目录一、引言

1.1 背景和意义

1.2 项目概述

二、单元学习目标

2.1 目标明确

2.2 学习任务

三、项目实施

3.1 技术架构

3.2 数据库设计

3.3 代码实现

3.4 界面展示

四、项目总结

4.1 项目成果

4.2 项目经验

4.3 未来展望

一、引言

1.1 背景和意义

现在我们生活在一个信息化时代，网络技术不断发展，Web应用已经成为人们日常工作和生活中必不可少的一种工具。WebUI作为Web应用的一个重要组成部分，为用户提供了非常方便的交互界面，已经广泛应用于各种Web应用系统的开发中。

Flask作为一种Python Web框架，具有轻量级、易用性和高度可定制化等优点，已经成为了Web应用开发的热门选择。本次项目旨在基于Flask构建一个简单的WebUI，通过一个输入框和三个选项，用户可以输入数据并将数据发送到后端，后端将记录数据并返回前端，最后在前端将记录展示出来。

1.2 项目概述

本项目基于Flask框架，使用Python语言进行开发。前端页面包括一个输入框和三个选项，用户可以输入数据并选择其中一个选项，将数据发送到后端。后端将记录数据，然后将记录返回前端进行展示。本项目旨在让我们掌握如何使用Flask框架构建一个简单的WebUI，学会如何处理表单数据，如何与后端交互，以及如何将数据保存在数据库中。

二、单元学习目标

2.1 目标明确

本单元的学习目标主要包括以下几个方面：

1. 了解WebUI的基本概念和架构原理；

2. 掌握Flask框架的使用方法；

3. 熟悉Web应用开发中常用的HTML、CSS和JavaScript等前端技术；

4. 学会如何使用Flask与数据库进行交互；

5. 实现一个简单的WebUI，让我们掌握如何处理表单数据，如何与后端交互，以及如何将数据保存在数据库中。

2.2 学习任务

1. 学习WebUI的基本概念和架构原理；

2. 学习Flask框架的使用方法；

3. 学习前端技术，包括HTML、CSS和JavaScript等；

4. 学习如何使用SQLAlchemy与数据库进行交互；

5. 实现一个简单的WebUI，包括前端页面设计、Flask框架搭建、表单数据处理、后端数据存储和展示等。

三、项目实施

3.1 技术架构

本项目的技术架构主要包括前端技术、后端技术和数据库技术。

前端技术主要包括HTML、CSS和JavaScript等。HTML用于构建网页的结构，CSS用于美化网页的样式，JavaScript用于实现网页的动态效果和交互功能。

后端技术主要包括Flask框架和Python语言。Flask是一个轻量级的Web框架，提供了路由、模板、表单处理等功能，可以快速构建一个Web应用。Python作为一种高级编程语言，已经成为了Web应用开发的热门选择。

数据库技术主要使用了SQLite数据库和SQLAlchemy工具。SQLite是一个轻量级的关系型数据库，适合于小型应用的开发。SQLAlchemy是一个Python的ORM（对象关系映射）工具，可以将Python对象映射到数据库表中，方便进行数据库操作。

3.2 数据库设计

本项目使用了SQLite数据库，包括一个数据表。具体的数据库结构如下：

Table: records

Columns:

id INTEGER PRIMARY KEY

content TEXT

type TEXT

created\_at TIMESTAMP

其中，id为记录的唯一标识符，content为记录内容，type为记录类型，created\_at为记录创建时间。

3.3 代码实现

本项目的代码实现主要包括前端设计和后端代码实现两部分。

前端设计使用了HTML、CSS和JavaScript等技术，采用了Bootstrap框架进行快速开发。具体的前端代码实现如下：

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>WebUI</title>

<link rel="stylesheet" href="https://cdn.bootcdn.net/ajax/libs/twitter-bootstrap/4.6.0/css/bootstrap.min.css">

<script src="https://cdn.bootcdn.net/ajax/libs/jquery/3.6.0/jquery.min.js"></script>

<script src="https://cdn.bootcdn.net/ajax/libs/twitter-bootstrap/4.6.0/js/bootstrap.min.js"></script>

</head>

<body>

<div class="container">

<div class="row mt-5">

<div class="col-md-6 offset-md-3">

<h1 class="text-center mb-5">WebUI</h1>

<form method="post" action="/">

<div class="form-group">

<label for="content">Content</label>

<input type="text" class="form-control" id="content" name="content" required>

</div>

<div class="form-group">

<label for="type">Type</label>

<select class="form-control" id="type" name="type" required>

<option value="Option 1">Option 1</option>

<option value="Option 2">Option 2</option>

<option value="Option 3">Option 3</option>

</select>

</div>

<button type="submit" class="btn btn-primary">Submit</button>

</form>

<div class="mt-5" id="records"></div>

</div>

</div>

</div>

<script>

$(function() {

function showRecords() {

$.getJSON('/records', function(data) {

var html = '';

$.each(data, function(i, item) {

html += '<div class="card mb-2">';

html += '<div class="card-body">';

html += '<p>' + item.content + '</p>';

html += '<p>' + item.type + '</p>';

html += '<p>' + item.created\_at + '</p>';

html += '</div>';

html += '</div>';

});

$('#records').html(html);

});

}

showRecords();

setInterval(showRecords, 5000);

});

</script>

</body>

</html>

后端代码主要使用了Flask框架和SQLAlchemy工具，具体的后端代码实现如下：

import os

from flask import Flask, render\_template, request, jsonify

from flask\_sqlalchemy import SQLAlchemy

from datetime import datetime

app = Flask(\_\_name\_\_)

app.config['SQLALCHEMY\_DATABASE\_URI'] = 'sqlite:///' + os.path.join(app.root\_path, 'data.db')

app.config['SQLALCHEMY\_TRACK\_MODIFICATIONS'] = False

db = SQLAlchemy(app)

class Record(db.Model):

id = db.Column(db.Integer, primary\_key=True)

content = db.Column(db.Text, nullable=False)

type = db.Column(db.String(50), nullable=False)

created\_at = db.Column(db.DateTime, default=datetime.now)

def to\_dict(self):

return {

'id': self.id,

'content': self.content,

'type': self.type,

'created\_at': self.created\_at.strftime('%Y-%m-%d %H:%M:%S'),

}

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

def index():

if request.method == 'POST':

content = request.form['content']

type = request.form['type']

record = Record(content=content, type=type)

db.session.add(record)

db.session.commit()

records = Record.query.order\_by(Record.created\_at.desc()).limit(10).all()

return render\_template('index.html', records=[r.to\_dict() for r in records])

@app.route('/records')

def records():

records = Record.query.order\_by(Record.created\_at.desc()).limit(10).all()

return jsonify([r.to\_dict() for r in records])

if \_\_name\_\_ == '\_\_main\_\_':

app.run()

3.4 界面展示

下图展示了本项目的前端页面界面：

![WebUI](https://i.imgur.com/AqQZjZh.png)

四、项目总结

4.1 项目成果

本项目基于Flask框架，使用Python语言进行开发。前端页面包括一个输入框和三个选项，用户可以输入数据并选择其中一个选项，将数据发送到后端。后端将记录数据，然后将记录返回前端进行展示。本项目旨在让我们掌握如何使用Flask框架构建一个简单的WebUI，学会如何处理表单数据，如何与后端交互，以及如何将数据保存在数据库中。

4.2 项目经验

通过本项目的实施，我们可以掌握如下技能：

1. 掌握WebUI的基本概念和架构原理；

2. 掌握Flask框架的使用方法；

3. 熟悉Web应用开发中常用的HTML、CSS和JavaScript等前端技术；

4. 学会如何使用SQLAlchemy与数据库进行交互；

5. 实现一个简单的WebUI，让我们掌握如何处理表单数据，如何与后端交互，以及如何将数据保存在数据库中。

4.3 未来展望

本项目只是一个简单的WebUI示例，还有很多可以改进的地方。例如，可以添加用户认证和授权功能，让用户只能访问自己的数据；可以添加数据筛选和排序功能，让用户更方便地查找自己需要的数据。在未来的学习中，我们可以继续完善本项目，让它更加完善和实用。