11.8

数据库设计

用户管理：

用户ID，用户名，用户密码，用户邮箱，用户手机号，用户头像，用户权限

可以通过用户名/手机号/邮箱+密码登录，或者通过手机号+验证码登录

博文管理：

博文ID，发布时间，发布用户ID，博文标题，博文内容，点赞数，回复数，浏览量，分类

博文由用户发布，删除，评论，修改；其中删除，修改仅限自己发布的内容

评论管理：

评论ID，评论时间，点赞数，发表用户ID，评论文章ID，评论内容，父评论ID

评论可以被发表，评论，以及被自己删除

分类管理：

分类ID，分类名称，父分类ID

关注管理：

用户ID，关注的人ID

留言管理：

被留言者ID，留言者ID，留言内容，留言ID

网站搭建过程：

安装python，Django等，通过运行django-admin startproject blog 创建项目

在阿里云上安装mysql：

$ sudo dnf install @mysql

#@mysql模块将安装MySQL及其所有依赖项。

#安装完成后，通过运行以下命令来启动MySQL服务并使它在启动时自动启动：

$ sudo systemctl enable --now mysqld

#要检查MySQL服务器是否正在运行，请输入：

$ sudo systemctl status mysqld

在服务器上安好之后，需要在服务器本地登录，修改可访问权限，允许远程登录

在云服务器安全组设置里开通3306端口

11.9

Icp申请备案

通过pymysql，orm建立项目需要的数据库内容：

修改settings里的databases部分；

\_\_init\_\_.py文件引入pymysql，创建实例

创建新模块：

Django-admin startapp account

加入settings的installed\_apps

在account.models里建立表结构

python操作数据库：

命令行运行python3 manage.py migrate

Python3 manage.py makemigrations account

Python3 manage.py migrate account

<https://www.runoob.com/wp-content/uploads/2020/05/Django-orm_6.png>

<https://www.runoob.com/wp-content/uploads/2020/05/Django-orm_7.png>

出现个问题，花了好些时间解决：

先是把managed设置为了False，导致没能在数据库中创建表。

改过来之后需要通过把migration表中app名为account的行删掉，然后再重新运行migrate命令。

但是在删除的过程中是通过右键后选删除的操作删除的，却没有在删除之后点击apply，导致没有删除成功，重新migrate指令时显示no changes detected。

好在最后解决了。

项目准备通过token进行权限控制，做成一个中间件进行使用

使用drf写接口，进行前后端分离开发。

11.10

看celery，准备使用celery实现任务异步化，定时任务

安装drf，配置

11.11

创建admin超级用户createsuperuser

可以在admin.py文件中注册模型，以便在后台管理

创建serializer.py文件，创建类，继承rest\_framework的serializers.ModelSerializer，进行序列化

编写api视图：

学习wsgi，asgi，websocket相关内容

学习计算机操作系统课程

11.12

学习计算机操作系统

学习gil，并发，协程，多线程，多进程等相关内容；

11.13

使用drf的@api\_view

使用格式后缀，format\_suffix\_patterns

使用基于类的视图

Generics.\*类简洁代码

改变返回字段的格式：

使用serializerMethodField，

使用嵌套序列化器；

反序列化时进行验证：

字段级别验证使用validate\_<field\_name>;

对象级别验证使用validate(self,data);

使用验证器，自定义函数，然后在字段中设置validator=[funcname]；

在内部的class Meta上声明验证器，如UniqueValidator，UniqueTogetherValidator等

一个序列化器创建两个模型对象：

重写serializer里的create，update方法

权限设置：

Permissions.\*各种自带权限类

自定义权限类，创建permission文件，

class IsOwnerOrReadOnly(permissions.BasePermission):

def has\_object\_permission(self, request, view, obj):

if request.method in permissions.SAFE\_METHODS:

return True

return obj.author == request.user

可以在setting里通过DEFAULT\_PERMISSION\_CLASSES全局设置默认权限

如果函数api，使用@permission\_classes()

如果你在生产环境下使用BasicAuthentication和TokenAuthentication认证，你必须确保你的API仅在https可用。

11.16

准备登陆权限功能换用drf自带的权限系统