**API**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| API | Method | Description | Parameter | Parameter Description | Result |
| /driver/find-driver-within-distance | GET | API to find driver within a given distance in Kilometer | distance : String | Distance in Kilometer | Integer array of driverId |
| locationX: float | For Location’s X |
| locationY: float | For Location’s Y |
| status : int (optional) | Driver Status |
| /booking/create-booking | POST | API to create booking | driverId : int | Driver selected for this booking | The Booking Object. |
| sourceX : float | For Source Location’s X |
| sourceY : float | For Source Location’s Y |
| destinationX: float | For Destination Location’s X |
| destinationY :float | For Destination Location’s Y |

**Business Logic**

**Booking Flow**

1. Create booking with particular driver id, booking\_status = pendingDriverPickUp

2. Driver to decide whether to accept booking

3. If Driver decided to pick up, Booking status will be updated to Booked, booking\_status = booked.

4. Else If Driver Reject the booking, booking\_status = rejected.

5. If Driver is not available for booking, system will reject the booking.

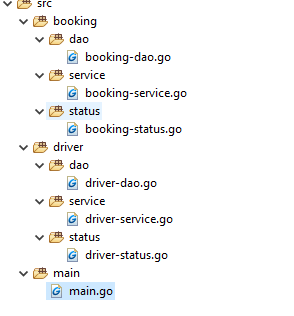
6. Once the booking is confirm booked, a record will be created in booking\_in\_progress table.

**Technical**

**DB: Postgresql**

**Programing Language : GoLang**

**Folder Structure**



Folder path can be interpret in following way

<module>/<function>/<.go>

For example,

booking/dao/booking-dao.go

|  |  |
| --- | --- |
| Function of folder | Description |
| Dao | Domain path. For Domain/DB access. |
| Service | For Business Logic |
| Status | State of Process |

**Formulae**

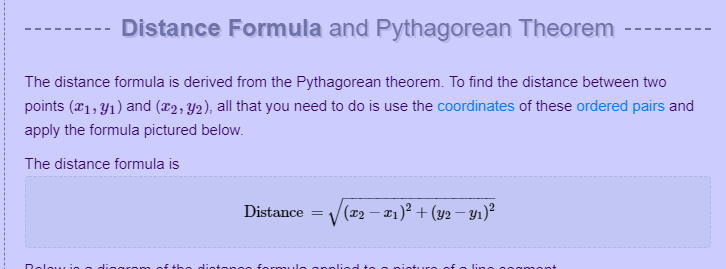
**Assumption**

**1 Decimal Degree = 111.32km**

*Reference : https://en.wikipedia.org/wiki/Decimal\_degrees*

I have created put the formula in database’s function to improve the performance of data retrieval and reduce the processing required in program.

The following formula had used to find the distance between the given point and the drivers.



*Reference: https://www.mathwarehouse.com/algebra/distance\_formula/index.php*

**Which had derived into below sql,**

(sqrt(power(abs(abs(location\_x) - source\_location\_x),2) + power(abs(abs(location\_y) - source\_location\_y),2)) \* kmInDecimalDegree) < distance

Another technic was used to eliminate the drivers that out of the search radius which allows to increase the efficiencies of the queries. This technic is required is because to find out the distance of all drivers in the database would be a heavy processing, thus, only the drivers that fall into the blue color square will be calculated. The drivers that fall into green color area will not be take into account.

**The technic in sql as below,**

location\_x >= (source\_location\_x - (distance / kmInDecimalDegree)) and location\_x <= (source\_location\_x + (distance \* kmInDecimalDegree))

and location\_y >= (source\_location\_y - (distance / kmInDecimalDegree)) and location\_y <= (source\_location\_y + (distance \* kmInDecimalDegree))

**The full Sql**

