MAD1 Bootcamp Day 2

Agenda

- SQLAlchemy (ORM)

what is ORM, and how to implement DB design in SQLAlchemy

- breaking main into files
- db + session = auth
- Jinja Filters can do what?

Tentative Schedule

10 - 11 SQLAlchemy Motivation

11 - 12 SQL Alchemy Implementation

12 - 12:30 Segregating code into files

12:30 - 14 break

14 - 15 M+V+C for auth

15 - 16 Jinja Filters, Jinja Layouts, inheritance

16 - 16:30 break

16:30 - 17 doubts, ending notes

SQLAlchemy

- Object Relational Mapping
- Use SQL without writing SQL
- Use familiar Python objects to interact with DB
- Define a single source of truth for DB schema in code
- then generate DB schema from it
- lets you handle relationships easily

sqlalchemy docs

relationships in SQLAlchemy

Spltting code into files

- Why?
- easier to manage, navigate, and debug
- How?
- create config.py, models.py, routes.py
- models -> MODEL, routes -> CONTROLLER
- jinja templates -> VIEW
- import each file correctly, avoid circular import
- explore blueprints (modern approach to modularization)

blueprints

db + session = auth

- on POST of register and login, use python logic to validate
- if valid, perform DB operations for register using SQLAlchemy
- for login, store the username in session using flask session
- use session to show error messages using flash

flask-session

SQLAlchemy Cascades

Jinja Layouts

- create a layout.html file, this holds all the common html
- for all pages, head tags and flash message markup
- for each html file, simply extend layout.html and add content block
- other blocks like style, script, title, are also used
- explore ‘include‘ as well, to have components

documentation

Jinja Filters

- use filters to format data in templates
- capitalize, lower, upper, title, etc
- explore other filters as well, where can we use them?

filters docs

Links

All in one document

code