# 系统架构设计

## 前端Web

### 框架：

前端Web采用AngularJS框架。它具有以下特性：

1. 数据绑定：这是angularJS最实用的特性，它能够帮助你避免书写大量的初始代码从而节约开发时间。一个典型的 Web 应用可能包含了 80% 的代码用来处理，查询和监听 DOM。数据绑定使得代码更少，你可以专注于你的应用。
2. 模板：在 AngularJS 中，一个模板就是一个 HTML 文件。但是 HTML 的内容扩展了，包含了很多帮助你映射 Model 到 View 的内容。HTML 模板将会被浏览器解析到 DOM 中。DOM 然后成为 AngularJS 编译器的输入。AngularJS 将会遍历 DOM 模板来生成一些指导，即directive（指令）。所有的指令都负责针对 View 来设置数据绑定。我们要理解 AuguarJS 并不把模板当做 String 来操作。输入 AngularJS 的是 DOM 而非 string。数据绑定是 DOM 变化，不是字符串的连接或者 innerHTML 变化。使用 DOM 作为输入，而不是字符串，是 AngularJS 区别于其它的框架的最大原因。使用 DOM 允许你扩展指令词汇并且可以创建你自己的指令，甚至开发可重用的组件。
3. MVC设计典范：针对客户端应用开发 AngularJS 吸收了传统的 MVC 基本原则。MVC 分层有助于管理复杂的应用程序，同时也让应用程序的测试更加容易。MVC 分层同时也简化了分组开发。不同的开发人员可同时开发视图、控制器逻辑和业务逻辑。方便适应新的需求，也方便团队开发。
4. 依赖注入：angularJS通过控制器引入依赖的方式引入丰富的组件。

### 与后端的交互：

AngularJS通过引入ngresource作为依赖，采用**RESTful接口**，以不同的方法或不同的目标接口实现与后台的交互。

## 前端App（王梓璇、苏博文）

微信小程序开发

## 后端

后端采用Flask框架，Flask是一个使用Python编写的轻量级Web应用框架。基于Werkzeug WSGI工具箱和Jinja2模板引擎。它的优势在于轻量，简单，并具有较好的兼容性。数据库我们使用Mysql。

## 接口设计

使用流程：

用户在访问网上生鲜超市时，可以在未登录的状态下浏览商品。购买商品时需要登录，若无账号则先注册，拥有账户ID、用户名和密码。注册账号后随时可以修改个人信息，包括用户名、密码、绑定的手机号、收货地址和头像。

将商品添加到购物车后可在购物车页面确认，修改或购买。购买后系统会生成账单。

根据以上操作设置以下接口：

## 1.用户

### 创建新用户

#### Reuqest

* Method: **POST**
* URL: /user
* Headers: Content-Type:multipart/form-data
* Body:

phone 111111

username newguy

password 112322

image file

#### Response

* Body

{

"code": 200,

"message": "new user has created"

}

### 更新用户信息

#### Reuqest

* Method: **PUT**
* URL: /user
* Headers: Content-Type:multipart/form-data
* Body:

phone 111111

username newguy

password 112322

image file

#### Response

* Body

{

"code": 200

}

### 用户登录

#### Reuqest

* Method: **GET**
* URL: /user
* Headers: Content-Type:multipart/form-data
* Body:

username newguy

password 112322

#### Response

* Body

{

"code": 200,

"data":{

"phone":"111111",

"userId":"newguy",

}

}

## 2.商品

### 获取商品信息

#### Reuqest

* Method: **GET**
* URL: /goods
* Headers:
* Body:

#### Response

* Body

{

"code": 200,

"data":[{

"goodsID":"111",

"category":"水果",

"name":"火龙果",

"word\_description":"12adfsfsdf",

"image":"http://1.1.1.1/2323.jpg",

"price":60.5,

"remaining":5

},{

"goodsID":"112",

"category":"🐟",

"name":"鲤鱼",

"word\_description":"12adfsfsdf",

"image":"http://1.1.1.1/2323.jpg",

"price":60.5,

"remaining":5

}],

}

## 3.购物车

### 添加商品到购物车

#### Reuqest

* Method: **POST**
* URL: /shopping\_cart
* Headers: Content-Type:multipart/form-data
* Body:

goodsID 1231

userID 112322

number 2

#### Response

* Body

{

"code": 200

}

### 查看购物车

#### Reuqest

* Method: **GET**
* URL: /shopping\_cart
* Headers:
* Body:

#### Response

* Body

{

"code": 200,

"data":[{

"goodsID":"111",

"category":"水果",

"name":"火龙果",

"word\_description":"12adfsfsdf",

"image":"http://1.1.1.1/2323.jpg",

"price":60.5,

"number":5,

"total\_amount":302.5

},{

"goodsID":"112",

"category":"🐟",

"name":"鲤鱼",

"word\_description":"12adfsfsdf",

"image":"http://1.1.1.1/2323.jpg",

"price":60.5,

"number":5,

"total\_amount":302.5

}],

}

### 删除购物车中的商品

#### Reuqest

* Method: **DELETE**
* URL: /shopping\_cart
* Headers: Content-Type:multipart/form-data
* Body:

goodsID 1231

userID 112322

number 2

#### Response

* Body

{

"code": 200

}

## 4.购物账单

### 清算购物车，产生购物账单

#### Reuqest

* Method: **POST**
* URL: /shopping\_bill
* Headers: Content-Type:application/json
* Body:

{

"goods":[{

"goodsID":"1231",

"number":5

},{

"goodsID":"1111",

"number":6

}],

"total\_amount":2220,

"userID":"122",

"shipping\_addressID":"122"

}

#### Response

* Body

{

"code": 200

}

### 获取购物账单

#### Reuqest

* Method: **GET**
* URL: /shopping\_bill
* Headers: Content-Type:multipart/form-data
* Body:

userID 11111

#### Response

* Body

{

"code": 200,

"data":[{

"shipping\_billID":"23134",

"goods":[{

"name":"火龙果",

"number":5,

"price":"60",

"total\_amount":300

},{

"name":"火龙果",

"number":5,

"price":"60",

"total\_amount":300

}],

"total\_amount":2220,

"shipping\_address":"北京市朝阳区××街道××小区5号楼201"

},{

"shipping\_billID":"23134",

"goods":[{

"name":"火龙果",

"number":5,

"price":"60",

"total\_amount":300

},{

"name":"火龙果",

"number":5,

"price":"60",

"total\_amount":300

}],

"total\_amount":2220,

"shipping\_address":"北京市朝阳区××街道××小区5号楼201"

}],

}

## 5.收货地址

### 添加收货地址

#### Reuqest

* Method: **POST**
* URL: /shipping\_address
* Headers: Content-Type:multipart/form-data
* Body:

userID 11111

city 北京

area 朝阳区

street ××街道

details ××小区5号楼201

#### Response

* Body

{

"code": 200

}

### 查看收货地址

#### Reuqest

* Method: **GET**
* URL: /shipping\_address
* Headers: Content-Type:multipart/form-data
* Body:

userID 11111

#### Response

* Body

{

"code": 200,

"data":[{

"shipping\_addressID":"1232",

"city":"北京",

"area":"朝阳区",

"street":"××街道",

"details":"××小区5号楼201"

}.{

"shipping\_addressID":"1232",

"city":"北京",

"area":"朝阳区",

"street":"××街道",

"details":"××小区5号楼201"

}]

}

### 删除收货地址

#### Reuqest

* Method: **DELETE**
* URL: /shipping\_address
* Headers: Content-Type:multipart/form-data
* Body:

userID 112322

shipping\_addressID 23423

#### Response

* Body

{

"code": 200

}

## 数据设计

数据字典：

**user**

| **字段** | **类型** | **注释** |
| --- | --- | --- |
| userID | String | 主码 |
| username | String | 用户名 |
| password | String | 密码 |
| phone | String | 手机号 |
| image | String | 头像的地址 |

**shipping\_address**

| **字段** | **类型** | **注释** |
| --- | --- | --- |
| shipping\_addressID | String | 主码 |
| city | String | 城市 |
| area | String | 区 |
| street | String | 街道 |
| details | String | 详细地址 |
| userID | String | 外键 |

**goods**

| **字段** | **类型** | **注释** |
| --- | --- | --- |
| goodsID | String | 主码 |
| category | String | 城市 |
| name | String | 名称 |
| word\_description | String | 商品的文字描述 |
| price | Float | 价格 |
| remaining | Integer | 剩余量 |

**shopping\_cart**

| **字段** | **类型** | **注释** |
| --- | --- | --- |
| shopping\_cartID | String | 主码 |
| goods | JSON | {商品:count} |
| userID | String | 名称 |

**shopping\_cart**

| **字段** | **类型** | **注释** |
| --- | --- | --- |
| shopping\_cartID | String | 主码 |
| goods | JSON | {商品:count} |
| total\_amount | Float | 总金额 |
| shipping\_addressID | String | 外键 |
| userID | String | 外键 |