МИНИСТЕРСТВО ТРАНСПОРТА РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА (МИИТ)»

Кафедра ЦТУТП

**Отчёт**

По лабораторной работе №6  
по дисциплине «Корпоративные информационные системы»

Тема: «Работа с библиотекой MyBatis»

Вариант №19

Выполнил: Любченкова О. А.

Группа: УИС-411

Преподаватель: доц. Кафедры ЦТУТП

Козьяков П. О.

­

Москва 2024 г.

**СОДЕРЖАНИЕ**

[ПОСТАНОВКА ЗАДАЧИ 3](#_Toc180325218)

[ХОД ВЫПОЛНЕНИЯ РАБОТЫ 4](#_Toc180325219)

[РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ 5](#_Toc180325220)

[КОД ПРОГРАММЫ 6](#_Toc180325221)

[ВЫВОД 9](#_Toc180325222)

# **ПОСТАНОВКА ЗАДАЧИ**

Описать класс ЖелезнаяДорога. В БД Postgres создать таблицу «железные дороги». С помощью MyBatis реализовать операции Добавления, Удаления, Изменения и Выборки данных из таблицы «железные дороги». Продемонстрировать операции работы с БД.

# **ХОД ВЫПОЛНЕНИЯ РАБОТЫ**

Для реализации сервиса управления данными о железнодорожных путях была разработана многослойная архитектура приложения на основе Spring Boot и MyBatis. В слое модели создан класс Railway, содержащий информацию о железной дороге, включая её идентификатор, название, длину в километрах, страну и статус эксплуатации. Для работы с базой данных определён интерфейс RailwayMapper, где прописаны операции вставки, обновления, выборки и удаления записей с помощью аннотаций MyBatis.

Реализация запросов для MyBatis дополнена XML-файлом маппинга, где использованы resultMap и SQL-запросы. На уровне сервиса RailwayService инкапсулирована логика обращения к mapper, предоставляя интерфейс для контроллера. Контроллер RailwayController обеспечивает REST API, позволяющий добавлять, обновлять, получать и удалять записи о железных дорогах через HTTP-запросы.

Приложение настроено на подключение к базе данных PostgreSQL с использованием конфигурационных параметров в файле application.properties. Включено детализированное логирование запросов MyBatis и взаимодействия с базой данных.

# **РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ**

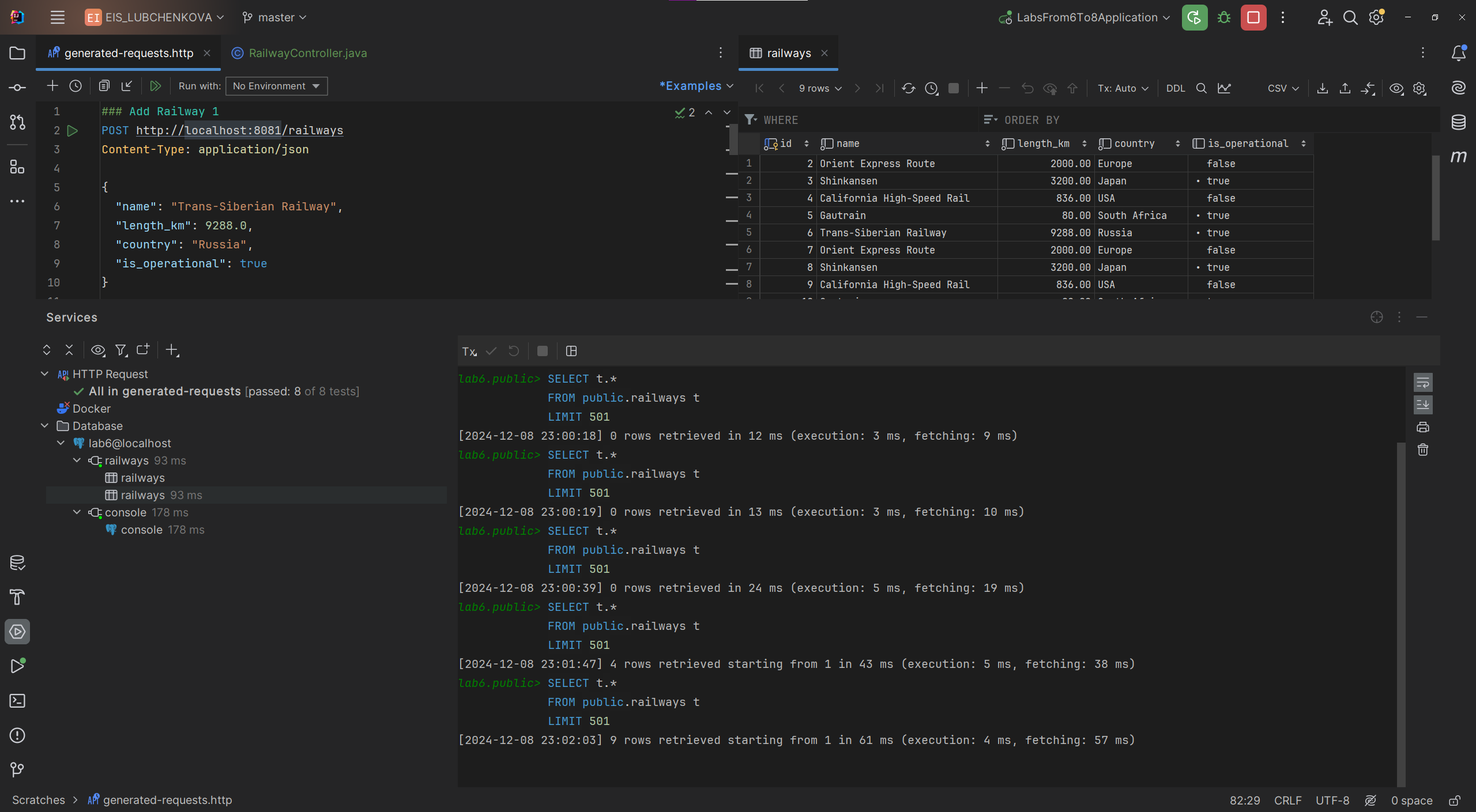


Рисунок 1 – Результат работы программы

# **КОД ПРОГРАММЫ**

package com.labsfrom6to8.Model;  
  
import com.fasterxml.jackson.annotation.JsonProperty;  
import lombok.AllArgsConstructor;  
import lombok.Data;  
import lombok.NoArgsConstructor;  
  
@Data  
@NoArgsConstructor  
@AllArgsConstructor  
public class Railway {  
  
 private Long id;  
  
 @JsonProperty("name")  
 private String name;  
  
 @JsonProperty("length\_km")  
 private Double lengthKm;  
  
 @JsonProperty("country")  
 private String country;  
  
 @JsonProperty("is\_operational")  
 private Boolean isOperational;  
}

package com.labsfrom6to8.Mapper;  
  
import com.labsfrom6to8.Model.Railway;  
import org.apache.ibatis.annotations.\*;  
  
import java.util.List;  
  
@Mapper  
public interface RailwayMapper {  
  
 @Insert("INSERT INTO railways (name, length\_km, country, is\_operational) VALUES (#{name}, #{lengthKm}, #{country}, #{isOperational})")  
 void insertRailway(Railway railway);  
  
 @Update("UPDATE railways SET name = #{name}, length\_km = #{lengthKm}, country = #{country}, is\_operational = #{isOperational} WHERE id = #{id}")  
 void updateRailway(Railway railway);  
  
 @Select("SELECT \* FROM railways")  
 @Results({  
 @Result(property = "lengthKm", column = "length\_km"),  
 @Result(property = "isOperational", column = "is\_operational")  
 })  
 List<Railway> selectAllRailways();  
  
 @Delete("DELETE FROM railways WHERE id = #{id}")  
 void deleteRailway(Long id);  
}

<?xml version="1.0" encoding="UTF-8" ?>  
<!DOCTYPE mapper  
 PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"  
 "http://mybatis.org/dtd/mybatis-3-mapper.dtd">  
  
<mapper namespace="com.example.Mapper.RailwayMapper">  
  
 <insert id="insertRailway" parameterType="com.example.Model.Railway">  
 INSERT INTO railways (name, length\_km, country, is\_operational)  
 VALUES (#{name}, #{lengthKm}, #{country}, #{isOperational})  
 </insert>  
  
 <update id="updateRailway" parameterType="com.example.Model.Railway">  
 UPDATE railways  
 SET name = #{name},  
 length\_km = #{lengthKm},  
 country = #{country},  
 is\_operational = #{isOperational}  
 WHERE id = #{id}  
 </update>  
  
 <resultMap id="RailwayResultMap" type="com.example.Model.Railway">  
 <result property="id" column="id"/>  
 <result property="name" column="name"/>  
 <result property="lengthKm" column="length\_km"/>  
 <result property="country" column="country"/>  
 <result property="isOperational" column="is\_operational"/>  
 </resultMap>  
  
 <select id="selectAllRailways" resultMap="RailwayResultMap">  
 SELECT \* FROM railways  
 </select>  
  
 <delete id="deleteRailway" parameterType="Long">  
 DELETE FROM railways  
 WHERE id = #{id}  
 </delete>  
</mapper>  
package com.labsfrom6to8.Service;  
import com.labsfrom6to8.Mapper.RailwayMapper;  
import com.labsfrom6to8.Model.Railway;  
import org.springframework.stereotype.Service;  
import java.util.List;  
@Service  
public class RailwayService {  
  
 private final RailwayMapper railwayMapper;  
  
 public RailwayService(RailwayMapper railwayMapper) {  
 this.railwayMapper = railwayMapper;  
 }  
  
 public void addRailway(Railway railway) {  
 railwayMapper.insertRailway(railway);  
 }  
  
 public void updateRailway(Railway railway) {  
 railwayMapper.updateRailway(railway);  
 }  
  
 public List<Railway> getAllRailways() {  
 return railwayMapper.selectAllRailways();  
 }  
  
 public void deleteRailwayById(Long id) {  
 railwayMapper.deleteRailway(id);  
 }  
}

package com.labsfrom6to8.Controller;  
  
import com.labsfrom6to8.Model.Railway;  
import com.labsfrom6to8.Service.RailwayService;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/railways")  
public class RailwayController {  
  
 private final RailwayService railwayService;  
  
 public RailwayController(RailwayService railwayService) {  
 this.railwayService = railwayService;  
 }  
  
 @PostMapping  
 public void addRailway(@RequestBody Railway railway) {  
 railwayService.addRailway(railway);  
 }  
  
 @PutMapping  
 public void updateRailway(@RequestBody Railway railway) {  
 railwayService.updateRailway(railway);  
 }  
  
 @GetMapping  
 public List<Railway> getAllRailways() {  
 return railwayService.getAllRailways();  
 }  
  
 @DeleteMapping("/{id}")  
 public void deleteRailway(@PathVariable Long id) {  
 railwayService.deleteRailwayById(id);  
 }  
}  
spring.application.name=labsFrom6To8  
  
spring.datasource.url=jdbc:postgresql://localhost:5432/lab6  
spring.datasource.username=postgres  
spring.datasource.password=123  
spring.datasource.driver-class-name=org.postgresql.Driver  
  
logging.level.org.mybatis=DEBUG  
logging.level.org.springframework.jdbc.core=DEBUG  
logging.level.org.springframework.jdbc.datasource=DEBUG  
logging.level.org.apache.ibatis.logging.stdout.StdOutImpl=TRACE  
mybatis.configuration.log-impl=org.apache.ibatis.logging.stdout.StdOutImpl  
  
spring.mvc.view.prefix=  
spring.mvc.view.suffix=  
server.port=8081

# **ВЫВОД**

Реализованное приложение предоставляет полнофункциональный сервис для управления информацией о железных дорогах, включая REST API для выполнения CRUD-операций. Приложение успешно подключается к базе данных PostgreSQL, использует MyBatis для маппинга объектов, а также логирование для отладки SQL-запросов. Решение демонстрирует модульную архитектуру, упрощающую расширение и сопровождение кода.