# 考勤管理系统《设计分析报告》

# 考勤管理系统

##### 项目整体设计

springboot + restful + data jbdc + mysql

##### 软件架构

前端+后端：

前端采用了一个网站模板，可以通过本地url进入前端页面

使用restful api来实现与前端的交互

设计了Controller层、Repository层、Config层之间的接口

设计了与数据库相接的DAO

数据库设计表项共5条，1个表

##### 开发环境要求

* java 8
* Maven
* Mysql
* springboot 2.3.0+

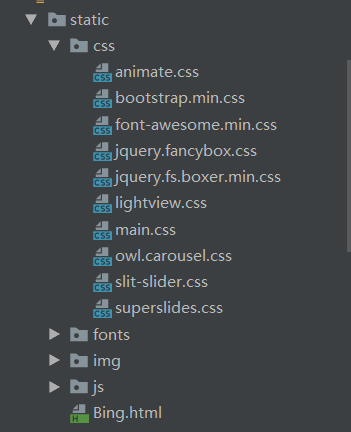
##### 启动程序的两个方法

1.maven 打包（pring-boot-maven-plugin提供了超简便的打包方法）

maven clean;     # maven 清理  
maven package   # 打包  
# 启动程序  
java -jar target/snails-0.1.jar

2.在IDEA中直接运行（示例视频采用的是这种方法，其它IDE应该都可）

前端代码设计：



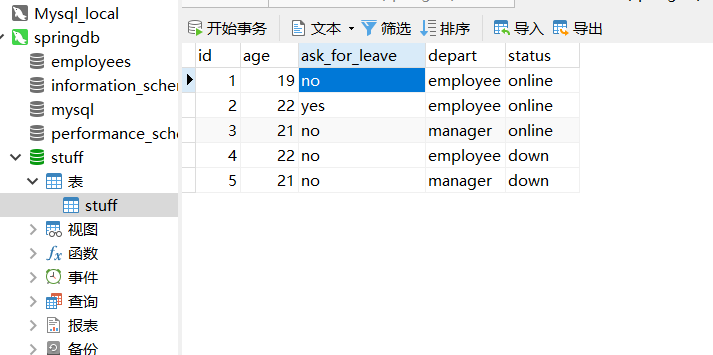
调用前端接口代码设计：

##### @Controller public class HelloController { @RequestMapping(value = "/hickey", method = RequestMethod.*GET*) public String index(){ return "index"; } }

##### restful API设计：

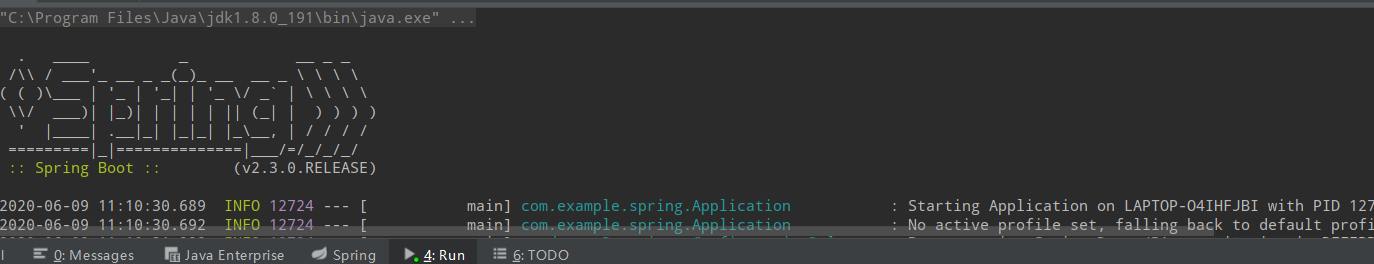
package com.example.spring;  
  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.web.bind.annotation.\*;  
  
import java.util.List;  
  
@RestController  
public class stuffController {  
  
 @Autowired  
 private StuffRepository stuffRepository;  
  
 /\*  
 获取所有职员列表  
 \*/  
 @GetMapping(value = "/stuffs")  
 public List<Stuff> stuffs(){  
 return stuffRepository.findAll();  
 }  
  
 /\*  
 添加一位职员  
 \*/  
  
 @PostMapping(value = "/stuffs")  
 public Stuff stuffAdd(@RequestParam("status") String status,  
 @RequestParam("age") int age,  
 @RequestParam("askForLeave") String aFL,  
 @RequestParam("depart") String depart) {  
 Stuff stuff = new Stuff();  
 stuff.setStatus(status);  
 stuff.setAge(age);  
 stuff.setAskForLeave(aFL);  
 stuff.setDepart(depart);  
  
 return stuffRepository.save(stuff);  
 }  
  
 /\*  
 通过id查询职员  
 \*/  
  
 @GetMapping(value = "/stuffs/{id}")  
 public Stuff findStuff(@PathVariable("id") int id){  
 if(stuffRepository.findById(id).isPresent()) {  
 return stuffRepository.findById(id).get();  
 }else return null;  
 }  
  
 /\*  
 通过年龄查询职员  
 \*/  
 @GetMapping(value = "/stuffs/age/{age}")  
 public List<Stuff> stuffListByAge(@PathVariable("age") int age){  
 return stuffRepository.findByAge(age);  
 }  
  
  
 /\*  
 通过职位查询职员  
 \*/  
 @GetMapping(value = "/stuffs/depart/{depart}")  
 public List<Stuff> stuffListByAge(@PathVariable("depart") String depart){  
 return stuffRepository.findByDepart(depart);  
 }  
  
 /\*  
 通过id更新职员信息  
 \*/  
 @PutMapping(value = "/update/{id}")  
 public Stuff stuffUpdate(@PathVariable("id") int id,  
 @RequestParam("status") String status,  
 @RequestParam("age") int age,  
 @RequestParam("askForLeave") String aFL,  
 @RequestParam("depart") String depart) {  
 Stuff stuff = new Stuff();  
 stuff.setId(id);  
 stuff.setStatus(status);  
 stuff.setAge(age);  
 stuff.setAskForLeave(aFL);  
 stuff.setDepart(depart);  
  
 return stuffRepository.save(stuff);  
 }  
  
  
  
 /\*  
 通过id删除一位职员  
 \*/  
  
 @DeleteMapping(value = "/stuffs/{id}")  
 public void deleteStuff(@PathVariable("id") int id){  
 stuffRepository.deleteById(id);  
 }  
}

##### 数据库设计：

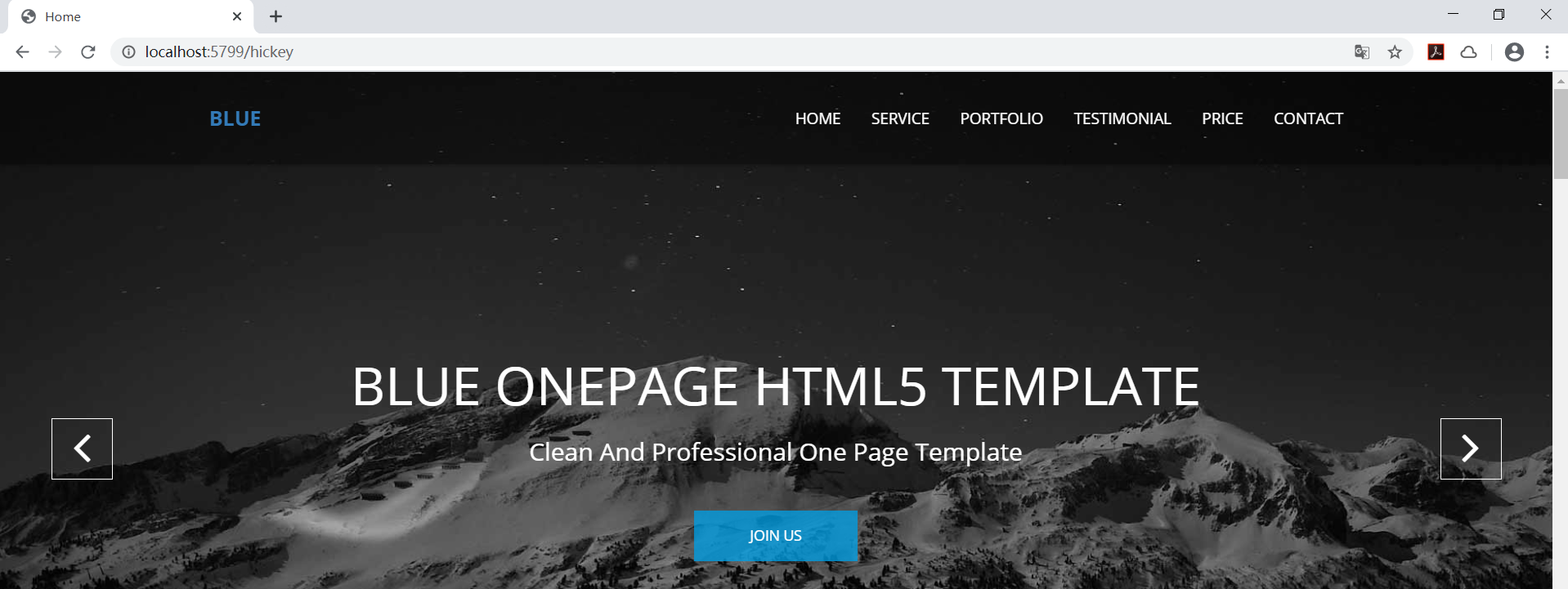


##### 测试方法

1.在项目目录下运行springboot



2.通过本地url来测试前端页面的展示



3.使用Postman来执行GET、POST、PUT、DELETE等操作。

