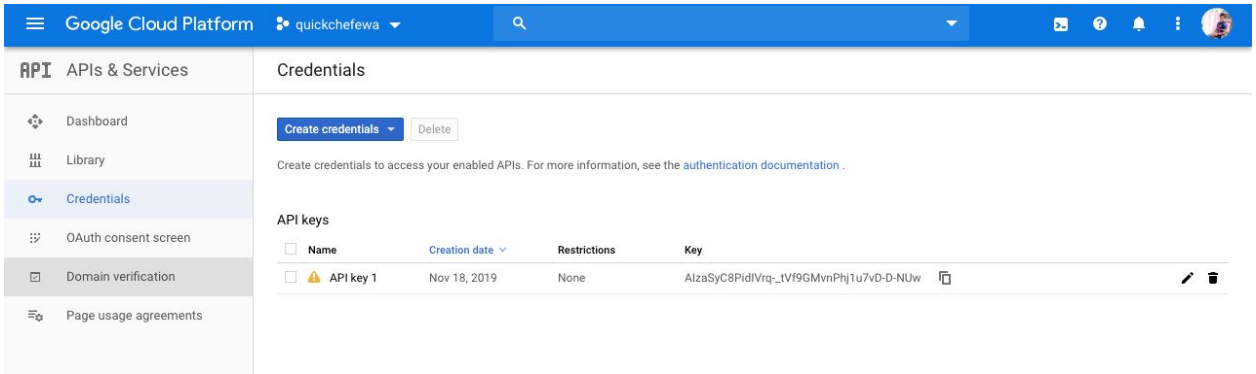
```
Quickchef;
        *       restaurants;
```

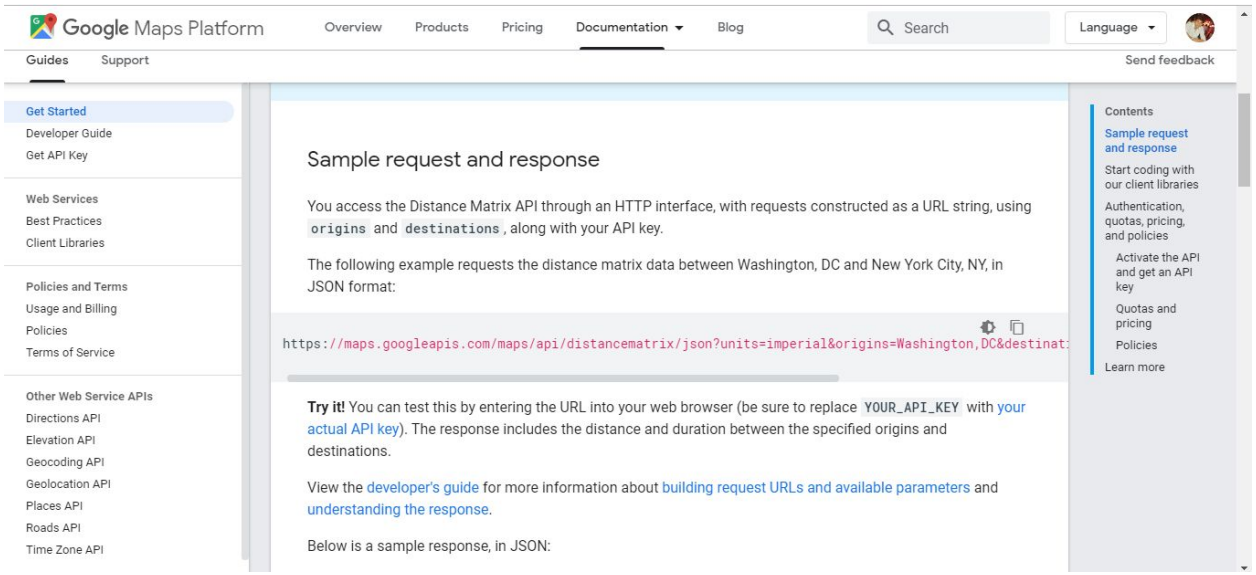
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id	name	image_url	rating	latitude	longitude	address
0eMo-MeawM-tmbwpzv3rDg	Carnitas Uruapan	BLOB	4.5	41.857612801225	-87.669703625051	1725 W 18th St Chicago, IL 60608
4oATsYP0OF0ciHz-_-c_Lw	1492 Cuban Fusion	BLOB	4.5	41.909976	-87.692779	2615 West North Ave Chicago, IL 60647
t2nXebM7aqTj_qq6saZ89g	Eataly - Chicago	BLOB	4.0	41.892335432963	-87.6262607945148	43 E Ohio St Chicago, IL 60611
IPRSekE7gtBLgo6NMhOtLQ	The Crack Shack - Little ...	BLOB	4.0	32.72776	-117.1713	2266 Kettner Blvd San Diego, CA 92101
OfjUGieDXyPB8jWa0pvehA	Homestyle Hawaiian	BLOB	4.0	32.801671968633	-117.159088104705	7524 Mesa College Dr San Diego, CA 92111

restaurants 2 | Read Only

View of MySQL database that stores the restaurants data that is fetched from the Yelp API.





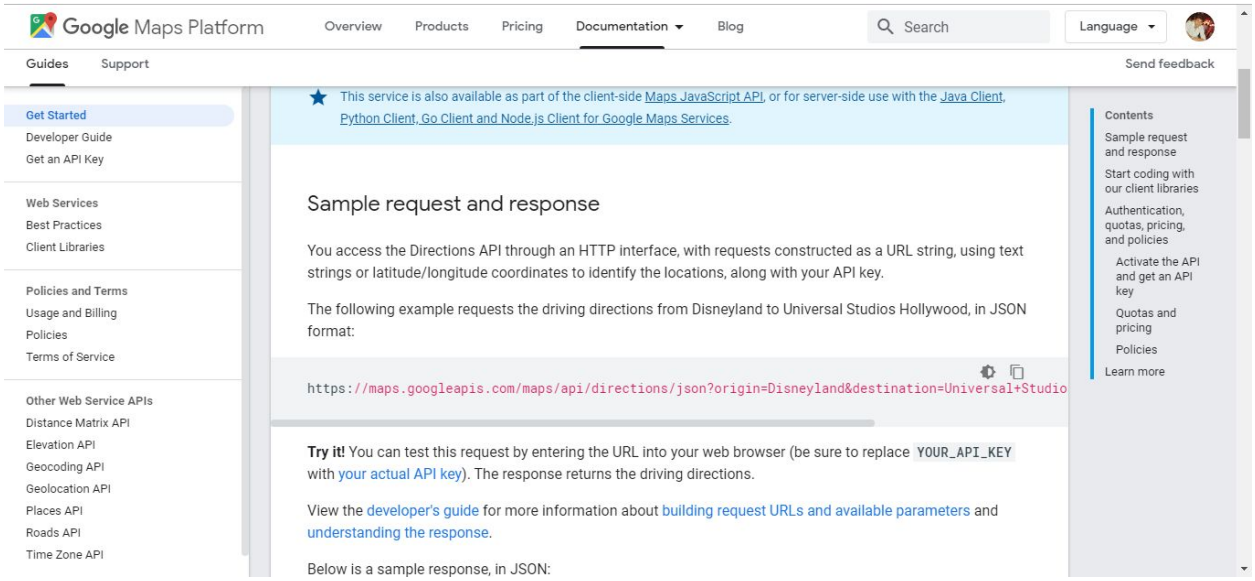

```
try{
    address = address.replaceAll("\\s", "+");
    String url = "https://maps.googleapis.com/maps/api/distancematrix/json?origins="+myaddress+"&destinations="+add
    URL obj = new URL(url);
    HttpURLConnection con = (HttpURLConnection) obj.openConnection();
con.setRequestMethod("GET");

con.setRequestProperty("User-Agent", "Mozilla/5.0");
int responseCode = con.getResponseCode();

BufferedReader in = new BufferedReader(
                new InputStreamReader(con.getInputStream()));
String inputLine;
StringBuffer response1 = new StringBuffer();
while ((inputLine = in.readLine()) != null) {
    response1.append(inputLine);
}
in.close();

JSONObject myResponse = new JSONObject(response1.toString());
```

Google Distance Matrix API is used to predict the time of arrival from the restaurant to the user's location



Google Maps API is used to provide map view of users and restaurant's location, distance and approximate time of arrival