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 №14

Выполнил: Сафонов Г. К.

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 НаселённыйПункт. В БД Postgres создать таблицу «населённые пункты». С помощью MyBatis реализовать операции Добавления, Удаления, Изменения и Выборки данных из таблицы «населённые пункты». Продемонстрировать операции работы с БД.

# **ХОД ВЫПОЛНЕНИЯ РАБОТЫ**

Создано приложение для управления данными о населённых пунктах с использованием Spring Boot и MyBatis. Модель Settlement представляет данные о населённых пунктах, включая идентификатор, название, численность населения, регион и статус активности. Аннотации Jackson используются для автоматического преобразования данных в JSON и обратно.

Для работы с базой данных PostgreSQL разработан интерфейс SettlementMapper, который предоставляет методы для вставки, обновления, выборки активных населённых пунктов и удаления записей. Реализация запросов производится с помощью аннотаций MyBatis, а также XML-файла маппинга, где используется resultMap для соответствия между полями модели и колонками таблицы базы данных.

Сервис SettlementService обрабатывает бизнес-логику и предоставляет интерфейс для взаимодействия с SettlementMapper. Это включает в себя добавление, обновление, получение списка активных населённых пунктов и удаление записей.

Контроллер SettlementController реализует REST API для обработки HTTP-запросов. Он поддерживает добавление новых населённых пунктов через POST-запросы, обновление данных через PUT-запросы, получение активных записей через GET-запросы и удаление записей через DELETE-запросы.

Конфигурация подключения к базе данных PostgreSQL выполнена в файле application.properties. Логирование запросов включено для удобства отладки. Приложение работает на порту 8081, предоставляя доступ через REST API.

# **РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ**

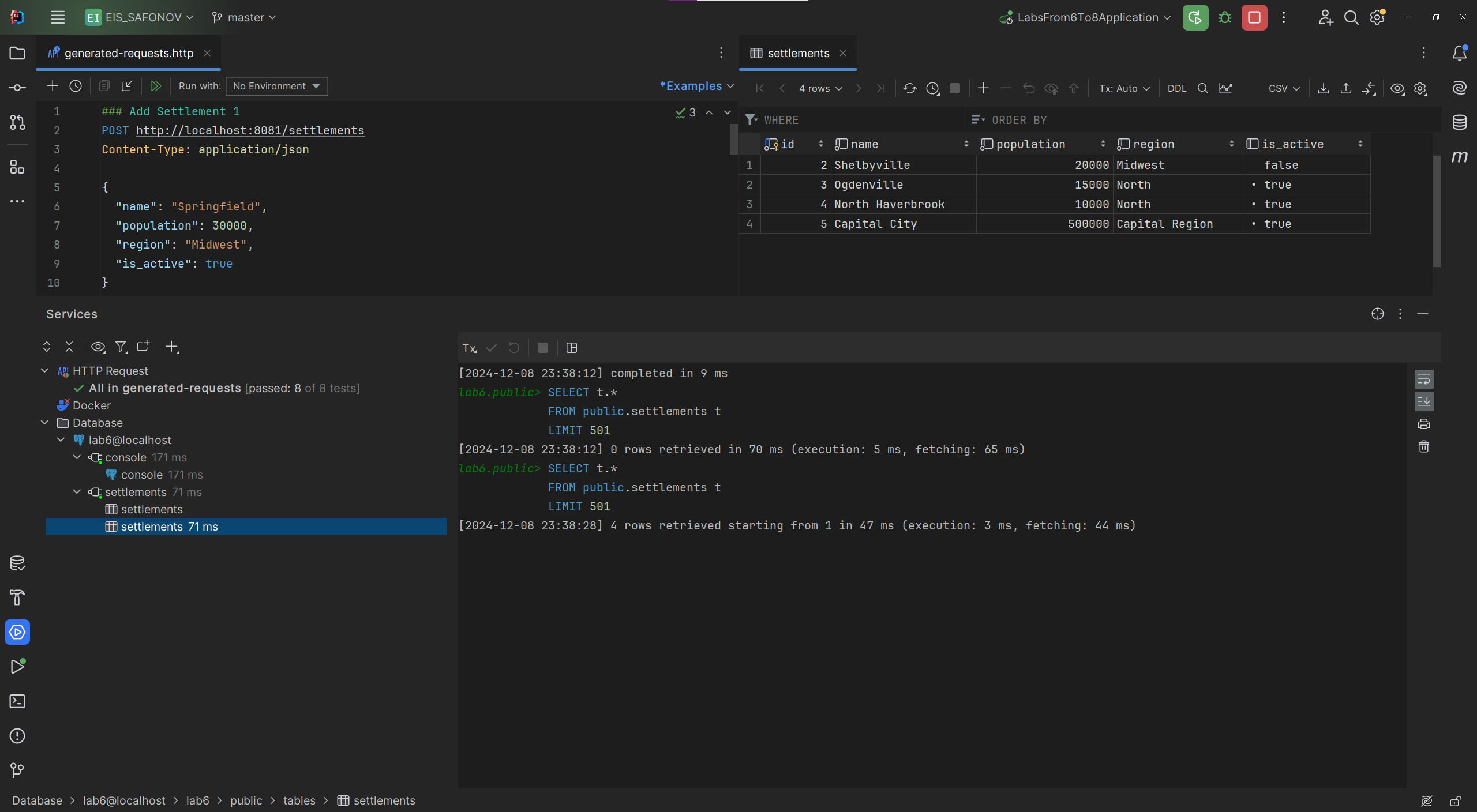


Рисунок 1 – Результат работы программы

# **КОД ПРОГРАММЫ**

package com.labsfrom6to8.Model;  
  
import lombok.AllArgsConstructor;  
import lombok.Data;  
import lombok.NoArgsConstructor;  
  
import com.fasterxml.jackson.annotation.JsonProperty;  
  
@Data  
@NoArgsConstructor  
@AllArgsConstructor  
public class Settlement {  
  
 private Long id;  
  
 @JsonProperty("name")  
 private String name;  
  
 @JsonProperty("population")  
 private Integer population;  
  
 @JsonProperty("region")  
 private String region;  
  
 @JsonProperty("is\_active")  
 private Boolean isActive;  
}

package com.labsfrom6to8.Mapper;  
  
import com.labsfrom6to8.Model.Settlement;  
import org.apache.ibatis.annotations.\*;  
  
import java.util.List;  
  
@Mapper  
public interface SettlementMapper {  
  
 @Insert("INSERT INTO settlements (name, population, region, is\_active) VALUES (#{name}, #{population}, #{region}, #{isActive})")  
 void insertSettlement(Settlement settlement);  
  
 @Update("UPDATE settlements SET name = #{name}, population = #{population}, region = #{region}, is\_active = #{isActive} WHERE id = #{id}")  
 void updateSettlement(Settlement settlement);  
  
 @Select("SELECT \* FROM settlements WHERE is\_active = TRUE")  
 List<Settlement> selectActiveSettlements();  
  
 @Delete("DELETE FROM settlements WHERE id = #{id}")  
 void deleteSettlement(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.SettlementMapper">  
  
 <insert id="insertSettlement" parameterType="com.example.Model.Settlement">  
 INSERT INTO settlements (name, population, region, is\_active)  
 VALUES (#{name}, #{population}, #{region}, #{isActive})  
 </insert>  
  
 <update id="updateSettlement" parameterType="com.example.Model.Settlement">  
 UPDATE settlements  
 SET name = #{name},  
 population = #{population},  
 region = #{region},  
 is\_active = #{isActive}  
 WHERE id = #{id}  
 </update>  
  
 <resultMap id="SettlementResultMap" type="com.example.Model.Settlement">  
 <result property="id" column="id"/>  
 <result property="name" column="name"/>  
 <result property="population" column="population"/>  
 <result property="region" column="region"/>  
 <result property="isActive" column="is\_active"/>  
 </resultMap>  
  
 <select id="selectActiveSettlements" resultMap="SettlementResultMap">  
 SELECT \* FROM settlements WHERE is\_active = TRUE  
 </select>  
  
 <delete id="deleteSettlement" parameterType="Long">  
 DELETE FROM settlements  
 WHERE id = #{id}  
 </delete>  
</mapper>

package com.labsfrom6to8.Service;  
  
  
import com.labsfrom6to8.Mapper.SettlementMapper;  
import com.labsfrom6to8.Model.Settlement;  
import org.springframework.stereotype.Service;  
  
import java.util.List;  
  
@Service  
public class SettlementService {  
  
 private final SettlementMapper settlementMapper;  
  
 public SettlementService(SettlementMapper settlementMapper) {  
 this.settlementMapper = settlementMapper;  
 }  
  
 public void addSettlement(Settlement settlement) {  
 settlementMapper.insertSettlement(settlement);  
 }  
  
 public void updateSettlement(Settlement settlement) {  
 settlementMapper.updateSettlement(settlement);  
 }  
  
 public List<Settlement> getActiveSettlements() {  
 return settlementMapper.selectActiveSettlements();  
 }  
  
 public void deleteSettlementById(Long id) {  
 settlementMapper.deleteSettlement(id);  
 }  
}

package com.labsfrom6to8.Controller;  
  
import com.labsfrom6to8.Model.Settlement;  
import com.labsfrom6to8.Service.SettlementService;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/settlements")  
public class SettlementController {  
  
 private final SettlementService settlementService;  
  
 public SettlementController(SettlementService settlementService) {  
 this.settlementService = settlementService;  
 }  
  
 @PostMapping  
 public void addSettlement(@RequestBody Settlement settlement) {  
 settlementService.addSettlement(settlement);  
 }  
  
 @PutMapping  
 public void updateSettlement(@RequestBody Settlement settlement) {  
 settlementService.updateSettlement(settlement);  
 }  
  
 @GetMapping  
 public List<Settlement> getActiveSettlements() {  
 return settlementService.getActiveSettlements();  
 }  
  
 @DeleteMapping("/{id}")  
 public void deleteSettlement(@PathVariable Long id) {  
 settlementService.deleteSettlementById(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**

Приложение предоставляет надёжный инструмент для управления данными о населённых пунктах. Использование MyBatis и Spring Boot делает решение масштабируемым и легко расширяемым. REST API позволяет эффективно интегрировать сервис с другими системами. Логирование и модульная структура приложения обеспечивают простоту поддержки и тестирования.