# Web 项目1-day01-课堂笔记



# 美多商城电商项目简介

## 1.1 【简单了解】项目介绍和需求分析

美多商城是一个B2C(企业对个人)模式的电商网站，旨在为个人用户提供一个方便快捷的网上购物渠道。其功能主要分为如下几个部分：

- 用户部分：注册、登录、个人中心、地址管理、订单查看
- 商品部分：网站首页、商品列表页、商品详情页、商品搜索
- 购物车部分：购物车记录的增、删、改、查
- 订单部分：商品结算、订单提交、订单支付、订单评论

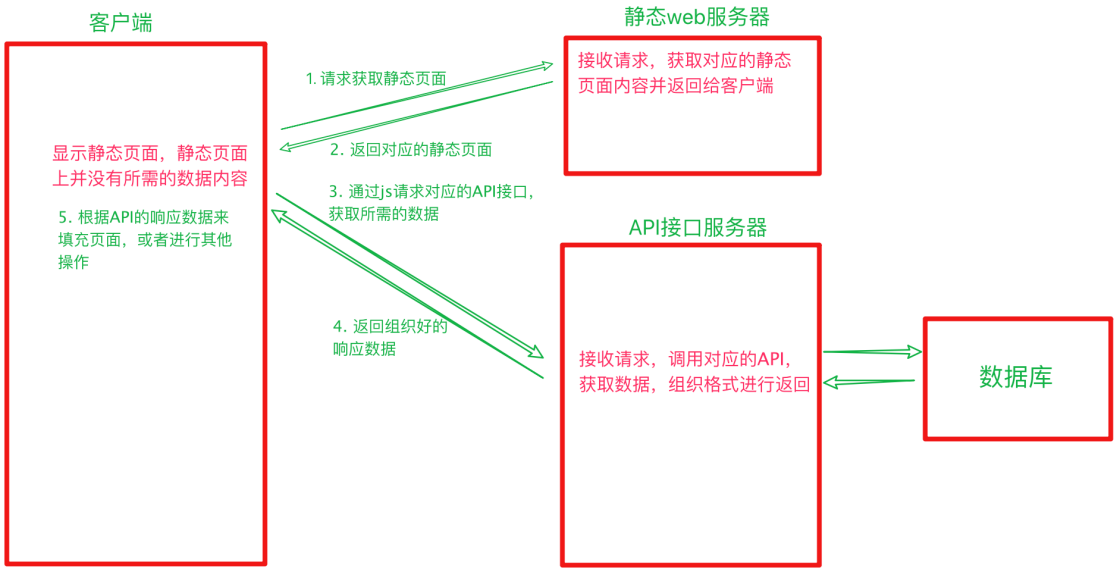
**项目的生命周期**：


## 1.2 【简单理解】美多商城项目架构设计

**项目架构设计**：

| 要点 | 说明 |
| --- | --- |
| 开发语言 | python |
| 开发模式 | 前后端分离 |
| 前端框架 | vue.js |
| 后端框架 | Django |
| 开发数据库 | mysql+redis |
| 关键技术 | celery异步、es搜索、第三方API对接... |

**前后端分离开发模式**：




## 1.3 【熟练常握】美多商城前端页面查看

**美多商城静态服务**：

作用：静态 Web 服务器，提供静态页面。

启动：

```
# 进入自己的虚拟环境
workon meiduo_site
# 注意：首先一定要进入 front_end_pc 目录，再执行下面的命令
python -m http.server 8080
```

**本地域名的设置**：

当我们采用域名访问网站时，会经过DNS的解析过程，DNS解析就是根据域名去查询对应的IP，最终根据IP去访问对应的网站。实际项目上线时，需要提前购买域名并进行备案，然后将域名和web主机IP配置绑定，那么用户才可以通过域名访问到对应的网站。

开发时如果想在自己本机模拟通过域名访问对应的服务，需要进行本地域名的设置：

1）liunx或Mac机器编辑本机的 `/etc/hosts` 文件，添加本地域名设置：

> 进行DNS解析之前，会先查找本地的hosts文件进行域名解析，查询不到才会进行DNS解析

```
# 将 www.meiduo.site 域名对应的 IP 地址设置为：127.0.0.1
127.0.0.1 www.meiduo.site
```

> 注：windows机器需编辑 c:\windows\system32\drivers\etc\hosts文件

# 美多商城电商项目框架搭建

## 2.1 【理解常握】美多商城项目代码仓库和项目创建

具体步骤参考项目讲义

## 2.2 【理解常握】美多商城项目配置文件调整

具体步骤参考项目讲义

## 2.3 【理解常握】美多商城项目模板文件目录配置

具体步骤参考项目讲义

## 2.4 【理解常握】美多商城项目 mysql 数据库配置

具体步骤参考项目讲义

## 2.5 【理解常握】美多商城项目 caches 和 session 存储配置

具体步骤参考项目讲义

## 2.6 【理解常握】美多商城项目日志存储配置

具体步骤参考项目讲义

# 用户模块-用户注册功能

## 3.1 【简单了解】RestAPI 接口设计风格

RestAPI 是一种接口设计风格的建议，目的是统一不同开发人员 API 接口的设计风格，使设计的接口风格更加相似。

**接口设计过程**：

```
设计接口的请求方式和URL地址
设计接口的请求参数和传递形式
设计接口的响应数据、格式和响应状态码
```

**关键点**：

```
要点1：URL地址尽量使用名词复数，不要使用动词。
要点2：访问同一个URL地址，采用不同的请求方式，代表要执行不同的操作。
要点3：过滤参数可以放在查询字符串中进行传递。
要点4：针对不同操作，服务器向用户返回不同的响应数据。
要点5：服务器返回的响应数据格式，应该尽量使用JSON。
要点6：服务器向客户端返回的状态码和提示信息。
要点7：如果接口出错，需要给客户端返回对应的错误信息。
```

## 3.2 【理解掌握】users 子应用创建和搜索包目录添加

为方便多个子应用的统一管理，将子应用统一创建在内层 meiduo_mall 下的 apps 目录中。调整了子应用的存放目录之后，再按之前的方式注册子应用会出错，提示找不到子应用路径，需要将 apps 目录添加到当前项目的 sys.path 中，即当前项目的 python 包搜索目录列表。

## 3.3 【理解掌握】自定义 User 模型类并生成用户数据表

注册用户信息保存，需要在数据库中创建一个用户数据表。Django 框架内置的 `django.contrib.auth` 子应用下其实有一个 User 模型类，迁移之后即可在数据库中生成一张用户表，但是其缺少 mobile 字段，我们需要将内置的 User 模型类进行替换。

**替换内置的 User 模型类**：

1）在 users 子应用 models.py 文件中，自定义 User 模型类：

```
from django.db import models
from django.contrib.auth.models import AbstractUser


class User(AbstractUser):
    """用户模型类"""
    # 增加 mobile 字段
    mobile = models.CharField(max_length=11, unique=True, verbose_name='手机
号')

    class Meta:
        # 表名
        db_table = 'tb_users'
        verbose_name = '用户'
```

2）在 dev.py 配置文件中，设置 AUTH_USER_MODEL 配置项

```
# 替换auth子应用中内置的User模型类
# 固定格式：AUTH_USER_MODEL = '子应用名.模型类名'
AUTH_USER_MODEL = 'users.User'
```

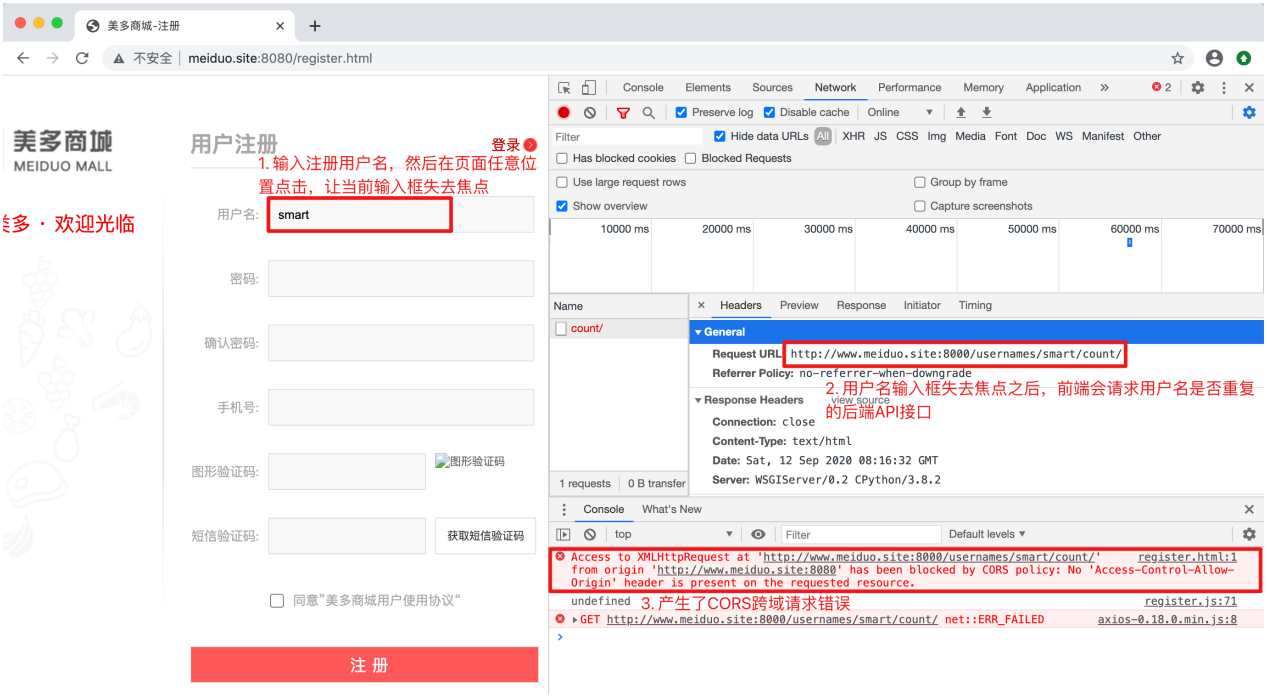
## 3.4 【理解掌握】用户名是否重复注册接口实现

**业务逻辑：**


**API接口设计：**

```
API: GET /usernames/(?P<username>[a-zA-Z0-9_-]{5,20})/count/
请求参数：
    通过url地址传递username用户名
响应数据：
    {
      "code": "响应码",
      "message": "提示信息",
      "count": "用户名数量"
    }
响应举例：
    {
      "code": 0,
      "message": "OK",
      "count": 1
    }
```

## 3.5 【理解掌握】CORS 跨域请求限制错误




**基本概念**：

1）同源地址

对于两个不同的url地址，如果其协议、IP和PORT完全一致，这样的地址就叫同源地址，否则就叫非同源地址。 例如：

```
http://www.meiduo.site:8080/register.html
http://www.meiduo.site:8000/usernames/smart/count/
```

2）跨域请求

当浏览器发起跨域请求时，如果`源请求页面地址`和`被请求地址`不是同源地址，**那么这个请求就是跨域请求**。

对于上面演示的跨域请求错误：

```
源请求地址：http://www.meiduo.site:8080/register.html
被请求地址：http://www.meiduo.site:8000/usernames/smart/count/
```

3）CORS跨域请求限制

浏览器在发起 异步跨域请求 时，默认会有CORS跨域请求限制。浏览器在请求头中携带 `Origin` 请求头，表明源请求地址，服务器在返回响应时，如果允许源请求地址对其进行跨域请求，需要在响应头中携带 `Access-Control-Allow-Origin` 响应头，浏览器在收到响应时，如果发现响应头中没有Access-Control-Allow-Origin响应头，浏览器会直接将请求驳回，产生CORS跨域请求限制错误。

**CORS限制错误解决**：

在前后端分离项目开发中，后端API服务需要允许静态web服务对其发起跨域请求，我们可以借助`django-cors-headers` 扩展包，在后端API服务器项目中添加跨域白名单设置即可。

## 3.6 【理解掌握】手机号是否重复注册接口实现

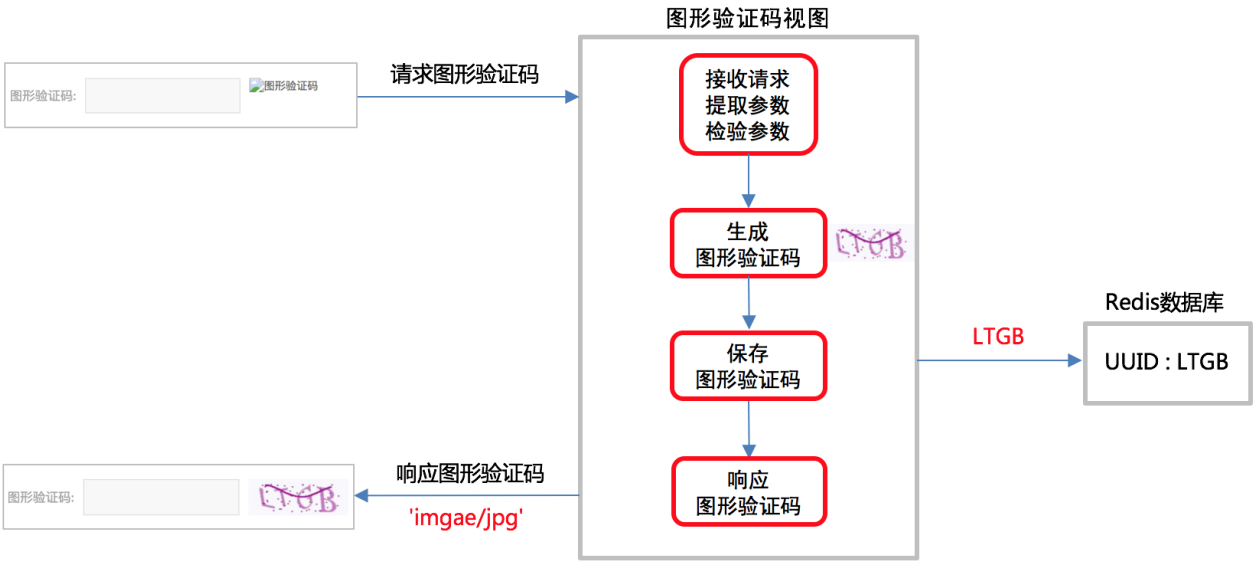
**业务逻辑**:


**API接口设计**:

```
API: GET /mobiles/(?P<mobile>1[3-9]\d{9})/count/
请求参数:
    通过url地址传递mobile手机号
响应数据:
    {
      "code": "响应码",
      "message": "提示信息",
      "count": "手机号数量"
    }
响应举例:
    {
      "code": 0,
      "message": "OK",
      "count": 1
    }
```

## 3.7 【理解掌握】图片验证码数据生成接口实现

**业务逻辑**:




**API接口设计**：

```
API: GET /image_codes/(?P<uuid>[\w-]+)/
请求参数：
    通过url地址传递唯一标识uuid
响应数据：
    验证码图片数据
```