# 设计模式实验报告

## 一、C/S模式

### 1.思路分析

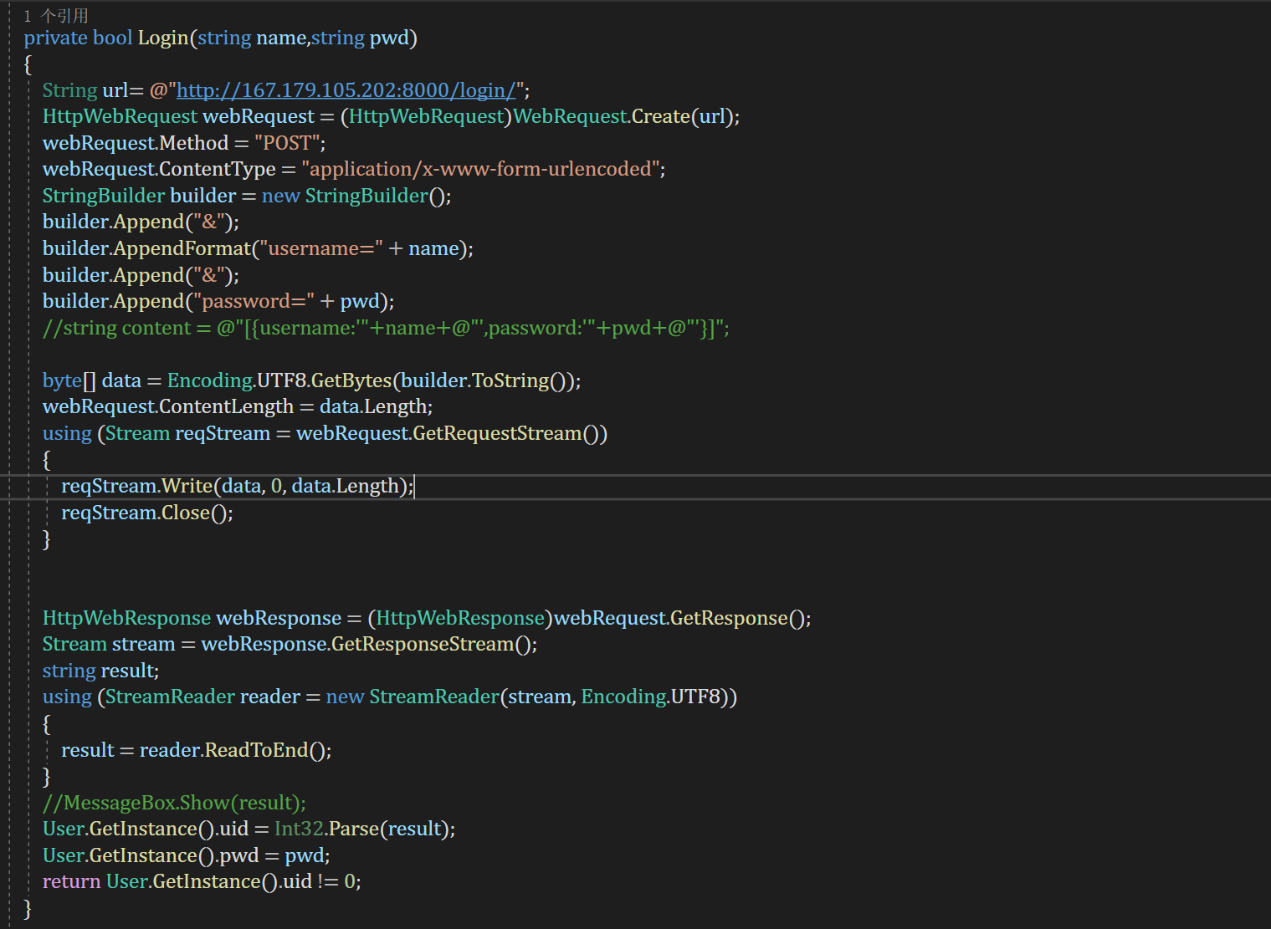
由于我们的软件是搭建用户之间交流的平台，因此，客户端/服务器模式十分符合这一项目。采取前后端分离的方式，前端使用C#窗体编写，后端使用Django框架构建。服务器与客户端通过ftp传送数据。

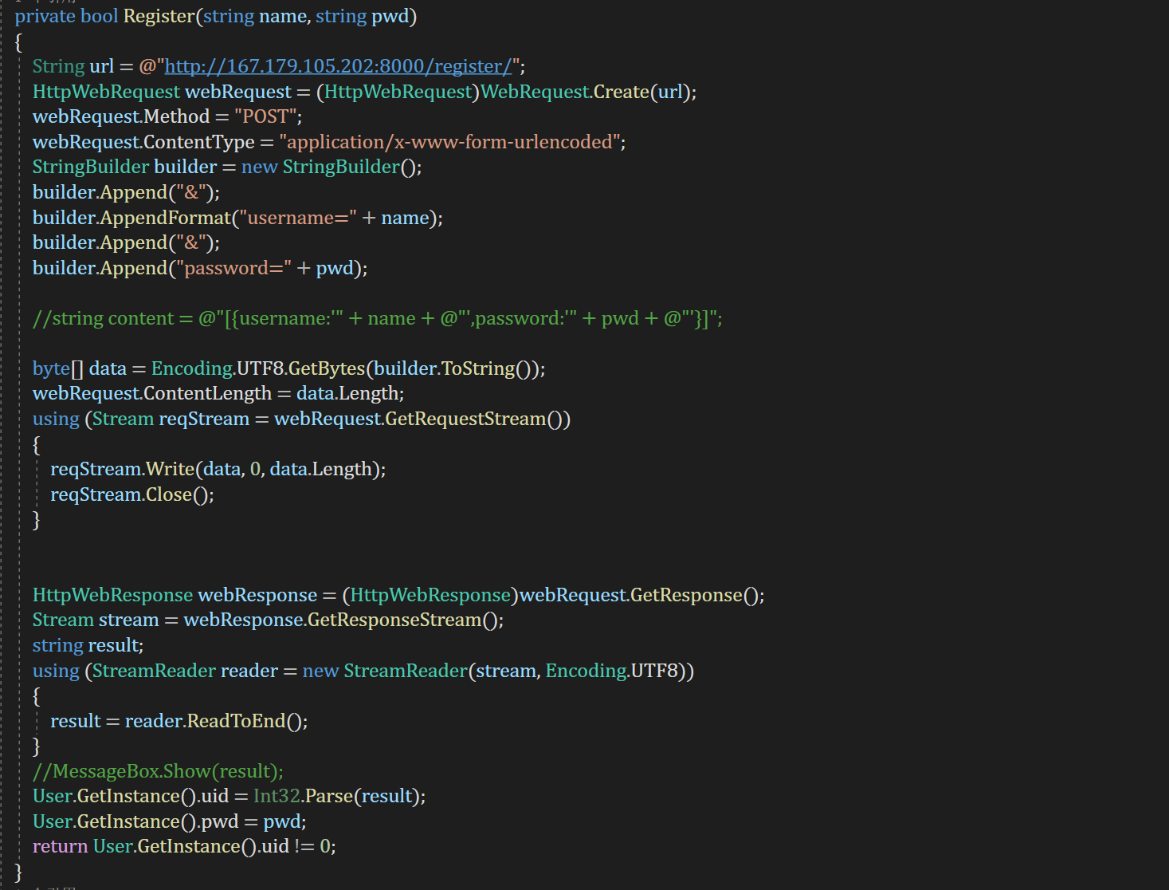
这些数据包括客户端向服务器发送的数据及服务器返回的数据。前者包含用户登录或注册时传递的账号密码信息、用户发布问题内容、用户评论内容等一系列用户进行的操作。后者包含比对数据库后返回的操作提示信息等。

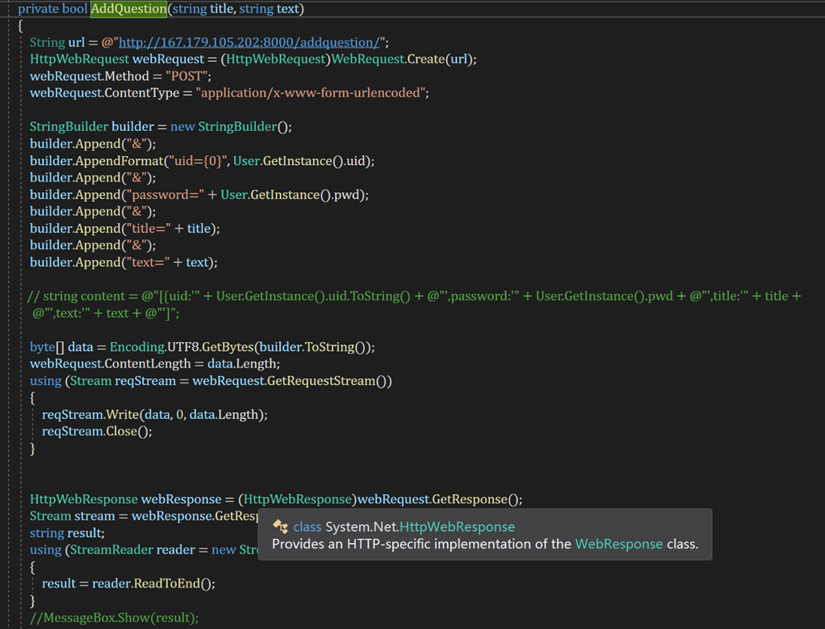
### 2.代码展示

#### 客户端

登录逻辑

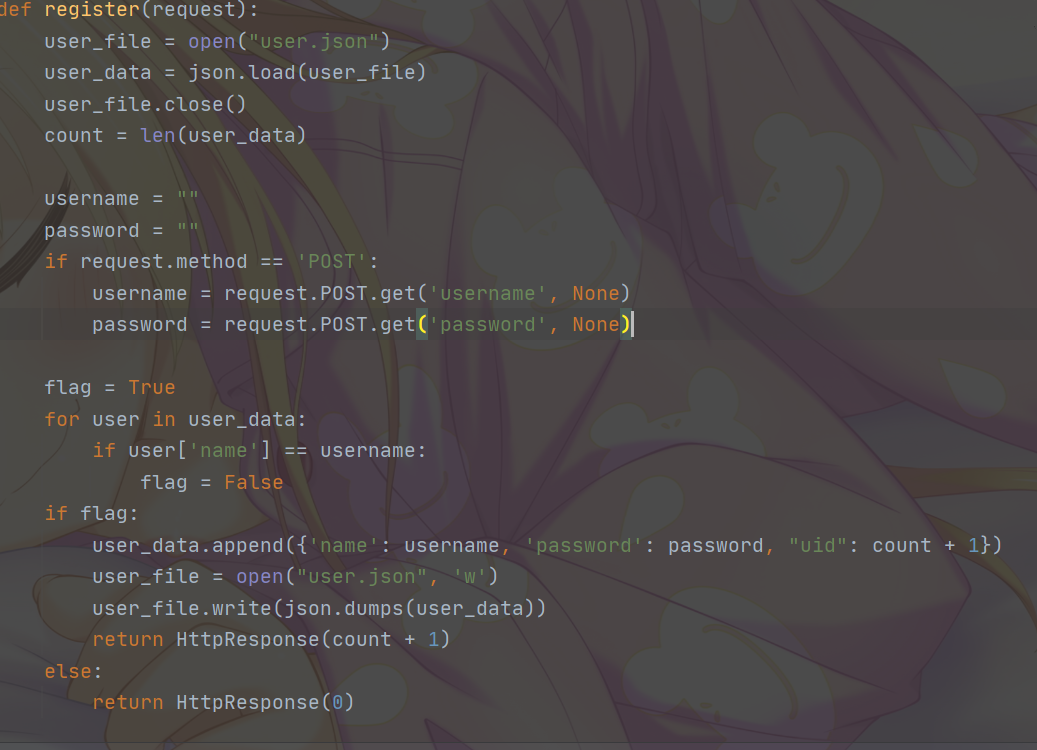


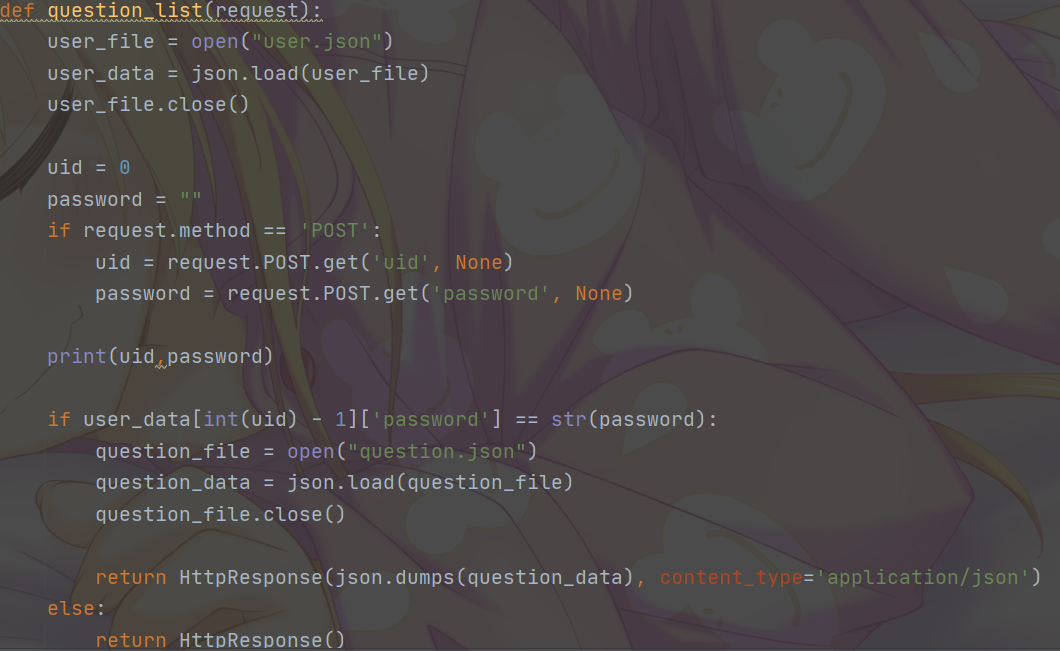
注册逻辑

发布问题逻辑

#### 服务器

登录逻辑

注册逻辑

问题列表

提交问题逻辑

### 3.模式优势

1、结构简单，客户端与服务器分别承担不同的响应任务。

2、易于编写，前后端可采用不同的语言，方便合作。

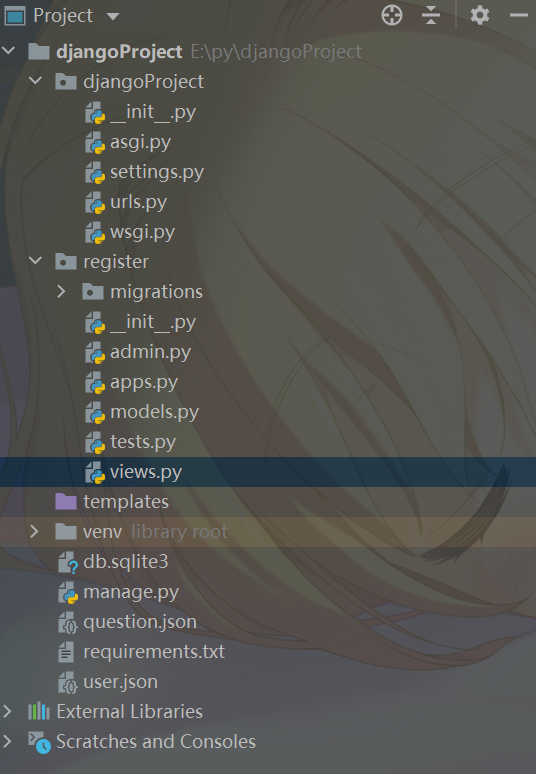
3、服务器管理系统数据库资源，用户仅获得接口的视图，维护了系统的安全性 。

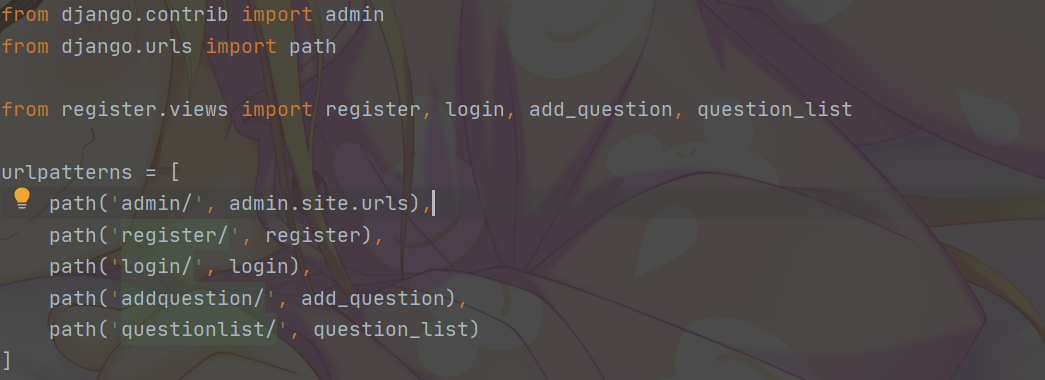
## 二、MVC模式

### 1.思路分析

要想实现多用户的平台，并维护他们的数据安全。首先要向用户呈现合适的操作界面，然后要对用户传递信息进行相应的处理,并根据用户要求来访问并操作数据库。根据这些需求我们可以在服务器端使用Django框架。合理利用了该框架中的MVC模式，即Model（模型）、View（视图）和Controller（控制）。

### 2.代码展示

Django框架，图中djangoProject文件夹内包含一些框架的基础设置。Register则是一个组件，实现服务端的具体功能，代码已在cs模式中进行了展示。

一些设置

### 3.模式优势

1. 低耦合，模型，视图，控制器的模式保持了各组件间的低耦合度。
2. 开发快捷，运用构建好的框架，同时也体现了高可重用性、部署方便的优点。
3. 维护成本低，由于代码耦合度较低，且分三层实现用户的需求，因此易于修改代码与维护数据。