第 5 章

CHAPTER 5

智慧照明云平台设计

智慧照明云平台在端、边、云结构中属于工业互联网的应用层，如图 5-1 所示，它可实现实验箱内大功率 LED 的管理、各本地控制器和主控制器的管理，为资源管理和运维管理提供数据。

5.1 智慧照明云平台架构

架构设计是从需求分析到软件实现的桥梁，也是决定软件质量的关键。本教材实现的智慧照明云平台架构是一种典型的物联网云平台。物联网云平台的系统架构采用的是 MVC 思想，它包括应用分层 Model(模型)、View(视图)和 Controller(控制器)3 个基本部分，这 3 个部分以最少的耦合协同工作，从而提高了应用的可扩展性及可维护性。MVC 要求应用分层，产品的应用通过模型体现。

物联网云平台的架构如图 5-1 所示。

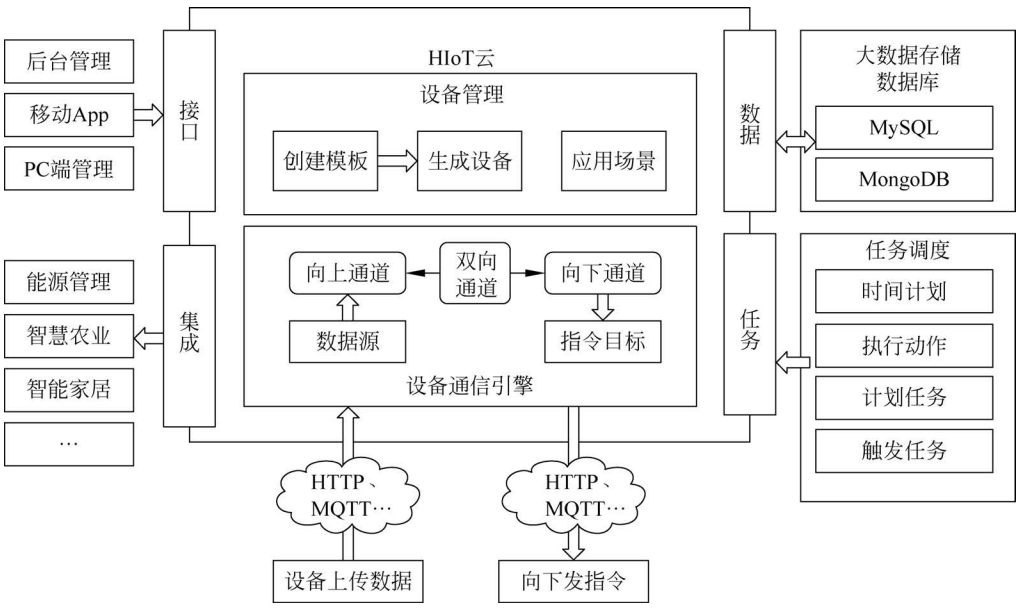

图 5-1 物联网云平台架构

5.1.1　物联网云平台架构简介

1. 物联网云平台的架构本质

架构的本质是呈现三大能力：系统如何面向最终用户提供支撑能力、如何面向外部系统提供交互能力、如何面向企业数据提供处理能力。

1）面向最终用户提供支撑能力

访问数据的窗口是表现层，用于展示与接收数据。物联网云平台的访问数据窗口有后台管理、移动 App 和 PC 端管理，它为客户提供交互的页面。PC 端的管理一般由前端工程师完成，移动 App 由安卓工程师完成。该系统的后台管理可以提供功能性接口，供兴趣爱好者在物联网云平台上进行二次开发，而移动 App 端提供给普通用户使用，普通用户通过 App 控制已有的智能设备并展示数据。移动 App 还可以提供不同场景下的组合控制设备组，组合控制又可以分为定时控制和一键控制等功能，使用户体验达到更加智能的状态。

PC 端管理是给智能设备开发者及智能设备生产商使用的，PC 端管理分为用户模块、设备模块等。用户模块的主要功能是注册登录、修改密码和找回密码等；设备模块则包括模板的创建、设备的创建、设备通道的生成、设备数据的显示以及设备持有者的 CRUD 等功能。

2）面向外部系统提供交互能力

物联网云平台是开发者使用的平台，开发者也可以通过该平台将开发完成的智能设备的信息传送给 PC 端或者移动 App，这样 PC 端或者移动 App 就可以绑定设备，从而控制智能设备。设备的所有数据都将经过物联网云平台进行存储和显示，物联网云平台提供 PC 端或者移动 App 获取设备上传信息及下发控制设备信息的接口。PC 与物联网云平台是通过 HTTP 进行交互的，而物联网云平台与智能硬件是通过 HTTP 或者 MQTT 协议进行通信的。

3）面向企业数据提供处理能力

物联网云平台通过订阅向上通道获取不同设备各种类型的数据，并能整理数据，开发者可以查看正在上传数据的智能设备，如果某些设备上传了预警信息或是触发事件的信息，开发者可以直接预先设置设备，同时，物联网云平台将预警消息发送给智能设备使用者的 PC 端或移动 App；利用物联网云平台中的仪表盘，通过输入设备信息，可以查看某个设备上传的数据及下发的数据。

2. 物联网云平台系统架构的优势

1）简化开发

物联网云平台架构减少了用户开发项目的工作量，用户无须构建复杂的网络，无须重构主机处理器代码，无须开发后台软件，无须学习特殊的编程或脚本语言，因此，开发难度较低，同时也降低了项目失败的风险。

2）加速产品上市

物联网云平台架构为用户节省了开发时间，加速了连接设备和移动 App 的开发。它可以连接设备和移动 App 并可独立连接到云端的抽象端点，这样在系统集成阶段就可减少很多问题。

3）降低成本

相比企业内部模式，物联网云平台架构按需索取平台处理能力，降低了建设成本。

本章根据物联网系统架构分析云平台的功能需求，并根据物联网的特点对云平台中的

架构设计进行介绍。根据云平台需求设计其分区架构及技术架构，并确定每个分区的功能及需要解决的问题。

物联网是利用各种传感设备将物体接入互联网，借助互联网实现物与物之间的信息交互和数据共享，达到对物品的智能化识别、定位、监控和管理的目的，最终实现人与人、人与物、物与物之间的便捷通信。

本书根据云平台的功能模块将云平台进行分区设计，设计模块分为服务器及后端程序模块设计、Web 模块设计。

服务器及后端程序模块是终端接入的通道入口，为用户端和设备端提供终端资源管理的接口，终端资源除了包含用户端和设备端的属性信息以及绑定关系以外，还包含传感数据存储规则、设备绑定关系规则等，都是通过服务器及后端程序模块对这些资源进行增删改查的操作，并提供一套开放的接口，实现用户和设备对终端资源的管理。

Web 模块为用户端，作用是为用户端及设备端之间提供通信及数据交互的服务，实现用户对设备控制命令的下发、远程参数设置、在线状态管理以及设备采集数据信息的实时获取等。

系统架构由云平台、用户端、设备端以及提供用户端和设备端上网的网络接入层四部分构成，如图 5-2 所示。设备通过和云平台建立连接形成可以通信的网络，并借助云平台实现用户到终端设备以及终端设备之间的数据交互，下面对各部分功能进行详细介绍。

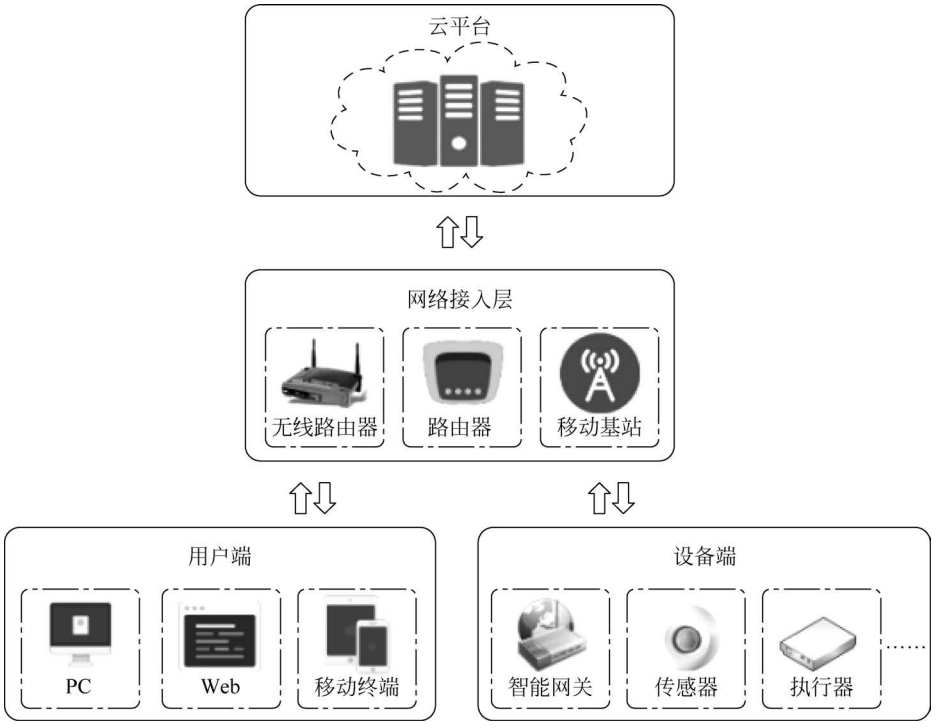

图 5-2　四层架构

云平台是为物联网系统提供终端统一管理、海量数据存储、远程即时通信、高性能计算分析等服务的数据业务中心。云平台能够实现有效的终端管理，提供终端的注册和授权管理，使用户能够快速获得设备端的信息和数据，并且能够灵活地向设备端推送消息和下发控制指令；同时，云平台能够提供高性能的数据处理及大规模数据管理服务，实现数量庞大的

终端连接的有效维护、数据可靠传输及海量数据的处理和存储。

用户端即搭载在移动设备、PC 上的应用软件，包括移动 App、网页界面、PC 软件终端程序等。用户软件是用户控制与管理云平台、硬件设备的交互工具，通过用户软件可实时掌握系统的状态，对设备进行集中化管理，实现对设备的远程控制、历史数据查询与统计、故障查询与诊断、个性化服务定制等功能。

设备端可分为节点设备、网关设备。节点设备是指能够直接与云平台建立连接的传感器、执行器等底层终端设备；网关设备是指能够将远程命令数据与现场命令数据相互转换的中间设备，网关设备能够汇聚各感知节点采集的数据，将其处理后上传至云平台，并接收云平台远程命令下发给底层设备。无论是网关设备还是节点设备都是互联网服务渗透到我们生活中各领域的智能化改造的重要成分，充分体现了物联网物物相连的特性。

接下来将分别对服务器及后端程序模块、Web 模块设计进行讲解。

5.1.2　智慧照明云平台需求分析

云平台作为智慧照明系统中的复杂数据和远端控制功能的技术核心，首先要为用户端与设备端的接入提供服务接口；其次需要为云与端、端与端之间的即时通信提供通道，同时需要实现对不同设备端产生的传感数据进行解析和存储。总结本云平台的主要功能，包括 LED 灯控制终端接入管理、终端监控设备、集控和分控三方面，下面对这三方面的需求进行详细分析。

1. 物联网终端接入管理

物联网终端包括用户端及设备端，用户端一般指一个管理主题，例如本智慧照明系统指共用一个网关的实验室，即以实验室为单位分配用户的使用权限；设备端在本书中指的是实验箱。终端接入管理包括终端的注册、绑定、授权等，终端接入管理过程必须保证物联网终端信息的安全性，以及终端之间绑定和授权过程的灵活性。下面对终端接入管理的具体要求进行详细介绍。

1）用户接入管理

用户接入管理是指智慧照明系统中的用户端获得一个合法的身份，并以此身份进行对云平台中资源的操作。为保障云平台中信息的安全，将用户角色分为“生产厂商”和“普通用户”，只有生产厂商拥有对所有终端资源的操作权限，而普通用户只可以在设备出厂后对设备进行处理和操作。不同物联网行业应用、不同的用户角色有不同操作平台的用户端需求，因此云平台要求支持不同平台的用户端的接入，提供手机号验证注册、邮箱验证注册两种账号注册方式，为不同用户端用户提供注册服务，使用户可以存储与使用物联网终端设备的数据与服务。另外，需要明确不同角色用户对云平台资源的操作权限，并实现灵活的资源授权和分享机制，保障物联网系统资源的安全性。

2）设备接入管理

设备分类管理：物联网系统中包含大量终端设备，其中某些设备隶属于同一个物联网应用系统，其功能需求和底层协议等均相同，为便于对这些相同项目中的设备进行统一的管理，将相同的系统或项目中的设备统称为“产品”。每个设备在出厂前便需要确定其功能及所属的产品，并保证设备与产品信息的从属关系不能被用户更改。用户与设备的绑定及授权管理：为实现用户对设备的管理，需要完成用户与设备信息的绑定，绑定完成后才可对设

备信息进行查看，为保障设备信息的安全，每个设备只能与唯一的用户建立绑定关系，称为设备的“管理员”，拥有对设备操作的所有权限，包括对设备的创建、获取、远程控制、数据监测、删除、分享、取消分享等操作。而其余用户需向该设备的管理员发送授权请求，授权成功后以“分享者”的角色只对设备具有一部分操作权限，包括设备信息获取、远程控制、数据监测等，而不具有创建、删除及分享的权限。

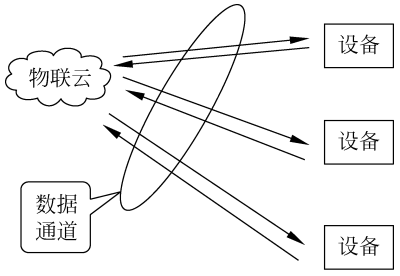

图 5-3　管理流程

2. 终端监控设备

用户与设备之间的通信实现了用户对设备的远程控制、参数设置、运行状态监测、实时数据获取等功能。物联网云平台根据设备模板批量生产设备，设备根据自身的数据通道上传数据至云平台，通过数据通道下发指令控制设备。具体管理流程如图 5-3 所示。

1）设备模板的概念

设备模板是向上提取设备的共性，并形成抽象的一类设备，我们将该抽象的一类设备定义为此类设备的模板。设备模板可用于批量生成设备。

2）设备的概念

IoT 中的 T 就是设备，是所有其他功能的基础。设备向下分配的是通道，向上整合的是场景。

3）数据通道的概念

向上通道：设备采集的数据通过向上通道上传至云端。

向下通道：云端通过向下通道推送指令、消息至设备端。

双向通道＝向上通道＋向下通道。

4）数据类型的概念

设备的上传数据是各种各样的，为方便起见，我们将数据类型分为数值型、布尔型、文本型、GPS 型 4 种类型。

3. 单控、分控和集控

本智慧照明系统支持单控、分控和集控功能。单控：对某个实验箱的某个 LED 灯实现调光、调色控制；分控：即分组控制，根据场景需要对某个实验箱或几个实验箱的单个灯或多个灯进行控制，用户可以根据系统需求将设备分到不同的组中，实现个性化控制，即要求分组能够动态更改；集控：实现对所有实验箱的所有灯的调光调色控制。

5.2　服务器及后端模块设计

本节主要介绍服务器及后端模块设计，该模块实现接收 Web 指令，并下发给智能网关的功能。本节将依次介绍服务器购买及配置、后端程序的开发流程、程序讲解。

5.2.1　服务器购买和配置

1. 注册腾讯云账号

1）官网快速注册

注册腾讯云账号：注册—腾讯云（tencent.com）。

2）按要求完成实名认证

注册完成后，登录腾讯云：登录—腾讯云（tencent.com），根据提示进行实名认证。

2. 预付费购买服务器

如图 5-4 所示，学生新用户可以使用“云＋校园 首单特惠”进行购买。

学生云服务器_学生云主机_学生云数据库_云＋校园特惠套餐-腾讯云(tencent.com)。

云产品首单秒杀活动：云产品首单秒杀_云服务器秒杀_云数据库秒杀-腾讯云(tencent.com)。

图 5-4　服务器购买流程

1）选择合适配置并付费

选择合适的套餐点击立即购买，选择活动地域、镜像、购买时长后单击“立即购买”按钮进行付费。

本书选择的实例配置是轻量应用服务器 2 核 4G，镜像是 Windows 腾讯云专享版，如图 5-5 所示。

图 5-5　配置选择

2）进入控制台

第一步，购买完成后，单击右上角“控制台”按钮，如图 5-6 所示。

图 5-6　腾讯云首页

第二步，打开腾讯云控制台后单击“云产品”按钮，单击“轻量应用服务器”按钮，如图 5-7 所示，进入控制台界面。

图 5-7　控制台界面

第三步，在轻量应用服务器控制台界面左侧导航栏中找到“服务器”，之前购买的服务器实例在页面下方显示，如图 5-8 所示。

图 5-8　服务器信息

3. 重置服务器密码

1）进入服务器

第一步，单击实例卡片，进入服务器实例控制台。

第二步，顶部导航栏选择“概要”，如图 5-9 所示。

图 5-9　服务器实例控制台

2）重置密码

第一步，在应用信息面板中，建议先手动单击“关机”按钮，再单击“重置密码”按钮，如图 5-10 所示。

图 5-10　重置密码

第二步，输入新密码，单击“下一步”按钮，如图 5-11 所示。

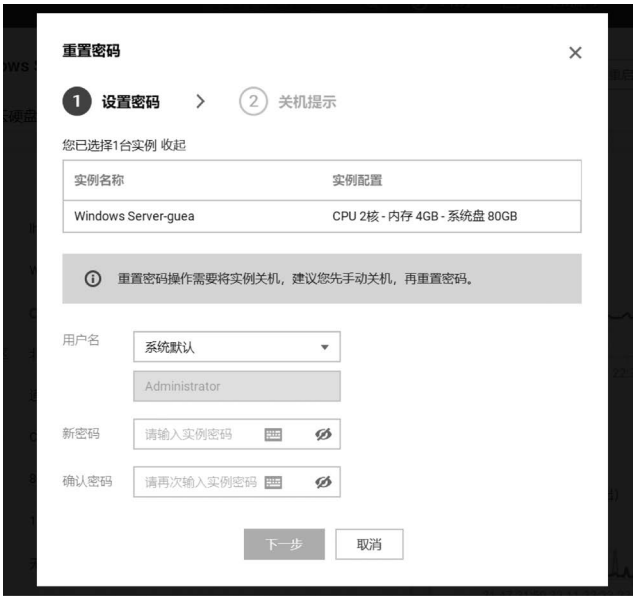

图 5-11　设置新密码

第三步，如果没有按第一步手动单击“关机”按钮，这一步勾选“同意强制关机”复选框，单击“重置密码”按钮，如图 5-12 所示。

图 5-12 “重置密码”对话框

4. 连接服务器

1）本地连接服务器

第一步，单击“连接”按钮，如图 5-13 所示。

图 5-13 远程桌面连接

第二步，单击“是”按钮，如图 5-14 所示。进入服务器图标的桌面，如图 5-15 所示。

2）进入宝塔面板首页

外网面板地址默认为服务器公网地址为：8888/tencentcloud/，也就是应用内软件信息中的面板首页地址，开启端口后，在浏览器输入外网面板地址，输入前面获取的用户名和密码，进入宝塔面板，如图 5-16 所示。

3）进入面板安装所需插件

进入面板后可以选择一键安装推荐的 LNMP 或 LAMP 套件，也可以选择自行安装，如图 5-17 所示。

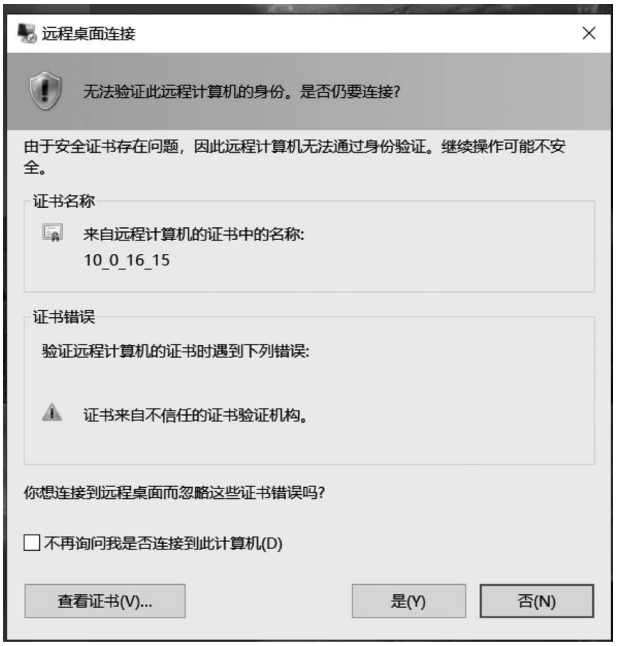

图 5-14　确认连接

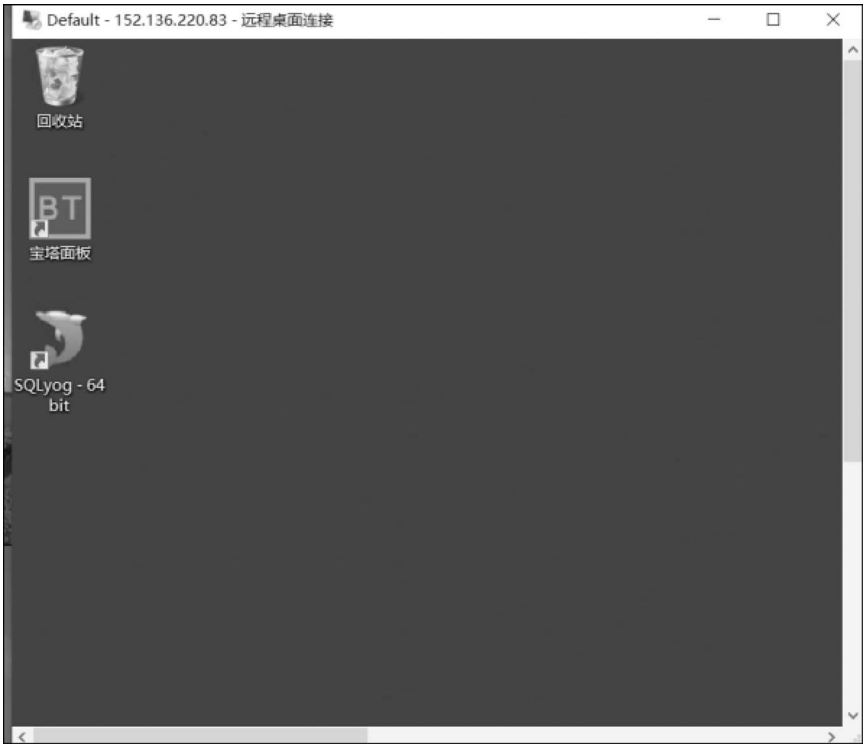

图 5-15　进入服务器图标的桌面

图 5-16　宝塔面板

图 5-17　安装套件

5.2.2　程序开发流程

基于本设备的需求，选择使用后端语言 Python、Flask 框架进行程序开发，实现接收 Web 指令，并下发给智能网关的功能。

Python 是一种解释型、面向对象、动态数据类型的高级程序设计语言。Python 由 Guido van Rossum 于 1989 年底发明，第一个公开发行版发布于 1991 年。像 Perl 语言一样，Python 源代码同样遵循 GPL(GNU General Public License)协议。该语言具有简单、易学、速度快、免费、可移植性强、可扩展性强、可嵌入性强等诸多优点。

Flask 是目前十分流行的 Web 框架，采用 Python 语言来实现相关功能。它被称为微框架(microframework)，“微框架”中的“微”并不是意味着把整个 Web 应用放到一个 Python

文件中，而是指 Flask 保持核心的简单，同时又易于扩展。

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，系统就会执行为该 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，其工作过程如图 5-18 所示。

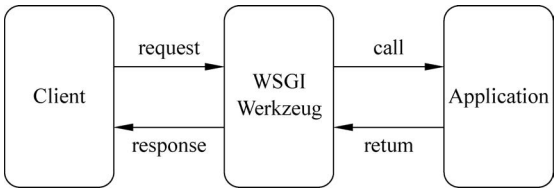

图 5-18　工作过程

本平台服务器程序可实现接收 HTTP 请求报文，解析报文，产生 request 传送给框架程序。

(1) 框架程序：接收 HTTP 请求对象 request，中间层处理(拦截请求)，路由处理，具体视图处理业务处理，再进行中间层处理(拦截响应)，返回给服务器程序的是一个 response 的对象。

(2) 服务器程序：通过 response 对象构造一个 HTTP 响应报文，再传回客户端。

功能实现逻辑简述：比如单控实验箱某个灯的开关功能，控制界面的单击按钮，JavaScript 发送请求报文，例如开灯(a)，后端接收此请求，连接智能网关，再次将指令转发给网关进行解析，实现控制开灯的功能。下面开始介绍框架搭建步骤和实现功能的代码。

第一步，下载 Flask 框架。

本平台后端程序用到的 Flask 工具包如图 5-19 所示。

图 5-19　Flask 框架的工具包

方法一：在终端命令行输入语句 pip install flask 即可下载 Flask 框架。

方法二：使用 PyCharm 开发工具新建一个解析器为 Flask 的项目文件。

第二步，导入所需的框架模块，如图 5-20 所示。

```
import socket
from threading import Thread
from functools import partial
from flask import Flask, request,redirect,url_for
from tcp_client import create_client, send_message, start_server
```

图 5-20　导入所需的框架模块

要注意下面三点。

(1) 连接智能网关部分，给定需要连接的智能网关 IP 地址，定义一个发送信息的函数，如图 5-21 所示。

```
client = create_client(host='192.168.200.2')  # 需要输入树莓派 IP
send_message = partial(send_message, client)

Thread(target=start_server, daemon=True).start()
```

图 5-21　连接智能网关

(2) 跨域访问支持(必不可少，程序固定)，如图 5-22 所示。

```
@app.after_request
def after_request(resp):
    resp.headers['Access-Control-Allow-Origin'] = '*'
    return resp
```

图 5-22　跨域访问支持

(3) Get 请求，通过装饰器设置路由，请求方式为 Get 或 Post 请求，本书采用 Get 方式。从 Web 得到命令信息，将设备号及命令下发给智能网关，如图 5-23 所示。

```
@app.route('/admin', methods=['GET'])
def get_admin():
    info_id = request.args.get("info_id")
    number = request.args.get('number', 25)
    message = f'{number},{info_id}'

    send_message(message)
```

图 5-23　Get 请求语句

5.3　Web 模块设计

本节主要介绍 Web 模块设计，Web 页面实现用户登录、首页产品介绍、设备控制等主要功能。使用基本的前端开发工具 HTML、JavaScript 等。将依次介绍前端 Web 开发流程及程序讲解。

HTML 的英文全称是 hyper text markup language，即超文本标记语言。JavaScript(简称 JS)是当前最流行、应用最广泛的客户端脚本语言，用于在网页中添加一些动态效果与交

互功能，在 Web 开发领域有着举足轻重的地位。

HTML 用来定义网页的内容，例如标题、正文、图像等。

CSS 用来控制网页的外观，例如颜色、字体、背景等。

JavaScript 用来实时更新网页中的内容，例如从服务器获取数据并更新到网页中，修改某些标签的样式或其中的内容等，可以让网页更加生动。

5.3.1　登录界面设计

用户登录模块部分程序介绍，如图 5-24 所示。

从图 5-25 可以看出登录界面有三层布局，最外层为线性布局，将内层水平和垂直都居中，内层为线性布局垂直分布，“欢迎登录”的 TextView 可以直接放在第二层，也可以单独加一层布局，用户和输入框，密码和输入框，登录和取消都是使用了相对布局。

```
    <title>智慧照明实验平台</title>
</head>
<body>

    <link href="jiemian.css" rel="stylesheet" type="text/css">
    <div class="login_box">

        <h1 class="title">欢迎登录</h1>

        <div>
            <input class="input_box" type="text" id="username" placeholder="请输入密码: " />
        </div>

        <div>
            <input class="input_box" type="password" id="passward" placeholder="请输入密码: " />
        </div>

        <div id="btns">
            <input class="button_box" type="button" value="登录" onclick="check()" />
            <input class="button_box" type="button" value="取消" onclick="reset()" />
        </div>
```

图 5-24　登录模块程序

图 5-25　登录界面

5.3.2 Web 首页设计与效果图

Web 首页设计了一项图片轮播模块，将产品（实验箱设备）介绍图进行轮播展示，并赋予文字讲解。

图 5-26 为实现 Web 首页显示的代码，图 5-27 为系统登录成功后的首页界面。

```
            <div>
                <span  class="fl"  style="width: 500px; margin-top: 45px" >边缘设备即现场智能照明硬件设备分为三层：传感器层即感知层
                    主控器层即远端管理层。本设计使用的是兆易创新GD32F3系列MCU，采用模块化设计思想，支持多种功能的传感器模块采集环境信息，
                    发送到中控器和主控制器，从而实现两层控制，即中控制器直接控制和主控制器控制。主控制器作为主控单元，发起通信，实现DALI、
                    DMX512、RS485、Zigbee和Wi-Fi这5种协议的转化，中控制器作为服务器端可以接收主控单元的命令执行并回传数据。中控制器完美地
                    体现了智能硬件的概念，可配置多种传感器，可以独立完成本终端LED灯的手动和自动调光调色控制，也可以与主控制器以及其他中控制器
                    组网，实现多种网络协议融合的调光调色功能。</span>
            </div>
        </div>
    </div>

    </body>
    <script type="text/javascript">
        var index=0;
        //效果
        function ChangeImg() {
            index++;
            var a=document.getElementsByClassName("img-slide");
            if(index>=a.length) index=0;
            for(var i=0;i<a.length;i++){
                a[i].style.display='none';
            }
            a[index].style.display='block';
        }
        //设置定时器，每隔两秒切换一张图片
        setInterval(ChangeImg,2000);
    </script>
    </html>
```

图 5-26　实现 Web 首页显示的代码

图 5-27　首页界面

5.3.3 集控功能的实现

集控功能是指一条指令可以同时控制一个实验箱的多个大功率 LED 灯，或控制多个实验箱的大功率 LED 灯。集控界面实现了对 25 个实验箱的集中控制。主要功能有：对 25

个实验箱的 RS485、WiFi、ZigBee、DMX512、DALI 模块集中控制调光、调色。

（1）集控界面设计对应的部分程序如图 5-28 所示。

```
    <div class="RS485-title" style="...">
        RS485集控按钮
    </div>
    <div class="RS485-open" style="...">
        <input id="button1" style="margin-left: 20px" type="button" value="打开设备" onclick="OpenLight(1)"/>
    </div>
    <div class="RS485-Red" style="...">
        <input id="button3" style="..." type="button" value="打开红灯" />
    </div>
    <div class="RS485-Green" style="...">
        <input id="button5" style="..." type="button" value="打开绿灯" />
    </div>
    <div class="RS485-Blue" style="...">
        <input id="button7" style="..." type="button" value="打开蓝灯" />
    </div>
    <div class="RS485-Close" style="...">
        <input id="button2" style="..." type="button" value="关闭设备" onclick="CloseLight(1)"/>
    </div>
</div>
```

图 5-28　集控程序代码

（2）向后端发送请求功能的部分程序：变量 a 的参数“01”代表集中控制；变量 number 代表实验箱编号，即 1～25 号实验箱；变量 info_id 代表控制每个灯光功能的指令，如 06 09 64 64 64 代表打开 25 个实验箱所有 RS485 模块的灯光。实验集控功能的关键代码如图 5-29 所示。

```
<script type="text/javascript">

    function send(a, number, info_id) {
        var httpRequest = new XMLHttpRequest();
        httpRequest.open('GET', `http://localhost:5000/admin?info_id=${info_id}&number=${number}&a=${a}`, true);
        httpRequest.send();
        httpRequest.onreadystatechange = function () {
            if (httpRequest.readyState == 4 && httpRequest.status == 200) {
                var json = httpRequest.responseText;
                console.log(json);
            }
        };
    }

    function OpenLight(number, info_id='06 09 64 64 64') {
        a='01';
        alert("打开");
        send(a, number, info_id)
    }
    function CloseLight(number, info_id='06 09 00 00 00') {
        alert("关闭");
        send(a, number, info_id)
    }
```

图 5-29　实现集控功能的关键代码

（3）Web 效果如图 5-30 所示，分别对应五个模块的集控功能。例如，RS485 模块可实现对 25 个实验箱里所有的 RS485 模块集中调光调色。

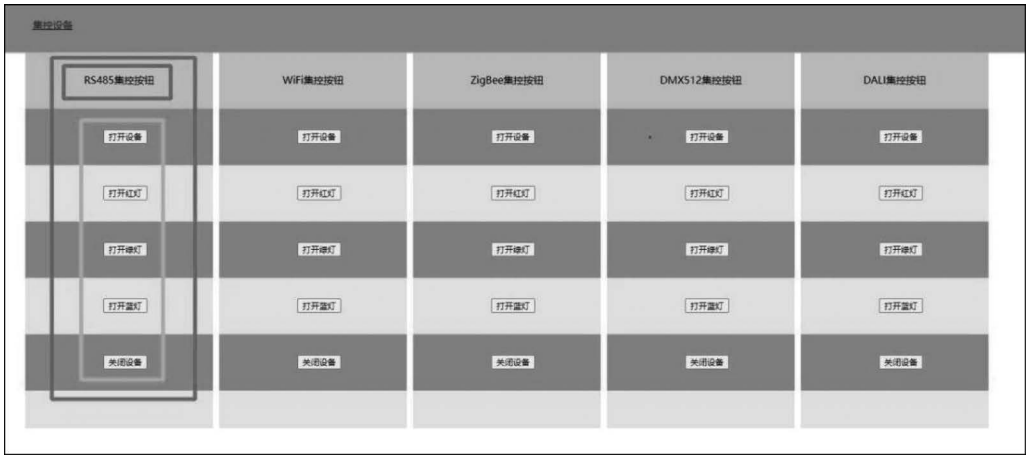

图 5-30　集控功能操作界面

5.3.4　单控功能

单控功能即一条指令只控制一个大功率 LED 灯，分别实现对每个实验箱里 RS485、WiFi、ZigBee、DMX512、DALI 模块的每一个灯光的调光、调色功能。

（1）单控界面设计对应的部分代码如图 5-31 所示。

```
<div class="one" >
    <input  style="margin-left: 60px" type="button" value="一号箱" />

    <div class="s"  style="border:solid #CCC;  width:250px; height: 70px" >
        <input style="color: #000000" type="button" value="RS485"  disabled/>
        <input id="button1" style="margin-left: 20px" type="button" value="开" onclick="RSopenLight(1)"/>
        <input id="button4" type="button" value="R" onclick="RSRedLight(1)"/>
        <input id="button2" type="button" value="G" onclick="RSGreenLight(1)"/>
        <input id="button3" type="button" value="B" onclick="RSBlueLight(1)"/>
        <input id="button5" type="button" value="关" onclick="RScloseLight(1)"/>
    </div >

    <div class="s" style="border: solid #CCC;  width:250px; height: 70px" >
        <input style="color: #000000" type="button" value="WIFI"  disabled/>
        <input id="button7" style="margin-left: 20px" type="button" value="开" onclick="WFopenLight(1)"/>
        <input id="button10" type="button" value="R" onclick="WFRedLight(1)"/>
        <input id="button8" type="button" value="G" onclick="WFGreenLight(1)"/>
        <input id="button9" type="button" value="B" onclick="WFBlueLight(1)"/>
        <input id="button11" type="button" value="关" onclick="WFcloseLight(1)"/>
    </div>
```

图 5-31　单控程序代码

（2）向后端发送请求功能的部分程序：同理，变量 a 的参数“00”代表单个控制；变量 number 代表实验箱编号，即 1～25 号实验箱；变量 info_id 代表控制每个灯光功能的指令，如 01 09 64 00 00 代表打开某个实验箱 RS485 模块的红灯，同理 01 09 00 64 00 代表打开绿灯。实现单控功能的关键代码如图 5-32 所示。

（3）Web 效果如图 5-33 所示，分别对应 25 个实验箱的单个控制功能。例如 1 号实验箱，可实现对 RS485、WiFi、ZigBee、DMX512、DALI 模块分别调光调色。R 代表打开红灯，G 代表打开绿灯，B 代表打开蓝灯。

```
<script type="text/javascript">

    function send(a, number, info_id) {
        var httpRequest = new XMLHttpRequest();
        httpRequest.open('GET', `http://localhost:5000/admin?info_id=${info_id}&number=${number}&a=${a}`, true);
        httpRequest.send();
        httpRequest.onreadystatechange = function () {
            if (httpRequest.readyState == 4 && httpRequest.status == 200) {
                var json = httpRequest.responseText;
                console.log(json);
            }
        };
    }

    function RSRedLight(number, info_id='01 09 64 00 00') {
        a='00';
        alert("打开红灯");
        send(a, number, info_id)
    }
    function RSGreenLight(number, info_id='01 09 00 64 00') {
        a='00';
        alert("打开绿灯");
        send(a, number, info_id)
    }
```

图 5-32　实现单控功能的关键代码

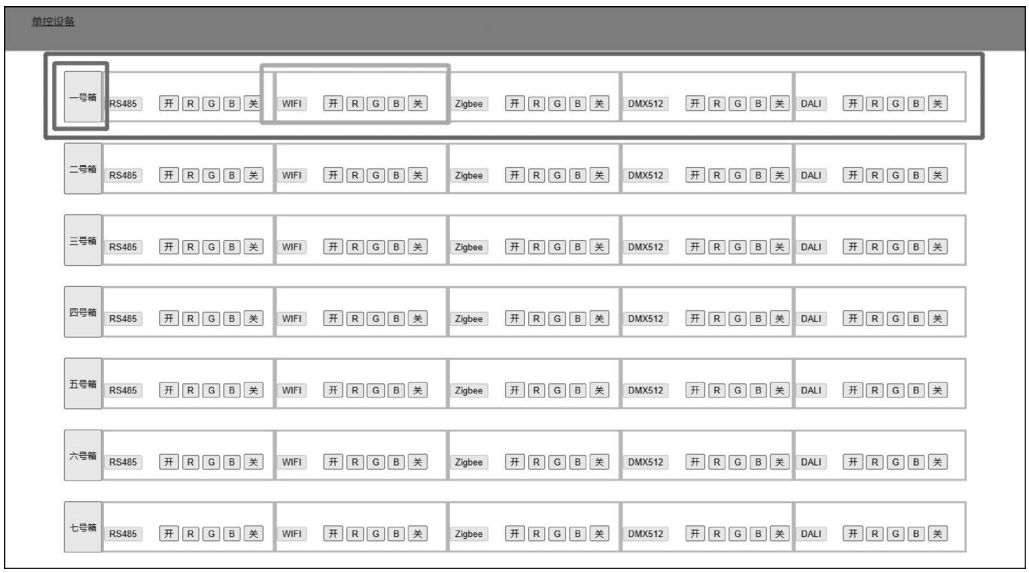

图 5-33　单控功能操作界面

5.4　智慧照明云平台使用方法

本章介绍该本系统 Web 平台执行文件的下载、安装和使用的方法。

5.4.1　下载教程

(1) 下载后解压，解压后的结果如图 5-34 所示。

(2) 确保智能网关已经启动的情况下，双击 App.exe，打开如图 5-35 所示界面，则视为

图 5-34　智慧照明平台安装文件解压后的文件名称

后端运行成功；如果出现“对方计算机拒绝请求”，检查综合控制器（智能网关）是否运行成功。

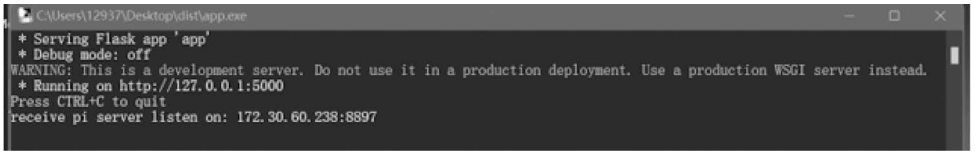

图 5-35　智能网关正常开启的显示内容

（3）在浏览器输入网址 http://localhost:5000/jiemian.html 后打开 Web 界面，如图 5-36 所示。

图 5-36　网址

Web 页面如图 5-37 所示。

图 5-37　登录界面

（4）输入用户名“admin”，密码“123456”，登录成功，进入首页，如图 5-38 所示。

5.4.2　基于 Web 的单灯控制

单灯控制即实现一个灯的控制。

（1）在首页单击界面上方菜单的单控设备选项，进入单控界面，如图 5-39 所示。

（2）界面里有 25 台实验箱的控制按钮，每台实验箱对应 5 个模块，选取要控制的灯对应的协议模块，每个模块分别有“开”“红灯”“绿灯”“蓝灯”“关”5 项调光调色功能，单击对应

图 5-38　首页

图 5-39　单控界面

的按钮，则触发下发事件，将控制指令通过综合控制器发送给对应实验箱的对应模块，完成控制功能。操作方法如图 5-40 所示。

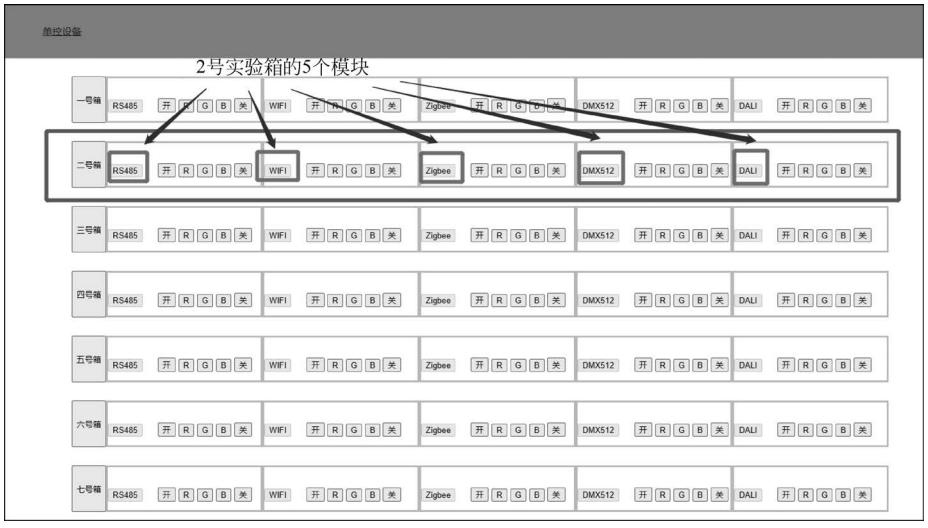

图 5-40　单控界面控制

(3) 单击按钮后,后台自动触发 Get 事件,完成控制指令的下发,指令发送到综合控制系统,综合控制系统对 LED 的地址进行解析,然后发送给对应实验箱的对应大功率 LED 灯模块,从而实现控制功能。

5.4.3 基于 Web 的集中控制

(1) 与单控功能相类似,选择首页上方菜单的集控设备选项,进入集控界面,如图 5-41 所示。

(2) 集控界面里分别对应 5 个模块的集控功能。例如 RS485 模块,可实现对 25 个实验箱里所有的 RS485 模块集中调光调色。如要打开 RS485 的红灯,则直接选择单击 RS485 集控按钮下方打开红灯按钮,后台会触发打开 RS485 红灯指令的下发,实现打开所有实验箱 RS485 模块大功率 LED 灯的功能。操作示例如图 5-41 所示。

图 5-41 集控界面