## **实验报告**

### **项目名称**

数字万年历

### **项目简介**

本项目是一个基于Flask的Web应用，提供用户注册、登录、日程管理和日记管理功能。用户可以通过注册创建账户，登录后可以添加、编辑和删除日程和日记。

### **技术栈**

· **后端**：Flask

· **数据库**：MySQL

· **前端**：HTML, CSS, Bootstrap

· **其他**：JavaScript

### **功能实现**

#### 1. 用户注册

· **功能描述**：用户可以通过注册页面创建新账户。

· **实现细节**：

· 检查用户名是否已存在。

· 验证密码格式为8-16位的字母加数字。

· 为新用户分配唯一的is\_admin值。

#### 2. 用户登录

· **功能描述**：用户可以通过登录页面访问其账户。

· **实现细节**：

· 验证用户名和密码。

· 登录成功后，显示用户名和注销按钮。

· 登录失败时，显示“用户名或者密码错误”。

#### 3. 日程和日记管理

· **功能描述**：用户可以添加、编辑和删除日程和日记。

· **实现细节**：

· 只有登录用户才能进行操作。

· 数据库中存储用户的日程和日记信息。

· 提供API接口供前端调用。

### **数据库设计**

· **用户表**：存储用户信息，包括用户名、密码和is\_admin。

· **日程表**：存储用户的日程信息。

· **日记表**：存储用户的日记信息。

### **代码结构**

· **PythonApplication1.py**：Flask应用的主文件，包含路由和业务逻辑。

· **templates/**：存放HTML模板文件。

· **static/**：存放CSS和JavaScript文件。

### **实验步骤**

· **环境配置**：安装Flask和MySQL，并配置数据库连接。

· **数据库初始化**：运行init\_db()函数创建必要的表。

· **功能开发**：实现注册、登录、日程和日记管理功能。

· **测试**：通过浏览器测试各个功能模块。

### **遇到的问题及解决方案**

· **问题**：用户名重复注册时报错。

· **解决方案**：在注册逻辑中添加用户名存在检查，并返回错误信息。

· **问题**：未登录用户可以访问日程和日记。

· **解决方案**：在相关路由中添加登录状态检查。

· **问题**：增删改查的时候对应id错误。

· **解决方案**：使用事务处理，同时执行或者回滚。

### **结论**

本项目实现了一个基本的用户管理和日程、日记管理系统，用户可以通过注册和登录管理个人信息。通过本次实验，熟悉了Flask框架的基本使用和Web应用的开发流程。

### **未来改进**

· 增加用户密码加密存储。

· 提供更丰富的日程和日记管理功能。

· 优化前端界面，提高用户体验。