Capstone Project

Name: Aditya Pawar

USN: Aditya_72233016J

1. User-Service

model

```
package com.infosys.User_Service.model;

import jakarta.persistence.*;
import lombok.Data;

@Data
@Entity
@Table(name = "ftr_user")
public class User {
    @Id
    @GeneratedValue(strategy =
    GenerationType.IDENTITY)
    private int userId;
    private String firstName;
    private String lastName;
    private String emailId;
    private Long mobileNumber;
    private String password;
    private String nationality;
    private String passportNumber;
    private String permanentAddress;
    private String officeAddress;
    private Long personalIdentificationNumber;
    private String assignedTerminalId;
}
```

repository

```
package com.infosys.User_Service.repository;

import com.infosys.User_Service.model.User;
import org.springframework.data.jpa.repository
.JpaRepository;
import org.springframework.stereotype.Repository;



@Repository
public interface UserRepo extends
JpaRepository<User, Integer> {
}
```

service

```
package com.infosys.User_Service.service;

import com.infosys.User_Service.feign.TerminalServiceClient;
import com.infosys.User_Service.model.User;
import com.infosys.User_Service.repository.UserRepo;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class UserService {

    @Autowired
    UserRepo userRepo;
```

```
@Autowired
TerminalServiceClient terminalClient;

public User createUser(User user) {
    return userRepo.save(user);
}

public User getUserById(int userId) {
    return userRepo.findById(userId).orElseThrow(() →
    new RuntimeException("User Id not Found"));
}

public User updateUser(int userId, User user) {
    User existingUser = getUserById(userId);

    existingUser.setFirstName(user.getFirstName());
    existingUser.setLastName(user.getLastName());
    existingUser.setEmailId(user.getEmailId());
    existingUser.setMobileNumber(user.getMobileNumber());
    existingUser.setPassword(user.getPassword());
    existingUser.setNationality(user.getNationality());
    existingUser.setPassportNumber(user.getPassportNumber());
    existingUser.setPermanentAddress(user.getPermanentAddress());
    existingUser.setOfficeAddress(user.getOfficeAddress());
    existingUser.setPersonalIdentificationNumber(user
    .getPersonalIdentificationNumber());

    return userRepo.save(existingUser);
}

public void deleteUser(int userId) {
    userRepo.deleteById(userId);
}

public void assignTerminalToUser(int userId, String terminalId) {
```

}

```
        User user = getUserById(userId);

        boolean isValidTerminal = terminalClient.validateTerminal
        (terminalId);
        if (!isValidTerminal) {
            throw new IllegalArgumentException("Invalid Terminal ID");
        }

        user.setAssignedTerminalId(terminalId);
        userRepo.save(user);
    }
}
```

Controller

```
package com.infosys.User_Service.controller;

import com.infosys.User_Service.model.User;
import com.infosys.User_Service.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

@RestController
@RequestMapping("/api/users")
public class UserController {

    @Autowired
    UserService userService;

    @PostMapping
    public ResponseEntity<User> createUser(@RequestBody  User user){
        return ResponseEntity.ok(userService.createUser(user));
    }
    @GetMapping("/{userId}")
    public ResponseEntity<User> getUserById(@PathVariable int userId){
        return ResponseEntity.ok(userService.getUserById(userId));
    }

    @PutMapping("/{userId}")
    public ResponseEntity<User> updateUser(@PathVariable int userId,
    @RequestBody User user){
        return ResponseEntity.ok(userService.updateUser(userId, user));
    }

    @DeleteMapping("/{userId}")
    public ResponseEntity<String> deleteUser(@PathVariable int userId) {
        userService.deleteUser(userId);
        return ResponseEntity.ok("Success");
    }

    @PostMapping("/{userId}/assign-terminal/{terminalId}")
    public ResponseEntity<Void> assignTerminal(
            @PathVariable int userId,
            @PathVariable String terminalId) {
        userService.assignTerminalToUser(userId, terminalId);
        return ResponseEntity.ok().build();
    }
}
```

config

```
package com.infosys.User_Service.config;

import feign.Logger;
import feign.RequestInterceptor;
import org.springframework.context.annotation.Bean;
```

```
public class FeignConfig {
    @Bean
    Logger.Level feignLoggerLevel() {
        return Logger.Level.FULL;
    }

    @Bean
    public RequestInterceptor requestInterceptor() {
        return requestTemplate → {
            requestTemplate.header("Content-Type", "application/json");
        };
    }
}
```

feign

```
package com.infosys.User_Service.feign;

import com.infosys.User_Service.config.FeignConfig;
import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.stereotype.Component;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;

@FeignClient(
        name = "terminal-service",
        configuration = FeignConfig.class,
        fallback = TerminalServiceFallback.class)
public interface TerminalServiceClient {

    @GetMapping("/api/terminals/{terminalId}/validate")
    boolean validateTerminal(@PathVariable String terminalId);
}

@Component
```

```
class TerminalServiceFallback implements TerminalServiceClient {
    @Override
    public boolean validateTerminal(String terminalId) {
        return true;
    }
}
```

UserServiceApplication.java

```
package com.infosys.User_Service;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;
import org.springframework.cloud.openfeign.EnableFeignClients;

@SpringBootApplication
@EnableDiscoveryClient
@EnableFeignClients(basePackages = "com.ftr.userservice.feign")
public class UserServiceApplication {

    public static void main(String[] args) {
        SpringApplication.run(UserServiceApplication.class, args);
    }

}
```

Application.properties

```
spring.application.name=User-Service

server.port=8081
```

```
spring.datasource.url=jdbc:postgresql://localhost:5432/freight_transport_region_
spring.datasource.username=postgres
spring.datasource.password=root
spring.datasource.driver-class-name=org.postgresql.Driver


spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true

eureka.client.service-url.defaultZone=http://localhost:8761/eureka/
eureka.client.register-with-eureka=true
eureka.client.fetch-registry=true
eureka.instance.prefer-ip-address=true

feign.client.config.default.connect-timeout=5000
feign.client.config.default.read-timeout=5000
feign.client.config.default.logger-level=basic
```

2. Terminal-Service

model

```
package com.infosys.Terminal_Service.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;
import jakarta.persistence.Table;
import lombok.Data;

@Data
```

```
@Entity
@Table(name = "ftr_terminals")
public class Terminal {
    @Id
    private String terminalId;
    private String terminalName;
    private String country;
    private String itemType;
    private String terminalDescription;
    private Integer capacity;
    private Integer availableCapacity;
    private String status;
    private String harborLocation;
}
```

repository

```
package com.infosys.Terminal_Service.repository;

import com.infosys.Terminal_Service.model.Terminal;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface TerminalRepo extends JpaRepository<Terminal, String> {
    List<Terminal> findByItemType(String itemType);
    boolean existsByTerminalId(String terminalId);
}
```

Service

```
package com.infosys.Terminal_Service.service;

import com.infosys.Terminal_Service.model.Terminal;
import com.infosys.Terminal_Service.repository.TerminalRepo;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class TerminalService {

    @Autowired
    TerminalRepo terminalRepo;

    public Terminal addTerminal(Terminal terminal) {
        if (terminalRepo.existsById(terminal.getTerminalId())) {
            throw new IllegalArgumentException("Terminal ID already exists");
        }
        return terminalRepo.save(terminal);
    }

    public Terminal getTerminalById(String terminalId) {
        return terminalRepo.findById(terminalId)
                .orElseThrow(() → new RuntimeException("Terminal not found"));
    }


    public List<Terminal> getTerminalsByType(String itemType) {
        return terminalRepo.findByItemType(itemType);
    }

    public Terminal updateTerminalStatus(String terminalId, String status) {
        Terminal terminal = getTerminalById(terminalId);
        terminal.setStatus(status);
        return terminalRepo.save(terminal);
    }

    public List<Terminal> getAllTerminals() {
        return terminalRepo.findAll();
    }

    public boolean existsById(String terminalId) {
        return terminalRepo.existsById(terminalId);
    }
}
```

controller

```
package com.infosys.Terminal_Service.controller;

import com.infosys.Terminal_Service.model.Terminal;
import com.infosys.Terminal_Service.service.TerminalService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/api/terminals")
public class TerminalController {
    @Autowired
    TerminalService terminalService;

    @PostMapping
    public ResponseEntity<Terminal> createTerminal(@RequestBody
    Terminal terminal) {
        return ResponseEntity.ok(terminalService.addTerminal(terminal));
    }
    @GetMapping("/{terminalId}")
    public ResponseEntity<Terminal> getTerminalById(@PathVariable
    String terminalId) {
        return ResponseEntity.ok(terminalService.getTerminalById(terminalId));
    }

    @GetMapping
    public ResponseEntity<List<Terminal>> getTerminalsByType(@RequestParam
    String itemType) {
        return ResponseEntity.ok(terminalService.getTerminalsByType(itemType));
    }

    @PutMapping("/{terminalId}/status")
    public ResponseEntity<Terminal> updateTerminalStatus(
            @PathVariable String terminalId,
            @RequestParam String status) {
        return ResponseEntity.ok(terminalService.updateTerminalStatus
        (terminalId, status));
    }

    @GetMapping("/{terminalId}/validate")
    public ResponseEntity<Boolean> validateTerminal
    (@PathVariable String terminalId) {
        boolean exists = terminalService.existsById(terminalId);
        return ResponseEntity.ok(exists);
    }


    @GetMapping("/all")
    public List<Terminal> getAllTerminal(){
        return terminalService.getAllTerminals();
    }
}
```

TerminalServiceApplication.java

```
package com.infosys.Terminal_Service;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;

@SpringBootApplication
@EnableDiscoveryClient
public class TerminalServiceApplication {

    public static void main(String[] args) {
        SpringApplication.run(TerminalServiceApplication.class, args);
    }

}
```

Application.properties

```
spring.application.name=Terminal-Service
server.port=8082

spring.datasource.url=jdbc:postgresql://localhost:5432/freight_transport_region_
spring.datasource.username=postgres
spring.datasource.password=root
spring.datasource.driver-class-name=org.postgresql.Driver

spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true

eureka.client.service-url.defaultZone=http://localhost:8761/eureka/
```

3. Vehicle-Service

model

```
package com.infosys.Vehicle_Service.model;

import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.Table;
import lombok.Data;

import java.util.Date;

@Data
@Entity
@Table(name = "ftr_vehicle")
public class Vehicle {
    @Id
    private String vehicleNumber;
    private String vehicleName;
    private Integer maxLiftingCapacity;
    private Date retireDate;
    private String vehicleStatus;
    private String country;
    private String harborLocation;
}
```

repository

```
package com.infosys.Vehicle_Service.repository;

import com.infosys.Vehicle_Service.model.Vehicle;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import java.util.List;

@Repository
public interface VehicleRepo extends JpaRepository<Vehicle, String> {
    List<Vehicle> findByVehicleStatus(String status);
    List<Vehicle> findByVehicleNameContainingIgnoreCase(String name);
    boolean existsByVehicleNumber(String vehicleNumber);
}
```

service

```
package com.infosys.Vehicle_Service.service;

import com.infosys.Vehicle_Service.model.Vehicle;
import com.infosys.Vehicle_Service.repository.VehicleRepo;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class VehicleService {

    @Autowired
    VehicleRepo vehicleRepo;


    public Vehicle addVehicle(Vehicle vehicle) {
        if (vehicleRepo.existsById(vehicle.getVehicleNumber())) {
            throw new IllegalArgumentException("Vehicle already exists");
        }
        return vehicleRepo.save(vehicle);
    }
    public List<Vehicle> getAllAvailableVehicles() {
        return vehicleRepo.findAll();
    }

    public Vehicle getVehicleByNumber(String vehicleNumber) {
        return vehicleRepo.findById(vehicleNumber)
                .orElseThrow(() → new RuntimeException("Vehicle not found"));
    }

    public List<Vehicle> getVehiclesByName(String name) {
        return vehicleRepo.findByVehicleNameContainingIgnoreCase(name);
    }

    public void updateVehicleStatus(String vehicleNumber, String status) {
        Vehicle vehicle = getVehicleByNumber(vehicleNumber);
        vehicle.setVehicleStatus(status);
        vehicleRepo.save(vehicle);
    }

    public String deleteVehicle(String number) {
        vehicleRepo.deleteById(number);
        return "deleted";
    }
}
```

controller

```
package com.infosys.Vehicle_Service.controller;

import com.infosys.Vehicle_Service.model.Vehicle;
import com.infosys.Vehicle_Service.service.VehicleService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import java.util.List;

@RestController
@RequestMapping("/api/vehicles")
public class VehicleController {

    @Autowired
    VehicleService vehicleService;

    @PostMapping
    public ResponseEntity<Vehicle> createVehicle(@RequestBody Vehicle vehicle
        return ResponseEntity.ok(vehicleService.addVehicle(vehicle));
    }

    @GetMapping
    public ResponseEntity<List<Vehicle>> getAllAvailableVehicles() {
        return ResponseEntity.ok(vehicleService.getAllAvailableVehicles());
    }

    @GetMapping("/search")
    public ResponseEntity<List<Vehicle>> getVehiclesByName(@RequestParam
    String name) {
        return ResponseEntity.ok(vehicleService.getVehiclesByName(name));
    }

    @GetMapping("/{vehicleNumber}")
    public ResponseEntity<Vehicle> getVehicleByNumber(@PathVariable
    String vehicleNumber) {
        return ResponseEntity.ok(
        vehicleService.getVehicleByNumber(vehicleNumber));
    }

    @PutMapping("/{vehicleNumber}/status")
    public ResponseEntity<Void> updateStatus(
            @PathVariable String vehicleNumber,
            @RequestParam String status) {
```

```
        vehicleService.updateVehicleStatus(vehicleNumber, status);
        return ResponseEntity.noContent().build();
    }

    @DeleteMapping("/{number}")
    public ResponseEntity<String> deleteVehicle(@PathVariable String number){
        String result = vehicleService.deleteVehicle(number);
        return ResponseEntity.ok(result);
    }
}
```

VehicleServiceApplication.java

```
package com.infosys.Vehicle_Service;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;

@SpringBootApplication
@EnableDiscoveryClient
public class VehicleServiceApplication {

    public static void main(String[] args) {
        SpringApplication.run(VehicleServiceApplication.class, args);
    }

}
```

Application.properties

```
spring.application.name=Vehicle-Service
server.port=8083
```

```
spring.datasource.url=jdbc:postgresql://localhost:5432/freight_transport_region_
spring.datasource.username=postgres
spring.datasource.password=root
spring.datasource.driver-class-name=org.postgresql.Driver

spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true


eureka.client.service-url.defaultZone=http://localhost:8761/eureka/
```

4. WorkItem-Service

model

```
package com.infosys.WorkItem_Service.model;

import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.Table;
import lombok.Data;

import java.util.Date;

@Data
@Entity
@Table(name = "ftr_workitem")
public class WorkItem {
    @Id
    private String workItemId;
    private Long userId;
    private String itemName;
    private String itemType;
    private String itemDescription;
    private String messageToRecipient;
    private String quantity;
    private String collectionCountry;
    private String destinationCountry;
    private String originHarborLocation;
    private String selectedHarborLocations;
    private Date shippingDate;
    private Integer amount;
    private String status;
    private String assignedTerminalId;
    private String assignedVehicleNumber;
}
```

```
package com.infosys.WorkItem_Service.model;

import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.Table;
import lombok.Data;

@Data
@Entity
@Table(name = "ftr_vehicle_workitem")
public class VehicleWorkItem {
    @Id
    private String vehicleNumber;
    private String workItemId;
    private String assignedWorkItemStatus;
}
```

```
package com.infosys.WorkItem_Service.model;

import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.Table;
import lombok.Data;

@Data
@Entity
@Table(name = "ftr_workitem_terminal")
public class WorkItemTerminal {
    @Id
    private String workItemId;
    private String terminalId;

}
```

repository

```
package com.infosys.WorkItem_Service.repository;

import com.infosys.WorkItem_Service.model.WorkItem;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface WorkItemRepo extends JpaRepository<WorkItem, String> {
    List<WorkItem> findByUserId(int userId);
    List<WorkItem> findByAssignedVehicleNumber(String vehicleNumber);
    boolean existsByWorkItemId(String workItemId);
}
```

service

```
package com.infosys.WorkItem_Service.service;

import com.infosys.WorkItem_Service.feign.TerminalServiceClient;
import com.infosys.WorkItem_Service.feign.UserServiceClient;
import com.infosys.WorkItem_Service.feign.VehicleServiceClient;
import com.infosys.WorkItem_Service.model.WorkItem;
import com.infosys.WorkItem_Service.repository.WorkItemRepo;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class WorkItemService {

    @Autowired
    WorkItemRepo workItemRepo;

    @Autowired
    UserServiceClient userClient;
    @Autowired
    private TerminalServiceClient terminalClient;

    @Autowired
    private VehicleServiceClient vehicleClient;

    public WorkItem createWorkItem(WorkItem workItem) {

        if (!userClient.validateUser(workItem.getUserId())) {
            throw new IllegalArgumentException("Invalid User ID");
        }
        return workItemRepo.save(workItem);
    }
```

```
public WorkItem getWorkItemById(String workItemId) {
    return workItemRepo.findById(workItemId)
            .orElseThrow(() → new RuntimeException("WorkItem not found"));
}

public void assignTerminal(String workItemId, String terminalId) {
    WorkItem workItem = getWorkItemById(workItemId);

    // Validate terminal via Feign
    if (!terminalClient.validateTerminal(terminalId)) {
        throw new IllegalArgumentException("Invalid Terminal ID");
    }

    workItem.setAssignedTerminalId(terminalId);
    workItemRepo.save(workItem);
}


public void allocateVehicle(String workItemId, String vehicleNumber) {
    WorkItem workItem = getWorkItemById(workItemId);

    if (!vehicleClient.validateVehicle(vehicleNumber)) {
        throw new IllegalArgumentException("Invalid Vehicle Number");
    }

    workItem.setAssignedVehicleNumber(vehicleNumber);
    workItemRepo.save(workItem);
}


public List<WorkItem> getWorkItemsByUser(int userId) {
    return workItemRepo.findByUserId(userId) ;
}
```

```
    public List<WorkItem> getWorkItemsByVehicle(String vehicleNumber) {
        return workItemRepo.findByAssignedVehicleNumber(vehicleNumber);
    }

    public List<WorkItem> geAllWorkItems() {
        return workItemRepo.findAll();
    }
}
```

controller

```
package com.infosys.WorkItem_Service.controler;

import com.infosys.WorkItem_Service.model.WorkItem;
import com.infosys.WorkItem_Service.service.WorkItemService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/api/workitems")
public class WorkItemController {

    @Autowired
    WorkItemService workItemService;

    @PostMapping
    public ResponseEntity<WorkItem> createWorkItem(@RequestBody
    WorkItem workItem) {
        return ResponseEntity.ok(workItemService.createWorkItem(workItem));
    }

    @PostMapping("/{workItemId}/assign-terminal/{terminalId}")
    public ResponseEntity<Void> assignTerminal(
            @PathVariable String workItemId,
            @PathVariable String terminalId) {
        workItemService.assignTerminal(workItemId, terminalId);
        return ResponseEntity.ok().build();
    }

    @PostMapping("/{workItemId}/allocate-vehicle/{vehicleNumber}")
    public ResponseEntity<Void> allocateVehicle(
            @PathVariable String workItemId,
            @PathVariable String vehicleNumber) {
        workItemService.allocateVehicle(workItemId, vehicleNumber);
        return ResponseEntity.ok().build();
    }

    @GetMapping("/user/{userId}")
    public ResponseEntity<List<WorkItem>> getWorkItemsByUser
    (@PathVariable int userId) {
        return ResponseEntity.ok(workItemService.getWorkItemsByUser(userId));
    }

    @GetMapping
    public List<WorkItem> getAllworkItems(){
        return workItemService.geAllWorkItems();
    }

    @GetMapping("/vehicle/{vehicleNumber}")
    public ResponseEntity<List<WorkItem>> getWorkItemsByVehicle(
            @PathVariable String vehicleNumber) {
        return ResponseEntity.ok(workItemService.getWorkItemsByVehicle
        (vehicleNumber));
    }
}
```

config

```
package com.infosys.WorkItem_Service.config;

import feign.Logger;
import feign.RequestInterceptor;
import org.springframework.context.annotation.Bean;

public class FeignConfig {
    @Bean
    Logger.Level feignLoggerLevel() {
        return Logger.Level.FULL;
    }

    @Bean
    public RequestInterceptor requestInterceptor() {
        return requestTemplate → {
            requestTemplate.header("Content-Type", "application/json");
        };
    }
}
```

feign

```
package com.infosys.WorkItem_Service.feign;

import com.infosys.WorkItem_Service.config.FeignConfig;
import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.stereotype.Component;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;

@FeignClient(
    name = "terminal-service",
    configuration = FeignConfig.class,
    fallback = TerminalServiceFallback.class
)
public interface TerminalServiceClient {
    @GetMapping("/api/terminals/{terminalId}/validate")
    boolean validateTerminal(@PathVariable String terminalId);
}

// Fallback
@Component
class TerminalServiceFallback implements TerminalServiceClient {
    @Override
    public boolean validateTerminal(String terminalId) {
        return true;
    }
}
```

```
package com.infosys.WorkItem_Service.feign;

import com.infosys.WorkItem_Service.config.FeignConfig;
import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.stereotype.Component;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;

@FeignClient(
    name = "user-service",
    configuration = FeignConfig.class,
    fallback = UserServiceFallback.class
)
public interface UserServiceClient {
    @GetMapping("/api/users/{userId}/validate")
    boolean validateUser(@PathVariable Long userId);
}

@Component
class UserServiceFallback implements UserServiceClient {
    @Override
    public boolean validateUser(Long userId) {
        return true;
    }
}
```

```
package com.infosys.WorkItem_Service.feign;

import com.infosys.WorkItem_Service.config.FeignConfig;
import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.stereotype.Component;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;

@FeignClient(
    name = "vehicle-service",
    configuration = FeignConfig.class,
    fallback = VehicleServiceFallback.class
)
public interface VehicleServiceClient {
    @GetMapping("/api/vehicles/{vehicleNumber}/validate")
    boolean validateVehicle(@PathVariable String vehicleNumber);
}

// Fallback
@Component
class VehicleServiceFallback implements VehicleServiceClient {
    @Override
    public boolean validateVehicle(String vehicleNumber) {
        return true;
    }
}
```

WorkItemServiceApplication.java

```
package com.infosys.WorkItem_Service;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;
import org.springframework.cloud.openfeign.EnableFeignClients;

@SpringBootApplication
@EnableDiscoveryClient
@EnableFeignClients(basePackages = "com.ftr.workitemservice.feign")
public class WorkItemServiceApplication {

    public static void main(String[] args) {
        SpringApplication.run(WorkItemServiceApplication.class, args);
    }

}
```

Application.properties

```
spring.application.name=WorkItem-Service
server.port=8084

spring.datasource.url=jdbc:postgresql://localhost:5432/freight_transport_region_
spring.datasource.username=postgres
spring.datasource.password=root
spring.datasource.driver-class-name=org.postgresql.Driver



spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
```

```
eureka.client.service-url.defaultZone=http://localhost:8761/eureka/
feign.client.config.default.connect-timeout=5000
feign.client.config.default.read-timeout=5000
```

Eureka Server

EurekaServerApplication.java

```
package com.infosys.EurekaServer;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.netflix.eureka.server.EnableEurekaServer;

@SpringBootApplication
@EnableEurekaServer
public class EurekaServerApplication {

    public static void main(String[] args) {
        SpringApplication.run(EurekaServerApplication.class, args);
    }

}
```

Application.properites

```
spring.application.name=EurekaServer

server.port=8761

eureka.client.register-with-eureka=false
eureka.client.fetch-registry=false

eureka.instance.prefer-ip-address=true
```

Api Gateway

ApiGatewayApplication.java

```
package com.infosys.ApiGateway;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.client.discovery.EnableDiscoveryClient;

@SpringBootApplication
@EnableDiscoveryClient
public class ApiGatewayApplication {

    public static void main(String[] args) {
        SpringApplication.run(ApiGatewayApplication.class, args);
    }

}
```

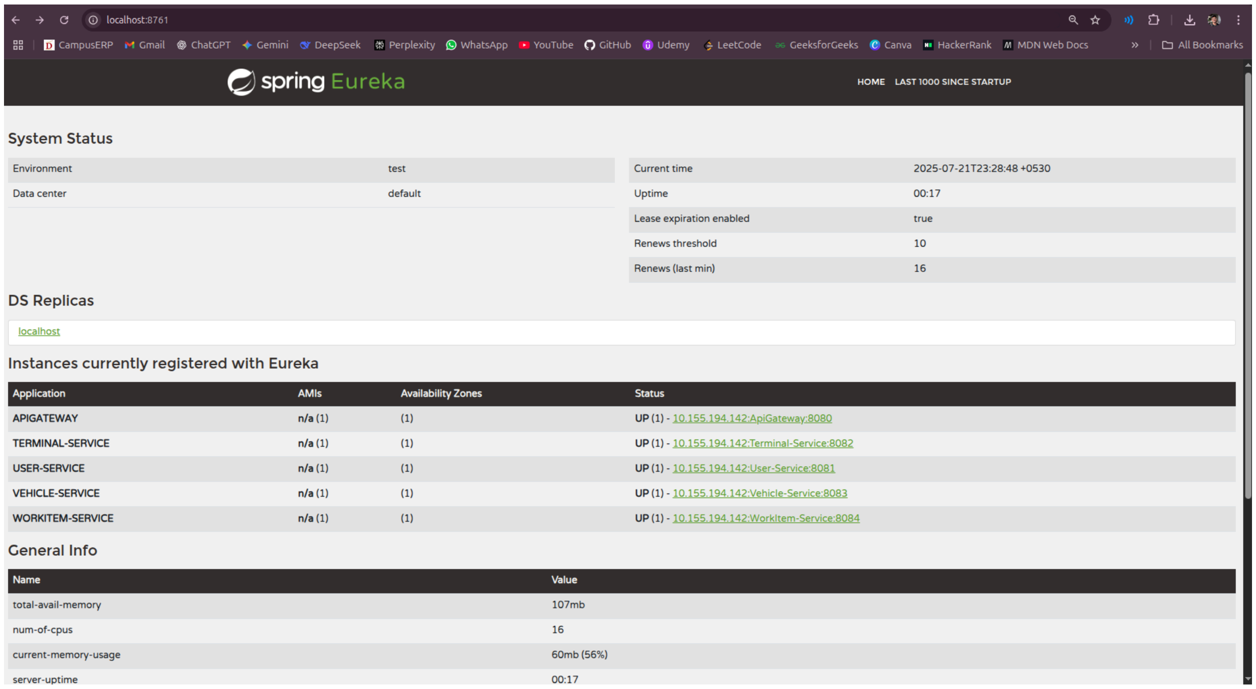
Application.properties

```
spring.application.name=ApiGateway

server.port=8080
```

```
# Eureka
eureka.client.service-url.defaultZone=http://localhost:8761/eureka/

# Gateway Routes
spring.cloud.gateway.routes[0].id=user-service
spring.cloud.gateway.routes[0].uri=lb://user-service
spring.cloud.gateway.routes[0].predicates[0]=Path=/api/users/**

spring.cloud.gateway.routes[1].id=terminal-service
spring.cloud.gateway.routes[1].uri=lb://terminal-service
spring.cloud.gateway.routes[1].predicates[0]=Path=/api/terminals/**

spring.cloud.gateway.routes[2].id=vehicle-service
spring.cloud.gateway.routes[2].uri=lb://vehicle-service
spring.cloud.gateway.routes[2].predicates[0]=Path=/api/vehicles/**

spring.cloud.gateway.routes[3].id=workitem-service
spring.cloud.gateway.routes[3].uri=lb://workitem-service
spring.cloud.gateway.routes[3].predicates[0]=Path=/api/workitems/**
```

Outputs:

Eureka Server



System Status

Environment	test
Data center	default

Current time	2025-07-21T23:28:48 +0530
Uptime	00:17
Lease expiration enabled	true
Renews threshold	10
Renews (last min)	16

DS Replicas

localhost

Instances currently registered with Eureka

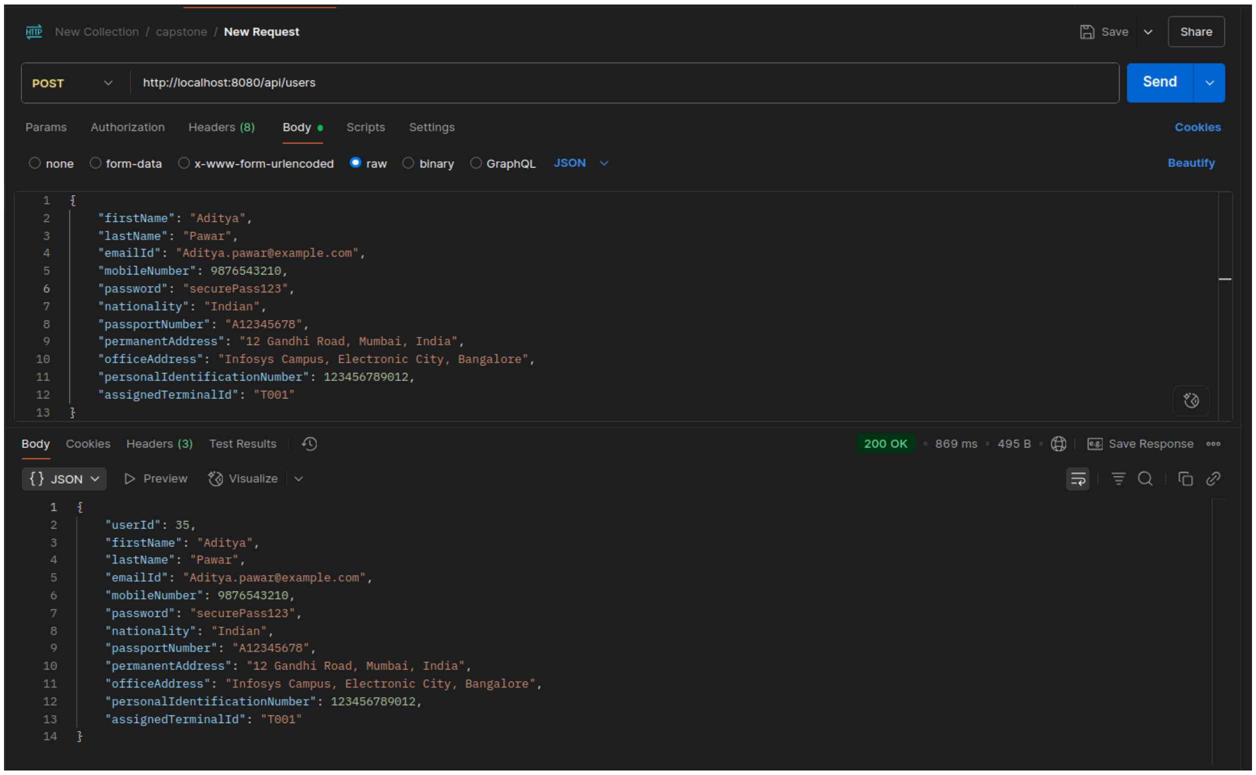
Application	AMIs	Availability Zones	Status
APIGATEWAY	n/a (1)	(1)	UP (1) - 10.155.194.142:ApiGateway:8080
TERMINAL-SERVICE	n/a (1)	(1)	UP (1) - 10.155.194.142:Terminal-Service:8082
USER-SERVICE	n/a (1)	(1)	UP (1) - 10.155.194.142:User-Service:8081
VEHICLE-SERVICE	n/a (1)	(1)	UP (1) - 10.155.194.142:Vehicle-Service:8083
WORKITEM-SERVICE	n/a (1)	(1)	UP (1) - 10.155.194.142:WorkItem-Service:8084

General Info

Name	Value
total-avail-memory	107mb
num-of-cpus	16
current-memory-usage	60mb (56%)
server-uptime	00:17

User

1. add user



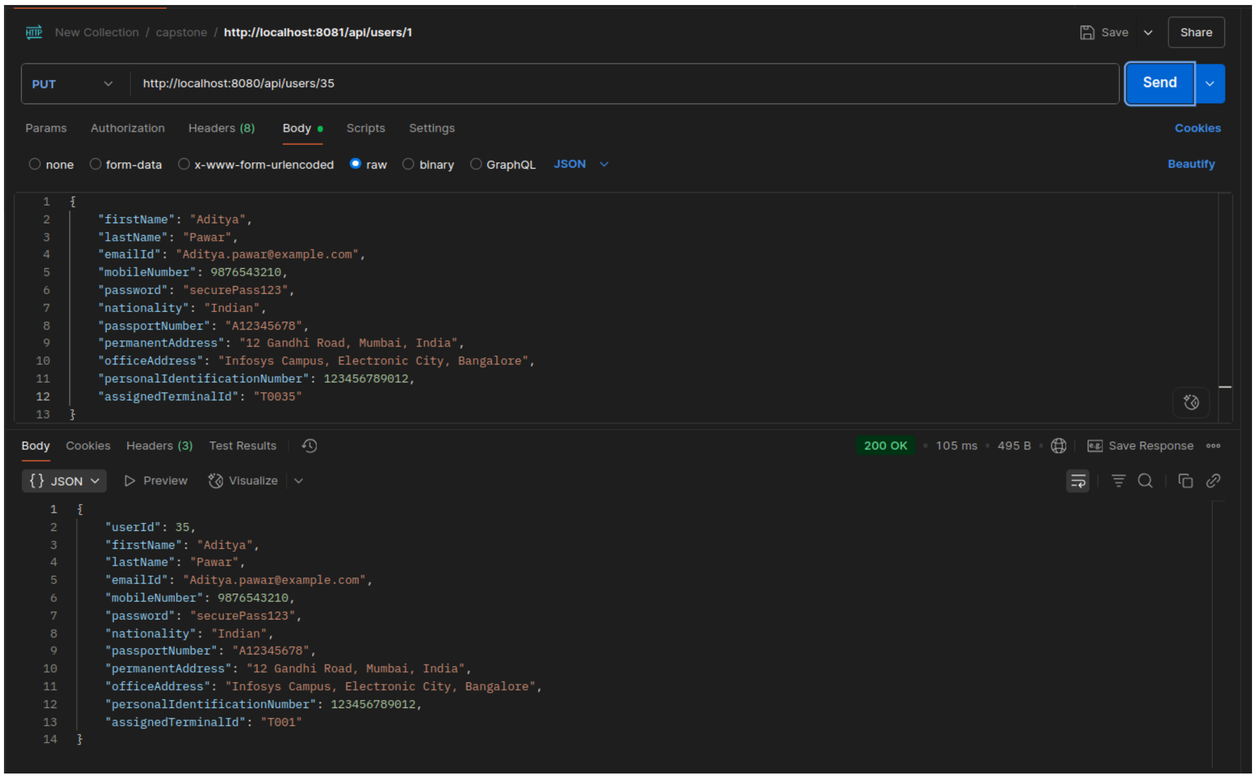
POST http://localhost:8080/api/users

```
{
    "firstName": "Aditya",
    "lastName": "Pawar",
    "emailId": "Aditya.pawar@example.com",
    "mobileNumber": 9876543210,
    "password": "securePass123",
    "nationality": "Indian",
    "passportNumber": "A12345678",
    "permanentAddress": "12 Gandhi Road, Mumbai, India",
    "officeAddress": "Infosys Campus, Electronic City, Bangalore",
    "personalIdentificationNumber": 123456789012,
    "assignedTerminalId": "T001"
}
```

Response: 200 OK, 869 ms, 495 B — JSON

```
{
    "userId": 35,
    "firstName": "Aditya",
    "lastName": "Pawar",
    "emailId": "Aditya.pawar@example.com",
    "mobileNumber": 9876543210,
    "password": "securePass123",
    "nationality": "Indian",
    "passportNumber": "A12345678",
    "permanentAddress": "12 Gandhi Road, Mumbai, India",
    "officeAddress": "Infosys Campus, Electronic City, Bangalore",
    "personalIdentificationNumber": 123456789012,
    "assignedTerminalId": "T001"
}
```

2. Update user


New Collection / capstone / http://localhost:8081/api/users/1
Save
Share
Send
Params Authorization Headers (8) Body Scripts Settings
Cookies
Beautify
Preview Visualize
Save Response

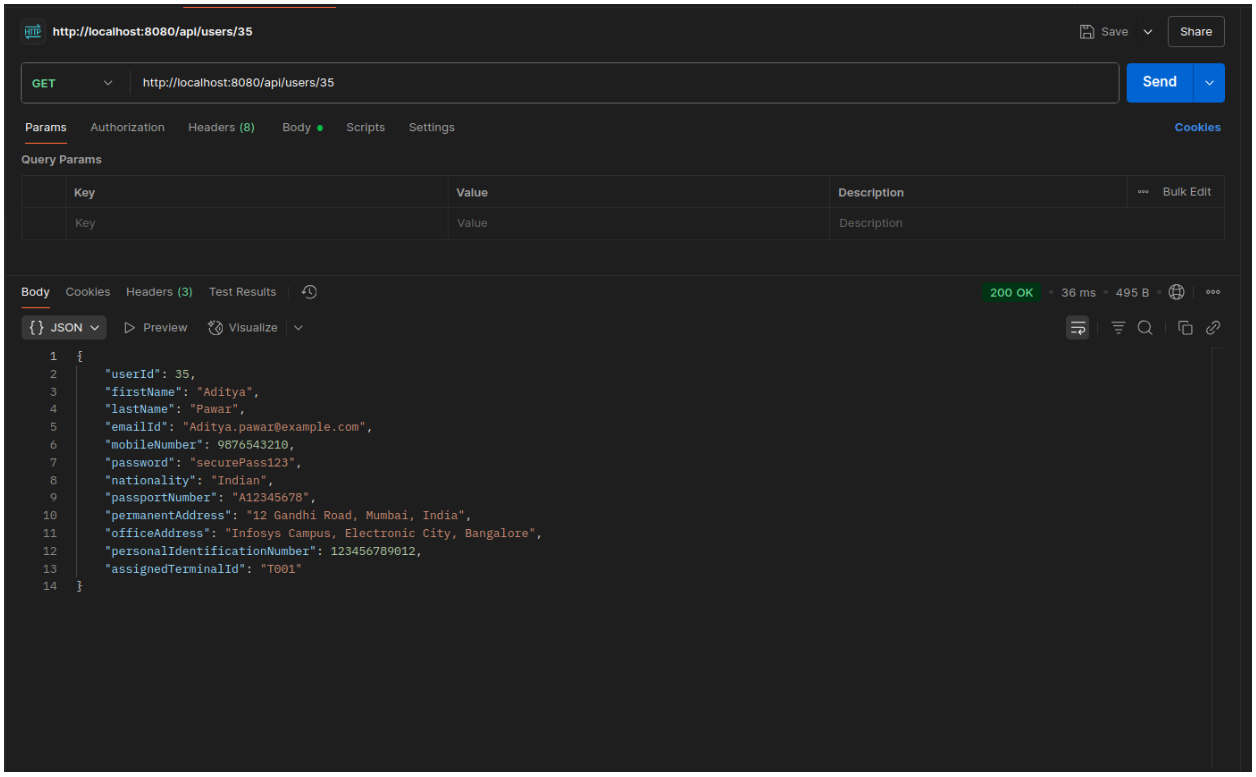
PUT http://localhost:8080/api/users/35

```
{
    "firstName": "Aditya",
    "lastName": "Pawar",
    "emailId": "Aditya.pawar@example.com",
    "mobileNumber": 9876543210,
    "password": "securePass123",
    "nationality": "Indian",
    "passportNumber": "A12345678",
    "permanentAddress": "12 Gandhi Road, Mumbai, India",
    "officeAddress": "Infosys Campus, Electronic City, Bangalore",
    "personalIdentificationNumber": 123456789012,
    "assignedTerminalId": "T0035"
}
```

Body Cookies Headers (3) Test Results — 200 OK • 105 ms • 495 B

```
{
    "userId": 35,
    "firstName": "Aditya",
    "lastName": "Pawar",
    "emailId": "Aditya.pawar@example.com",
    "mobileNumber": 9876543210,
    "password": "securePass123",
    "nationality": "Indian",
    "passportNumber": "A12345678",
    "permanentAddress": "12 Gandhi Road, Mumbai, India",
    "officeAddress": "Infosys Campus, Electronic City, Bangalore",
    "personalIdentificationNumber": 123456789012,
    "assignedTerminalId": "T001"
}
```

3. view user

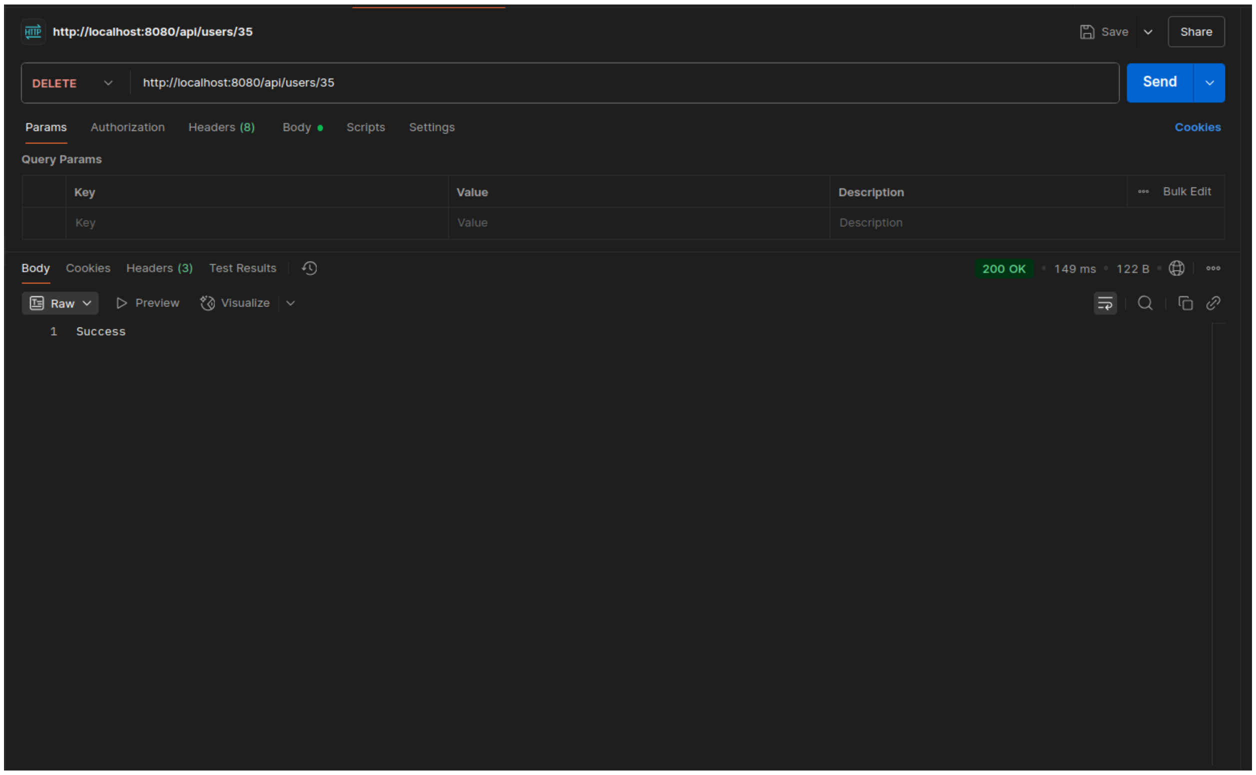

http://localhost:8080/api/users/35

GET http://localhost:8080/api/users/35

200 OK · 36 ms · 495 B

```
{
    "userId": 35,
    "firstName": "Aditya",
    "lastName": "Pawar",
    "emailId": "Aditya.pawar@example.com",
    "mobileNumber": 9876543210,
    "password": "securePass123",
    "nationality": "Indian",
    "passportNumber": "A12345678",
    "permanentAddress": "12 Gandhi Road, Mumbai, India",
    "officeAddress": "Infosys Campus, Electronic City, Bangalore",
    "personalIdentificationNumber": 123456789012,
    "assignedTerminalId": "T001"
}
```

4. Delete User

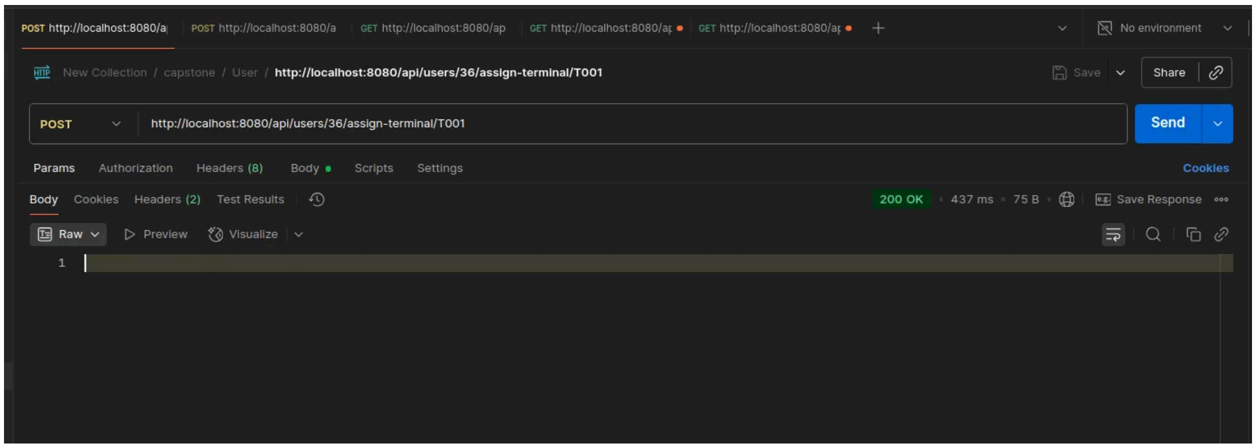

http://localhost:8080/api/users/35

DELETE http://localhost:8080/api/users/35

200 OK · 149 ms · 122 B

```
Success
```

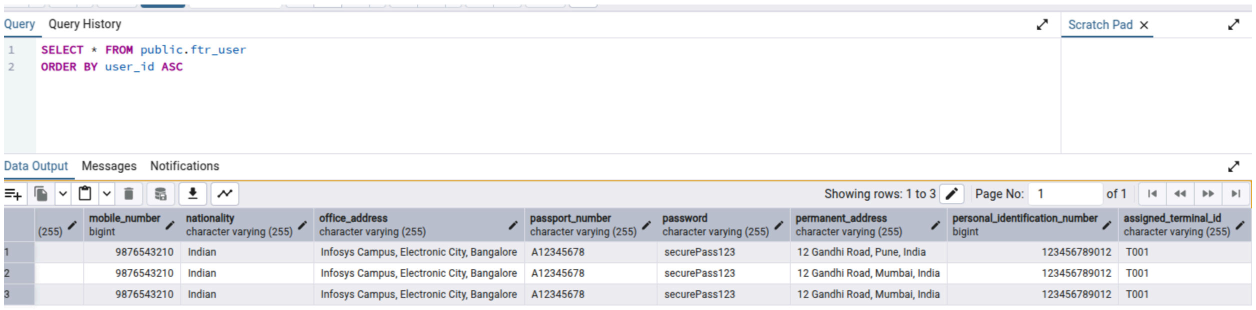
5. Assign Terminal



New Collection / capstone / User / http://localhost:8080/api/users/36/assign-terminal/T001

POST http://localhost:8080/api/users/36/assign-terminal/T001

200 OK • 437 ms • 75 B



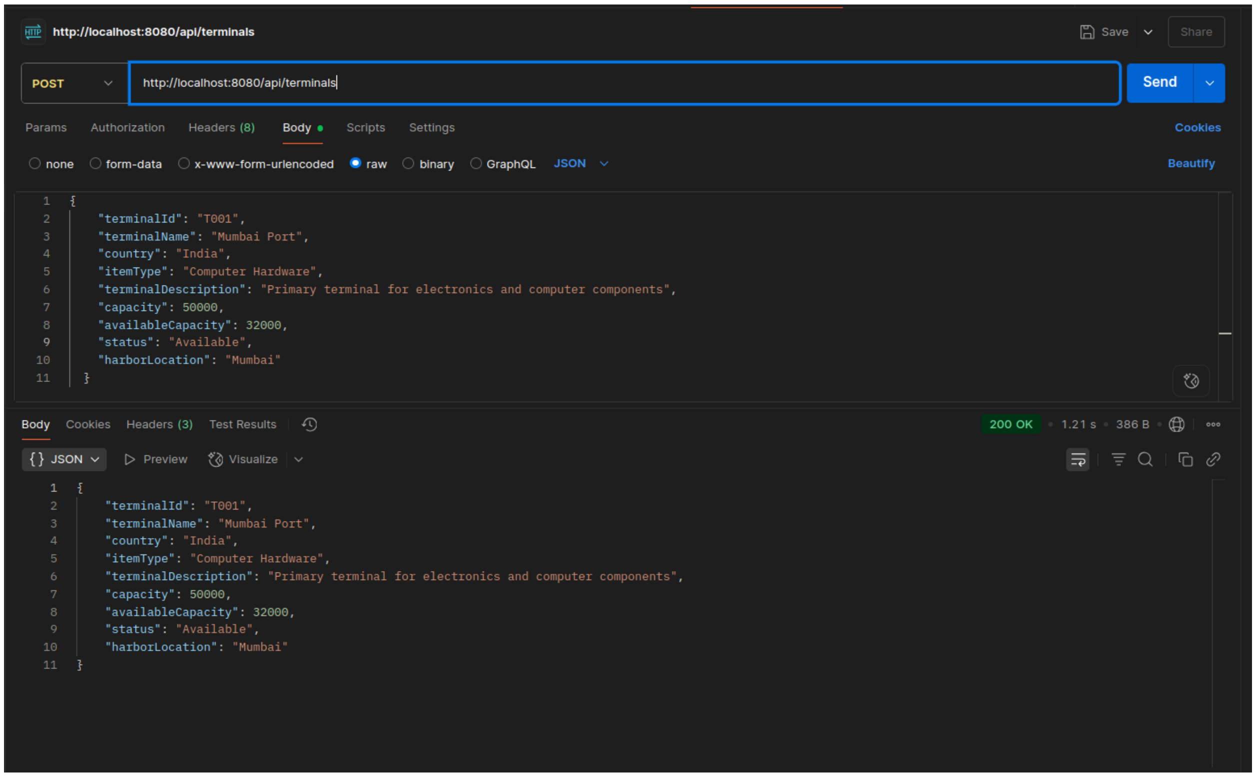
```
SELECT * FROM public.ftr_user
ORDER BY user_id ASC
```

Showing rows: 1 to 3 Page No: 1 of 1

	(255)	mobile_number bigint	nationality character varying (255)	office_address character varying (255)	passport_number character varying (255)	password character varying (255)	permanent_address character varying (255)	personal_identification_number bigint	assigned_terminal_id character varying (255)
1		9876543210	Indian	Infosys Campus, Electronic City, Bangalore	A12345678	securePass123	12 Gandhi Road, Pune, India	123456789012	T001
2		9876543210	Indian	Infosys Campus, Electronic City, Bangalore	A12345678	securePass123	12 Gandhi Road, Mumbai, India	123456789012	T001
3		9876543210	Indian	Infosys Campus, Electronic City, Bangalore	A12345678	securePass123	12 Gandhi Road, Mumbai, India	123456789012	T001

Terminals

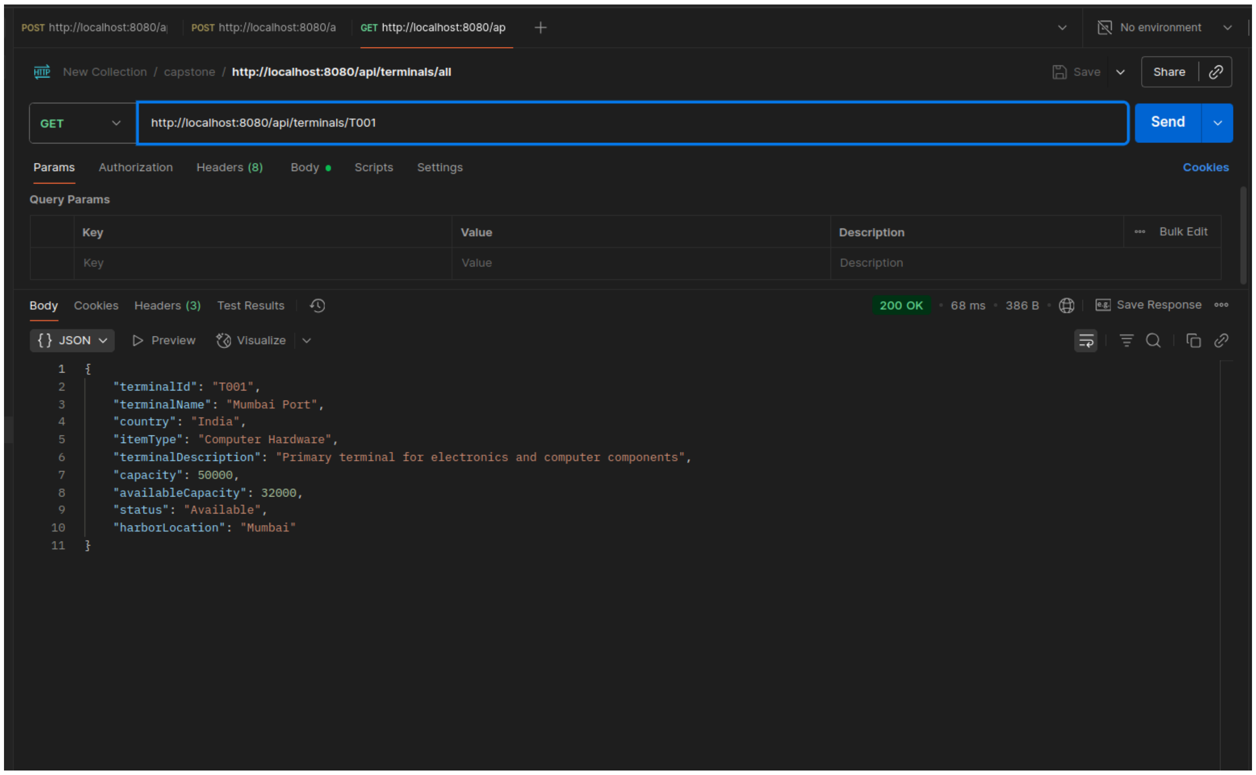
1. Add terminal



```
{
    "terminalId": "T001",
    "terminalName": "Mumbai Port",
    "country": "India",
    "itemType": "Computer Hardware",
    "terminalDescription": "Primary terminal for electronics and computer components",
    "capacity": 50000,
    "availableCapacity": 32000,
    "status": "Available",
    "harborLocation": "Mumbai"
}
```

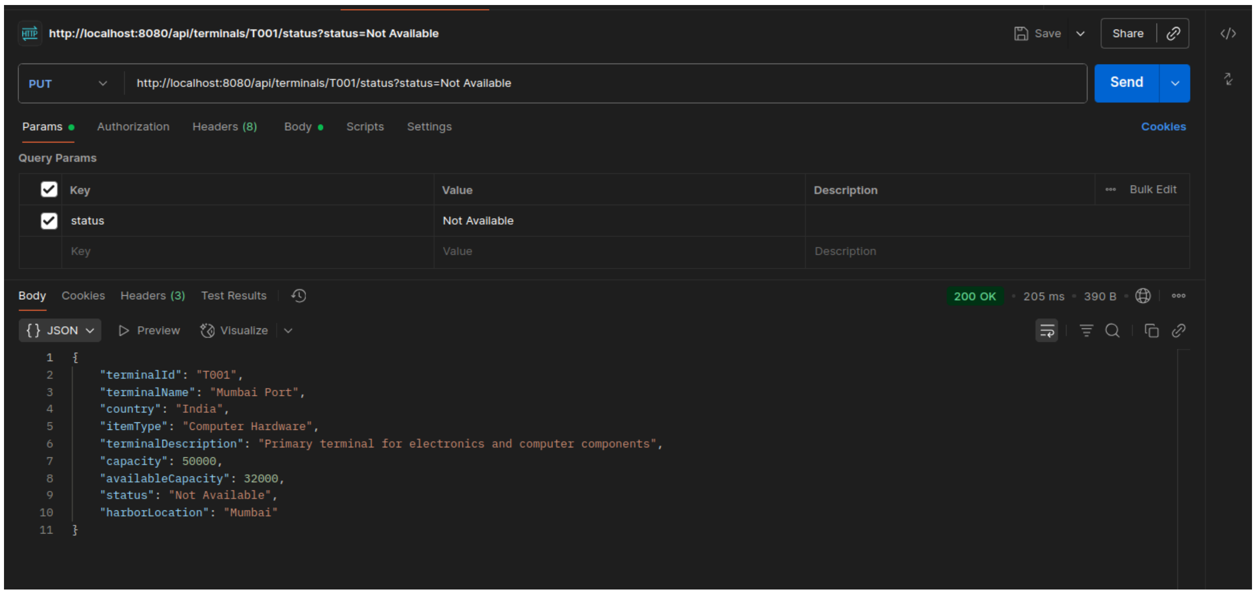
```
{
    "terminalId": "T001",
    "terminalName": "Mumbai Port",
    "country": "India",
    "itemType": "Computer Hardware",
    "terminalDescription": "Primary terminal for electronics and computer components",
    "capacity": 50000,
    "availableCapacity": 32000,
    "status": "Available",
    "harborLocation": "Mumbai"
}
```

2. Fetch terminal by Id



```
{
    "terminalId": "T001",
    "terminalName": "Mumbai Port",
    "country": "India",
    "itemType": "Computer Hardware",
    "terminalDescription": "Primary terminal for electronics and computer components",
    "capacity": 50000,
    "availableCapacity": 32000,
    "status": "Available",
    "harborLocation": "Mumbai"
}
```

3. Update Terminal Status


http://localhost:8080/api/terminals/T001/status?status=Not Available

PUT http://localhost:8080/api/terminals/T001/status?status=Not Available

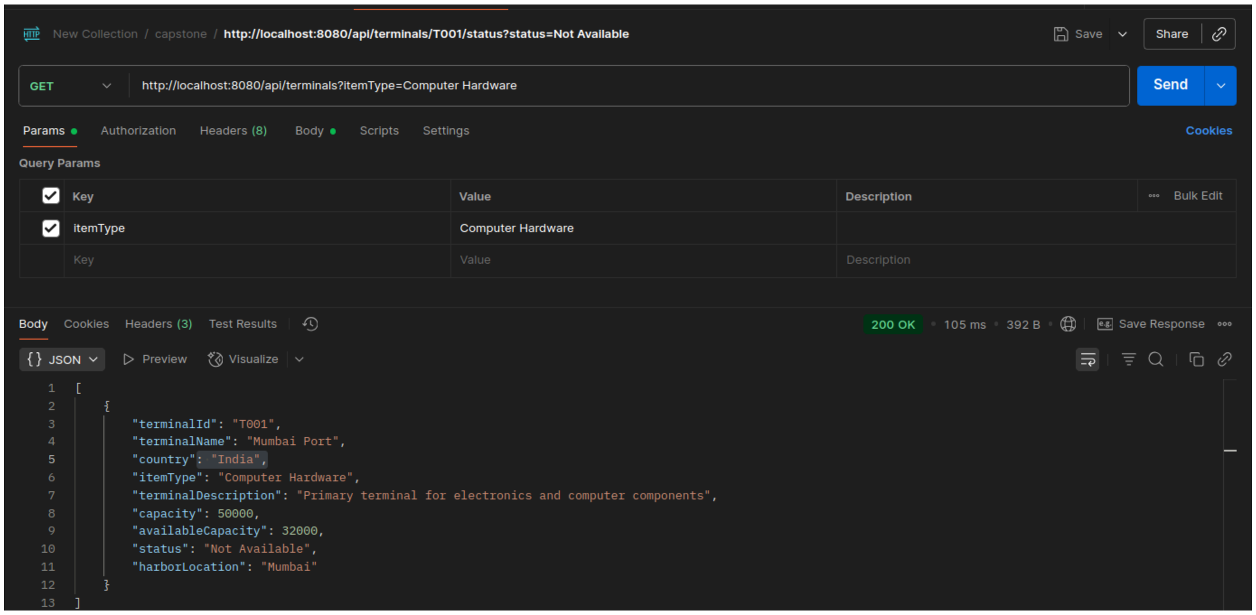
Query Params

Key	Value	Description
status	Not Available	

200 OK · 205 ms · 390 B

```
{
    "terminalId": "T001",
    "terminalName": "Mumbai Port",
    "country": "India",
    "itemType": "Computer Hardware",
    "terminalDescription": "Primary terminal for electronics and computer components",
    "capacity": 50000,
    "availableCapacity": 32000,
    "status": "Not Available",
    "harborLocation": "Mumbai"
}
```

4. Fetch terminal by Item Type


New Collection / capstone / http://localhost:8080/api/terminals/T001/status?status=Not Available

GET http://localhost:8080/api/terminals?itemType=Computer Hardware

Query Params

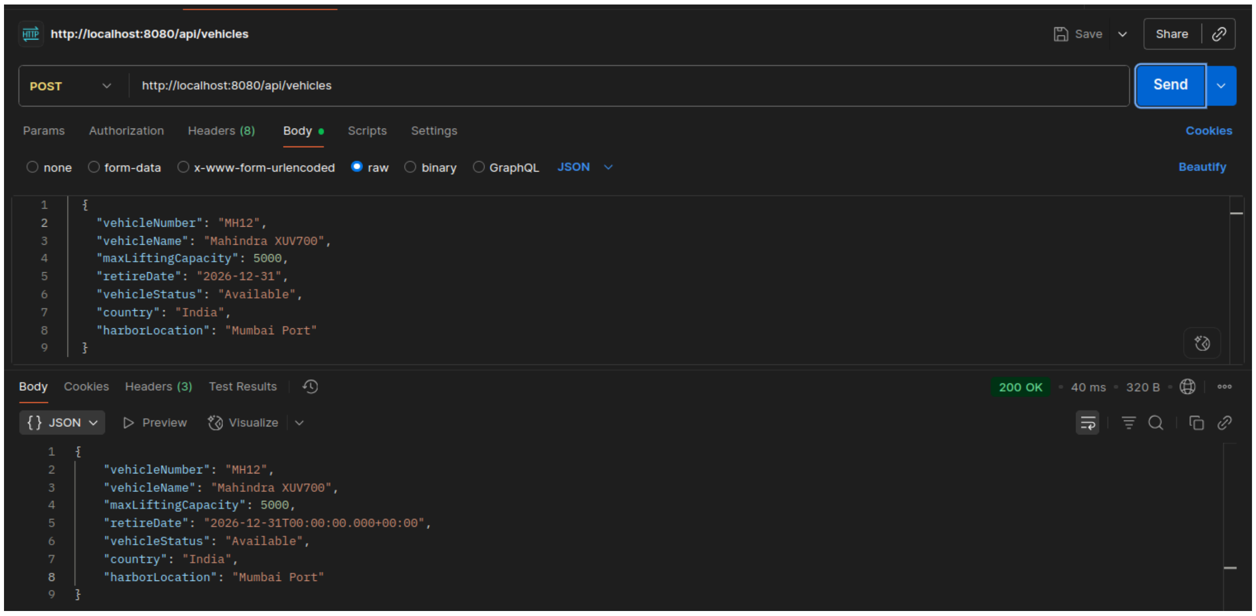
Key	Value	Description
itemType	Computer Hardware	

200 OK · 105 ms · 392 B

```
[
    {
        "terminalId": "T001",
        "terminalName": "Mumbai Port",
        "country": "India",
        "itemType": "Computer Hardware",
        "terminalDescription": "Primary terminal for electronics and computer components",
        "capacity": 50000,
        "availableCapacity": 32000,
        "status": "Not Available",
        "harborLocation": "Mumbai"
    }
]
```

Vehicle

1. Insert vehicle


http://localhost:8080/api/vehicles

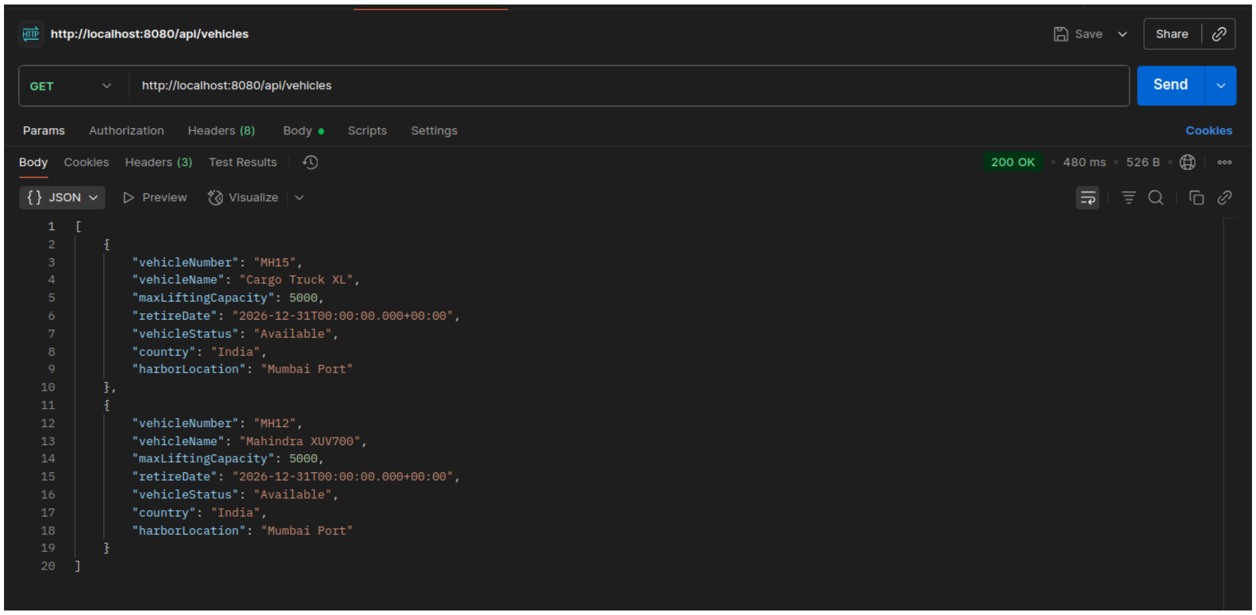
POST http://localhost:8080/api/vehicles

```
{
  "vehicleNumber": "MH12",
  "vehicleName": "Mahindra XUV700",
  "maxLiftingCapacity": 5000,
  "retireDate": "2026-12-31",
  "vehicleStatus": "Available",
  "country": "India",
  "harborLocation": "Mumbai Port"
}
```

200 OK · 40 ms · 320 B

```
{
  "vehicleNumber": "MH12",
  "vehicleName": "Mahindra XUV700",
  "maxLiftingCapacity": 5000,
  "retireDate": "2026-12-31T00:00:00.000+00:00",
  "vehicleStatus": "Available",
  "country": "India",
  "harborLocation": "Mumbai Port"
}
```

2. Get All vehicles Details

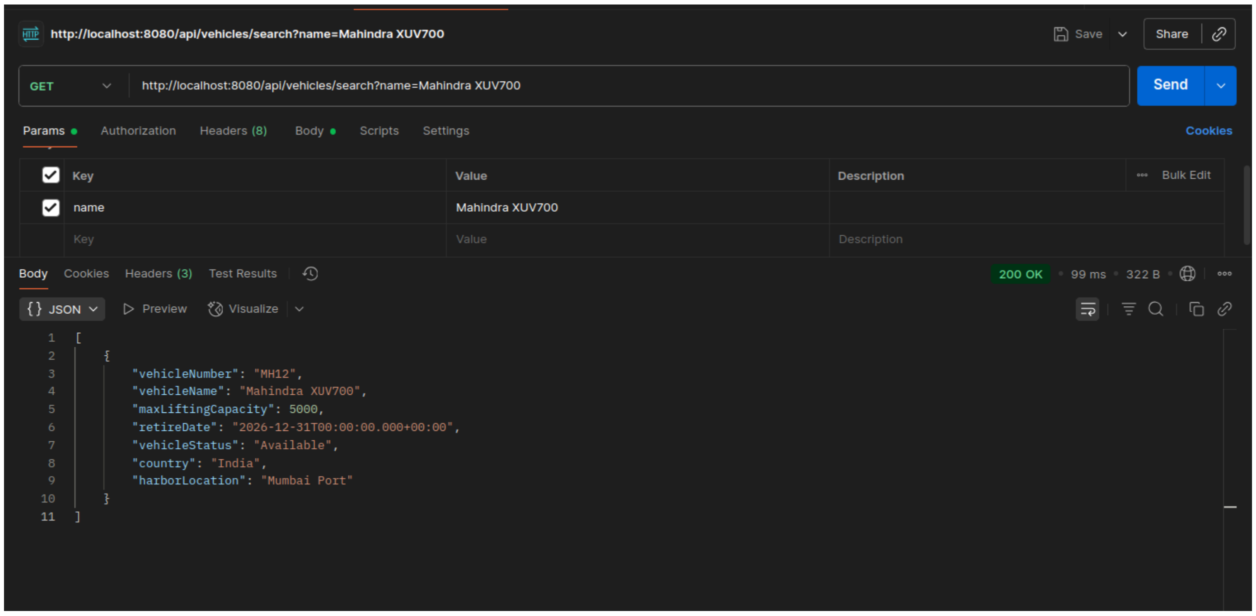

http://localhost:8080/api/vehicles

GET http://localhost:8080/api/vehicles

200 OK · 480 ms · 526 B

```
[
    {
        "vehicleNumber": "MH15",
        "vehicleName": "Cargo Truck XL",
        "maxLiftingCapacity": 5000,
        "retireDate": "2026-12-31T00:00:00.000+00:00",
        "vehicleStatus": "Available",
        "country": "India",
        "harborLocation": "Mumbai Port"
    },
    {
        "vehicleNumber": "MH12",
        "vehicleName": "Mahindra XUV700",
        "maxLiftingCapacity": 5000,
        "retireDate": "2026-12-31T00:00:00.000+00:00",
        "vehicleStatus": "Available",
        "country": "India",
        "harborLocation": "Mumbai Port"
    }
]
```

3. Get Vehicles by Name


http://localhost:8080/api/vehicles/search?name=Mahindra XUV700

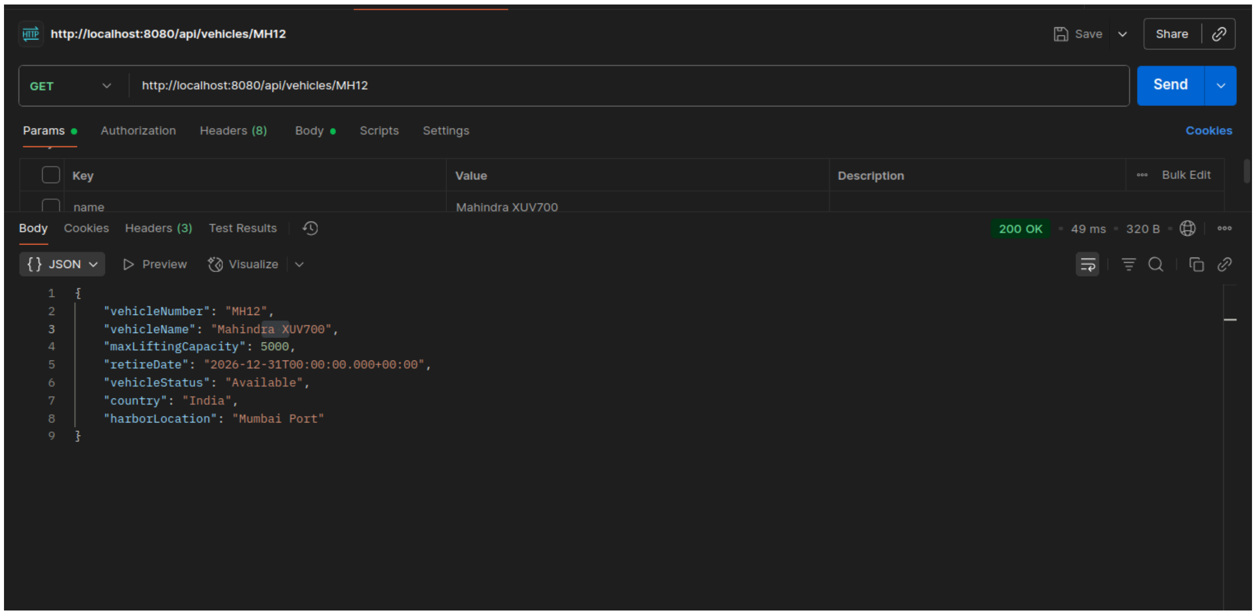
GET http://localhost:8080/api/vehicles/search?name=Mahindra XUV700

Key	Value	Description
name	Mahindra XUV700	

200 OK · 99 ms · 322 B

```
[
    {
        "vehicleNumber": "MH12",
        "vehicleName": "Mahindra XUV700",
        "maxLiftingCapacity": 5000,
        "retireDate": "2026-12-31T00:00:00.000+00:00",
        "vehicleStatus": "Available",
        "country": "India",
        "harborLocation": "Mumbai Port"
    }
]
```

4. Get Vehicles By Number


http://localhost:8080/api/vehicles/MH12

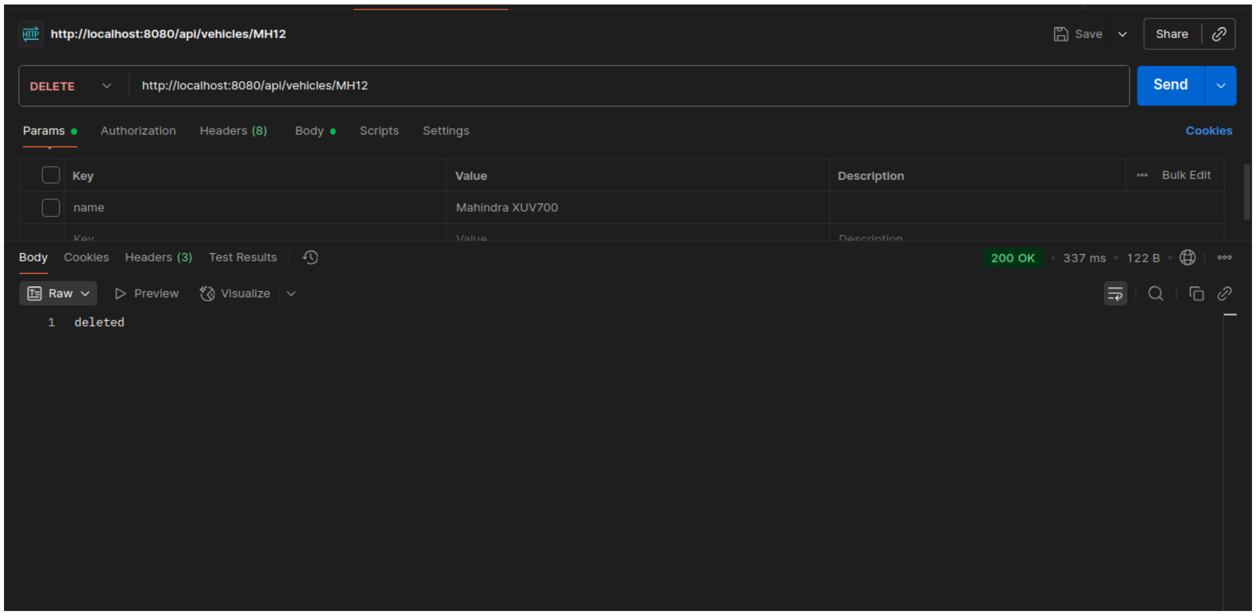
GET http://localhost:8080/api/vehicles/MH12

Key	Value	Description
name	Mahindra XUV700	

200 OK · 49 ms · 320 B

```
{
    "vehicleNumber": "MH12",
    "vehicleName": "Mahindra XUV700",
    "maxLiftingCapacity": 5000,
    "retireDate": "2026-12-31T00:00:00.000+00:00",
    "vehicleStatus": "Available",
    "country": "India",
    "harborLocation": "Mumbai Port"
}
```

5. delete Vehicle



DELETE http://localhost:8080/api/vehicles/MH12

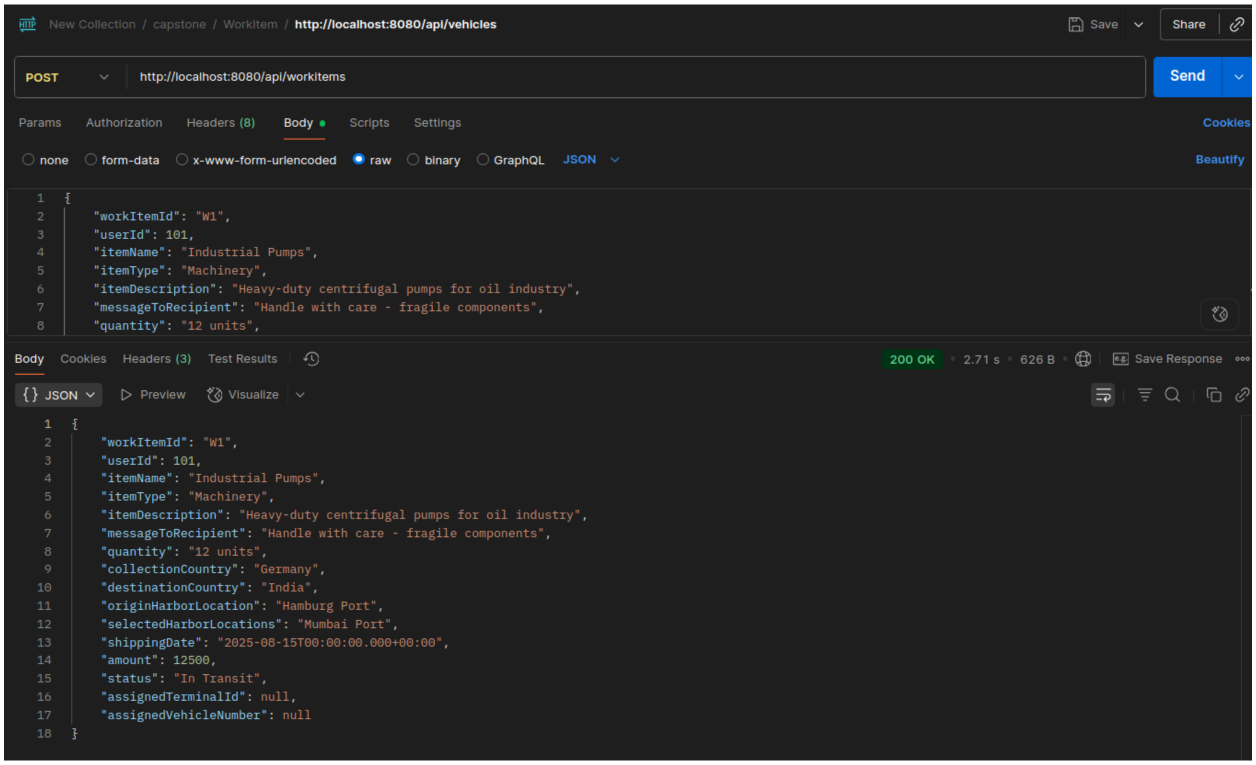
Key	Value	Description
name	Mahindra XUV700	
Key	Value	Description

200 OK · 337 ms · 122 B

```
deleted
```

WorkItem

1. \create workItem



POST http://localhost:8080/api/workitems

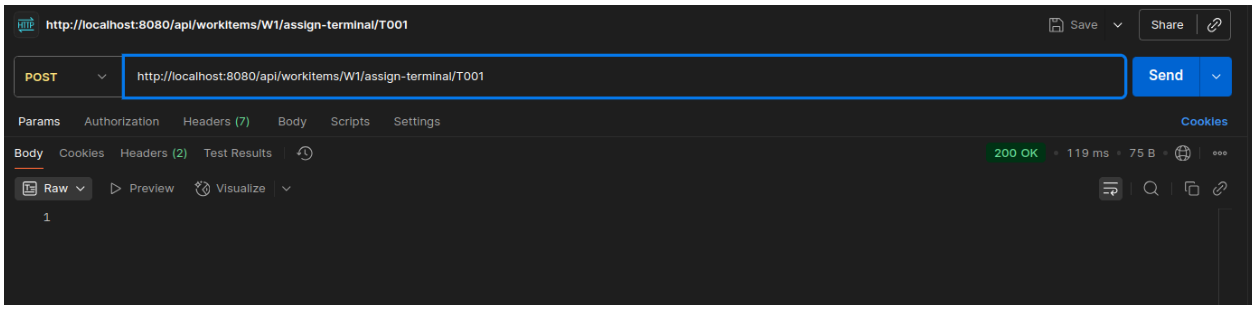
none form-data x-www-form-urlencoded [✓ raw] binary GraphQL JSON

```
{
    "workItemId": "W1",
    "userId": 101,
    "itemName": "Industrial Pumps",
    "itemType": "Machinery",
    "itemDescription": "Heavy-duty centrifugal pumps for oil industry",
    "messageToRecipient": "Handle with care - fragile components",
    "quantity": "12 units",
```

200 OK · 2.71 s · 626 B

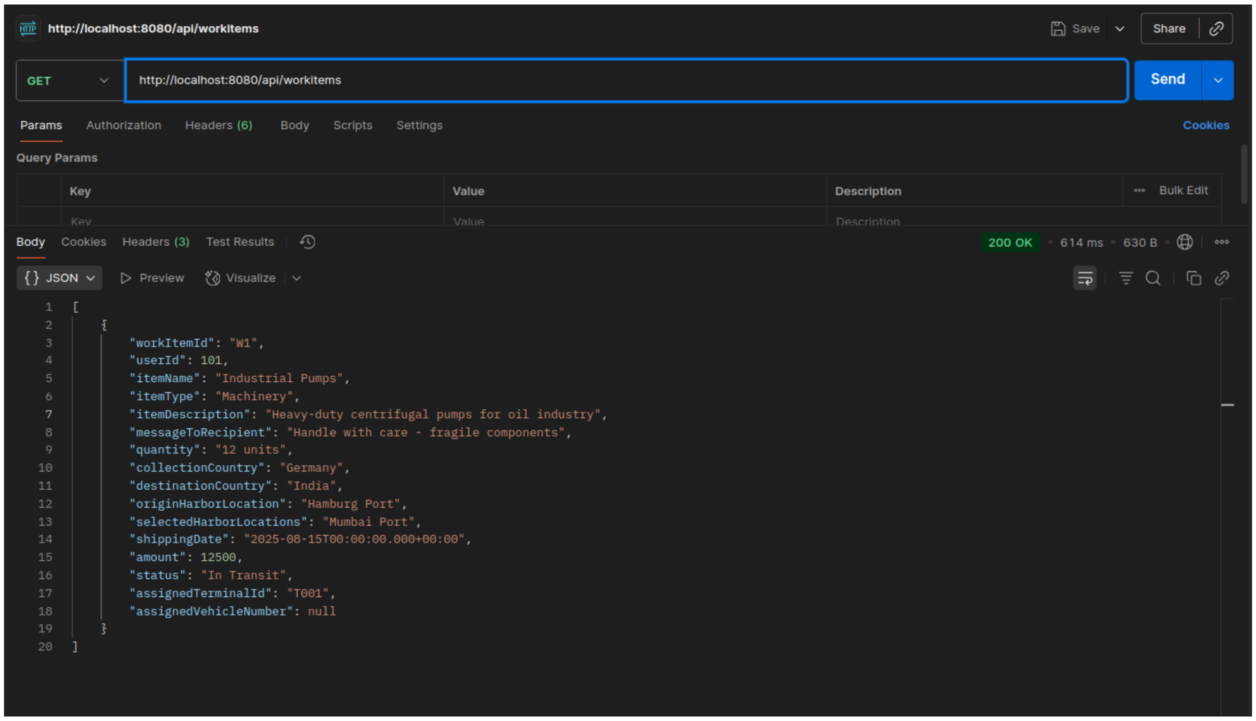
```
{
    "workItemId": "W1",
    "userId": 101,
    "itemName": "Industrial Pumps",
    "itemType": "Machinery",
    "itemDescription": "Heavy-duty centrifugal pumps for oil industry",
    "messageToRecipient": "Handle with care - fragile components",
    "quantity": "12 units",
    "collectionCountry": "Germany",
    "destinationCountry": "India",
    "originHarborLocation": "Hamburg Port",
    "selectedHarborLocations": "Mumbai Port",
    "shippingDate": "2025-08-15T00:00:00.000+00:00",
    "amount": 12500,
    "status": "In Transit",
    "assignedTerminalId": null,
    "assignedVehicleNumber": null
}
```

2. Assign Terminal


POST http://localhost:8080/api/workitems/W1/assign-terminal/T001

200 OK · 119 ms · 75 B

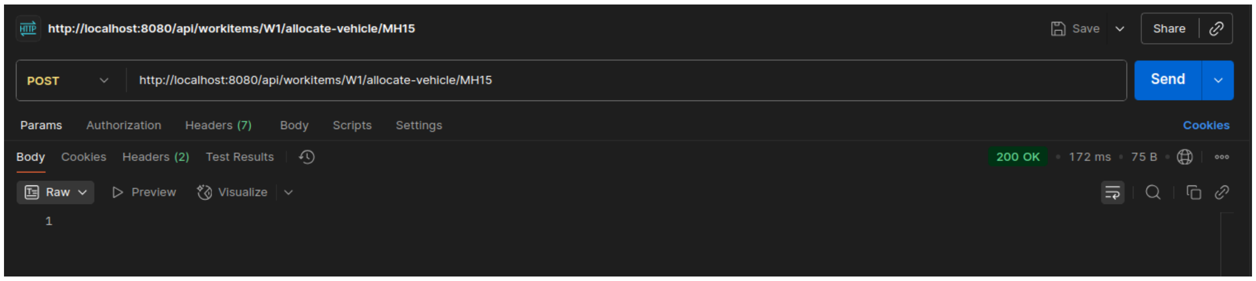
```
1
```


GET http://localhost:8080/api/workitems

200 OK · 614 ms · 630 B

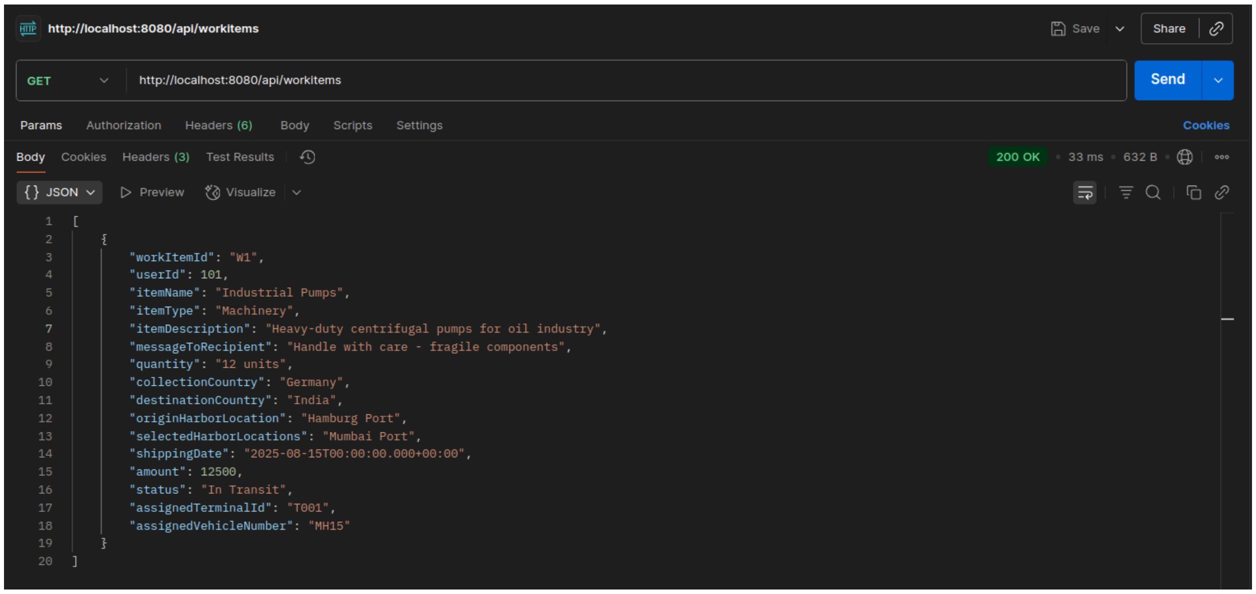
```
[
    {
        "workItemId": "W1",
        "userId": 101,
        "itemName": "Industrial Pumps",
        "itemType": "Machinery",
        "itemDescription": "Heavy-duty centrifugal pumps for oil industry",
        "messageToRecipient": "Handle with care - fragile components",
        "quantity": "12 units",
        "collectionCountry": "Germany",
        "destinationCountry": "India",
        "originHarborLocation": "Hamburg Port",
        "selectedHarborLocations": "Mumbai Port",
        "shippingDate": "2025-08-15T00:00:00.000+00:00",
        "amount": 12500,
        "status": "In Transit",
        "assignedTerminalId": "T001",
        "assignedVehicleNumber": null
    }
]
```

3. Allocate Vehicle


POST http://localhost:8080/api/workitems/W1/allocate-vehicle/MH15

200 OK · 172 ms · 75 B

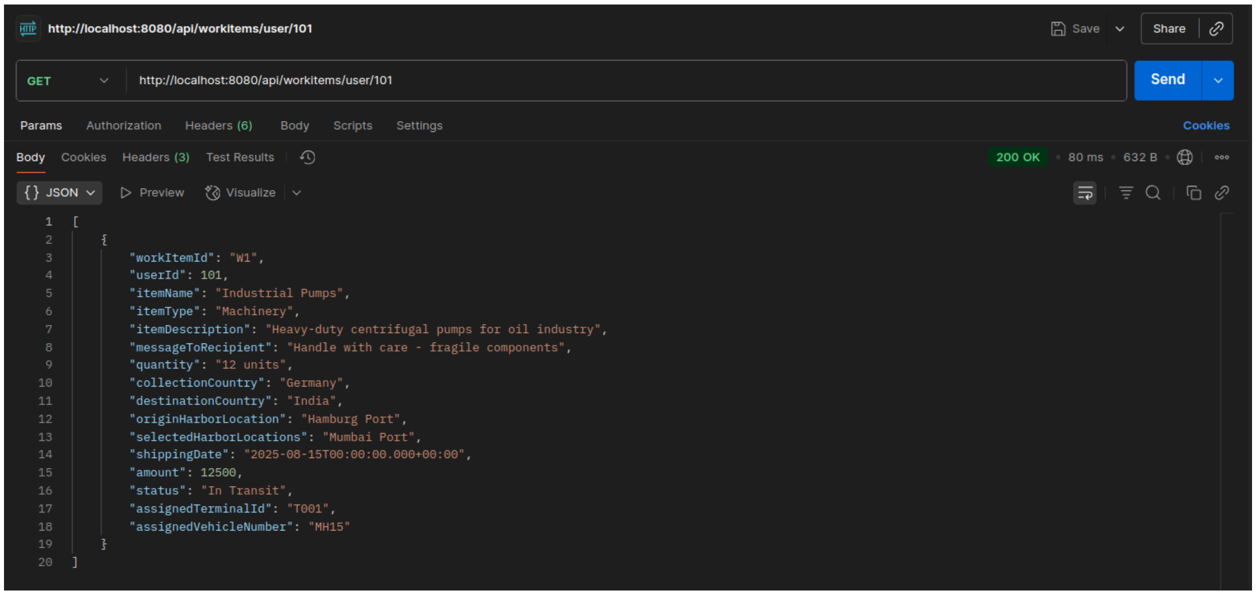
```
1
```



http://localhost:8080/api/workitems

```
[
    {
        "workItemId": "W1",
        "userId": 101,
        "itemName": "Industrial Pumps",
        "itemType": "Machinery",
        "itemDescription": "Heavy-duty centrifugal pumps for oil industry",
        "messageToRecipient": "Handle with care - fragile components",
        "quantity": "12 units",
        "collectionCountry": "Germany",
        "destinationCountry": "India",
        "originHarborLocation": "Hamburg Port",
        "selectedHarborLocations": "Mumbai Port",
        "shippingDate": "2025-08-15T00:00:00.000+00:00",
        "amount": 12500,
        "status": "In Transit",
        "assignedTerminalId": "T001",
        "assignedVehicleNumber": "MH15"
    }
]
```

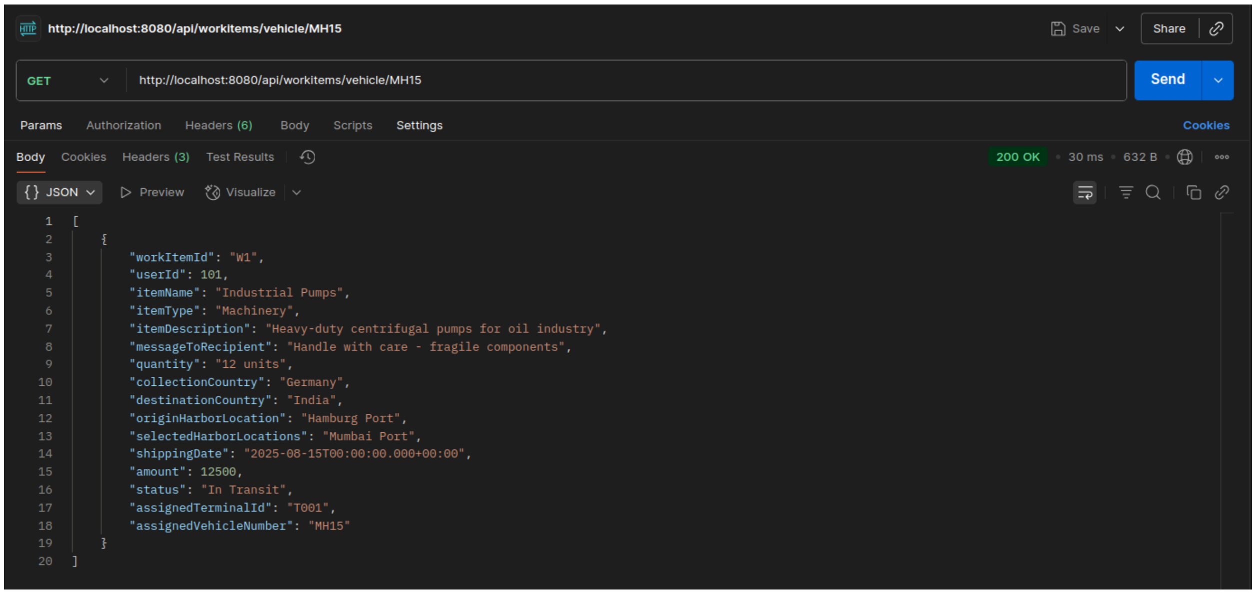
4. Fetch workitem by user



http://localhost:8080/api/workitems/user/101

```
[
    {
        "workItemId": "W1",
        "userId": 101,
        "itemName": "Industrial Pumps",
        "itemType": "Machinery",
        "itemDescription": "Heavy-duty centrifugal pumps for oil industry",
        "messageToRecipient": "Handle with care - fragile components",
        "quantity": "12 units",
        "collectionCountry": "Germany",
        "destinationCountry": "India",
        "originHarborLocation": "Hamburg Port",
        "selectedHarborLocations": "Mumbai Port",
        "shippingDate": "2025-08-15T00:00:00.000+00:00",
        "amount": 12500,
        "status": "In Transit",
        "assignedTerminalId": "T001",
        "assignedVehicleNumber": "MH15"
    }
]
```

5. Fetch workitem by vehicle

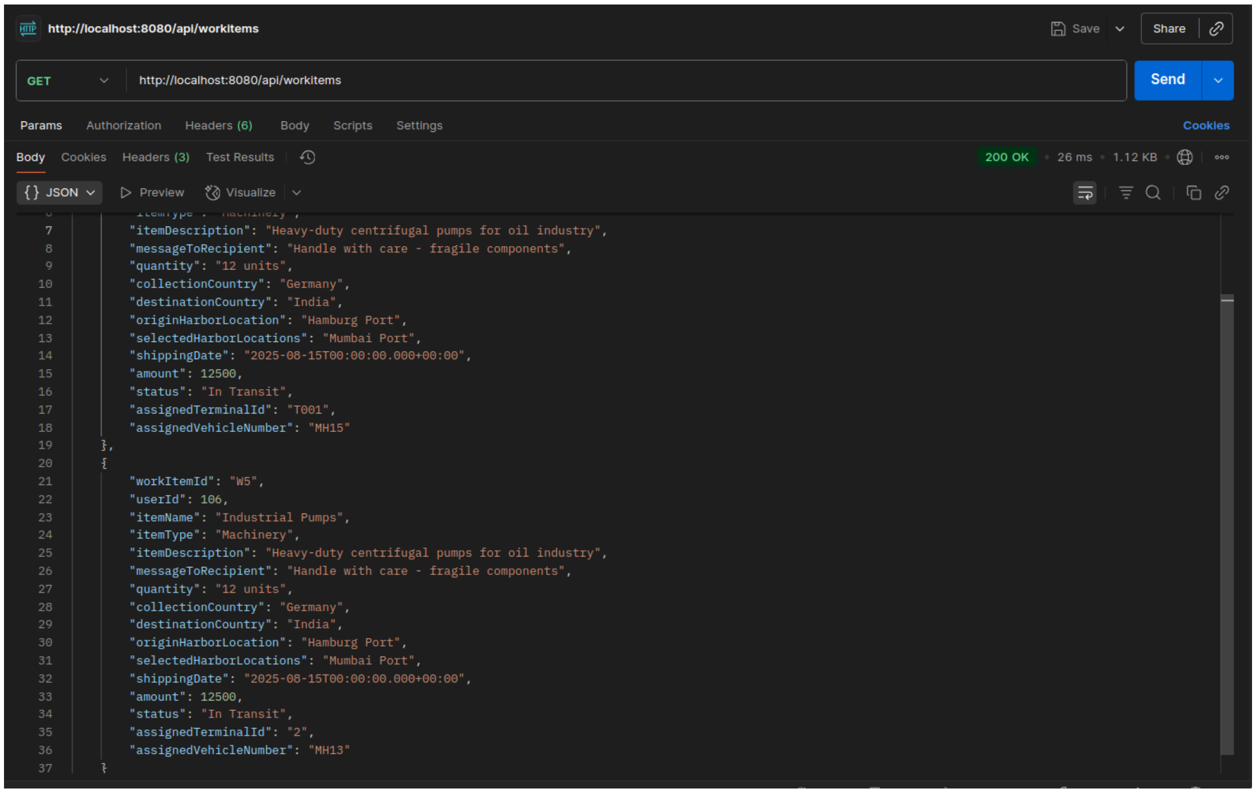


GET http://localhost:8080/api/workitems/vehicle/MH15

200 OK · 30 ms · 632 B

```json
[
    {
        "workItemId": "W1",
        "userId": 101,
        "itemName": "Industrial Pumps",
        "itemType": "Machinery",
        "itemDescription": "Heavy-duty centrifugal pumps for oil industry",
        "messageToRecipient": "Handle with care - fragile components",
        "quantity": "12 units",
        "collectionCountry": "Germany",
        "destinationCountry": "India",
        "originHarborLocation": "Hamburg Port",
        "selectedHarborLocations": "Mumbai Port",
        "shippingDate": "2025-08-15T00:00:00.000+00:00",
        "amount": 12500,
        "status": "In Transit",
        "assignedTerminalId": "T001",
        "assignedVehicleNumber": "MH15"
    }
]
```

6. Get All Details



GET http://localhost:8080/api/workitems

200 OK · 26 ms · 1.12 KB

```json
        "itemDescription": "Heavy-duty centrifugal pumps for oil industry",
        "messageToRecipient": "Handle with care - fragile components",
        "quantity": "12 units",
        "collectionCountry": "Germany",
        "destinationCountry": "India",
        "originHarborLocation": "Hamburg Port",
        "selectedHarborLocations": "Mumbai Port",
        "shippingDate": "2025-08-15T00:00:00.000+00:00",
        "amount": 12500,
        "status": "In Transit",
        "assignedTerminalId": "T001",
        "assignedVehicleNumber": "MH15"
    },
    {
        "workItemId": "W5",
        "userId": 106,
        "itemName": "Industrial Pumps",
        "itemType": "Machinery",
        "itemDescription": "Heavy-duty centrifugal pumps for oil industry",
        "messageToRecipient": "Handle with care - fragile components",
        "quantity": "12 units",
        "collectionCountry": "Germany",
        "destinationCountry": "India",
        "originHarborLocation": "Hamburg Port",
        "selectedHarborLocations": "Mumbai Port",
        "shippingDate": "2025-08-15T00:00:00.000+00:00",
        "amount": 12500,
        "status": "In Transit",
        "assignedTerminalId": "2",
        "assignedVehicleNumber": "MH13"
    }
```

Thank You !