课堂点名服务系统

目录

一．可行性研究报告

1. 引言

1.1 报告撰写的目的

课堂服务系统是针对当下点名占用课堂时间的问题而设计，是一个可以初步完成教师与学生注册、登录功能的系统。

此可行性研究报告，是为实现在最短的时间内以最小的代价确定设计该系统的问题是否可以解决，从而确定进一步对系统进行开发的流程步骤。

1.2 项目背景

为杜绝学生逃课，不少教师会采用课堂点名的方法。这种方法效率较低，且会极大地占用课堂的时间。因此需要开发一个可以节约课堂时间的在线点名系统。

2. 可行性分析

2.1 系统基本要求

2.1.1 采用框架

利用 Django 采用的 MVC 框架模式，使用 python、html、css 搭建后台。

2.1.2 主要功能要求

主要分为三大模块功能：用户注册、用户登录、系统管理。

(1) 用户注册功能：首次使用的用户进行注册。

(2) 用户登录功能：已注册的用户再次使用时需登录。

(3) 系统管理功能：完成用户管理和系统维护功能。

2.2 系统开发要求

2.2.1 网站实现

开发周期：计划时期 3 天、学习实践阶段 7 天、具体开发测试阶段 7 天、修改完善阶段：5 天

2.2.2 环境搭建

开发采用 Django 内置服务器和 SQLite 数据库引擎

2.2.3 费用开支与效应分析

(1) 环境搭建采用开源免费的平台

(2) 代码开发，由于项目为个人练习项目，规模较小，旨在实现基本功能，因袭所需开发与维护人员仅为本人。

综上，各项费用开支基本为 0，同时项目为非盈利网站，纯经济效益为 0。

2.3 其他因素的可行性分析

2.3.1 技术可行性分析

网络上有关创建网站和制作网页的教程很多并且很详细，因此开发该项目的学习资源很丰富。

2.3.2 操作可行性

主要考虑时间精力问题。进行可行性分析时为本学期第 3 周，其他专业相关课程还没有进入完成大作业的阶段。与大约一个月后的期中考试阶段以及后半学期的

完成大作业的阶段相比，最近几周相对有充足的时间来完成项目。

2.4 可行性分析总结

通过上述可行性分析，该项目具备进一步进行需求分析与后续开发的条件。

二．需求规格说明书

1.项目功能需求

1.1 基本需求分析

本系统主要实现用户登录/注册功能。考虑到后续开发的可能性，在条件允许的情况下，还可以考虑扩大规模（如使用专业数据库系统）和教师友好（期末学生到课情况邮件发送）功能。不过本阶段以实现基本功能为主（考虑到开发周期的因素）。

1.2 系统设计方法

系统为浏览器用户提供登录登出页面，为开发者提供管理页面。

层次	职责
模型（Model）,即数据存取层	处理与数据相关的所有事务：如何存取、如何验证有效性、包含哪些行为以及数据之间的关系等。
视图（View）,即表现层	处理与表现相关的决定：如何在页面或其他类型文档中进行显示
模板(Template),即业务逻辑层	存取模型及调取恰当模板的相关逻辑,模型与模板之间的桥梁

1.3 功能性分析

网页浏览、用户注册、用户登录、管理员导出数据开发环境、硬件及软件的需求

2.非功能性需求

2.1 方便操作，操作流程合理

尽量从用户的角度出发，以方便使用本产品。

2.2 控制必录入项

本系统能够对必需录入的信息进行控制，使用户能够确定信息录入的完整。同时对必录入项进行有效的统一的提示。

2.3 用户操作手册

系统提供用户操作手册，提供熟练使用本系统所有操作的指南。

三．概要设计说明书

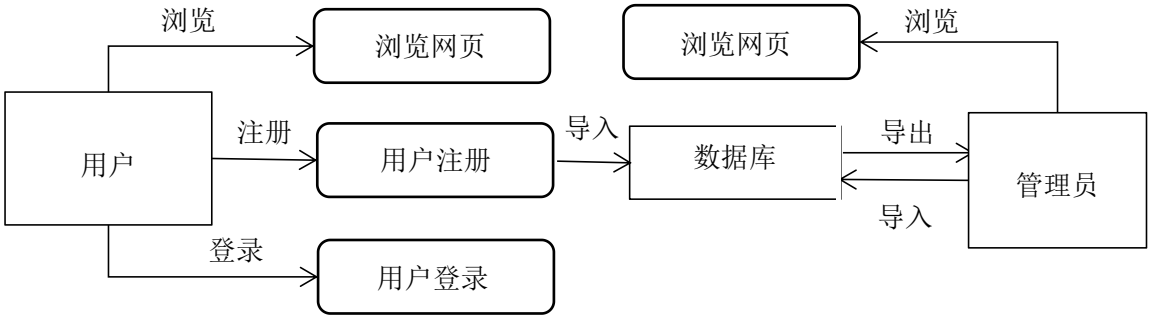
1.开发环境、硬件及软软件的需求

开发工具：PyCharm
编程语言：Python、HTML
软件与硬件需求：本地客户机普通 PC、使用 Google Chrome 进行调试
Web 服务器：Django 内置服务器

2.系统的结构化功能建模、分层数据流图

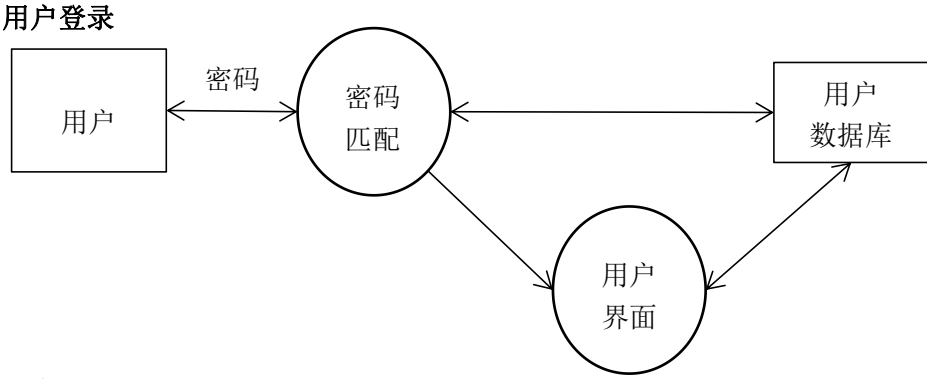
(1) 顶层数据流图

(2) 一层数据流图

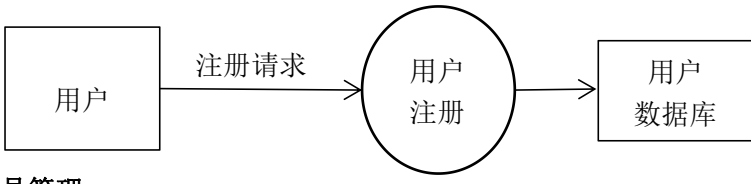


(3) 二层数据流图

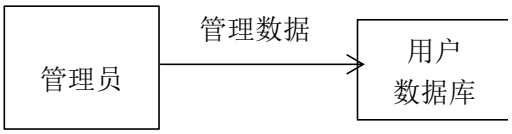
i. 用户登录



ii. 用户注册



iii. 管理员管理



3. 数据字典描述

(1) 顶层数据字典

实体名称	数据流	简述
用户	浏览信息、登录、注册	用户可以浏览网页内容，可以注册登录
管理员	浏览信息、导入导出信息	管理员可以发起对数据的管理操作

(2) 一层数据字典

系统各功能名称	对应实体	数据流	简述
浏览网页	用户	浏览信息	用户浏览网站，并记录下访客的浏览信息，如 IP 地址等
登录注册	用户	登录注册（输入信息）	用户可以注册并登录网页
浏览站点	管理员	浏览信息	除了可以浏览网页内容外，还可以浏览站点后台页面，网站信息
管理数据	管理员	数据信息管理	管理员可以看到用户的注册列表并导出注册用户名单

(3) 二层数据字典

i. 用户登录

加工名	输入数据流	输出数据流	加工逻辑
密码匹配	用户输入的密码	匹配成功、匹配失败	与数据库中存储的用户密码相同则匹配成功，否则匹配失败，用户要登录则要重新输入密码。

ii. 用户注册

加工名	输入数据流	输出数据流	加工逻辑
注册申请	用户输入自定的用户名和密码（重复输入两次）	数据库中的用户名和密码	确认用户输入无误后将用户的信息导入数据库

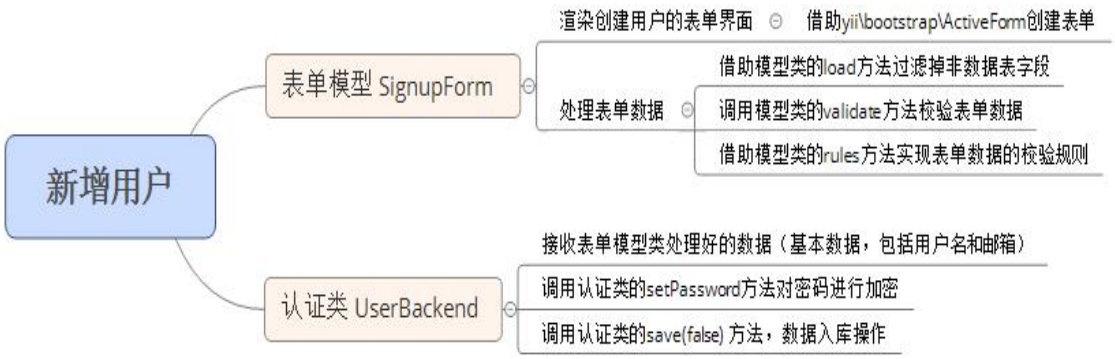
iii. 管理员管理

加工名	输入数据流	输出数据流	加工逻辑
添加用户	用户名和密码（重复输入两次）	数据库中的用户名和密码	将用户的信息导入数据库
删除用户	用户名	--	将用户的数据从数据库中移出
信息管理	管理信息事务、网站信息、站长信息	网站信息、站长信息	站长发起管理事务，可对自身或网站信息进行添加、修改、删除等操作，并应用到系统网站页面上

4. 系统行为建模、状态图示例

以该系统部分行为为例进行结构行为建模：

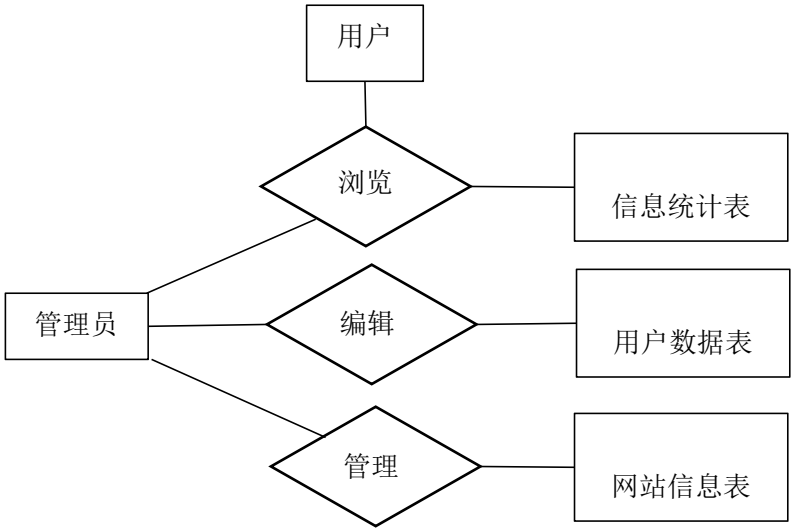
(1) 用户注册



(2) 用户登录

5. 系统数据建模、E-R 模型图、关系模式表示

5.1 总体 E-R 模型图：



5.2 关系模式表示

管理员信息（登录名，密码，管理员说明）
主码：登录名 唯一
外码：登录名

网站信息（序号，站名，URL）
主码：序号 唯一
外码：序号

四．详细设计说明书

1．系统结构

模块划分：

前端 UI：系统界面部分，负责接收用户输入，显示系统输出。
后端:

用户部分，实现用户注册和网页登录功能。
管理员部分，实现数据库管理及用户相关信息导出。

数据库：SQLite

2. 模块设计说明

2.1 模块一：前端 UI

2.1.1 模块描述

实现用户界面的文件，表现为 templates 文件夹内的 5 个 HTML 文件。

2.1.2 功能

负责接受用户输入、显示系统输出的前端网页，并其他模块间的协调调用。

2.1.3 交互的模块

My_login/login/templates/views.py 引用

HTML 文件中链接外部式.css 文件

2.1.4 模块设计

该模块中的主要文件以及功能交互分析如下：

- index.html:用户登录成功后返回的页面
- regist.html：用户注册页面，需要输入用户注册的相关信息
- share.html：用户申请注册后返回的信息页面（注册失败/注册成功）
- login.html：用户登录页面，需要用户输入用户名以及密码
- logout.html：用户登出页面

交互分析（My_login/login/templates/views.py 文件中与之关联的相关函数）：

- def index(request):

  ……

  #登录成功

  return render_to_response('index.html',{})
- def regist(request) :

  ……

  #返回相关信息页面

  return render_to_response('share.html',{})
- def share(request):

  ……

  #根据注册是否成功返回相应信息

  return render_to_response('share.html',{})
- def login(request):

  ……

  #登录成功

  return response

  #信息比较未成功

  return HttpResponseRedirect('/login/')
- def logout(request):

  ……

  return response

2.1.5 其他

模块编写者：NHY

模块编写日期：2018-4
模块修订日期：2018-4
模块测试者：NHY

2.2 模块二：后端 Django 框架

2.2.1 模块描述
后端框架，django 模板内的 PY 文件+db.sqlite3 数据库
2.2.2 功能
后端网页框架文件，调用前端的 HTML 文件。
2.2.3 交互的模块
前端 HTML 文件
2.2.4 模块框架及设计
该部分的主要文件为 django 框架中的 PY 文件：

```
My_login/
├── db.sqlite3              #框架自动创建的数据库
├── login                   #新建的应用文件夹
|   ├── migrations
|   |   ├── 0001_initial.py
|   |   ├──__init__.py
|   ├── templates           #模板目录，需要自己创建
|   |   ├── index.html
|   |   ├── regist.html
|   |   ├── share.html
|   |   ├── login.html
|   |   ├── logout.html
|   ├── __init__.py
|   ├── admin.py
|   ├── apps.py
|   ├── models.py           #在此创建项目的用户类  class User(models.Model)
|   ├── tests.py
|   └── views.py            #视图处理脚本，主要修改的文件,包含 index(request)、
|                           regist(request)、share(request)、login(request)、
|                           logout(request)函数
├── My_login                #整个项目的配置文件夹
|   ├── __init__.py
|   ├── settings.py         #全局设置脚本，在 INSTALL_APPS 添加应用 login
|   ├── urls.py             #在此配置项目相关的 url
|   └── wsgi.py
├── manage.py               #项目管理脚本，用于创建应用和启动项目
```

2.2.5 其他
模块编写者：NHY
模块编写日期：2018-4

模块修订日期：2018-4
模块测试者：NHY

2.3 模块三：SQLite 数据库相关

2.3.1 模块描述
进程内的软件库
2.3.2 功能
用于存储网页用户的相关信息及导出相关信息
2.3.3 交互的模块
django 后端框架，PY 文件中有相关接口
2.3.4 模块接口及连接数据库

```
import sqlite3
conn = sqlite.connect('Users.db')
```

2.3.5 模块内相关操作

```
#创建表
    c = conn.cursor()
    c.execute("'CREATE TABLE USERS()'")
    #用户信息
    create Users
    (
        User_name               int auto_increment not null,
        User_password           varchar(20) not null,
        User_id                 numeric(20) not null,
        User_sex                varchar(2) not null,
        User_email              varchar(50) not null,
        User_register           date not null,
        primary key (User_name)
    )
    #课程信息
    create table Course
    (
        Course_id               varchar(20) not null,
        Course_Stu_number       smallint(4) not null,
        Course_Stu              varchar(20) not null,
        Course_time             datetime not null,
        Course_teacher          varchar(20) not null,
        primary key (Course_id),
    )
#插入新数据
c.excute(''INSERT INTO USERS ()\VALUES()")
#导出数据库中的内容（终端操作）
```

>.output User.db

>.dump

2.3.6 其他

模块编写者：NHY

模块编写日期：2018-4

模块修订日期：2018-4

模块测试者：NHY