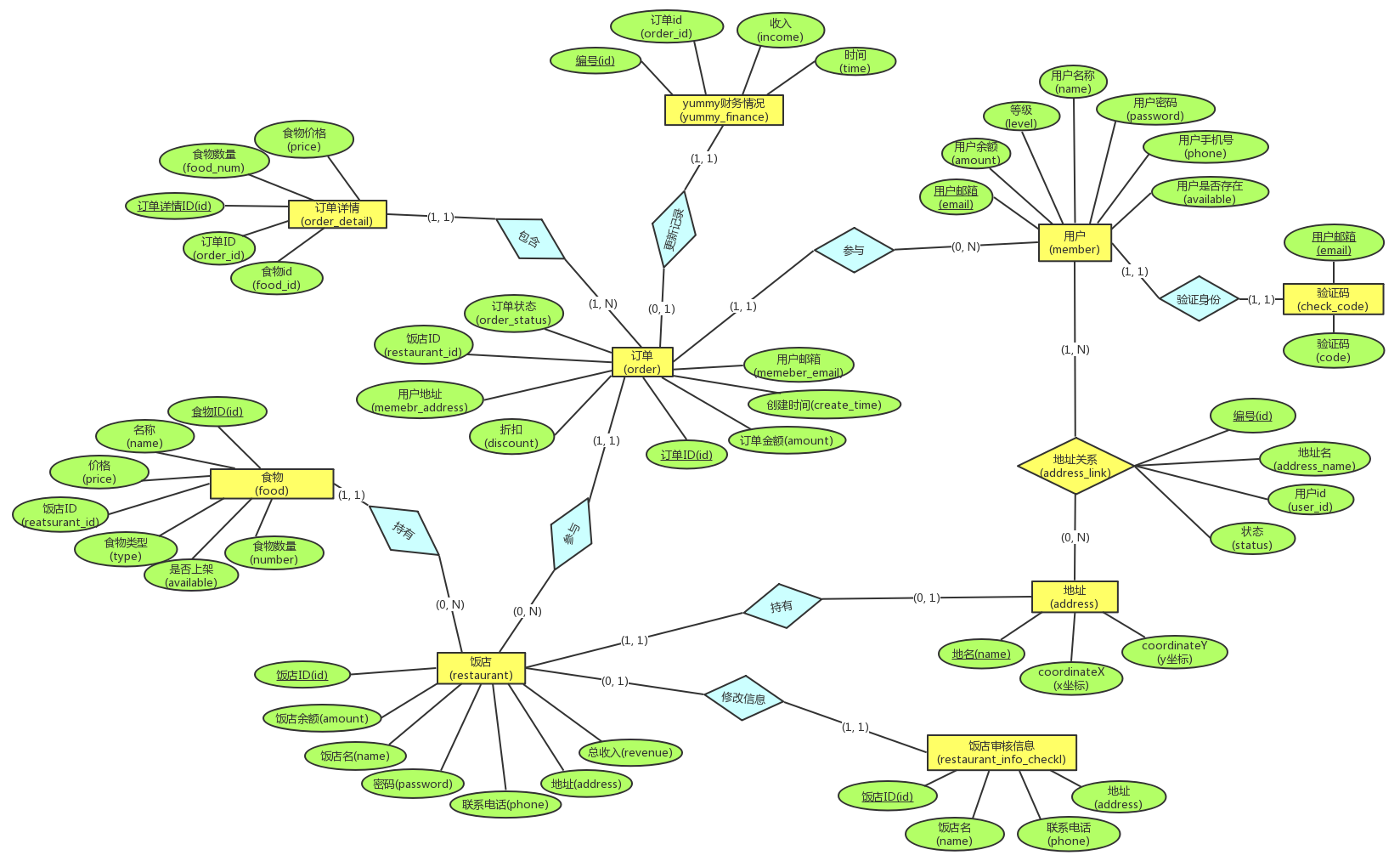
**J2EE**说明文档

1. 数据库设计

**1.1 ER**图

**1.2** 数据库表

**1.2.1** 用户（**member**）

用户（member）表保存了普通用户的基本信息，当用户注册的时候会在用户表中插入一条新的记录，但是当用户注销账号的时候，不会从表中删除该用户，而是将用户表的available属性设置为false。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| email（主键） | VARCHAR(255), PRIMARY KEY | 用户的邮箱 |
| amount | DOUBLE | 用户的余额（元） |
| level | INT(11) | 用户的等级 |
| name | VARCHAR(255) | 用户的名称 |
| password | VARCHAR(255) | 用户的密码 |
| phone | VARCHAR(255) | 用户的手机号 |
| available | BIT(1) | 用户是否存在，注销的场合为否 |

**1.2.2** 验证码（**check\_code**）

验证码（check\_code）表保存了用户对应的验证码，当用户注册的时候必须先获取验证码，只有当正确填写发送至邮箱的验证码才能成功注册。当用户注册后用户的邮箱不能重新注册。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| email（主键） | VARCHAR(255), PRIMARY KEY | 用户的邮箱 |
| code | VARCHAR(255) | 发送的验证码 |

**1.2.3** 地址（**address**）

地址（address）保存了各个地点的信息，其主键为地址名称。每个地址包含X坐标和Y坐标，可以根据X和Y坐标计算两个地址之间的距离，这在订单下单的时候会用到。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| name（主键） | VARCHAR(255), PRIMARY KEY | 用户的邮箱 |
| coordinateX | INT(11) | 地址的X坐标 |
| coordinateY | INT(11) | 地址的Y坐标 |

**1.2.4** 饭店（**restaurant**）

饭店（restaurant）保存了饭店的基本信息，如名称、余额、电话等。饭店因为只有一个地址，所以不在address\_link表中单独管理，而是把address作为饭店的一个属性。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| id（主键） | VARCHAR(255), PRIMARY KEY | 饭店的7位编号 |
| amount | DOUBLE | 饭店的余额 |
| name | VARCHAR(255) | 饭店名称 |
| passsword | VARCHAR(255) | 饭店密码 |
| phone | VARCHAR(255) | 饭店的电话 |
| address | VARCHAR(255) | 饭店的地址名称 |
| revenue | DOUBLE | 饭店的营业额 |

**1.2.5** 地址关系（**address\_link**）

地址关系（address\_link）保存了用户和用户对应的地址。因为一个用户可以有多个地址，一个地址可以属于多个用户，所以用一个专门的表来保存。status属性保存了地址的状态，是普通地址/默认地址。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| id（主键） | INT(11), PRIMARY KEY | 编号 |
| address\_name | VARCHAR(255) | 地址的名称 |
| user\_id | VARCHAR(255) | 用户id |
| status | INT(11) | 地址状态 |

**1.2.6** 食物（**food**）

食物（food）表保存了食物的信息，包括名称、价格、饭店id、类型和数量等。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| id（主键） | INT(11), PRIMARY KEY | 编号 |
| name | VARCHAR(255) | 食物的名称 |
| price | DOUBLE | 食物价格 |
| restaurant\_id | VARCHAR(255) | 饭店id |
| type | VARCHAR(255) | 食物类型 |
| available | BIT(1) | 食物是否上架 |
| number | INT(11) | 食物数量 |

**1.2.7** 饭店审核信息（**restaurant\_info\_check**）

饭店审核信息（restaurant\_info\_check）保存了饭店提交的待审核的信息，包括地址、名称、电话。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| id（主键） | VARCHAR(255), PRIMARY KEY | 饭店编号 |
| name | VARCHAR(255) | 饭店新名称 |
| address | VARCHAR(255) | 饭店新地址 |
| phone | VARCHAR(255) | 饭店新电话 |

**1.2.8** 订单（**order**）

订单（order）表保存了每一笔订单的记录，包括用户id、饭店id、用户地址、折扣和订单状态等，订单状态是一个int值，不同的int值分别表示“未支付/已支付/配送中/成功/已取消”几个状态之一。折扣discount由用户的等级来计算。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| id（主键） | VARCHAR(255), PRIMARY KEY | 订单编号 |
| amount | DOUBLE | 订单金额 |
| create\_time | DATETIME | 订单创建时间 |
| member\_email | VARCHAR(255) | 用户邮箱 |
| address | VARCHAR(255) | 用户地址 |
| restaurant\_id | VARCHAR(255) | 饭店地址 |
| order\_status | INT(11) | 订单状态 |
| discount | DOUBLE | 订单折扣 |

**1.2.9** 订单详情（**order\_detail**）

订单详情（order\_detail）表了一笔订单中某种食物的数量和价格。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| id（主键） | INT(11), PRIMARY KEY | 编号 |
| food\_id | INT(11) | 食物id |
| food\_num | INT(11) | 食物数量 |
| order\_id | VARCHAR(255) | 订单编号 |
| price | DOUBLE | 食物单价 |

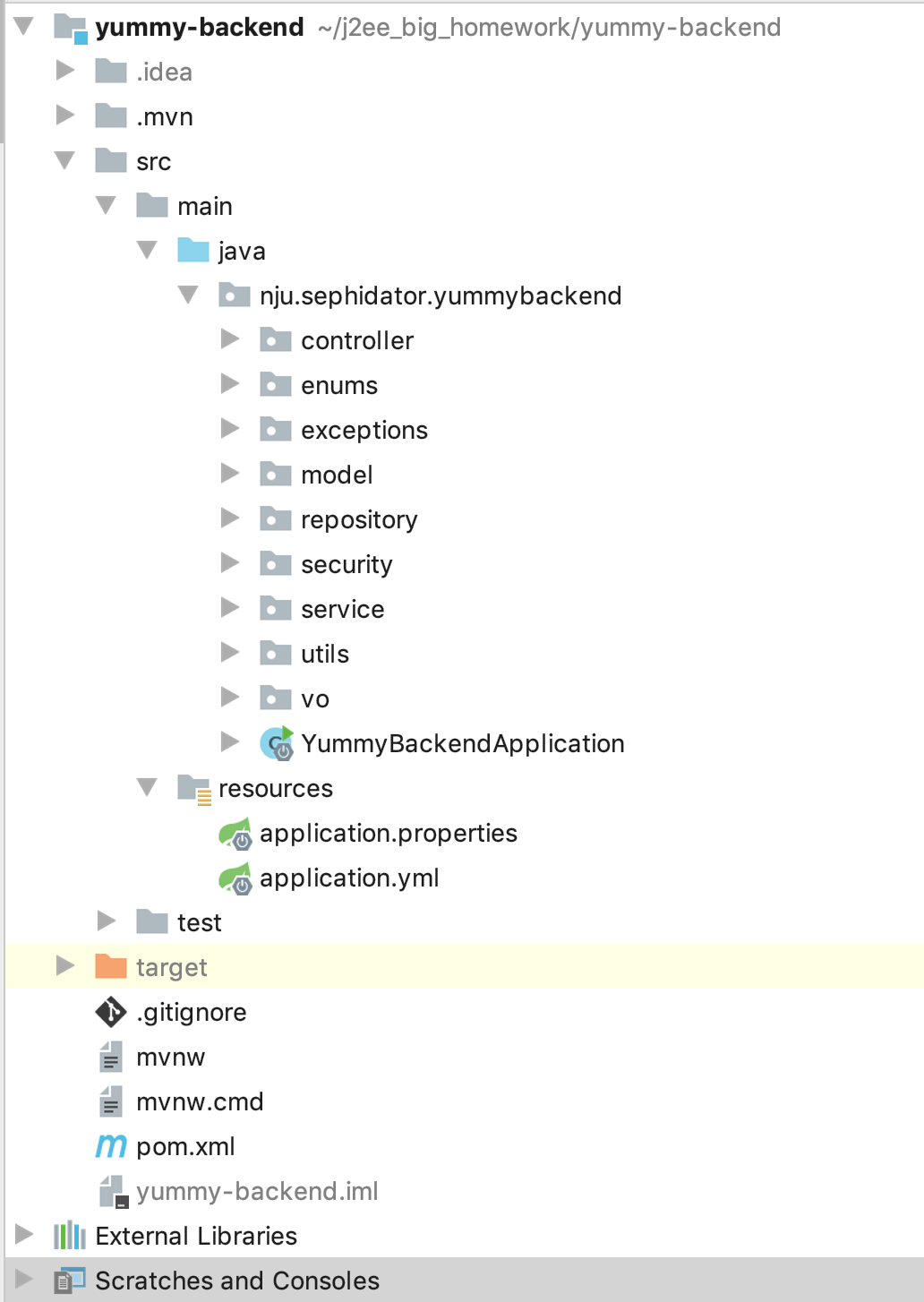
**1.2.10 yummy**财务情况（**yummy\_finance**）

yummy财务情况（yummy\_finance）表记录了每一笔订单带给yummy的收入或者订单退款带来的支出。

| 属性 | 类型 | 含义 |
| --- | --- | --- |
| id（主键） | INT(11), PRIMARY KEY | 编号 |
| income | DOUBLE | 订单带来的收入/支出 |
| order\_id | VARCHAR(255) | 订单编号 |
| time | DATETIME | 发生的时间 |

1. 架构设计

**2.1** 项目结构

****

**2.2** 框架的使用

后端使用Spring Boot + JPA框架，使用MySQL数据库

前端使用Vue.js框架，使用antd for vue组件库

1. 类设计

**3.1** 类的名称和职责

**3.1.1 controller**

| 名称 | 职责 |
| --- | --- |
| AddressController | 处理和地址相关的请求 |
| AdminController | 处理和管理员相关的请求 |
| FoodController | 处理和食物相关的请求 |
| MemberController | 处理和用户相关的请求 |
| OrderController | 处理和订单相关的请求 |
| RestaurantController | 处理和饭店相关的请求 |

**3.1.2 enums**

| 名称 | 职责 |
| --- | --- |
| AddressStatus | 地址状态的枚举 |
| OrderStatus | 订单状态的枚举 |
| Paystatus | 支付状态的枚举 |
| UserType | 用户状态的枚举 |

**3.1.3 model**

| 名称 | 职责 |
| --- | --- |
| Address | 地址的实体对象 |
| AddressLink | 地址关系的实体对象 |
| CheckCode | 验证码的实体对象 |
| Food | 食物的实体对象 |
| Member | 普通用户的实体对象 |
| Order | 订单的实体对象 |
| OrderDetail | 订单细节的实体对象 |
| Restaurant | 饭店的实体对象 |
| RestaurantInfoCheck | 饭店审核信息的实体对象 |
| YummyFinance | Yummy财务情况的实体对象 |

**3.1.4 exceptions**

| 名称 | 职责 |
| --- | --- |
| FoodInsufficientException | 自定义错误：食物不足 |
| MemberAmountException | 自定义错误：用户余额不足 |
| RestaurantAmountException | 自定义错误：饭店余额不足 |

**3.1.5 repository**

| 名称 | 职责 |
| --- | --- |
| AddressJPA | 访问地址实体对象的repository类 |
| AddressLinkJPA | 访问地址关系实体对象的repository类 |
| CheckCodeJPA | 访问验证码实体对象的repository类 |
| FoodJPA | 访问食物实体对象的repository类 |
| MemberJPA | 访问普通用户实体对象的repository类 |
| OrderJPA | 访问订单实体对象的repository类 |
| OrderDetailJPA | 访问订单细节实体对象的repository类 |
| RestaurantJPA | 访问饭店实体对象的repository类 |
| RestaurantInfoCheckJPA | 访问饭店审核信息实体对象的repository类 |
| YummyFinanceJPA | 访问Yummy财务情况实体对象的repository类 |

**3.1.5 service**

| 名称 | 职责 |
| --- | --- |
| AddressService(Impl) | 处理和地址相关的业务逻辑 |
| AdminSewrvice(Impl) | 处理和管理员相关的业务逻辑 |
| FoodService(Impl) | 处理和食物相关的业务逻辑 |
| MemberService(Impl) | 处理和用户相关的业务逻辑 |
| OrderService(Impl) | 处理和订单相关的业务逻辑 |
| RestaurantService(Impl) | 处理和饭店相关的业务逻辑 |
| UpdateService(Impl) | 处理和订单更新相关的业务逻辑 |

**3.1.6 utils**

| 名称 | 职责 |
| --- | --- |
| KeyUtil | 生成验证码/饭店的7位编号 |
| MathUtil | 计算距离/数字操作 |
| ResultVOUtil | 生成返回给前端的对象 |
| TimeUtil | 时间格式化 |

**3.2** 前端界面

| 名称 | 功能 |
| --- | --- |
| 登陆界面 | 用户/饭店/管理员登陆 |
| 用户主界面 | 显示系统内的饭店 |
| 用户购物车界面 | 显示某家饭店的信息和菜品，用户下单 |
| 用户订单界面 | 用户查看和编辑订单 |
| 用户信息界面 | 用户编辑个人信息 |
| 饭店订单界面 | 饭店查看和编辑自身参与的订单 |
| 饭店信息界面 | 饭店修改自身信息或者菜品 |
| 管理员信息界面 | 显示管理员信息和待审批的饭店信息 |
| 财务统计界面 | 显示财务统计情况 |
| 用户统计界面 | 显示用户统计情况 |

1. 其他

**4.1** 开发环境

后端：Java8, Spring Boot 2.1, MySQL 5.23

前端：ES6, Vue.js 2.6.6, node 10.12.0, ant-design-vue 1.3.5

**4.2** 开发心得

在这次作业中，通过前后端协同开发较大的项目，我探索了很多之前未深入的细节，学到了很多配置和使用方面的实践经验，同时在开发过程中，为了实现一些功能需求不断地查资料，我掌握了很多以前不知道的界面/代码方面的技巧，实在是受益颇多。