

# Common Express Login Confusion

- What does a route (path) do?
- When do I redirect?
- MVC
- Sessions

# This changes in the future

These answers address **server-generated pages**

- Some nuance when it comes to services
  - Later in semester
  - Common concepts, learn now!

We will cover this here

- Then move on to SPA + services

# Better than working

Working code may not be good code!

- Lots of things "work"
- We want the best options
  - Be understood
    - Not just lines, **concepts**
  - Handle change with minimal complexity

# What is the path?

Remember that Web is Request/Response

- Paths can be confusing

A login path

- The path with the form?
- The path that PROCESSES the form?

Answer: Neither

- The path is what you REQUEST

# Path is what you request

User makes request

- May get what expected
- May get something else

Example: Request main page 

- May get data and a form to change it
- May get a login form

# Redirects aren't flow control

Don't use redirects like calling a function

- (BAD) "redirect to /error"

Based on a good instinct! But

- Redirects involve overhead
- Error pages are weird "pages"
  - Reload?
  - Search for?

# **Instead of redirecting for flow control**

Common functionality in functions on server

- Ex: return HTML from a function
  - From anywhere you need that response
  - Can pass params to function!
    - For message text, etc

# Redirecting POST results

When do we redirect?

- Redirecting POST results often good
  - Keeps user from reloading a page
    - that would change app state
  - Reduces repeat logic
- Redirects are always GET

Ex:

- GET `/` shows login form without session
- POST `/login` redirects to `/` on success



# Maintaining info

Another common case of redirecting:

- Sending to a complex login process
  - Maintaining desired path
- Not in assignment

When you have many pages that require login

- Redirect to/return a login form
- Pass requested URL in query param
- After successful login
  - User is redirected to original request path

# Imagine how you would implement this

- GET /login displays a login form
  - Will POST to /login
- These exist: GET /users, GET /stuff, GET /more
  - They only display for logged in users
  - Other users are redirected
    - To `/login?url=THE_PATH_THEY_REQUESTED`
    - Example: `/login?url=/users`
- After a successful POST to /login
  - If url was supplied, redirect user to that path
  - If no url query param, redirect user to `/`

# MVC Confusion

## Goal of MVC

- Separation of Concerns
- Principle of Least Knowledge
  - Can Change Part A "safely"
    - Part B unaffected

# Benefits of MVC

Convert Big Tasks into more, smaller tasks

- Common patterns of behavior
- Situations as:
  - Data
  - Changes to Data
  - Presentation of Data

# What is NOT MVC

- Naming files with or without `-web`
  - Names aren't the MVC decision
    - They should reflect the roles
    - `model.js`, `view.js` just as easily
      - Often many such
- A separate `foo.js` for each `/foo` url
  - Separation, but not of Concerns

# **We will see MVC Again and Again**

This was server-side HTML generation MVC

- We'll later see separately with front end JS
- And then within React, while React itself is a View

Learning and applying concepts behind MVC

- A process, not a single lesson

# Session Id vs username

Session is user+browser

- You *can* use same session id for multiple browsers
  - Usually a bad idea
  - "Logout" destroys session on server
    - Treats all browsers as logged out
    - Frustrates/Confuses users
- Best to have each login generate new session id

# **Section Summary - Server Generated HTML**

- Server generated sites
  - Can mingle static+dynamic pages
- Ultimately generate HTML
  - Lots of tedium that libraries can make easier



# Section Summary - Session Data

## session

- Connects multiple requests from a user+browser
  - Same user may have many different sessions
- Stored on server
- Often based on session id
  - Connects session to other data
  - Shared with browser in cookie

# Section Summary - State

## state

- Values that can change in app
  - static files don't depend on state
- session data is just one kind of state
- When server-generated HTML
  - Data on server