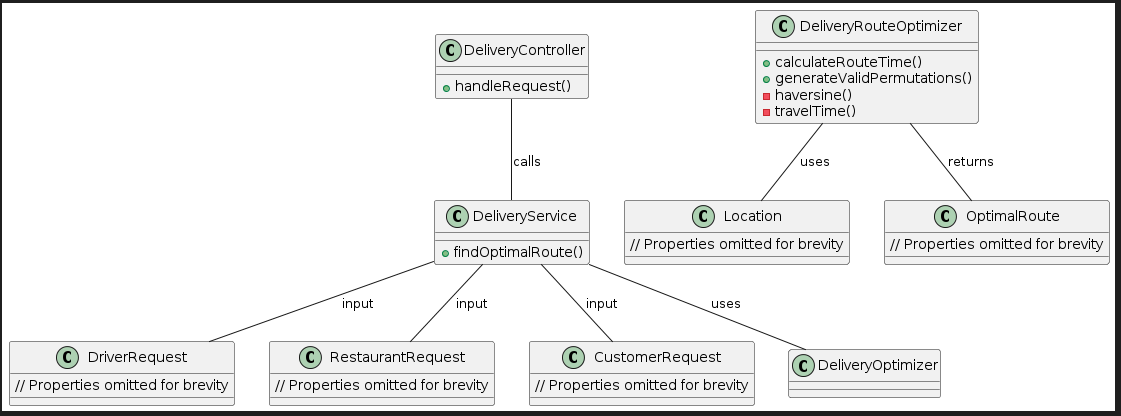
**Best Route Problem**

Best Route Problem is a cool Java Spring Boot app that figures out the best delivery route for a delivery person. It looks at a bunch of restaurant and customer spots and finds the quickest way to deliver everything.

**How It Works**

The Delivery Optimizer application calculates the optimal delivery route using the following steps:

1. Calculate the Haversine distance between two geographical points.
2. Compute travel time based on the distance and a predefined speed.
3. Generate all valid permutations of restaurant and customer visits.
4. Evaluate each permutation to find the one with the minimum total delivery time.
5. Return the optimal route and time in a JSON response.



**API Endpoint**

GET /api/v1/bestRoute

**Payload:**

{

"driver": {

"latitude": 12.345,

"longitude": 67.890

},

"restaurants": [

{

"latitude": 34.567,

"longitude": 89.012,

"preparationTime": 0.5

},

{

"latitude": 45.678,

"longitude": 23.456,

"preparationTime": 0.7

},

{

"latitude": 56.789,

"longitude": 34.567,

"preparationTime": 0.4

}

],

"customers": [

{

"latitude": 78.901,

"longitude": 12.345

},

{

"latitude": 89.012,

"longitude": 45.678

},

{

"latitude": 90.123,

"longitude": 56.789

}

]

}

**Response:**

{

"time": 707.6892162389636,

"path": [

"R1",

"R2",

"R3",

"C1",

"C2",

"C3"

]

}

**Sample Curl:**

You can use below mention curl to hit the API

curl --location --request GET 'http://localhost:8080/api/v1/bestRoute' \

--header 'Content-Type: application/json' \

--data '{

"driver": {

"latitude": 12.345,

"longitude": 67.890

},

"restaurants": [

{

"latitude": 34.567,

"longitude": 89.012,

"preparationTime": 0.5

},

{

"latitude": 45.678,

"longitude": 23.456,

"preparationTime": 0.7

},

{

"latitude": 56.789,

"longitude": 34.567,

"preparationTime": 0.4

}

],

"customers": [

{

"latitude": 78.901,

"longitude": 12.345

},

{

"latitude": 89.012,

"longitude": 45.678

},

{

"latitude": 90.123,

"longitude": 56.789

}

]

}

'

**Classes And Methods**

**Delivery Controller**

Handles the /api/v1/bestRoute endpoint, converting JSON input into appropriate objects and calling the DeliveryService to calculate the optimal route.

**Delivery Service**

Contains the findOptimalRoute method that uses the DeliveryRouteOptimizer utility to compute the optimal delivery route

**DeliveryRouteOptimizer**

Contains all the core logic for calculating distances, travel times, and generating permutations to find the optimal route.

**Models**

DriverRequest

RestaurantRequest

CustomerRequest

Location

OptimalRoute

These classes represent the input and output data structures used by the API.

**Constants**

Defines constants such as EARTH\_RADIUS = 6371 and SPEED = 20.0 used in the distance and time calculations.