《Java EE Web 组件开发》实验二

html+Jpa+Mysql

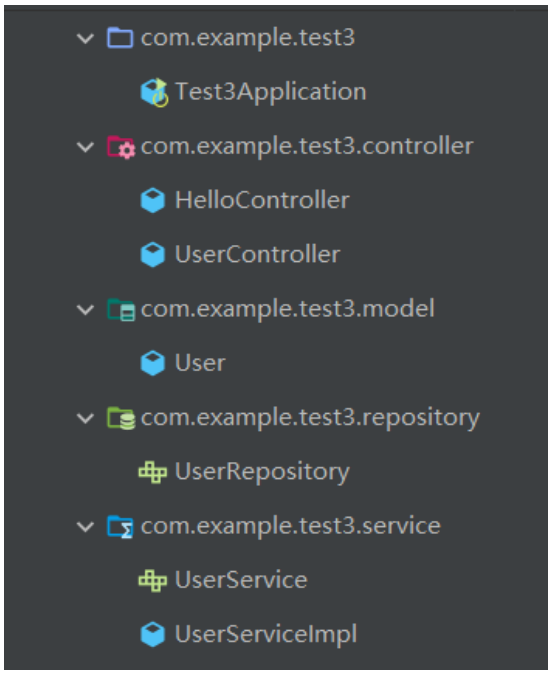
[实验任务] 实现用户信息的增删改显示功能

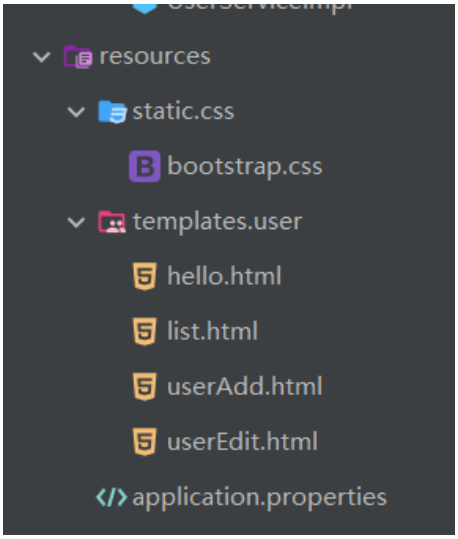
[任务介绍]

通过 thymeleaf 模板引擎实现用户信息提交，显示用户信息功能，达到前后端连通，并将数据更新到数据表的目的。

本任务要求，先建立 SpringBoot 项目，利用 thymeleaf 模板引擎语法知识，实现用户信息的提交，并通过后端数据的处理，将提交的内容加入到数据表。选择一条信息，对数据进行更新，或者删除一条记录

备注：本任务需要编写 3 个 html 页面（可以参考下图）、对应模型层、数据访问层、业务逻辑层和控制层，参考下图。





Pom 中的依赖如下：

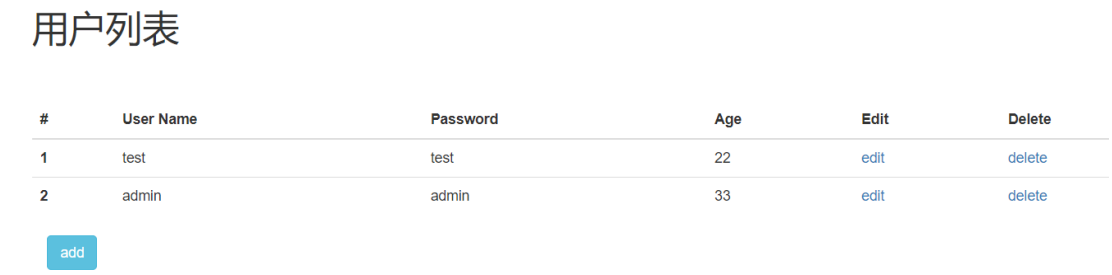
```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-thymeleaf</artifactId>
    </dependency>
```

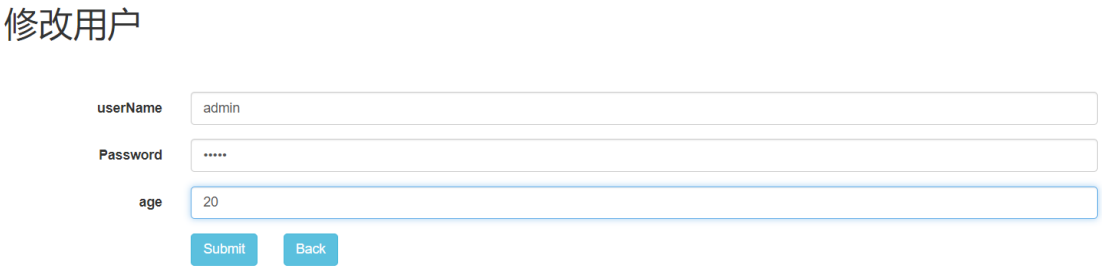
```
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>

    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>5.1.47</version>

    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>
```

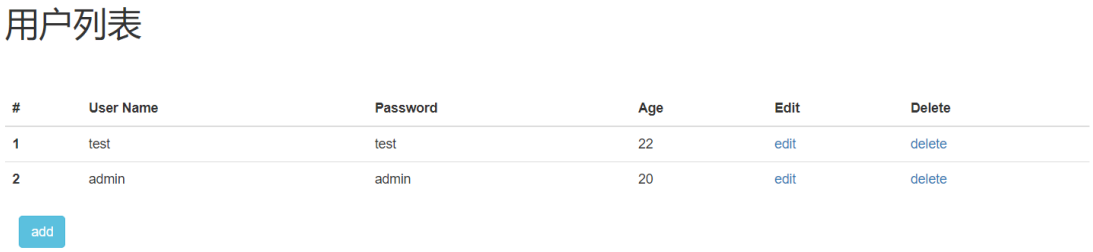
数据库配置文件；

```
spring.datasource.url=jdbc:mysql://localhost:3306/test2?characterEnco
ding=utf8
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.jdbc.Driver

spring.jpa.properties.hibernate.hbm2ddl.auto=create
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5I
nnoDBDialect
spring.jpa.show-sql= true

spring.thymeleaf.cache=false
```

结果截图：







[任务目标]

- 掌握 Jpa 接口对数据的访问；
- 链接数据库知识；
- 熟练前后端链接实现用户的增加、删除、修改等功能

[实现思路]

使用 MySQL 技术创建数据表 User 用于储存用户数据。list 页面用于展示用户数据，userAdd 页面用于添加用户，userEdit 页面用于修改用户信息。

[实现代码以及运行效果]

UserController.java

```
@Controller
public class UserController {

    @Resource
    UserService userService;


    @RequestMapping("/")
    public String index() {
        return "redirect:/list";
```

```
    }

    @RequestMapping("/list")
    public String list(Model model) {
        List<User> users=userService.getUserList();
        model.addAttribute("users", users);
        return "user/list";
    }

    @RequestMapping("/toAdd")
    public String toAdd() {
        return "user/userAdd";
    }

    @RequestMapping("/add")
    public String add(User user) {
        userService.save(user);
        return "redirect:/list";
    }

    @RequestMapping("/toEdit")
    public String toEdit(Model model,Long id) {
        User user=userService.findUserById(id);
        model.addAttribute("user", user);
        return "user/userEdit";
    }

    @RequestMapping("/edit")
    public String edit(User user) {
        userService.edit(user);
        return "redirect:/list";
    }


    @RequestMapping("/delete")
    public String delete(Long id) {
        userService.delete(id);
        return "redirect:/list";
    }
}
```

UserServiceImp.java

```
@Service
public class UserServiceImpl implements UserService{
    @Autowired
```

```
    private UserRepository userRepository;

    @Override
    public List<User> getUserList() {
        return userRepository.findAll();
    }

    @Override
    public User findUserById(long id) {
        return userRepository.findById(id);
    }

    @Override
    public void save(User user) {
        userRepository.save(user);
    }

    @Override
    public void edit(User user) {
        userRepository.save(user);
    }

    @Override
    public void delete(long id) {
        userRepository.deleteById(id);
    }
}
```

[总结或感悟]（对运行结果所作的分析以及本次调试程序所取得的经验。如果程序未能通过，分析其原因。）

本次实验所用到的知识点较多，即用到了之前学的 thymeleaf 模板引擎的知识，又用到了新学的 MySQL 数据库和 Jpa 方面的知识。在实验过程中，遇到了一些问题，运行程序时到创建数据表这一步就一直报错，好在最后通过参考老师给的程序成功解决了该问题，完成了本次实验。通过本次实验，加深了我对连接数据库和 Jpa 接口的了解。