SDEV 2401

Rapid Backend Development

Introduction to Backend Web Development

A LEADING POLYTECHNIC COMMITTED TO YOUR SUCCESS

Agenda

On the right is what we will cover today.

What is Backend Web Development?
Backend Development Frameworks Options
Why Use Django for Backend Development?
Companies using Django
Setting Up Your Environment For Django Development
Starting Your Django Project
Running the Development Server
Key Project Files Explained
Exploring Settings for Your Recipe App Example

Expectations

- No Late Assignments

- No Cheating

- Be a good classmate

- Don’t waste your time

- Show up to class

What is Backend Web Development?

- Backend web development refers to the server-side of web development. It focuses on databases, scripting, and website architecture. This is hosted on a server that you folks will deploy and maintain.
  - This is in contrast to frontend development, which focuses on the user interface and user experience.
  - Frontends code is on the client side device, usually in a web browser (HTML, CSS, JavaScript).
- Every application that you use whether it's a web app, mobile app, or desktop app has a backend that powers it.
  - The backend is responsible for storing and managing data, processing requests, and ensuring that the application functions correctly.

Backend Development Framework Options

There's a lot of different languages and frameworks that you can use for backend development. Some popular options include:

- Ruby on Rails (Ruby)
- .NET (C#)
- Laravel (PHP)
- Spring Boot (Java)
- Express.js (JavaScript)
- Flask or FastAPI (Python) - lightweight backend framework
- Django (Python) - what we will be using in this course.

All of these frameworks have their own strengths and weaknesses, and the choice of which one to use will depend on the specific requirements of your project, your team's expertise, and other factors.

Why Use Django for Backend Development?

- Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design.
  - Since you folks are already familiar with Python for from the first course, using Django will allow you to leverage your existing knowledge.
- It also has a stable API and a large community, which means that there are plenty of resources and libraries available to help you build your applications quickly and efficiently.
- It is a "batteries-included" framework, meaning it comes with a lot of built-in features that make common web development tasks easier.
  - Some of these features include an admin interface, an ORM (Object-Relational Mapping) system, authentication, form validation, and a templating engine.
- Good for small to large projects, and scales well as your application grows.

Companies using Django

A few well-known companies that use Django include:

- Instagram
- Pinterest
- Eventbrite
- Spotify (used in parts of the microservices)
- Mozilla (MDN)
- The Washington Times
- Bitbucket
- Dropbox (account and team management)

Setting Up Your Environment For Django Development

1. Create a virtual environment:
   - Windows: `python -m venv venv`
   - Activate: `.\venv\Scripts\activate`
2. Install Django:
   - `pip install django==5.2`
3. Save dependencies:
   - `pip freeze > requirements.txt`

Starting Your Django Project

Let's take a look at how to create a new Django project called `myrecipes`:

- Run: `django-admin startproject myrecipes`
- Project structure:

```
myrecipes/
    manage.py
    myrecipes/
        __init__.py
        settings.py
        urls.py
        asgi.py
        wsgi.py
```

Running the Development Server

Navigate to the `myrecipes` directory and start the server:

- `python manage.py runserver`
- Visit `http://localhost:8000/` in your browser to see the welcome page.

Key Project Files Explained

- `manage.py` : Main entry point for project commands.
- `myrecipes/settings.py` : Configure database, apps, static files, and more.
- `myrecipes/urls.py` : Map URLs to views for your app.
- `myrecipes/wsgi.py` & `asgi.py` : Entry points for production servers (not needed for now).

Exploring Settings for Your Recipe App

In `settings.py`, you'll find:

- `INSTALLED_APPS`: Add your recipe app here.
- `DATABASES`: Configure your database (default is SQLite).
- `TEMPLATES`: Set up HTML templates for recipe pages.
- `STATIC_URL`: Manage static files like images for recipes.
- `ALLOWED_HOSTS`: Control which hosts can access your app.
- `MIDDLEWARE`: Add custom request/response processing.

Example

Let’s work through an example together.