# Home Delivery Application

# Question Scenario

The ***Home Delivery Application*** is built using Spring Boot*.* The application is used to deliver the products to customer safely to their addresses. The application consists of 3 entities; Delivery Guy Details, User and Delivery Address.

Admin wants to assign the delivery of user ***Robert* to** a delivery person whose name is ***James.*** He wants the order to be delivered between ***5:00 PM*** to ***8:00 PM.*** For that, he needs to check the ***availability*** of ***James*** for the required duration. His availability is from ***4:00 PM*** to ***9:00 PM.***

Since ***James*** is available, assign the delivery to him and change his availability to ***8:00 PM*** to ***01:00 AM*** (Since his total duration of availability is ***5 hours***). Return the response which contains the details of ***Robert*** and ***James.***

If ***James*** is not available, admin has to assign the delivery to others. For that, he needs to check whether or not the location of others is same as ***Robert’s*** delivery address, also their availability. If he finds any delivery person with required criteria assign the delivery to him and change his ***availability***. If he finds multiple delivery guys with required criteria, assign the delivery to the person with the highest ***ratings***. Return the response which contains the details of ***Robert’s*** and ***Delivery*** ***Guy.***

If there are no delivery guys available for requirement, throw an exception with the message saying,

**Sorry...! James is already occupied with other delivery. We are unable to assign delivery of your product to other delivery persons also right now.**

# Objectives

1. ***Database*** layer with required code exists in the application. Complete the logic of methods present in ***Presentation*** and ***Business*** layers. You are allowed to create the ***Persistence***layer according to the requirement. You are not allowed to change the method signature of ***Presentation*** and ***Business*** layers and response should be as mentioned in json format.

1. We have defined below method in ***Presentation*** Layer. You must complete the body of the function mentioned below.

**@PostMapping(“assign/delivery”)**

**public ResponseEntity<ResponseMaster> assignDelivery (@RequestBody AssignDeliveryRequestDTO assignDeliveryRequestDTO):**

This function should read the request from the client and navigate it to ***Business*** layer. On a successful execution, it should return the object of **ResponseMaster.class** by updating the all its variables.

|  |  |
| --- | --- |
| **AssignDeliveryRequestDTO:** This class consists of details such as | userId, deliveryPartnerId, userAddress, |

requiredFrom and requiredTo

|  |  |
| --- | --- |
| **ResponseMaster:** This class consists of details such as error, | message and data |

1. We have defined below method in ***Business***Layer. You must complete the body of the function as mentioned below.

**public AssignDeliveryResponseDTO assignDelivery (AssignDeliveryRequestDTO assignDeliveryRequestDTO):**

This function should make use of all the variables of AssignDeliveryRequestDTO class. You should check the availability of specified delivery partner for given criteria (His location should be same as user address; wherein user address is provided in **AssignDeliveryRequestDTO**.) If the delivery partner is available, update his ***availableFrom*** and ***availableTo*** as mentioned in Question scenario section, map the user object to delivery object partner and save it in database.

|  |  |
| --- | --- |
| This function should return the object of **AssignDeliveryResponseDTO class** by updating the all its | |
| variables on a successful execution. |  |

|  |  |
| --- | --- |
| **AssignDeliveryResponseDTO:** This class consists of location and List< | DeliveryPartnerDTO |

>

|  |  |  |  |
| --- | --- | --- | --- |
| **DeliveryPartnerDTO:** This class consists of details of delivery partner such as | | | deliveryPartnerId, |
| deliveryPartnerName, location, | availableFrom, availableTo and rating |  | |

1. If he is not available, you should check the availability of other delivery partners whose data already exist in database for given criteria. If you find multiple delivery partners, map the user to delivery partner with the highest ***ratings***, update his ***availableFrom*** and ***availableTo*** and save it.
2. If no delivery partner is available, indicate the end user with appropriate exception.

1. You are allowed to create custom JPA methods according to the requirement.
2. You have to create the custom exception classes to cover all the possible scenarios and handle them with controller advice class.

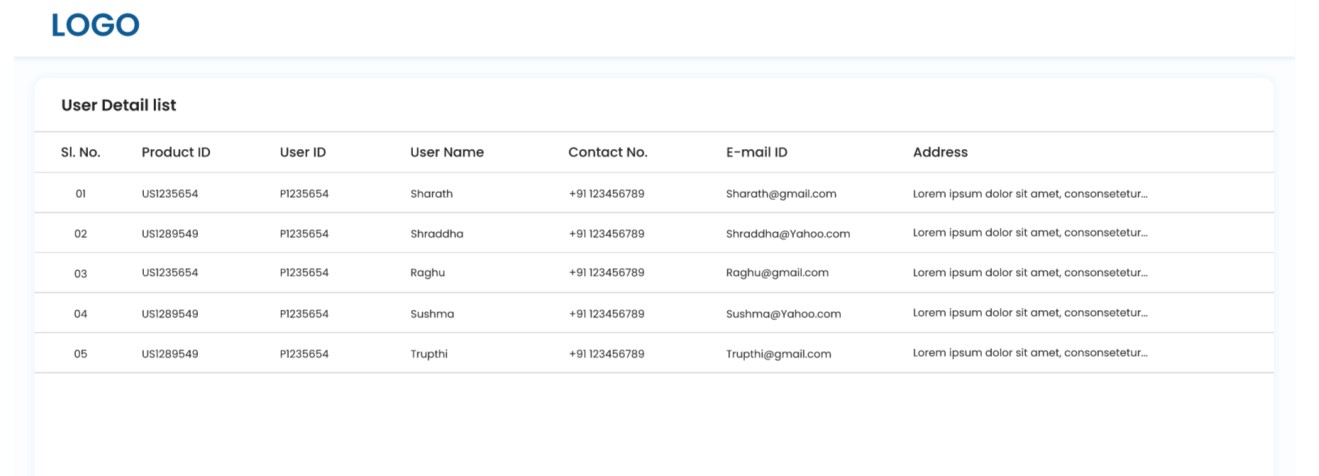




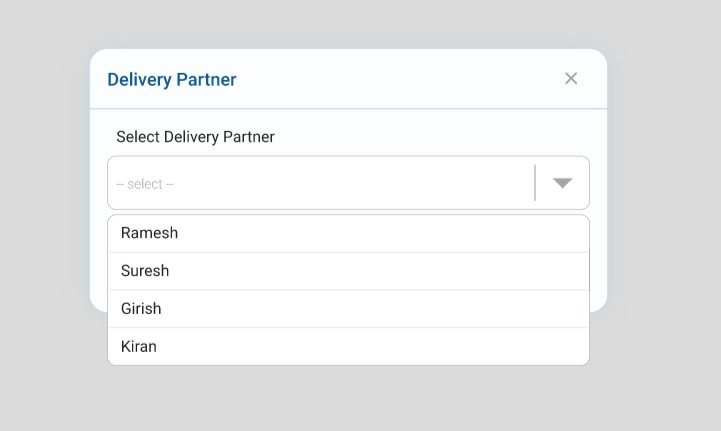


# Sample Screens

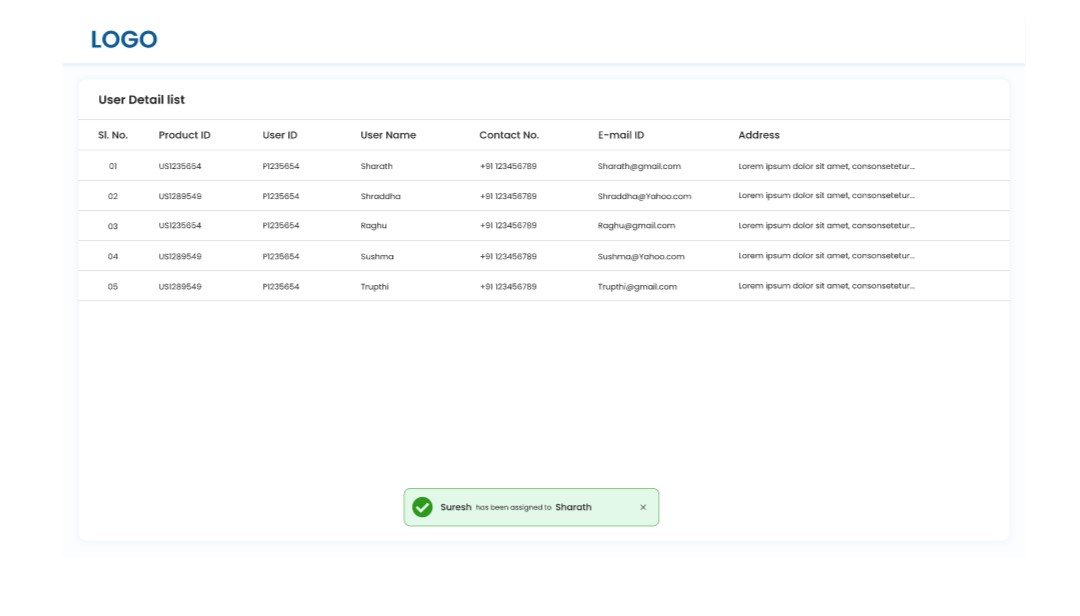
1. **Home Page Screen:** This page contains the details of list of users whose product need to be delivered. Admin can click on the specified user to assign a delivery partner to him.



1. **Assign delivery to delivery partner:** When admin clicks on any user, details of all the delivery partners whose location is same as location of user will be displayed in dropdown. Admin can select any of the delivery partner from the dropdown to assign the delivery.

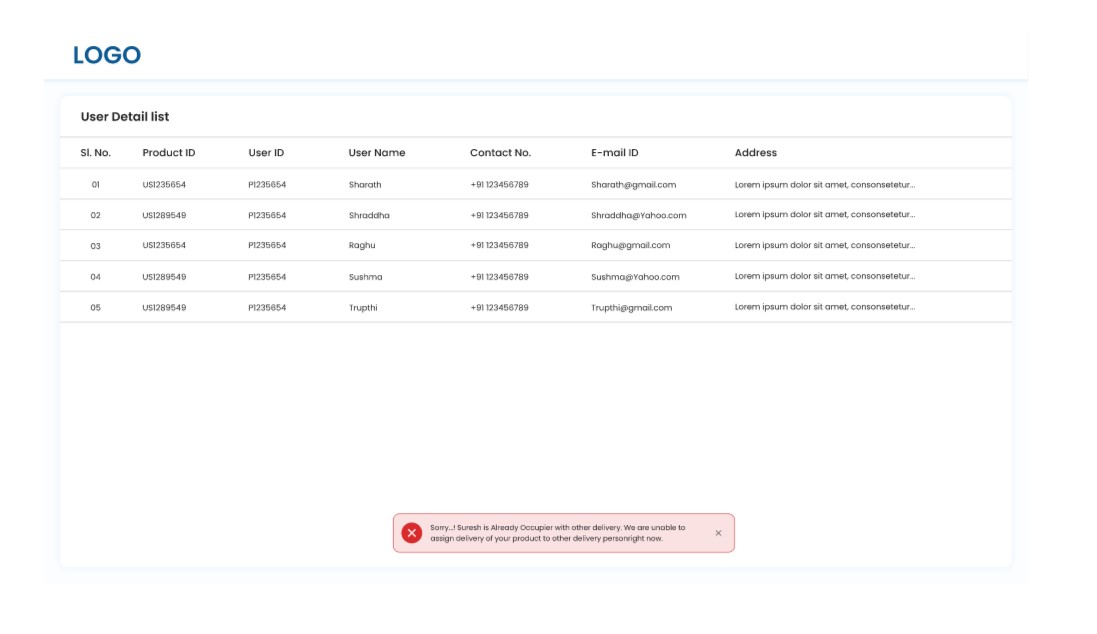


1. If the delivery partner is available, success message will be displayed as below.



1. If none of the delivery partner is available, error message will be displayed as below

**Instructions**



You are not allowed to make any changes in below files.

1. application.properties
2. pom.xml

You are not allowed to make any changes in the files present in below packages.

1. com.tyss.homedelivery.pojo
2. com.tyss.homedelivery.dto

You are not allowed to insert new data to the database.

# Output

**If you find the any delivery guy with required criteria, the output json should be as below,**

{

"error": **false**,

"message": "Delivery for the user: Robert is assigned to the delivery partner:

James successfully.",

"data": {

"userDTO": {

"userId": 1,

"userName": "Robert",

"addressDTOList": []

},

"deliveryPartnerDTO": {

"deliveryPartnerId": 3,

"deliveryPartnerName": "James",

"location": "USA",

"availableFrom": "25-07-2022 06:30:00",

"availableTo": "25-07-2022 07:25:00",

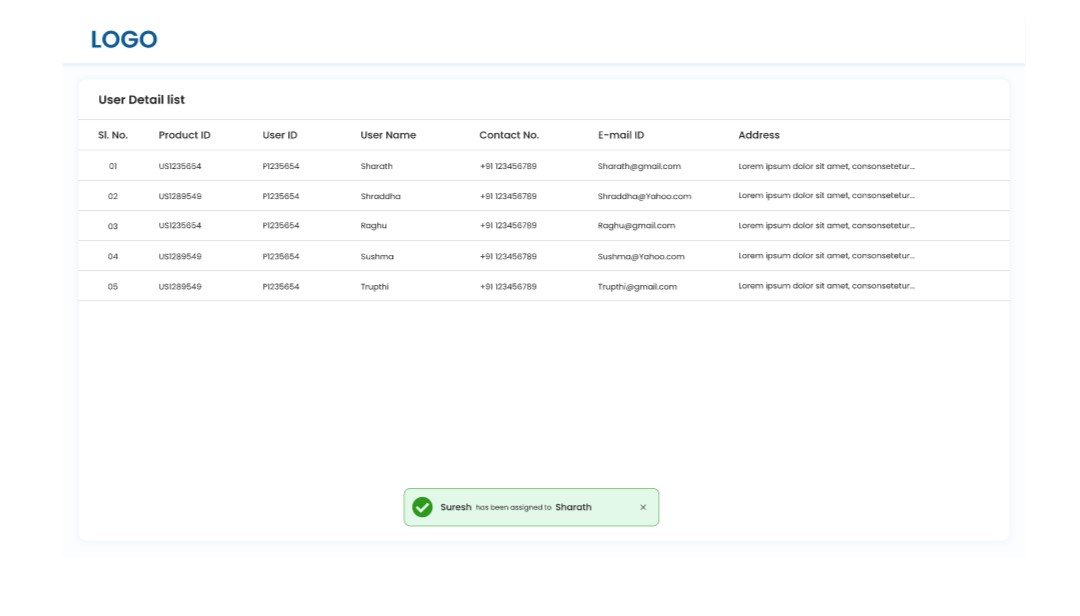
"rating": 4.5

}

}

}

**the final UI image should be as below,**



**If you did not find the any delivery guy with required criteria, throw the exception as below,**

{

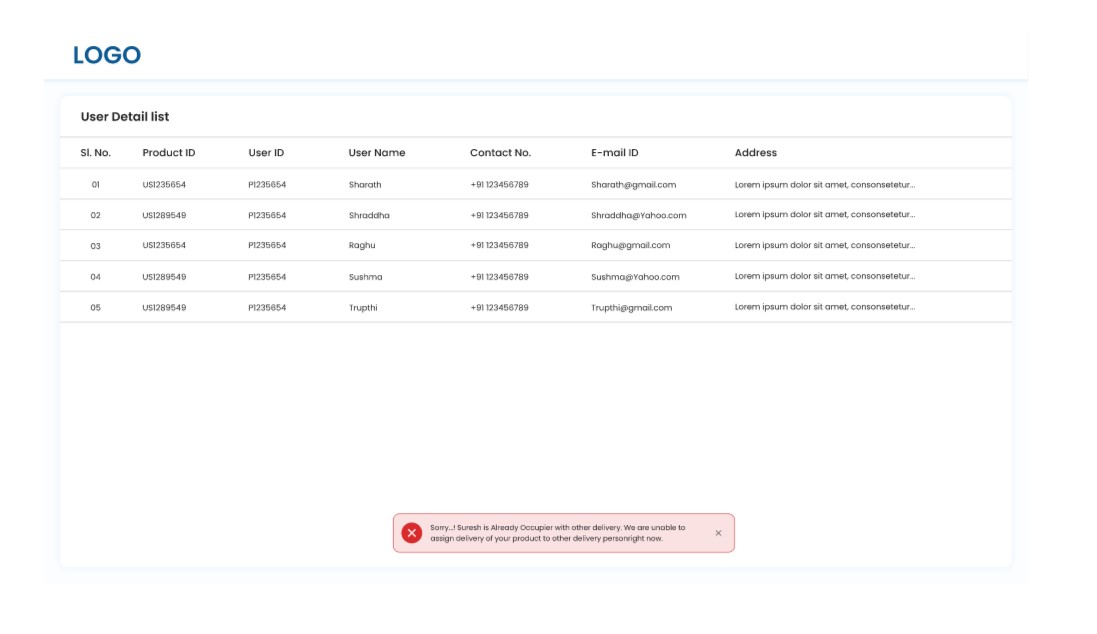
"timestamp": "2022-07-28T14:13:34.825+00:00",

"message": "Sorry..!James is occupied with other delivery. Also, No delivery p artner is available at this time.",

"details": "uri=/delivery-partner/assign/delivery",

"error": **true**

}



# Build, Deploy and Run

1. Load the project in the IDE
2. Update the maven dependency
3. mvn life cycle to build & deploy
4. Run the project

# Project Files

You can only work upon below files. You are allowed to create custom exception classes and controller advice class.

1. HomeDeliveryController.java
2. HomeDeliveryService.java
3. DeliveryGuyRepository.java
4. DeliveryAddressRepository.java