Delivery Route Optimization

1. Problem Statement

To allow small retailers and logistics providers to optimize their delivery costs by providing an optimized route for delivery of items.

2.Delivery Route Optimizer

The route-optimizer that we have built, is a service that uses powerful computer science algorithms to find out an optimal route, given a set of input coordinates. It internally uses OSRM (Open Source Routing Machine) which is a routing engine that computes routes based on map data available from **OpenStreetMap**.

Our route optimizer is currently a publicly available service that hosts a list of APIs for small retailers and logistics providers to consume in their respective applications. The APIs are intended to return optimal routes based on

- criteria such as distance (or) duration
- profile such as driving etc.

Before we dive deep into the route-optimizer service, let's understand the following use-personas.

3.User Personas and Scenarios

User Persona 1

Gupta Kirana Store, is a small retailer, who is not connected to any logistic provider. The store delivers products to the customers on their own i.e. they have their own delivery person employed. On any day, the store receives a set of orders which are handed over to the employed delivery person who would want to find out an optimal route between the set of delivery locations so that the cost or time is subsidized.

For the above user persona, we have our first API which returns an optimal route given a set of input coordinates (longitude and latitude).

Basic route optimization API

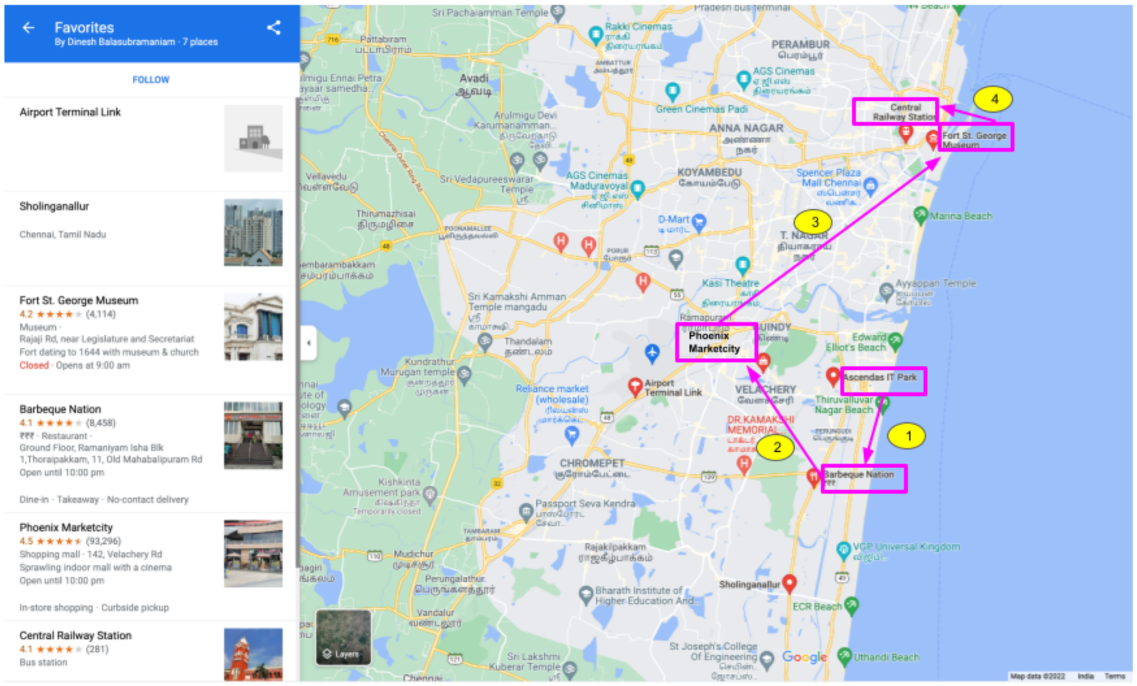
Sample request:

```
route/optimize/driving?locations=80.24604698888692,12.985064343962454;80.28734723901339,13.080515090143628;80.21699786001115,12.99076713804524;80.23815781524532,12.944759186768877;80.27621237532618,13.083670338758505
```

In the above request, driving is the profile and locations is a ; separated string of longitudes, and latitudes i.e. <longitude_1,latitude_1>**;**<longitude_2,latitude_2>.

Sample response:

```
[
   {
      "longitude": 80.24604698888692,
      "latitude": 12.985064343962454
   },
   {
      "longitude": 80.23815781524532,
      "latitude": 12.944759186768877
   },
   {
      "longitude": 80.21699786001115,
      "latitude": 12.99076713804524
   },
   {
      "longitude": 80.28734723901339,
      "latitude": 13.080515090143628
   },
   {
      "longitude": 80.27621237532618,
      "latitude": 13.083670338758505
   }
]
```



Favorites
By Dinesh Balasubramaniam · 7 places

FOLLOW

Airport Terminal Link

Sholinganallur
Chennai, Tamil Nadu

Fort St. George Museum
4.2 ★★★★☆ (4,114)
Museum ·
Rajaji Rd, near Legislature and Secretariat
Fort dating to 1644 with museum & church
Closed · Opens at 9:00 am

Barbeque Nation
4.1 ★★★★☆ (8,458)
₹₹₹ · Restaurant ·
Ground Floor, Ramaniyam Isha Blk
1,Thoraipakkam, 11, Old Mahabalipuram Rd
Open until 10:00 pm

Dine-in · Takeaway · No-contact delivery

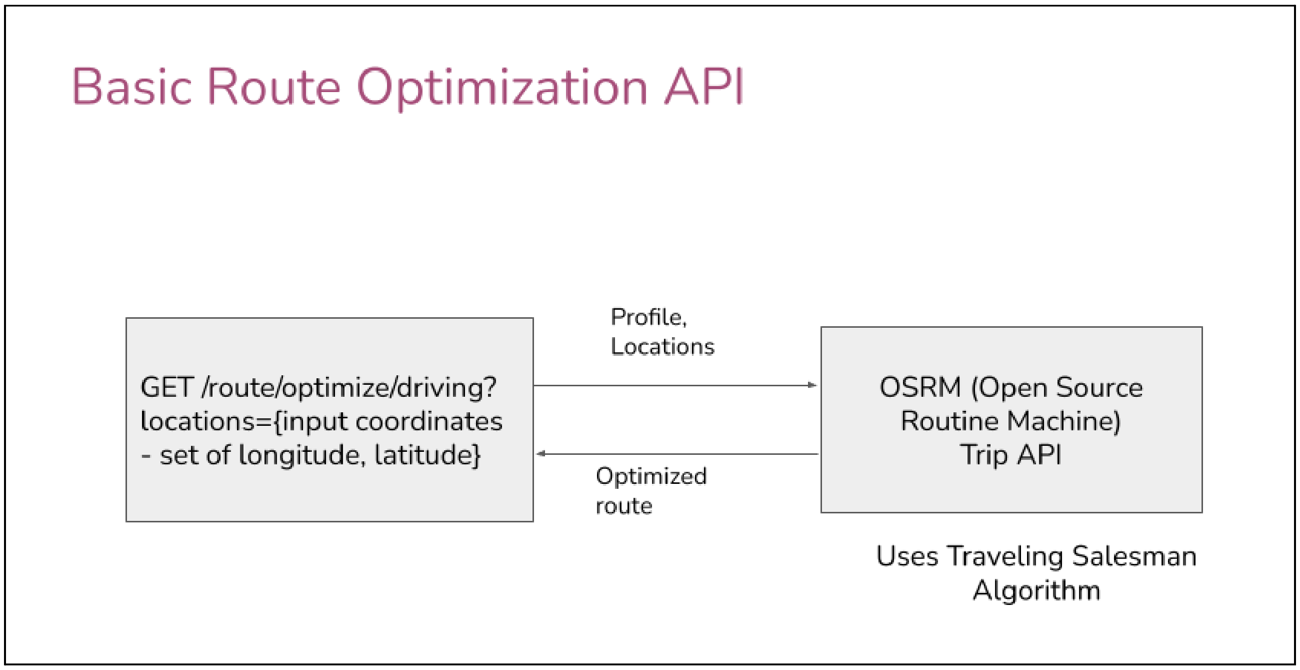
Phoenix Marketcity
4.5 ★★★★★ (93,296)
Shopping mall · 142, Velachery Rd
Sprawling indoor mall with a cinema
Open until 10:00 pm

In-store shopping · Curbside pickup

Central Railway Station
4.1 ★★★★☆ (281)
Bus station

The response is an ordered set of locations which when visited in the same order would produce optimal results for the delivery person.

This API, internally uses OSRM's Route API to fetch the optimal route based on the Traveling salesman algorithm.


Basic Route Optimization API
GET /route/optimize/driving? locations={input coordinates - set of longitude, latitude}
Profile, Locations
OSRM (Open Source Routine Machine) Trip API
Optimized route
Uses Traveling Salesman Algorithm

User Persona 2

Bharat Stores, is a retailer, who is predominantly online based. The store receives online orders and runs its own logistics service to meet the high delivery needs. The logistics service associated with Bharat Stores would like to group the orders optimally and assign them to different delivery agents such that each delivery agent picks up multiple orders which have close delivery points.

Example:

- *Delivery Points A, B and C are nearer to each other. Delivery Points D and E are closer to each other but are farther from A, B and C.*
- *In this case, the logistics service would want to group A, B and C orders and assign them to Delivery Person P1. Similarly, they would want to group D and E and assign them to Delivery Person P2.*
- *Also, Between A, B and C, the delivery person P1 would want to find an optimal route to travel and deliver the orders.*

This is serviced by our second API which groups nearby places based on the number of delivery agents available and returns an optimal route within each group. The API also accepts criteria as an input parameter which can either be distance or duration. This means that the resultant route is either optimized by distance or duration.

Multi-Agent Route optimization API

Sample request:

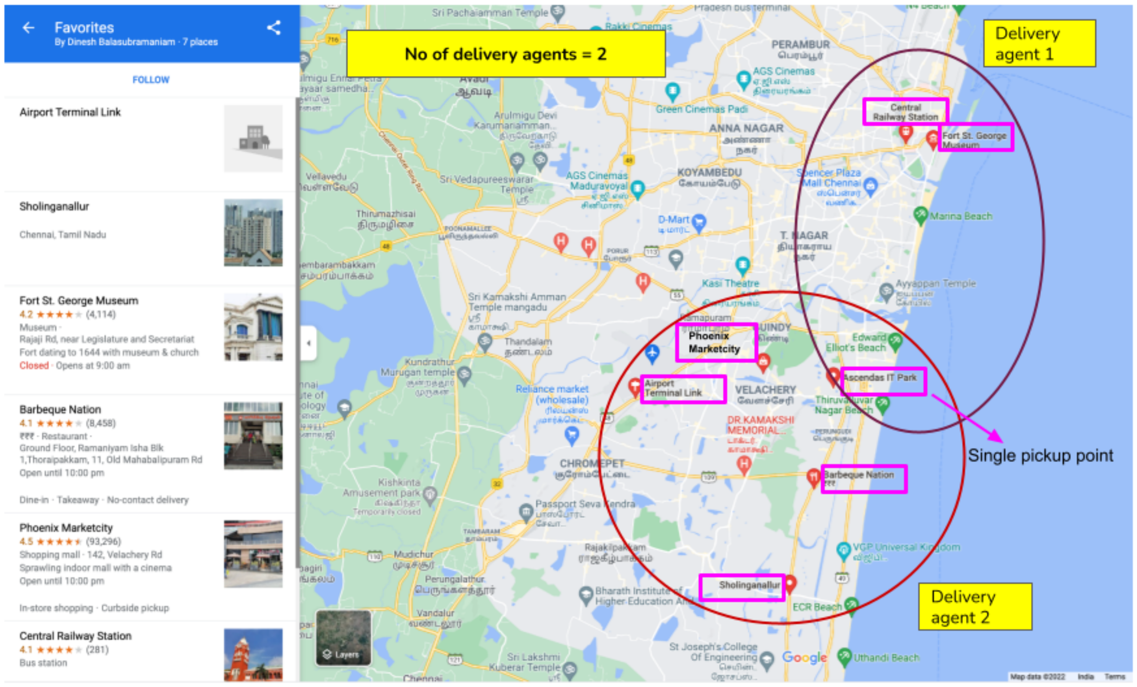
```
POST route/optimize/multi-agent/driving

{
   "orderLocations": [
      {
         "address": "Bharat Stores",
         "longitude": 80.24604698888692,
         "latitude": 12.985064343962454
      },
      {
         "address": "Fort museum",
         "longitude": 80.28734723901339,
```

```json
            "latitude": 13.080515090143628
        },
        {
            "address": "Phoenix Market City Velachery",
            "longitude": 80.21699786001115,
            "latitude": 12.99076713804524
        },
        {
            "address": "Thoraipakkam BBQ",
            "longitude": 80.23815781524532,
            "latitude": 12.944759186768877
        },
        {
            "address": "Central Railway station",
            "longitude": 80.27621237532618,
            "latitude": 13.083670338758505
        },
        {
            "address": "Sholinganallur",
            "longitude": 80.22841984085167,
            "latitude": 12.901309933009536
        },
        {
            "address": "Airport",
            "longitude": 80.17201995204957,
            "latitude": 12.998083269085368
        }
    ],
    "criteria": "duration",
    "no_of_agents": 2
}
```

Sample response:

```
[
// Group 1 - to be serviced by Delivery Person 1
   [
      {
         "address": "Bharat Stores",
         "longitude": 80.24604698888692,
         "latitude": 12.985064343962454
      },
      {
         "address": "Fort museum",
         "longitude": 80.28734723901339,
         "latitude": 13.080515090143628
      },
      {
         "address": "Central Railway station",
         "longitude": 80.27621237532618,
         "latitude": 13.083670338758505
      }
   ],
// Group 2 - to be serviced by Delivery Person 2
   [
      {
         "address": "Bharat Stores",
         "longitude": 80.24604698888692,
         "latitude": 12.985064343962454
      },
      {
         "address": "Sholinganallur",
         "longitude": 80.22841984085167,
         "latitude": 12.901309933009536
      },
      {
         "address": "Thoraipakkam BBQ",
         "longitude": 80.23815781524532,
         "latitude": 12.944759186768877
      },
        {
          "address": "Phoenix Market City Velachery",
          "longitude": 80.21699786001115,
          "latitude": 12.99076713804524
        },
        {
          "address": "Airport",
          "longitude": 80.17201995204957,
          "latitude": 12.998083269085368
        }
      ]
    ]
```


Favorites
By Dinesh Balasubramaniam · 7 places
FOLLOW

Airport Terminal Link

Sholinganallur
Chennai, Tamil Nadu

Fort St. George Museum
4.2 ★★★★☆ (4,114)
Museum ·
Rajaji Rd, near Legislature and Secretariat
Fort dating to 1644 with museum & church
Closed · Opens at 9:00 am

Barbeque Nation
4.1 ★★★★☆ (8,458)
₹₹₹ · Restaurant ·
Ground Floor, Ramaniyam Isha Blk
1,Thoraipakkam, 11, Old Mahabalipuram Rd
Open until 10:00 pm

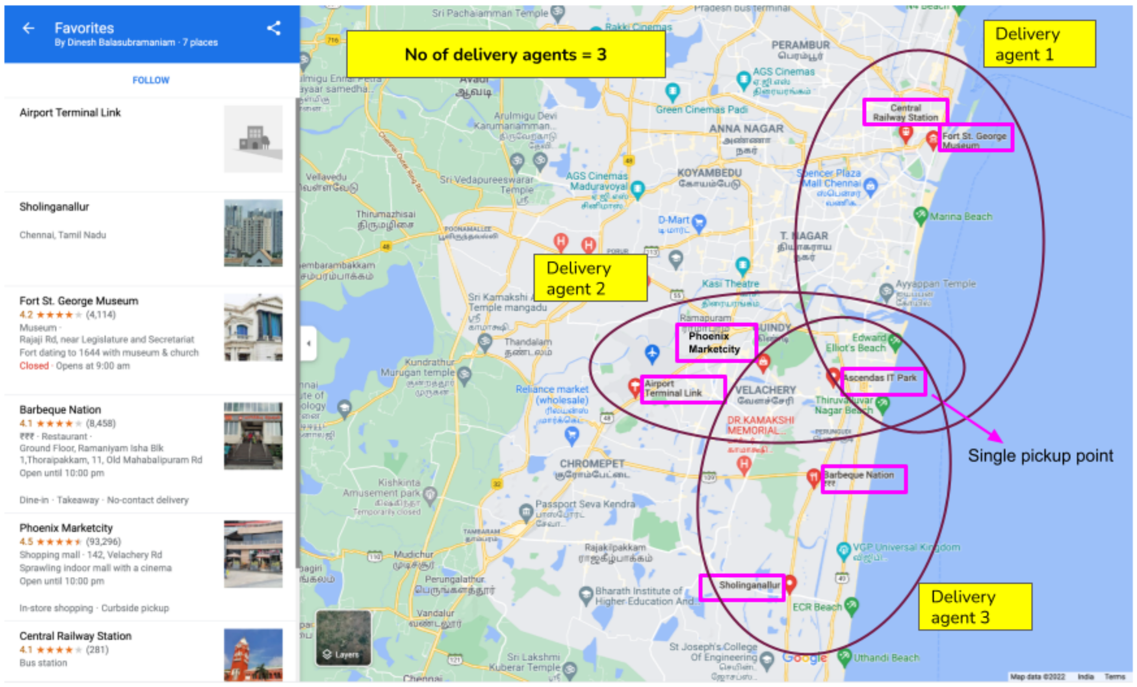
Dine-in · Takeaway · No-contact delivery

Phoenix Marketcity
4.5 ★★★★★ (93,296)
Shopping mall · 142, Velachery Rd
Sprawling indoor mall with a cinema
Open until 10:00 pm

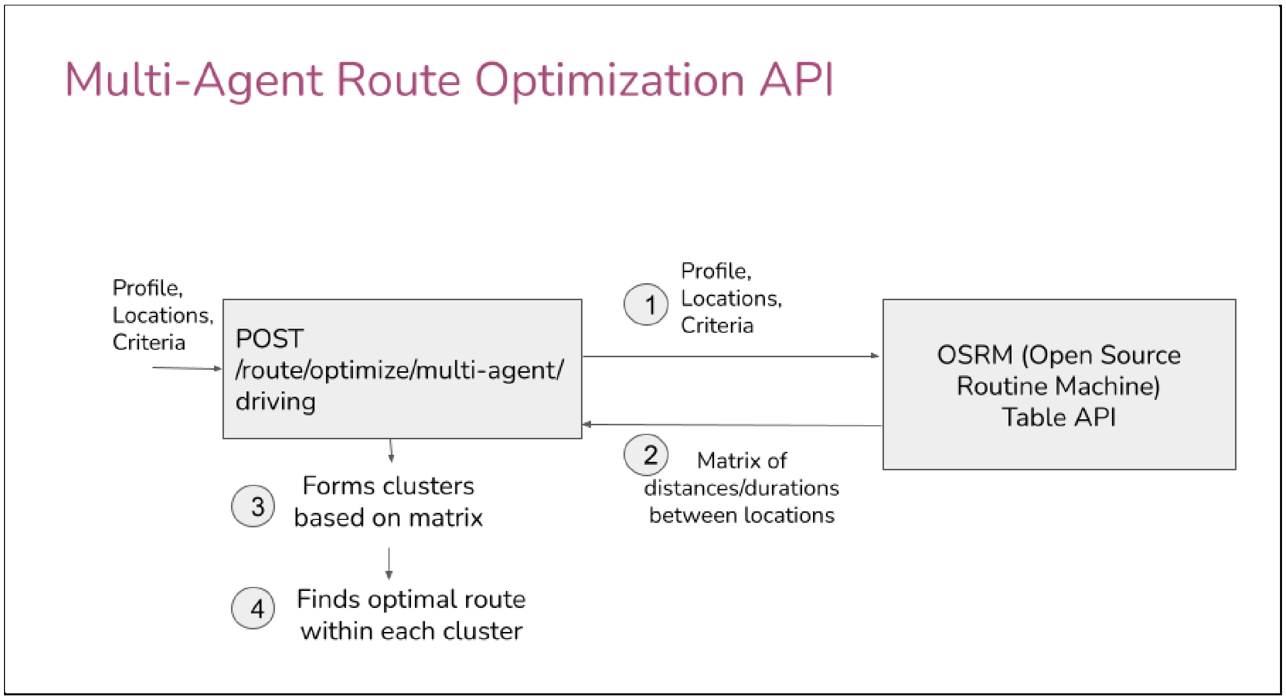
In-store shopping · Curbside pickup

Central Railway Station
4.1 ★★★★☆ (281)
Bus station

8



This API internally uses OSRM's Table API which returns a matrix of distances between each place. On the output matrix, we run the [k-spanning tree] algorithm and find out clusters/groups of closely located places. Each of the output clusters are then subjected to a route optimization algorithm to find an optimal route within them.


Multi-Agent Route Optimization API
Profile, Locations, Criteria
POST /route/optimize/multi-agent/driving
1 Profile, Locations, Criteria
OSRM (Open Source Routine Machine) Table API
2 Matrix of distances/durations between locations
3 Forms clusters based on matrix
4 Finds optimal route within each cluster

User Persona 3

I-Delivery, is a logistics provider, who picks up orders from various retailers and delivers them doorstep to their respective customers. Any delivery person working in I-Delivery would want to find an optimal route to pick up orders from various shops and deliver them to their customers.

This is serviced by our third API which accepts a list of pickup and drop locations and finds out an optimal route between them.

Example:
Given (pickup_1, drop_1), (pickup_2, drop_2), (pickup_3, drop_3),

An optimal route could be - a delivery person picking up an order from pickup_1 and then proceeding to, say, pickup_2 which is nearer to pickup_1. Then the person finds drop_2 to be closer and hence delivers at drop_2. The next closest stop is pickup_3 and hence the person picks up order and finds out that drop_1 is nearer to pickup_3. Now after delivering to drop_1, the person finally arrives at drop_3 to deliver.

pickup_1 → pickup_2 → drop_2 → pickup_3 → drop_1 → drop_3.

The difference from the previous API is that, here **the drop points can be visited only after visiting their corresponding pickup points.** Hence, there is a dependency in the order of locations while finding the optimal route.

This is a very common use-case where a single delivery agent could pick up orders from various places and deliver them respectively.

Multi-Point Delivery Route optimization API

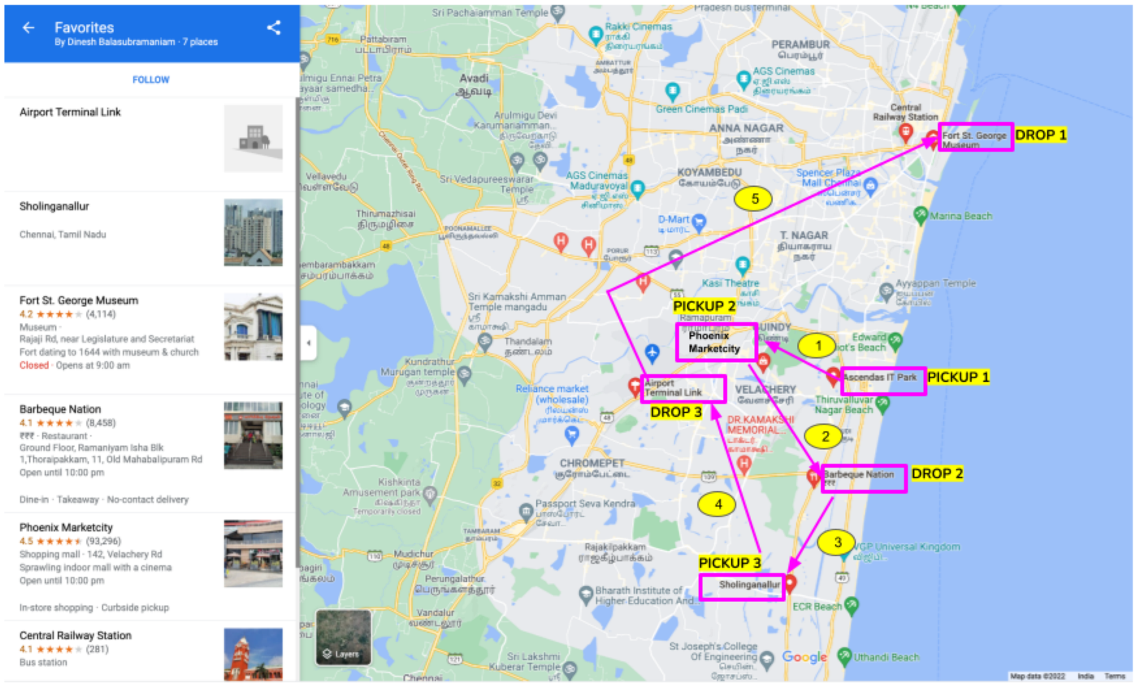
Sample request:

```
{
   "orderLocations": [
      {
         "pickup": {
            "address": "Pickup 1",
            "longitude": 80.24604698888692,
            "latitude": 12.985064343962454
```

```
        },
        "drop": {
            "address": "Drop 1",
            "longitude": 80.28734723901339,
            "latitude": 13.080515090143628
        }
    },
    {
        "pickup": {
            "address": "Pickup 2",
            "longitude": 80.21699786001115,
            "latitude": 12.99076713804524
        },
        "drop": {
            "address": "Drop 2",
            "longitude": 80.23815781524532,
            "latitude": 12.944759186768877
        }
    },
    {
        "pickup": {
            "address": "Pickup 3",
            "longitude": 80.22841984085167,
            "latitude": 12.901309933009536
        },
        "drop": {
            "address": "Drop 3",
            "longitude": 80.17201995204957,
            "latitude": 12.998083269085368
        }
    }
  ],
  "criteria": "distance"
}
```

Sample response:

```
[
   {
      "address": "Pickup 1",
      "longitude": 80.24604698888692,
      "latitude": 12.985064343962454
   },
   {
      "address": "Pickup 2",
      "longitude": 80.21699786001115,
      "latitude": 12.99076713804524
   },
   {
      "address": "Drop 2",
      "longitude": 80.23815781524532,
      "latitude": 12.944759186768877
   },
   {
      "address": "Pickup 3",
      "longitude": 80.22841984085167,
      "latitude": 12.901309933009536
   },
  {
      "address": "Drop 3",
      "longitude": 80.17201995204957,
      "latitude": 12.998083269085368
   },
   {
      "address": "Drop 1",
      "longitude": 80.28734723901339,
      "latitude": 13.080515090143628
   },
]
```


Favorites
By Dinesh Balasubramaniam · 7 places
FOLLOW

Airport Terminal Link

Sholinganallur
Chennai, Tamil Nadu

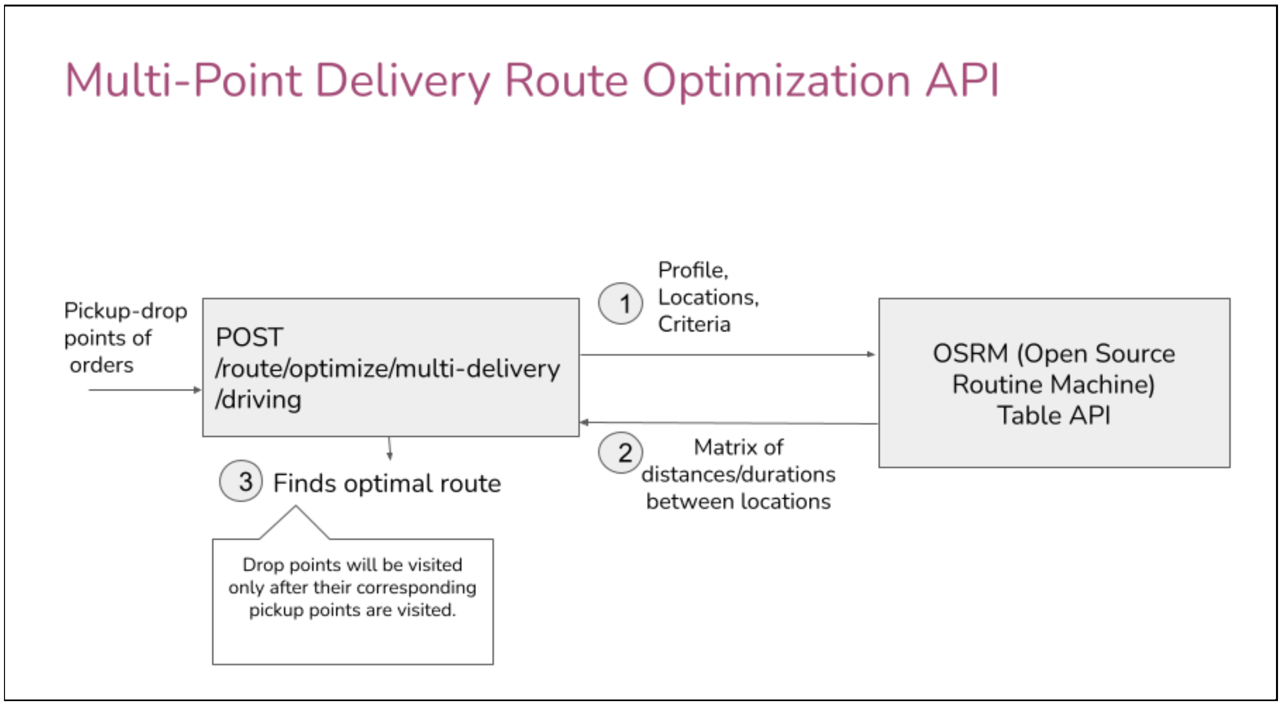
Fort St. George Museum
4.2 ★★★★☆ (4,114)
Museum ·
Rajaji Rd, near Legislature and Secretariat
Fort dating to 1644 with museum & church
Closed · Opens at 9:00 am

Barbeque Nation
4.1 ★★★★☆ (8,458)
₹₹₹ · Restaurant ·
Ground Floor, Ramaniyam Isha Blk
1,Thoraipakkam, 11, Old Mahabalipuram Rd
Open until 10:00 pm
Dine-in · Takeaway · No-contact delivery

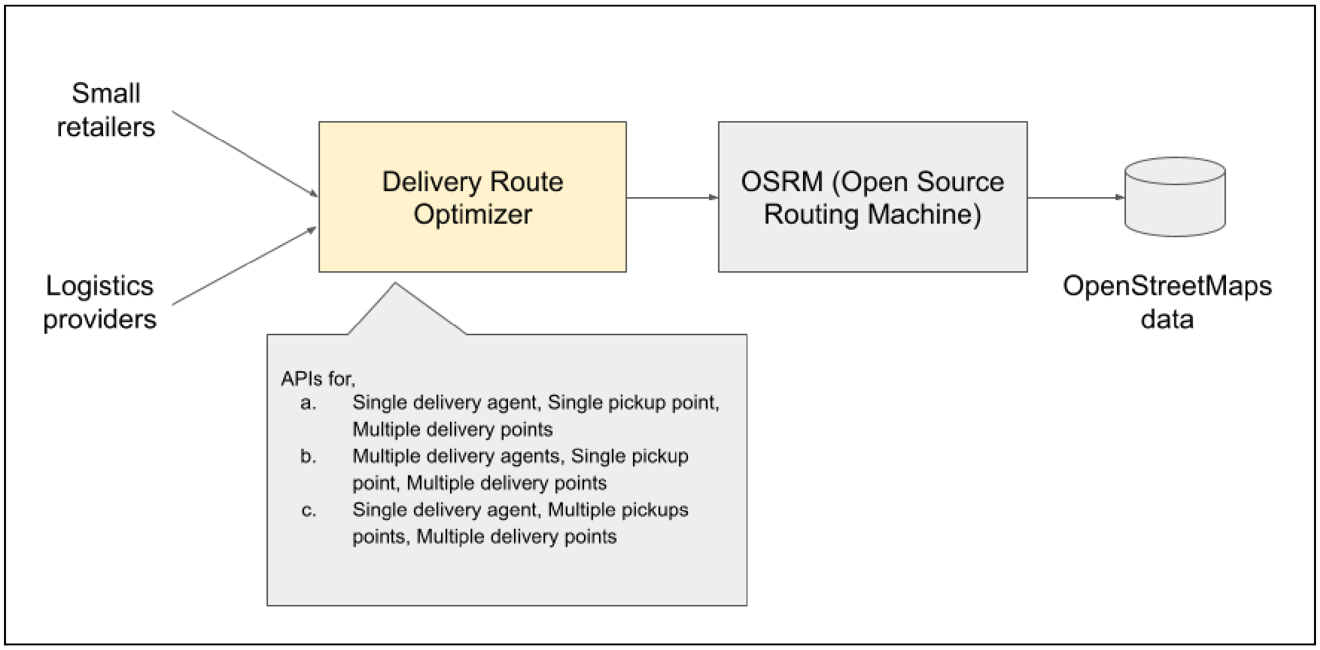
Phoenix Marketcity
4.5 ★★★★★ (93,296)
Shopping mall · 142, Velachery Rd
Sprawling indoor mall with a cinema
Open until 10:00 pm
In-store shopping · Curbside pickup

Central Railway Station
4.1 ★★★★☆ (281)
Bus station

The API internally uses OSRM's Table API to obtain a matrix of distances/durations between different pickup and drop points. From the obtained distance/duration matrix, our route optimizer computes an optimal route by keeping in mind the dependency in the order of visit between pickup and drop points.


Multi-Point Delivery Route Optimization API
Pickup-drop points of orders
POST /route/optimize/multi-delivery /driving
1 Profile, Locations, Criteria
OSRM (Open Source Routine Machine) Table API
2 Matrix of distances/durations between locations
3 Finds optimal route
Drop points will be visited only after their corresponding pickup points are visited.

4.Architecture Diagram


Small retailers
Logistics providers
Delivery Route Optimizer
OSRM (Open Source Routing Machine)
OpenStreetMaps data
APIs for,
a. Single delivery agent, Single pickup point, Multiple delivery points
b. Multiple delivery agents, Single pickup point, Multiple delivery points
c. Single delivery agent, Multiple pickups points, Multiple delivery points

5.Accessing the APIs

The delivery route optimizer service is deployed in Google Cloud Platform (GCP). The APIs are publicly available here.

6.Underlying Algorithms

- ### Traveling Salesman Problem:

  The core algorithm that is used by the route-optimizer for finding the optimized route, is the Traveling Salesman problem. The algorithm aims at visiting every place only once, in an optimized manner such that the total cost involved in traveling is the lowest. Here, the cost could either be *distance* or *duration*.

- k-Spanning tree:

  The algorithm that is used to group delivery locations and map them to multiple delivery agents is k-Spanning tree. The k-spanning tree is a graph clustering algorithm used to identify close clusters in a connected graph. We have used it to find out nearby places so that they can be grouped together and assigned to the same delivery person.

7. Features planned as enhancement

As described in the above 3 APIs, currently, the route optimizer supports the following use-cases in short.

a. Single delivery agent, Single pickup point → Multiple delivery points
b. Multiple delivery agents, Single pickup point → Multiple delivery points
c. Single delivery agent, Multiple pickups points → Multiple delivery points

All the above use-cases support criteria as a parameter such that the route could either be optimized by distance or duration.

Enhancement 1	The next use-case which we would like to address is a. Multiple delivery agents, Multiple pickup points → Multiple delivery points. We aim to reuse the existing components developed as part of the above APIs and enhance the solution to accommodate multi delivery agents picking up from multi points and delivering to multiple locations.
Enhancement 2	We aim to create an SDK for the mobile apps to integrate in order to communicate with our APIs.
Enhancement 3	Use the open source based osrm-frontend, to visually show the optimized route with driving instructions/steps
Enhancement 4	Considering factors such as road closures, water logging etc. while finding optimal routes.

8.How to setup and run locally

- The APIs are implemented using Java’s spring boot framework.
- Build tool used is gradle.

In order to setup locally, the prerequisites are,

Either

a. Java 8 or more
b. Gradle

In order to run the application,

```
./gradlew run
```

Or

a. Docker

In order to run the application,

```
docker run -it -p 8080:8080 dineshba/route-optimizer
```

In order to build locally and run the application,

```
docker build -t route-optimizer .
docker run -it -p 8080:8080 route-optimizer
```

Thank you

Team - **The Quad**

Madhumitha Prabakar
Dinesh Balasubramaniam
Kandan Muthukumar
Kavitha Thiagarajan