Application Initialisation :

Database ‘eventstore’ and ‘readdb’ are created for the application with no table at initial stage

A screen shot of a computer

Description automatically generated

Tables will be created by the application on startup as we mentioned following hibernate property (in both microservices)

spring.jpa.hibernate.ddl-auto=create-drop

After a successful start of application , tables got created as shown.

A screenshot of a computer

Description automatically generated

Topic :

“vehicle\_location “ topic is created in Kafka as a message broker to exchange messages as shown.

A black screen with text

Description automatically generated

Requests :

1. Post request for creating the locations (both latitude and longitude) of a vehicle.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

As on when we inserted different locations for the same vehicle both to database (eventstore) and topic (kafka) , consumer (cabbooking) microservice will fetch these records and get updated its database (readdb).The latest location details will be saved in the table so while booking a cab, these location cordinates will be considered while booking a cab.

As user cannot view the cab location details, we won’t expose any api’s which can be save,update the cab details.They can only book a cab by providing their locations and if any vehicle is nearby their location else booking cant be done.

A screenshot of a computer

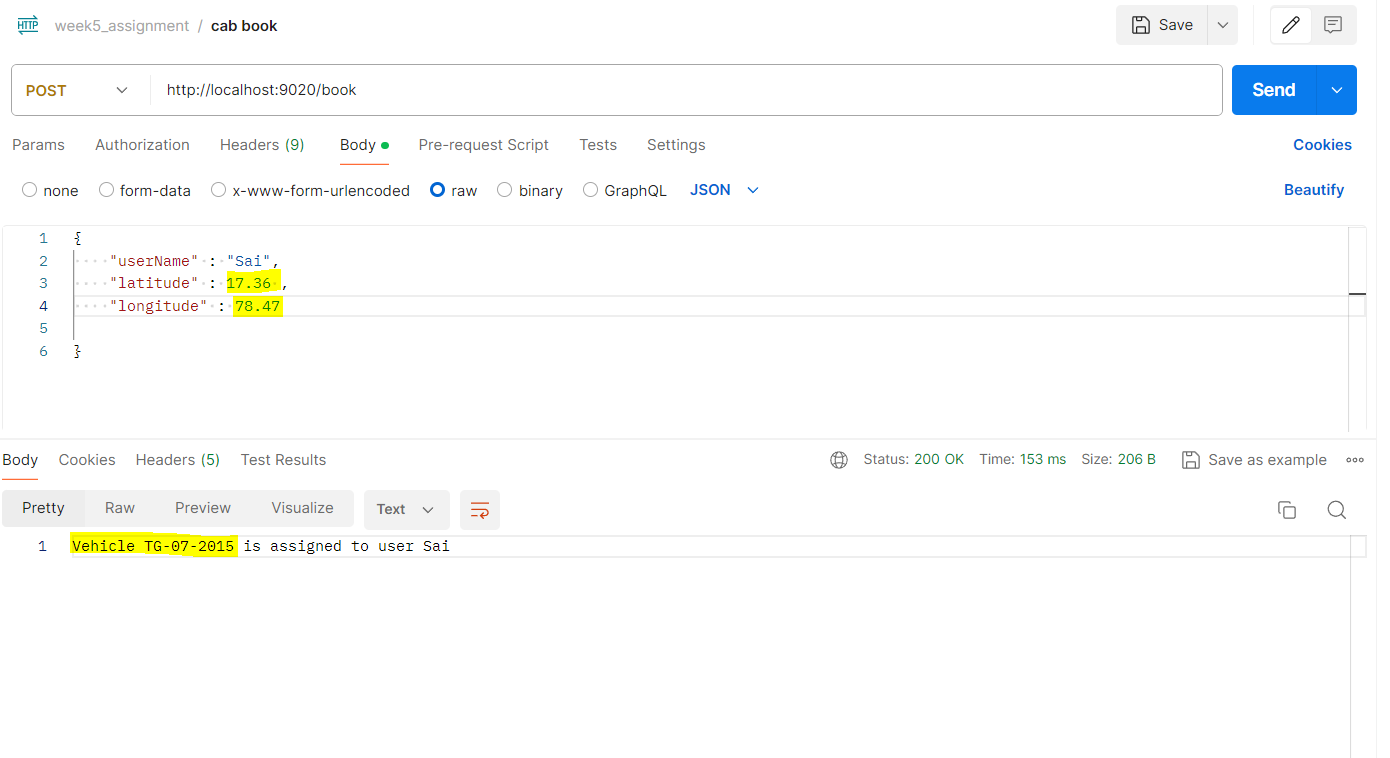
Description automatically generated

1. We can view the locations covered by a vehicle as shown.

A screenshot of a computer

Description automatically generated

1. Book a cab based on the user latitude and longitude provided.



Here the latitude and longitude of the user are 17.36 and 78.47 respectively, whereas the cab details are 17.352 and 78.4825. As per our close logic,

17.352 and 78.4825 -- > vehicle

17.36 and 78.47 🡪 User ( close vehicle can be assigned as the variation in decimal places is +/- 0.01 for 2 digits).

A screenshot of a computer

Description automatically generated

Here the latitude and longitude of the user are 19.46 and 68.72 respectively, whereas the cab details are 19.468 and 68.721. As per our close logic,

19.468 and 68.721 -- > vehicle

19.46 and 68.72 🡪 User ( close vehicle can be assigned as the variation in decimal places is +/- 0.01 for rounding of 2 digits).

A screenshot of a computer

Description automatically generated

Here the latitude and longitude of the user are 16.36 and 77.47 respectively, whereas the cab details are 17.352 and 78.4825. As per our close logic,

17.352 and 78.4825 -- > vehicle

16.36 and 77.47 🡪 User (vehicle cannot be assigned as the integral part itself varies even though decimal places are close.)

A screenshot of a computer

Description automatically generated

Here the latitude and longitude of the user are 17.32 and 78.48 respectively, whereas the cab details are 17.352 and 78.4825. As per our close logic,

17.352 and 78.4825 -- > vehicle

17.32 and 78.48 🡪 User (vehicle cannot be assigned even though integral part is same but decimal places variation is bigger than the range of +/- 0.01).