LECTURE 38

WEB APP WRAP-UP

MCS 275 Spring 2022
Emily Dumas

LECTURE 38: WEB APP WRAP-UP

Course bulletins:

- Work on Project 4.

- Project 4 is due 6pm CDT Friday 29 April

- Autograder (basically QuizBot) opens Monday 25 April.

- Course evaluations open Monday. Please complete!

IF GEO STRIKE BEGINS MONDAY

I hope UIC will avert a strike by reaching an agreement with GEO.

If strike starts on Monday 18 April, then until it ends:

- MCS 275 lectures move online (zoom) to avoid crossing picket line
- MCS 275 labs won't be held
- Homework collected as usual, grading begins when strike ends
- No TA office hours or email
- Instructor availability unchanged (office hours, email, discord)

TODAY

This is the last in our contiguous lecture series focused on writing a Flask+SQLite application.

(We may revisit this topic a bit in the last week.)

WORKSNAP TODO LIST

☑ HTML + CSS mockups

☑ Database

☑ Flask application with worker view

☑ New work order form

☑ Activate "take assignment" button

☐ Activate other buttons

☐ Style the submission form

☐ Date/time formatting

☐ Work order status page

☐ Make all actions redirect to natural destinations

☐ DB initialization and connection cleanup

☐ Detect and handle errors (e.g. failure to take assignment)

WORKSNAP TODO LIST

☑ HTML + CSS mockups
☑ Database
☑ Flask application with worker view
☑ New work order form
☑ Activate "take assignment" button
☑ **Activate other buttons**
☑ **Style the submission form**
☐ Date/time formatting
☐ Work order status page
☐ Make all actions redirect to natural destinations
☐ DB initialization and connection cleanup
☐ Detect and handle errors (e.g. failure to take assignment)

WORKSNAP TODO LIST

☑ HTML + CSS mockups

☑ Database

☑ Flask application with worker view

☑ New work order form

☑ Activate "take assignment" button

☑ Activate other buttons

☑ Style the submission form

☐ **Date/time formatting**

☐ **Work order status page**

☐ **Make all actions redirect to natural destinations**

☐ **DB initialization and connection cleanup**

☐ **Detect and handle errors (e.g. failure to take assignment)**

ROUTES

- **/worker/<name>/** - (GET) worker's view of orders
- **/wo/new/** - (GET) form for new order
- **/wo/post/** - (POST) form submission destination
- **/wo/<int:woid>/** - (GET) work order status
- **/wo/<int:woid>/assign_to/<name>/** - (GET[*]) take assignment
- **/wo/<int:woid>/unassign_from/<name>/** - (GET[*]) unassign
- **/wo/<int:woid>/complete_by/<name>/** - (GET[*]) mark complete

[*] These should really be POST but we would need to use javascript or a different button markup to do it.

LAST INSERTED ROW

After a single-row `INSERT`, how to get the primary key of the new row?

```
SELECT last_insert_rowid();
```

Implicitly refers to the most-recently executed `INSERT` on this connection.

DID UPDATE CHANGE ANYTHING?

An UPDATE might match any number of rows (e.g. 0, 1, 50). How can we check the actual number?

```
SELECT changes();
```

Implicitly refers to the most-recently executed UPDATE on this connection.

FLASK FUNCTIONS

All are in the `flask` module:

- **`redirect(url)`** - *Returning* this object from a route will cause the HTTP server to issue a 302 response code, telling client to load `url` instead.

- **`abort(http_error_code)`** - Immediately stop and return a HTTP error code (usually 400 bad request, 401 not authorized, 403 forbidden, or 404 not found).

RETROSPECTIVE

Some of the things you'd do differently in a "real" application:

- **Action history**: We have a single column for WO creation time. We should probably log every action that changes a work order in a separate table.

- **Accounts, roles, cookies**: login page checks credentials against DB, sets browser *cookie*. Auth-required pages check for it, redirect to login page if not found.

- **JavaScript**: e.g. to check for new messages in real time, post new message without loading a new page, make buttons perform POST requests without forms.

- **Pagination**: Links to show next/prev page of messages or posts.

REFERENCES

- jsfiddle - Write and test HTML+CSS quickly in browser

- HTML tutorial from w3schools

- CSS tutorial from w3schools

- The Flask tutorial

REVISION HISTORY

- 2022-04-15 Initial publication