

1 技术框架

Spring Boot + MyBatis Plus + Thymeleaf + MySQL

2 前端原型页面

静态页面，没有连接后台

```
package com.southwind.controller;

import
org.springframework.stereotype.Controller;
import
org.springframework.web.bind.annotation.GetMapping;
import
org.springframework.web.bind.annotation.PathVariable;
import
org.springframework.web.bind.annotation.ResponseBody;

@Controller
public class RedirectController {

    @GetMapping("/{url}")
    public String
    redirect(@PathVariable("url") String url){
```

```
        return url;
    }

    @GetMapping("/")
    public String main(){
        return "redirect:/main";
    }

    @GetMapping("favicon.ico")
    @ResponseBody
    void returnNoFavicon() {
    }

}
```

3 数据库

cart、orders、order_detail、product、
product_category、user、user_address

4 项目开发

- 1、创建 Spring Boot 工程，选择对应的依赖。
- 2、拷贝前端静态资源（html页面、js、css等等）html页面放到 templates 下，其他静态资源放到 static 下。
- 3、创建 RedirectController，实现页面的映射。

4、是 MyBatis Plus 连接数据库，自动生成代码 (Controller、Service、Entity、Mapper) 。

```
<dependency>
    <groupId>com.baomidou</groupId>
    <artifactId>mybatis-plus-boot-
starter</artifactId>
    <version>3.4.3.1</version>
</dependency>

<dependency>
    <groupId>com.baomidou</groupId>
    <artifactId>mybatis-plus-
generator</artifactId>
    <version>3.3.2</version>
</dependency>

<dependency>
    <groupId>org.apache.velocity</groupId>
    <artifactId>velocity</artifactId>
    <version>1.7</version>
</dependency>
```

```
package com.southwind;

import
com.baomidou.mybatisplus.annotation.DbType;
import
com.baomidou.mybatisplus.annotation.FieldFi
11;
```

```
import
com.baomidou.mybatisplus.generator.AutoGene
rator;
import
com.baomidou.mybatisplus.generator.config.D
ataSourceConfig;
import
com.baomidou.mybatisplus.generator.config.G
lobalConfig;
import
com.baomidou.mybatisplus.generator.config.P
ackageConfig;
import
com.baomidou.mybatisplus.generator.config.S
trategyConfig;
import
com.baomidou.mybatisplus.generator.config.p
o.TableFill;
import
com.baomidou.mybatisplus.generator.config.r
ules.NamingStrategy;

import java.util.ArrayList;
import java.util.List;

public class Main {
    public static void main(String[] args)
    {
        AutoGenerator autoGenerator = new
AutoGenerator();
```

```
        DataSourceConfig dataSourceConfig =
new DataSourceConfig();

        dataSourceConfig.setDbType(DbType.MYSQL);

        dataSourceConfig.setDriverName("com.mysql.
cj.jdbc.Driver");

        dataSourceConfig.setUsername("root");

        dataSourceConfig.setPassword("root");

        dataSourceConfig.setUrl("jdbc:mysql://loca
lhost:3306/maall?
useUnicode=true&characterEncoding=UTF-8");

        autoGenerator.setDataSource(dataSourceConf
ig);

        GlobalConfig globalConfig = new
GlobalConfig();
        globalConfig.setOpen(true);

        globalConfig.setOutputDir(System.getProper
ty("user.dir")+"/src/main/java");
        globalConfig.setAuthor("admin");

        globalConfig.setServiceName("%sService");

        autoGenerator.setGlobalConfig(globalConfig
);
```

```
PackageConfig packageConfig = new
PackageConfig();

packageConfig.setParent("com.southwind");
packageConfig.setEntity("entity");
packageConfig.setMapper("mapper");

packageConfig.setController("controller");

packageConfig.setService("service");

packageConfig.setServiceImpl("service.impl
");

autoGenerator.setPackageInfo(packageConfig
);

StrategyConfig strategyConfig = new
StrategyConfig();

strategyConfig.setEntityLombokModel(true);

strategyConfig.setNaming(NamingStrategy.underline_to_camel);

strategyConfig.setColumnNaming(NamingStrategy.underline_to_camel);

List<TableFill> list = new
ArrayList<>();
```

```
        TableFill tableFill1 = new
TableFill("create_time", FieldFill.INSERT);
        TableFill tableFill2 = new
TableFill("update_time",FieldFill.INSERT_UP
DATE);
        list.add(tableFill1);
        list.add(tableFill2);

strategyConfig.setTableFillList(list);

autoGenerator.setStrategy(strategyConfig);

        autoGenerator.execute();
    }
}
```

```
spring:
  datasource:
    url: jdbc:mysql://localhost:3306/mmall?
useUnicode=true&characterEncoding=UTF-8
    username: root
    password: root
    driver-class-name:
com.mysql.cj.jdbc.Driver
mybatis-plus:
  configuration:
    log-impl:
org.apache.ibatis.logging.stdout.StdOutImpl
  mapper-locations:
classpath:com/southwind/mapper/xml/*.xml
```

```
package com.southwind;

import
org.mybatis.spring.annotation.MapperScan;
import
org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.Spr
ingBootApplication;

@SpringBootApplication
@MapperScan("com.southwind.mapper")
public class MmallApplication {
```



```
    public static void main(String[] args)
    {

        SpringApplication.run(MmallApplication.class, args);
    }

}
```

5 注册

- 1、前端进行各种校验，提交表单
- 2、后端再一次进行数据校验（非空校验 & 格式校验）

```
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-
validator</artifactId>
    <version>5.2.4.Final</version>
</dependency>
<dependency>
    <groupId>javax.validation</groupId>
    <artifactId>validation-api</artifactId>
    <version>1.1.0.Final</version>
</dependency>
<dependency>
    <groupId>org.jboss.logging</groupId>
    <artifactId>jboss-logging</artifactId>
    <version>3.4.1.Final</version>
```

</dependency>

```
package com.southwind.form;

import lombok.Data;
import
org.hibernate.validator.constraints.NotEmpty;

import
javax.validation.constraints.NotNull;

@Data
public class UserRegisterForm {
    @NotEmpty(message = "登录名不能为空")
    private String loginName;
    @NotEmpty(message = "用户名不能为空")
    private String userName;
    @NotEmpty(message = "密码不能为空")
    private String password;
    @NotNull(message = "性别不能为空")
    private Integer gender;
    @NotEmpty(message = "邮箱不能为空")
    private String email;
    @NotEmpty(message = "手机不能为空")
    private String mobile;
}
```

- 1、进行数据校验
- 2、进行数据存储

```
package com.southwind.exception;

import com.southwind.result.ResponseEnum;
import
org.springframework.web.bind.annotation.Exc
eptionHandler;
import
org.springframework.web.bind.annotation.Res
tControllerAdvice;
import
org.springframework.web.servlet.ModelAndVie
w;

@RestControllerAdvice
public class UnifiedExceptionHandler {

    @ExceptionHandler(value =
MMAllException.class)
    public ModelAndView
handlerException(MMAllException e){
        ModelAndView modelAndView = new
ModelAndView();

        modelAndView.setViewName("register");
        ResponseEnum responseEnum =
e.getResponseEnum();
        switch (responseEnum.getCode()){
            case 301:
```

```
        modelAndView.addObject("emailError",
responseEnum.getMsg());
            break;
        case 302:

            modelAndView.addObject("mobileError",
responseEnum.getMsg());
            break;
        case 303:

            modelAndView.addObject("userNameExists",
responseEnum.getMsg());
            break;
    }
    return modelAndView;
}
}
```

```
package com.southwind.service.impl;

import
com.baomidou.mybatisplus.core.conditions.qu
ery.QueryWrapper;
import com.southwind.entity.User;
import
com.southwind.exception.MMailException;
import com.southwind.form.UserRegisterForm;
import com.southwind.mapper.UserMapper;
```

```
import com.southwind.result.ResponseEnum;
import com.southwind.service.UserService;
import
com.baomidou.mybatisplus.extension.service.
impl.ServiceImpl;
import com.southwind.utils.MD5Util;
import
com.southwind.utils.RegexValidateUtil;
import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.BeanUtils;
import
org.springframework.beans.factory.annotation.Autowired;
import
org.springframework.stereotype.Service;

/**
 * <p>
 * 服务实现类
 * </p>
 *
 * @author admin
 * @since 2021-11-22
 */
@Service
@Slf4j
public class UserServiceImpl extends
ServiceImpl<UserMapper, User> implements
UserService {
```

```
@Autowired
private UserMapper userMapper;

@Override
public User register(UserRegisterForm
userRegisterForm) {
    //用户名是否可用
    QueryWrapper<User> queryWrapper =
new QueryWrapper<>();
    queryWrapper.eq("login_name",
userRegisterForm.getLoginName());
    User one =
this.userMapper.selectOne(queryWrapper);
    if(one != null){
        log.info("【用户注册】用户名已存
在");
        throw new
MallException(ResponseEnum.USERNAME_EXISTS
);
    }
    //邮箱格式校验

    if(!RegexValidateUtil.checkEmail(userRegis
terForm.getEmail())){
        log.info("【用户注册】邮箱格式错
误");
        throw new
MallException(ResponseEnum.EMAIL_ERROR);
    }
    //手机格式校验
```

```
        if(!RegexValidateUtil.checkMobile(userRegisterForm.getMobile())){
            log.info("【用户注册】手机格式错误");

            throw new
            MMallException(ResponseEnum.MOBILE_ERROR);
        }
        //存储数据
        User user = new User();

        BeanUtils.copyProperties(userRegisterForm,
        user);

        user.setPassword(MD5Util.getSaltMD5(user.getPassword()));
        int insert =
        this.userMapper.insert(user);
        if(insert != 1){
            log.info("【用户注册】添加用户失败");

            throw new
            MMallException(ResponseEnum.USER_REGISTER_ERROR);
        }
        return user;
    }
}
```

6 登录

[illegible]


```
        <span th:text="${passwordError}"
style="color: red;text-align: center">
    </span>
    </td>
</tr>
```

```
@PostMapping("/login")
public ModelAndView login(@valid
UserLoginForm userLoginForm,BindingResult
bindingResult){
    //非空校验
    if(bindingResult.hasErrors()){
        log.info("【用户登录】用户信息不能为
空");
        throw new
MmallException(ResponseEnum.USER_INFO_NULL)
;
    }
    User login =
this.userService.login(userLoginForm);
    ModelAndView modelAndView = new
ModelAndView();
    modelAndView.setViewName("main");
    return modelAndView;
}
```

7 首页

加载三类数据：

1、购物车（判断是否登录）



2、商品分类



3、商品信息（一级分类）




```
@GetMapping("/main")
public ModelAndView main(HttpSession session){
```

```
ModelAndView modelAndView = new
ModelAndView();
modelAndView.setViewName("main");
//封装商品分类菜单
modelAndView.addObject("list",
this.productCategoryService.buildProductCat
egoryMenu());
//判断是否为登录用户
User user = (User)
session.getAttribute("user");
if(user == null){
//未登录
modelAndView.addObject("cartList",
new ArrayList<>());
}else{
//登录用户
//查询该用户的购物车记录
}
return modelAndView;
}
```

8 商品列表

加载三类数据：

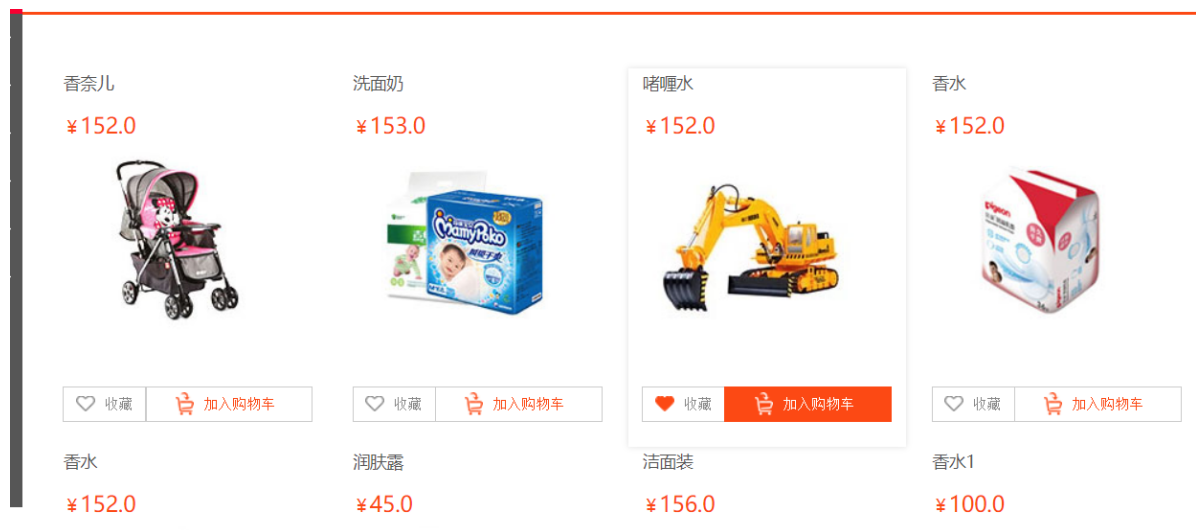
1、购物车（判断是否登录）

 购物车 [0] ▼

2、商品分类



3、所选商品分类对应的商品信息



```
@GetMapping("/list/{type}/{id}")
public ModelAndView list(
    @PathVariable("type") Integer type,
    @PathVariable("id") Integer
    productId,
    HttpSession session
){
    if(type == null || productId ==
    null){
        log.info("【商品列表】参数为空");
```

```

        throw new
        MMA11Exception(ResponseEnum.PARAMETER_NULL)
        ;
    }
    ModelAndView modelAndView = new
    ModelAndView();

    modelAndView.setViewName("productList");
    modelAndView.addObject("productList",
    this.productService.findAllByTypeAndProduct
    CategoryId(type, productId));
    //判断是否为登录用户
    User user = (User)
    session.getAttribute("user");
    if(user == null){
        //未登录
        modelAndView.addObject("cartList",
    new ArrayList<>());
    }else{
        //登录用户
        //查询该用户的购物车记录
    }
    modelAndView.addObject("list",
    this.productCategoryService.buildProductCat
    egoryMenu());
    return modelAndView;
}

```

9 商品详情

加载 3 类数据:

1、商品信息

2、购物车

3、商品分类

```
@GetMapping("/detail/{id}")
public ModelAndView
detail(@PathVariable("id") Integer
id, HttpSession session){
    if(id == null){
        log.info("【商品详情】参数为空");
        throw new
        MmallException(ResponseEnum.PARAMETER_NULL)
        ;
    }
    ModelAndView modelAndView = new
    ModelAndView();

    modelAndView.setViewName("productDetail");
    //判断是否为登录用户
    User user = (User)
    session.getAttribute("user");
    if(user == null){
        //未登录
        modelAndView.addObject("cartList",
        new ArrayList<>());
    }else{
        //登录用户
```

```
        //查询该用户的购物车记录
    }
    //商品分类
    modelAndView.addObject("list",
this.productCategoryService.buildProductCategoryMenu());
    //商品详情
    modelAndView.addObject("product",
this.productService.getById(id));
    return modelAndView;
}
```

10 添加购物车

首先要判断是否为登录用户，使用过滤器来实现

```
package com.southwind.filter;

import javax.servlet.*;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import java.io.IOException;

public class UserFilter implements Filter {
    @Override
```

```

    public void doFilter(ServletRequest
servletRequest, ServletResponse
servletResponse, FilterChain filterChain)
throws IOException, ServletException {
        HttpServletRequest request =
(HttpServletRequest) servletRequest;
        HttpServletResponse response =
(HttpServletResponse) servletResponse;
        HttpSession session =
request.getSession();
        if(session.getAttribute("user") ==
null){

            response.sendRedirect("/login");
        }else{

            filterChain.doFilter(servletRequest,servle
tResponse);
        }
    }
}

```

```

package com.southwind.configuration;

import com.southwind.filter.UserFilter;
import
org.springframework.boot.web.servlet.Filter
RegistrationBean;

```



```
import
org.springframework.context.annotation.Bean
;
import
org.springframework.context.annotation.Conf
iguration;

@Configuration
public class FilterConfiguration {

    @Bean
    public FilterRegistrationBean
registrationBean(){
        FilterRegistrationBean
filterRegistrationBean = new
FilterRegistrationBean();

        filterRegistrationBean.setFilter(new
UserFilter());

        filterRegistrationBean.addUrlPatterns("/ca
rt/*");
        return filterRegistrationBean;
    }
}
```

添加完购物车数据之后，还有进行商品减库存的操作。

11 修改购物车

```
/**
 * 更新购物车
 * @return
 */
@PostMapping("/update/{id}/{quantity}/{cost}")
@ResponseBody
public String update(
    @PathVariable("id") Integer id,
    @PathVariable("quantity") Integer
quantity,
    @PathVariable("cost") Float cost,
    HttpSession session
){
    if(id == null || quantity == null ||
cost == null){
        log.info("【更新购物车】参数为空");
        throw new
MallException(ResponseEnum.PARAMETER_NULL)
;
    }
    //判断是否为登录用户
    User user = (User)
session.getAttribute("user");
    if(user == null){
        log.info("【更新购物车】当前为未登录状
态");
        throw new
MallException(ResponseEnum.NOT_LOGIN);
    }
}
```

```
        if(this.cartService.update(id,
quantity, cost)) return "success";
        return "fail";
    }
}
```

12 删除购物车

```
/**
 * 删除购物车
 * @param id
 * @param session
 * @return
 */
@GetMapping("/delete/{id}")
public String delete(@PathVariable("id")
Integer id,HttpSession session){
    if(id == null){
        log.info("【更新购物车】参数为空");
        throw new
MmAllException(ResponseEnum.PARAMETER_NULL)
;
    }
    //判断是否为登录用户
    User user = (User)
session.getAttribute("user");
    if(user == null){
        log.info("【更新购物车】当前为未登录状
态");
    }
}
```

```
        throw new
        MmallException(ResponseEnum.NOT_LOGIN);
    }
    boolean delete =
    this.cartService.delete(id);
    if(delete) return "redirect:/cart/get";
    return null;
}
```

13 确认结算

- 1、购物车 数据库
- 2、收货人信息 Session
- 3、收货地址 数据库

点击确认结算，要将当前登录用户的购物车记录转换为订单记录

- 1、删除当前用户的购物车数据
- 2、创建订单表数据（主从表）

```
/**
 * 确认订单
 * @param userAddress
 * @param session
 * @return
 */
@PostMapping("/commit")
```

```

public ModelAndView commit(String
userAddress,HttpSession session){
    if(userAddress == null){
        log.info("【更新购物车】参数为空");
        throw new
MmallException(ResponseEnum.PARAMETER_NULL)
;
    }
    //判断是否为登录用户
    User user = (User)
session.getAttribute("user");
    if(user == null){
        log.info("【更新购物车】当前为未登录状
态");
        throw new
MmallException(ResponseEnum.NOT_LOGIN);
    }
    ModelAndView modelAndView = new
ModelAndView();

    modelAndView.setViewName("settlement3");
    if(this.cartService.commit(userAddress,
user)) return modelAndView;
    return null;
}

```

如果选择新地址：

- 1、将新地址作为订单的收货地址
- 2、将新地址存入数据库

14 管理中心

1、我的订单

2、用户信息

3、地址管理