# Python05

**前后端分离项目** 和 **前后端不分离项目**

* 前后端分离: **目前主流**
  + 面临的问题: **跨域**
  + 优点: 加载速度快, 降低服务器压力
  + 缺点: 不利于SEO优化--搜索优化
* 前后端不分离: **之前主流**
  + java , php 的常见做法.
  + 优点: SEO优化更好
  + 确定: 加载速度慢, 服务器压力大

## ORM

**ORM** 对象关系映射: 是一种使用 面向对象的类 来操作数据库的方式; 取代了SQL;

**出现原因**:

数据库的品牌有很多种-- mysql, sql server, sqlite, sql anywhere, oracle, mongodb...

这些数据库都是用 SQL 进行操作, 但是细节上有所不同.

**面临的挑战**:

如果项目更换数据库, 则SQL需要微调. 微调所有sql 工作量极大!

**解决**:

ORM, 把数据库语句使用 面向对象进行封装. 然后利用面向对象进行数据库操作, 底层会根据不同的数据库 转化成不同的 sql 语句.

### 模块安装

python需要安装 mysqlclient 模块才能拥有 ORM 操作mysql的功能

有极大概率无法自动安装成功, 则可以使用**手动方式**

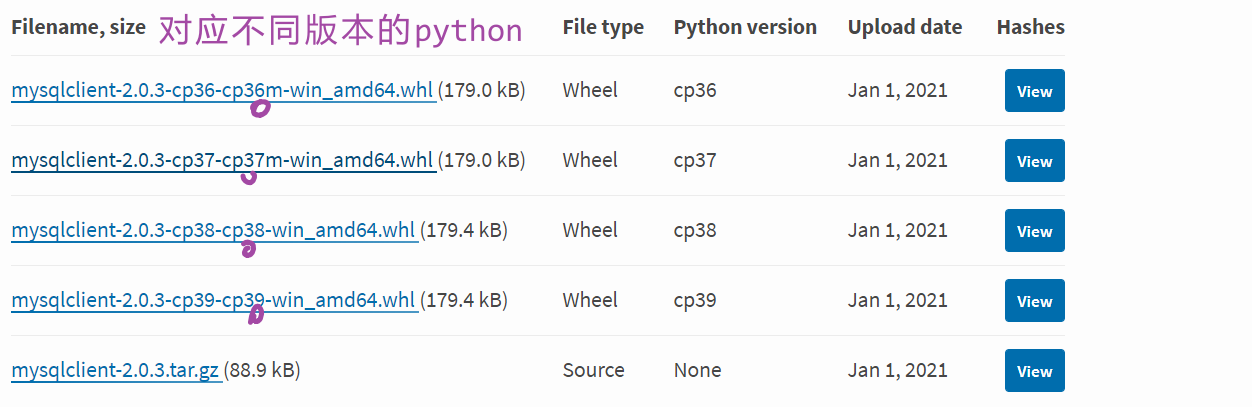
pip install mysqlclient

查看: pip list

#### 手动下载

https://pypi.org/project/mysqlclient/#files

使用py -V 查看python版本



安装方式: 文件下载之后所在的目录下, 打开cmd, 执行 pip install xxxx.whl



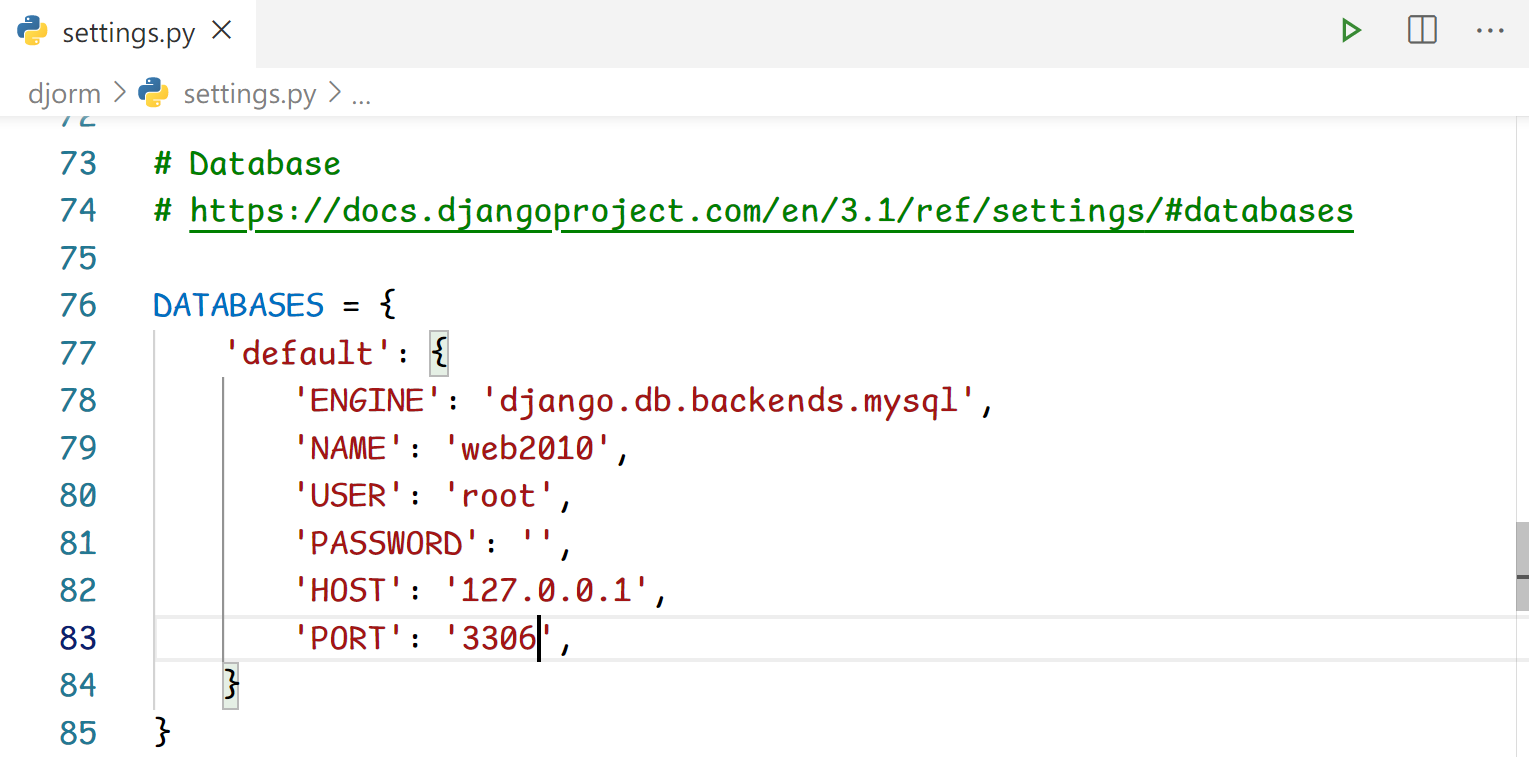
image-20210326091711670

新建项目包: django-admin startproject djorm

### 配置项

ORM不需要sql语句就可以操作数据库, 其中搭建数据库链接的操作 由系统自动完成;

我们需要把数据库的连接相关配置进行书写.



ORM的引入, 要求项目更加规范:

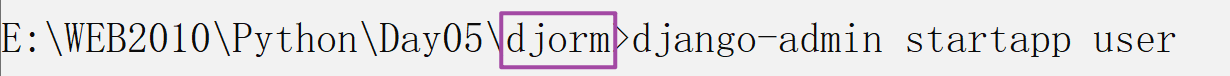
通常一个项目要分很多个模块: 广告 用户 商品 购物车 ...

不同的模块分开书写

创建模块的命令行: 必须在**项目下**执行

django-admin startapp 模块名

例如: django-admin startapp user



配置文件中, 加载子模块

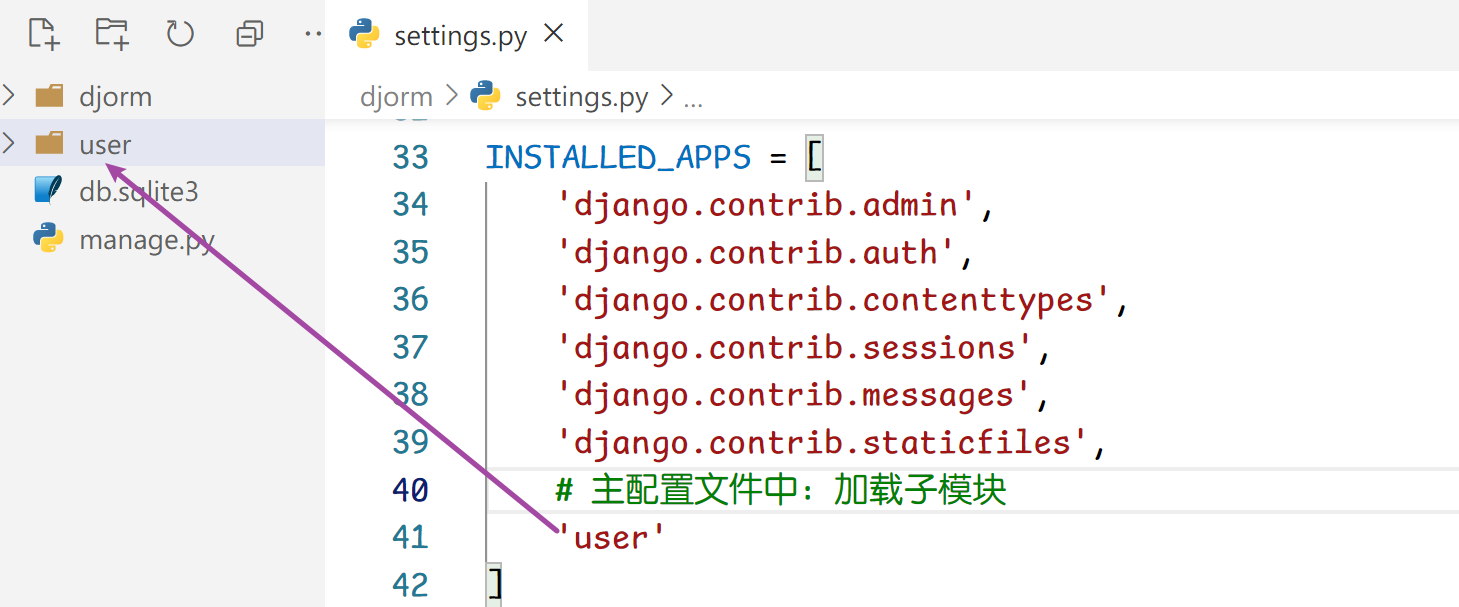


image-20210326093625410

models.py

from django.db import models
  
   
# Create your models here.
  
   
# ORM 对象关系映射
  
'''
  
程序员发现 面向对象写法 和 表结构非常相似
  
   
表名 类名
  
表字段 类属性
  
表数据 实例化对象
  
   
例如: 用户表 User
  
CREATE TABLE IF NOT EXISTS user(
  
 id INT UNSIGNED PRIMARY KEY AUTO\_INCREMENT,
  
 name VARCHAR(255) NOT NULL,
  
 age TINYINT UNSIGNED NOT NULL,
  
 phone CHAR(11) NOT NULL UNIQUE,
  
 email VARCHAR(255) NOT NULL UNIQUE,
  
 gender TINYINT UNSIGNED NOT NULL,
  
) CHARSET=utf8
  
'''
  
   
# 实现表->类的转化
  
   
   
class User(models.Model):
  
 # 必须继承父类, 才是合格的ORM类
  
 # id: 由于主键是必备的, 所以父类中已经默认具备此属性. 子类无需书写
  
 name = models.CharField(max\_length=255, null=False)
  
 # 数字类型 都自带无符号特性
  
 age = models.SmallIntegerField(null=False)
  
 phone = models.CharField(max\_length=11, null=False)
  
 email = models.CharField(max\_length=255, null=False)
  
 gender = models.SmallIntegerField(null=False)
  
   
   
'''
  
需要通过 django 提供的命令, 让 ORM 参考 User 类 生成对应的表
  
   
$ 是linux的风格, 代表接下来的是命令行
  
   
检测是否有 ORM 类变更
  
$ py manage.py makemigrations
  
   
把检测的变更应用到数据库
  
$ py manage.py migrate
  
   
'''

ORM生成表命令, 必须在项目下执行, 即 manage.py 文件所在目录

需要通过 django 提供的命令, 让 ORM 参考 User 类 生成对应的表
  
$ 是linux的风格, 代表接下来的是命令行
  
   
检测是否有 ORM 类变更
  
$ py manage.py makemigrations
  
   
把检测的变更应用到数据库
  
$ py manage.py migrate

