LECTURE 36

USING FLASK

MCS 275 Spring 2022

David Dumas

LECTURE 36: USING FLASK

Course bulletins:

- Please install **Flask**, e.g. with

```
python3 -m pip install Flask
```

  in preparation for using it in upcoming assignments.
- Start work on Project 4

TODAY'S GOAL

Build the worker view template

Create the database and link it to the Flask application

WORKSNAP DB SCHEMA

Table **`orders`** has columns:

- **`woid`** - integer uniquely identifying a WO
- **`description`** - string
- **`assigned_to`** - username this WO is assigned to (NULL if not assigned)
- **`time_created`** - `time.time()` when created

GENERATING WORKER VIEW

Route `/worker/ddumas/` results in:

- SELECT query to get `ddumas` assigned WOs
- SELECT query to get unassigned WOs
- Rearrange/format query data for use in template
- Render a page template to format the data as in mockup

Note part of the URL is an argument. The pattern is `/worker/<username>/`

ACTIONS

Buttons on the worker view form need to perform actions.

We'll make Flask routes for this, e.g. `/wo/58/assign_to/ddumas/`.

Requesting that URL should perform an action. But what content will it serve?

REDIRECTS

Every HTTP request results in a numerical status.

So far, we've seen 404 (NOT FOUND) and 200 (OK, generated by Flask when we return HTML).

There are also codes that instruct the browser to load another resource, e.g. 302 (FOUND).

Return `flask.redirect("destination")` to make this happen.

REFERENCES

- jsfiddle - Write and test HTML+CSS quickly in browser
- HTML tutorial from w3schools
- CSS tutorial from w3schools
- The Flask tutorial

REVISION HISTORY

- 2022-04-11 Initial publication