# 《中间件技术课程报告》

姓名：孙远远

学号：2018104139

## 基于Flask框架的在线视频网站系统

### 一、Flask背景分析和介绍

Flask是一个使用 Python 编写的轻量级 Web 应用框架。其 WSGI 工具箱采用 Werkzeug，模板引擎则使用Jinja2。Flask使用 BSD 授权。

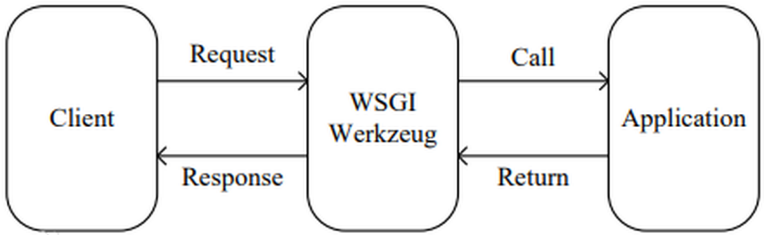
Flask也被称为 “microframework”，因为它使用简单的核心，用 extension 增加其他功能。Flask没有默认使用的数据库、窗体验证工具。

**简介**

Flask是一个轻量级的可定制框架，使用Python语言编写，较其他同类型框架更为灵活、轻便、安全且容易上手。它可以很好地结合MVC模式进行开发，开发人员分工合作，小型团队在短时间内就可以完成功能丰富的中小型网站或Web服务的实现。另外，Flask还有很强的定制性，用户可以根据自己的需求来添加相应的功能，在保持核心功能简单的同时实现功能的丰富与扩展，其强大的插件库可以让用户实现个性化的网站定制，开发出功能强大的网站。

**基本模式**

Flask的基本模式为在程序里将一个视图函数分配给一个URL，每当用户访问这个URL时，系统就会执行给该URL分配好的视图函数，获取函数的返回值并将其显示到浏览器上，其工作过程见图。



IT运维的基本点为安全、稳定、高效，运维自动化的目的就是为了提高运维效率，Flask[9]开发快捷的特点正好符合运维的高效性需求。在项目迭代开发的过程中，所需要实现的运维功能以及扩展会逐渐增多，针对这一特点更是需要使用易扩展的Flask框架。另外，由于每个公司对运维的需求不同，所要实现的功能也必须有针对性地来设计，Flask可以很好地完成这个任务

**特点**

目前，我国市场上大部分智能交通系统控制平台采用的都是C/S模式，对终端要求较高，且安装繁琐。部分的平台也有采用基于B/S模式传统框架，但这些框架的一些功能大多被固定，缺乏灵活性。采用Flask微型框架做服务器的后台开发，Flask是基于Python开发的框架，类似的框架还有Django、Tornado等，之所选择Flask来开发，原因如下：

一、后续的基于机器学习的车辆检测与属性识别算法研究，主要开发语言也是应用Python，整个系统统一开发语言，便于开发和后期维护。

二、Flask因为灵活、轻便且高效的特点被业界认可，同时拥有基于Werkzeug、Jinja2等一些开源库，拥有内置服务器和单元测试，适配RESTful，支持安全的cookies，而且官方文档完整，便于学习掌握。

三、Flask中拥有灵活的Jinja2模板引擎，提高了前端代码的复用率。这样可以提高开发效率和有利于后期开发与维护。在现有标准中，Flask算是微小型框架。Flask有两个主要依赖：路由、调试和Web服务器网关接口（WebServerGatewayInterface，WSGI）子系统由Werkzeug提供；模板系统由Jinja2提供。Werkzeug和Jinja2都是由Flask的核心开发者开发而成。对于数据库访问、验证Web表单和用户身份认证等一系列功能，Flask框架是不支持的。这些功能都是以扩展组件的方式进行实现，然后再与Flask框架集成。开发者可以根据项目的需求进行相应的扩展，或者自行开发。这与大型框架恰恰相反，大型框架本身做出了大部分决定，难以灵活改变方案。

### 二、用到的Flask知识

1.使用整形，浮点型，路径型，字符串型下正则表达式路由转化器

2.使用GET与POST请求，上传文件，cookie获取与响应，404处理

3.使用模板自动转义，定义过滤器，定义全局上下文处理器，Jinja2语法，包含，继承，定义宏

4.使用flask-wtf定义表单模型，字段类型，字段验证，视图处理表单，模板使用表单

5.使用flask-sqlachemy定义数据库模型，对数据进行增删查改，数据迁移

6.使用蓝图优化项目结构，实现微电影网站前台与后台业务逻辑

7.flask部署，安装编译Nginx,安装Mysql服务以及通过Nginx反向代理对视频流媒体限制下载速率，限制单个IP能发起的播放连接数

### 三、用到的视频技术

* jwplayer播放器插件
* 视频限速限IP访问
* FLV,MP4视频格式支持
* Nginx点播实现

**四、环境说明**

开发系统：Win 10(与数据库虚拟机以NAT方式连接，IP地址为：192.168.81.3)

python版本：3.6.3

开发工具：pycharm 专业版

数据库：mysql

**五、项目模型设计**

使用Flask的蓝图Blueprint规划项目结构

使用Flask sqlalchemy定义和业务需求相关的数据库模型

结合MySQL数据库生成数据表

**前端搭建**

实现前后台html布局页面搭建

实现Jinja2引擎语法

引入静态资源文件，404错误页面的处理

**后端开发**

Flask sqlalchemy结合MySQL数据表进行增删改查操作

Flask数据分页查询，路由装饰器定义，模板中变量调用，登录会话机制，上传文件

Flask wtforms表单验证，Flask自定义应用上下文，自定义权限装饰器对管理系统进行基于角色权限的访问控制

Flask的多表关联查询，关键字模糊查询等

**网站部署**

实现在win10系统上搭建Nginx+MySQL+python环境

使用Nginx反向代理多端口多进程部署微电影网站

配置Nginx流媒体访问限制参数

**五、前后台项目目录分析：**

**前台(home):**

数据模型：models.py

表单处理：home/forms.py

模版目录：templates/home

静态目录：static

**后台(admin):**

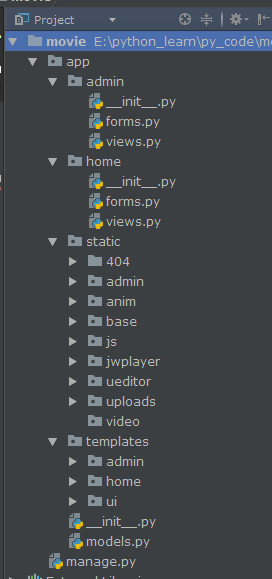
数据模型：models.py

表单处理：admin/forms.py

模版目录：templates/admin

静态目录：static

整个项目目录如下图所示：



**六、使用蓝图构建项目目录**

**什么是目录**

一个应用中或跨应用制作组件和通用的模式,类似于Django中的app

**蓝图的作用**

将不同的功能模块化

构建大型应用

优化项目结构

增强可读性，易于维护

**蓝图构建项目目录**

**定义蓝图**

在home目录下的\_\_init\_\_.py文件中

from flask import Blueprint

home=Blueprint('home',\_\_name\_\_)

import app.home.views

在admin目录下的\_\_init\_\_.py文件中

from flask import Blueprint

admin=Blueprint('admin',\_\_name\_\_)

import app.admin.views

**注册蓝图**

修改app目录下的\_\_init\_\_.py文件

from app.admin import admin as admin\_blueprint

from app.home import home as home\_blueprint

app.register\_blueprint(home\_blueprint)

app.register\_blueprint(admin\_blueprint, url\_prefix='/admin')

**调用蓝图**

home目录下的views.py文件

from . import home

@home.route("/register/", methods=["GET", "POST"])

def register():

pass

@home.route("/login/", methods=['GET', 'POST'])

def login():

pass

admin目录下的views.py文件

from . import admin

@admin.route('/')

def index():

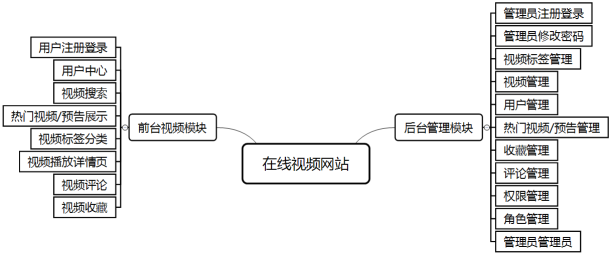
pass

@admin.route('/login/', methods=['GET', 'POST'])

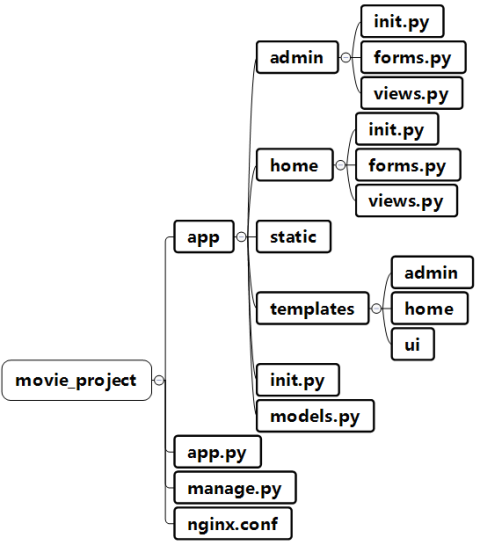
def login():

pass

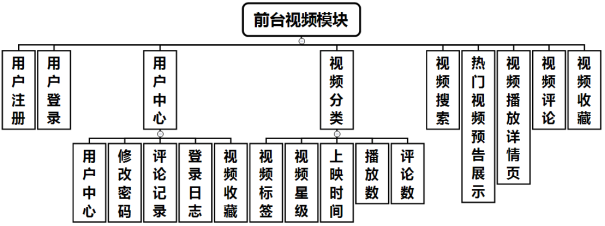
**七、结构图展示**



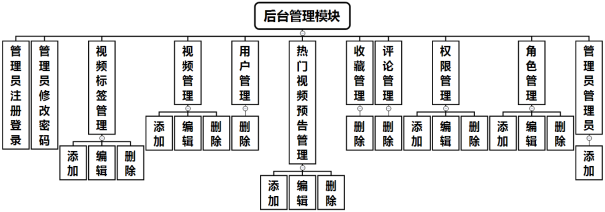
7.1整体架构设计图



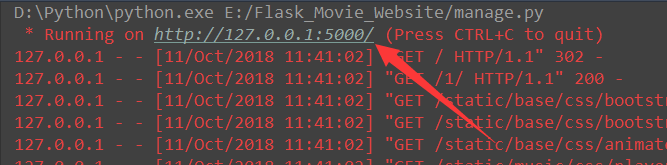
7.2项目目录结构图



7.3前台功能模块图



7.4后台功能模块图



7.5本地运行图