Лекция 14

Здравствуйте, уважаемые слушатели!
Тема нашей лекции – Введение в веб-разработку с использованием Flask

План лекции:
1. Основные концепции Flask
2. Маршрутизация и Представления
3. Работа с формами
4. Доступ к базам данных
5. Заключение

Введение

Flask является одним из наиболее популярных микрофреймворков для веб-разработки на языке Python. Он позволяет разрабатывать веб-приложения быстро и эффективно, обеспечивая при этом высокий уровень гибкости и контроля над процессом разработки. Flask поддерживает расширения, которые могут добавлять дополнительные функциональные возможности, такие как работа с формами, аутентификация пользователей, управление базами данных и многое другое. Эта лекция представляет собой введение в веб-разработку с использованием Flask, рассматривая его основные концепции, структуру приложения и практические аспекты разработки.

1. Основные концепции Flask

1.1 Что такое Flask?

Flask — это легковесный веб-фреймворк для Python, который предоставляет инструменты и функции для создания веб-приложений. Он был разработан как простой, но расширяемый инструментарий, который делает возможным разработку веб-приложений от простых до сложных. Flask основан на двух ключевых компонентах: Werkzeug (WSGI утилита для Python) и Jinja2 (система шаблонов).

1.2 Установка и настройка

Для начала работы с Flask его необходимо установить. Это можно сделать с помощью инструмента управления пакетами pip:

bash
Копировать код
pip install Flask

1.3 "Hello, World!" в Flask

Создание базового веб-приложения в Flask начинается с нескольких строк кода. Ниже приведен пример минимального приложения:

python
Копировать код
```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def hello_world():
    return 'Hello, World!'
```

Здесь мы импортируем класс Flask и создаем экземпляр этого класса. Декоратор @app.route() используется для связывания функции с URL.

2. Маршрутизация и Представления

2.1 Маршрутизация

Маршрутизация относится к определению URL-маршрутов приложения и функций, которые вызываются при обращении к этим маршрутам. Flask позволяет легко настраивать маршруты через декораторы.

2.2 Динамические маршруты

Flask поддерживает переменные в маршрутах, что позволяет создавать более гибкие URL-адреса:

python
Копировать код
```
@app.route('/user/<username>')
def show_user_profile(username):
    return f'User {username}'
```

2.3 Представления и шаблоны

Представления в Flask обрабатывают бизнес-логику приложения и отвечают за формирование ответа. Flask интегрируется с системой шаблонов Jinja2 для генерации HTML из шаблонов.

python
Копировать код
```
from flask import render_template

@app.route('/hello/<name>')
def hello(name=None):
```

```
    return render_template('hello.html', name=name)
```

3. Работа с формами

Обработка форм является обыденной задачей в веб-разработке. Flask поддерживает работу с формами через Flask-WTF, расширение, которое интегрирует работу с формами и валидацию данных.

3.1 Установка Flask-WTF

bash
Копировать код
```
pip install Flask-WTF
```

3.2 Пример формы

python
Копировать код
```
from flask_wtf import FlaskForm
from wtforms import StringField, SubmitField
from wtforms.validators import DataRequired

class MyForm(FlaskForm):
    name = StringField('What is your name?', validators=[DataRequired()])
    submit = SubmitField('Submit')
```

4. Доступ к базам данных

Для работы с базами данных в Flask часто используется расширение Flask-SQLAlchemy, которое предоставляет удобные инструменты для работы с базами данных через ORM SQLAlchemy.

4.1 Установка Flask-SQLAlchemy

bash
Копировать код
```
pip install Flask-SQLAlchemy
```

4.2 Пример модели

python
Копировать код
```
from flask_sqlalchemy import SQLAlchemy

db = SQLAlchemy(app)
```

```
class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True, nullable=False)
    email = db.Column(db.String(120), unique=True, nullable=False)

    def __repr__(self):
        return f'<User {self.username}>'
```

5. Практические аспекты использования Flask

5.1 Развертывание

Развертывание приложений Flask может быть выполнено на различных платформах, таких как Heroku, AWS и других. Важно убедиться, что все зависимости корректно настроены и окружение безопасно.

5.2 Отладка и тестирование

Flask предоставляет встроенные средства для отладки и тестирования приложений. Режим отладки можно включить, установив app.debug = True.

Заключение

Flask представляет собой мощный и гибкий инструмент для создания веб-приложений на Python. Благодаря своей простоте и расширяемости, он позволяет быстро создавать и развертывать веб-приложения, поддерживая при этом сложные и масштабируемые проекты. Надлежащее понимание и использование Flask может значительно улучшить качество веб-разработки и ускорить процесс создания веб-приложений.

Литературы:

1. Программирование на Python для начинающих. Райтман М.А. Москва-ЭМОСКО-2015 стр 96-102
2. Python для «чайников». Джон Полль Мюллер. Диалектика-2022 стр. 99-113