IN4MATX 285:
Interactive Technology Studio

Practice: Developer Handoffs

Today’s goals

By the end of today, you should be able to…

- Follow a few principles for handing off mockups to a developer
  - Technical principles
  - Social principles
- Reflect back on the course learning objectives

Caveats

Caveats

- Most of my industry development experience was in backend roles, where I was pretty separate from our UX team
- Your course staff crowdsourced some advice from UX and Dev folks in industry
- Today will offer some guidance, but keep asking this question!

Caveats

- A lot of this advice is *very* context-dependent
- Does your team have a PM? How are you communicating with them?
- Do you have a frontend developer, or are your developers full-stack?
  - You may need to involve developers in multiple roles

Handing off

Handing off

- Technical, but also social
- Technical: what you include in your design and how you document it
- Social: how you involve your development team in the conversation
- Handing off is probably thought of as *UX design*, but one should hand off *UX research* as well
  - Or at least share results, more on that later

Handing off

- Programmers, especially in school, are used to having “right” and “wrong” answers
  - “Write some code that produces X”
  - Part of the challenge is doing this efficiently and effectively
- Programmers are good at identifying ambiguity, but may be resistant to resolving it

Problem C
Less Dice
Time limit: 1 second

You have a single fair n-sided die with faces numbered 1 through n. Find the expected (average) value of a single roll multiplied by two. It can be shown that this will always be a whole number.

Input

The first line contains one integer n, ($1 \leq n \leq 100000$), the number of faces on the die.

Output

Print two times the expected value of a die roll.

Sample Input 1	Sample Output 1
2	3

Sample Input 2	Sample Output 2
5	6

Sample Input 3	Sample Output 3
100000	100001

Handing off UX Design: Technical

Handing off UX Design: Technical

Some basics

- What tools are your development team using?
- How can your design be compatible with those tools?
  - E.g., if they're using Bootstrap to implement responsivity, your mockups should too
  - If they're using a hybrid framework like Ionic to implement for mobile, follow that design system

Handing off UX Design: Technical

Some basics

- Support effective exports
  - Spacing, typography, color
  - Tools help surface these, but don’t deviate from your spec
- It’s easy to import and implement these standards

Handing off UX Design: Technical

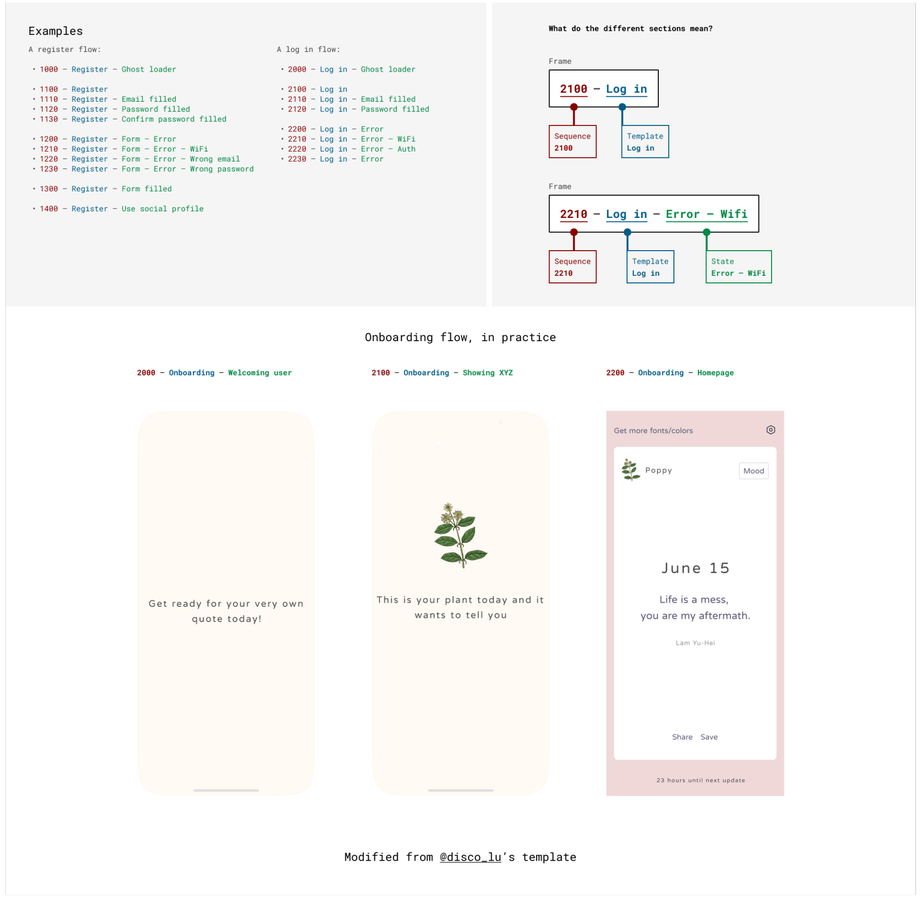
Some basics

- Support the full range of supported devices
  - Don’t just design for mobile, create a mockup for desktop & tablet too
- And ideally, don’t start from scratch
  - Follow a layout which would be easy/possible to implement responsively

Handing off UX Design: Technical

Some basics

- Effectively label your mockups
  - Screens, and relationships between screens
  - Layers
- Walkthrough videos can help, too



https://www.smashingmagazine.com/2023/05/designing-better-design-handoff-file-figma/

Handing off UX Design: Technical

Some basics

- Support accessibility
  - Write the alt text for images
  - Think about button labels and read order

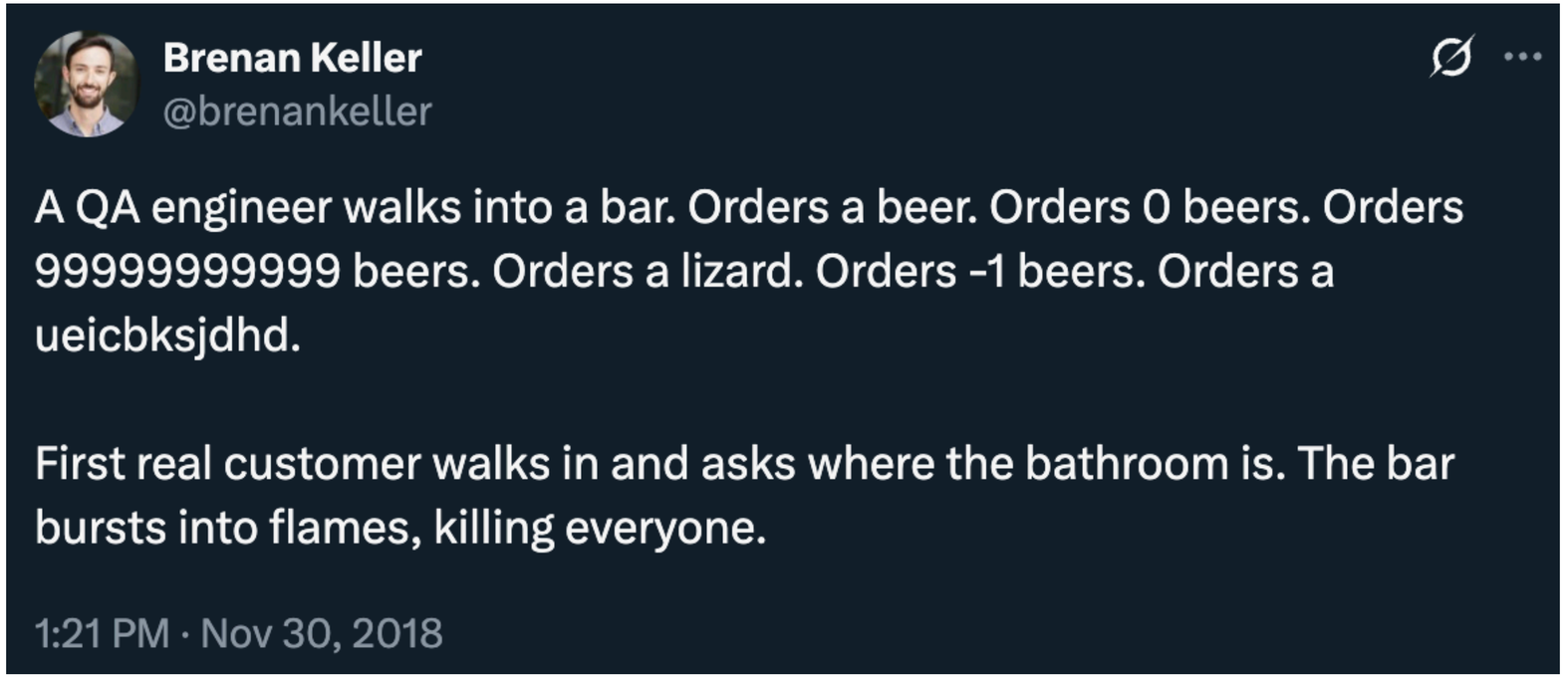
Handing off UX Design: Technical

Some basics

- It’s inevitable that your design will change over time
  - And often it should, in response to developer comments
- Set up clear versioning strategies so your developer is using the right one

Handing off UX Design: Technical

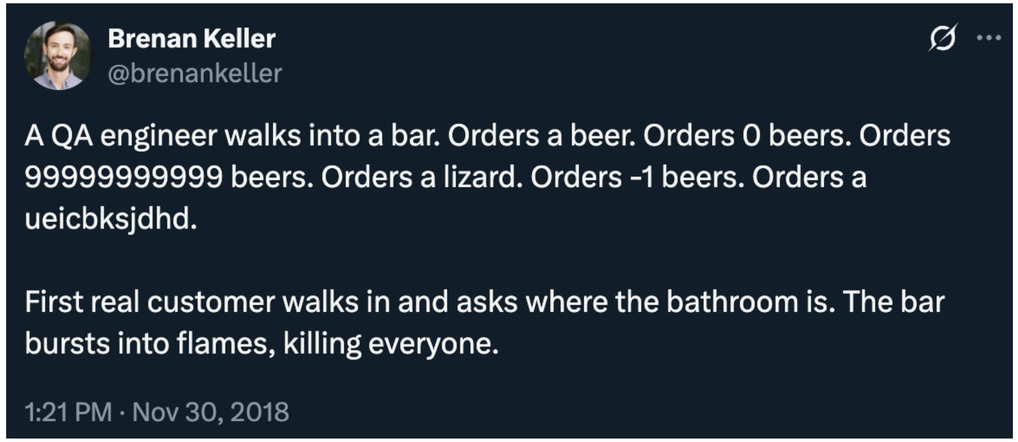
Edge cases



Handing off UX Design: Technical

Edge cases

- It’s one thing to test whether your code “breaks”
- It’s another thing to design what to do when it does
  - How should the design respond to invalid input? Misconfigured URLs?
  - Write the copy text



Handing off UX Design: Technical

Missing states

- Design mockups often fail to design all of the states of inputs
- One example: `<a>` (link) tag
  - Hover over? Active? Visited or not?
- What should transitions look like?
- Knowing what component states are available can help

In addition, links can be styled differently depending on what **state** they are in.

The four links states are:

- `a:link` - a normal, unvisited link
- `a:visited` - a link the user has visited
- `a:hover` - a link when the user mouses over it
- `a:active` - a link the moment it is clicked

Example

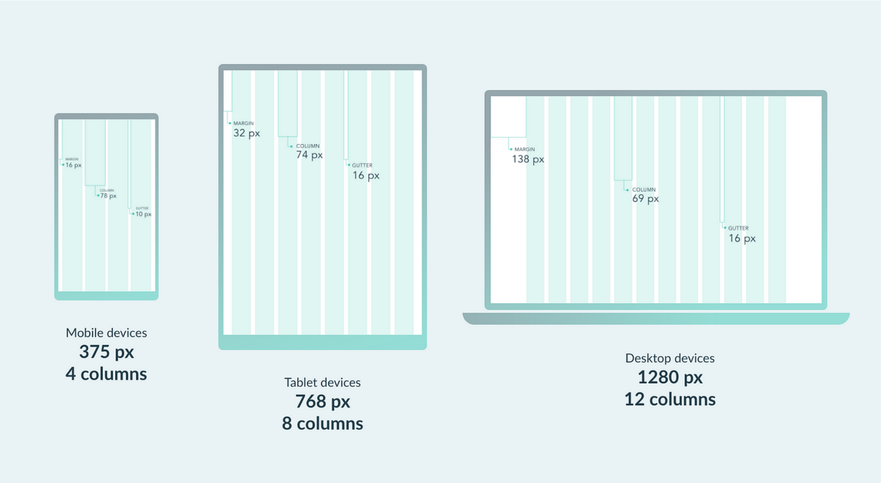
```
/* unvisited link */
a:link {
  color: red;
}

/* visited link */
a:visited {
  color: green;
}
```

https://www.w3schools.com/css/css_link.asp

Handing off UX Design: Technical

Dig into specifications

- Being *really* familiar with the ins and outs of CSS can help with creating implementable designs
  - Padding versus margin, alignment
  - Grid systems
  - Or, whatever similar concepts if not using HTML/CSS/JavaScript



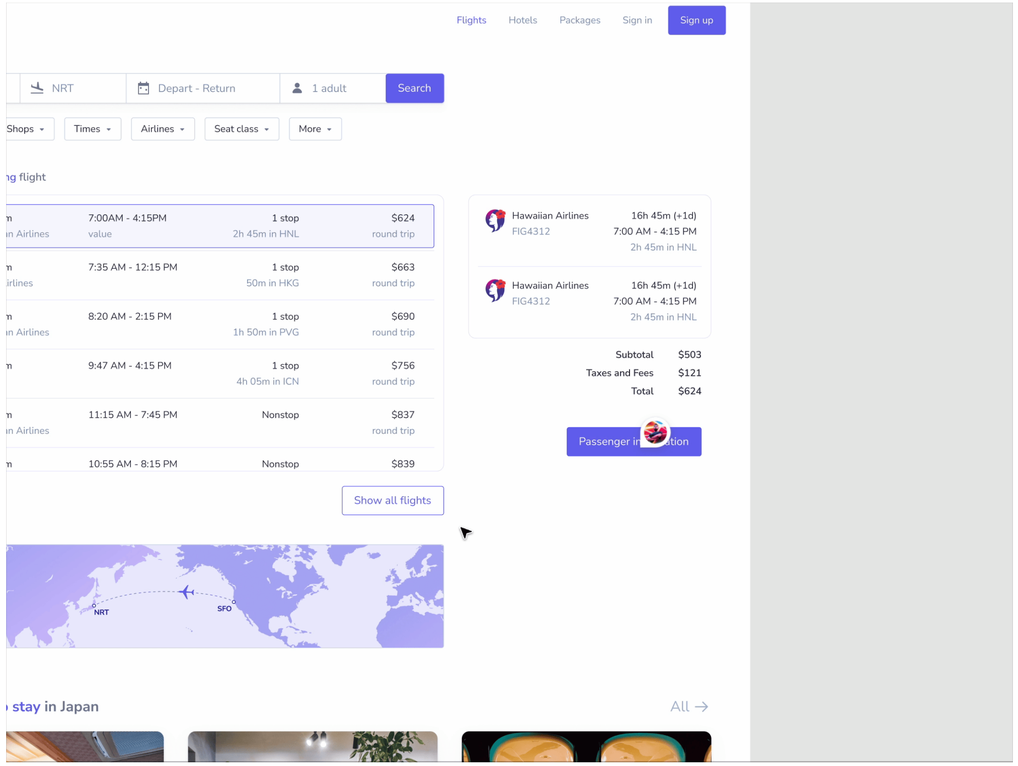
Handing off UX Design: Social

Handing off UX Design: Social

- A bad workflow: you finalize your design, throw it over the wall to your developers and PMs
  - Your design may be impractical to implement in some way
  - It may not align with the business needs in some way
- Instead, conversations early and often can help

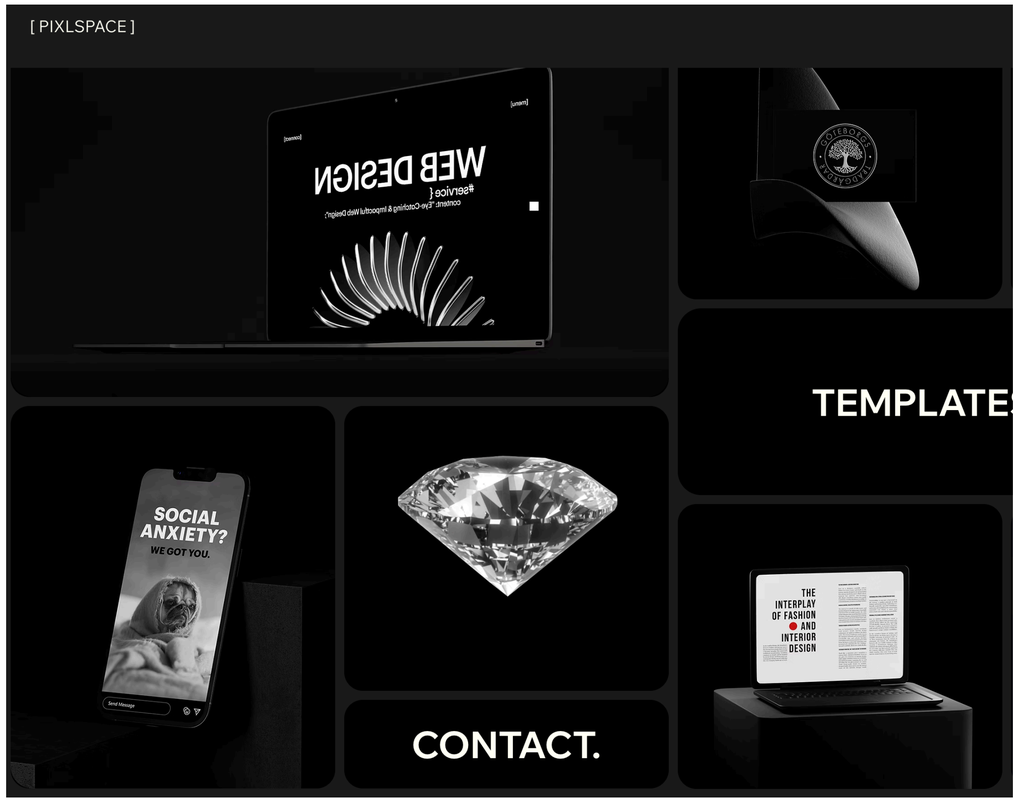
Handing off UX Design: Social

- Develop good feedback workflows
- For example, try to encourage collaboration within your tools



Handing off UX Design: Social

- Evaluating how much to innovate is a challenge
- A flashy, cutting-edge design may be more appealing, but harder to implement
- How much leeway do you have?



https://medium.com/codeart-mk/ux-ui-trends-2025-818ea752c9f7

Handing off UX Design: Social

- But ultimately, you have to read the room of your team, and what your team expects
- My hunch is that UX design often gets bossed around by development timelines and business priorities
  - You may have to use whatever software etc. that the development team expects
  - Your mileage may vary, of course

Handing off UX Research

Handing off UX Research

- Share your *rationale* alongside your *insights*
  - Some data, though be selective
  - Graphs, quotes, user study videos, etc.
- Sharing your problem statement, design objectives, etc. can help ensure that the team is on the same page
  - Developers might not read it, but if they do it’ll help them understand your thinking

Reflecting on course goals

This class won’t teach you to be a developer.

We have a different Masters program for that: Software Engineering!

Instead, we will learn how to work with developers.

We will learn to appreciate what they do, and converse on their topics.

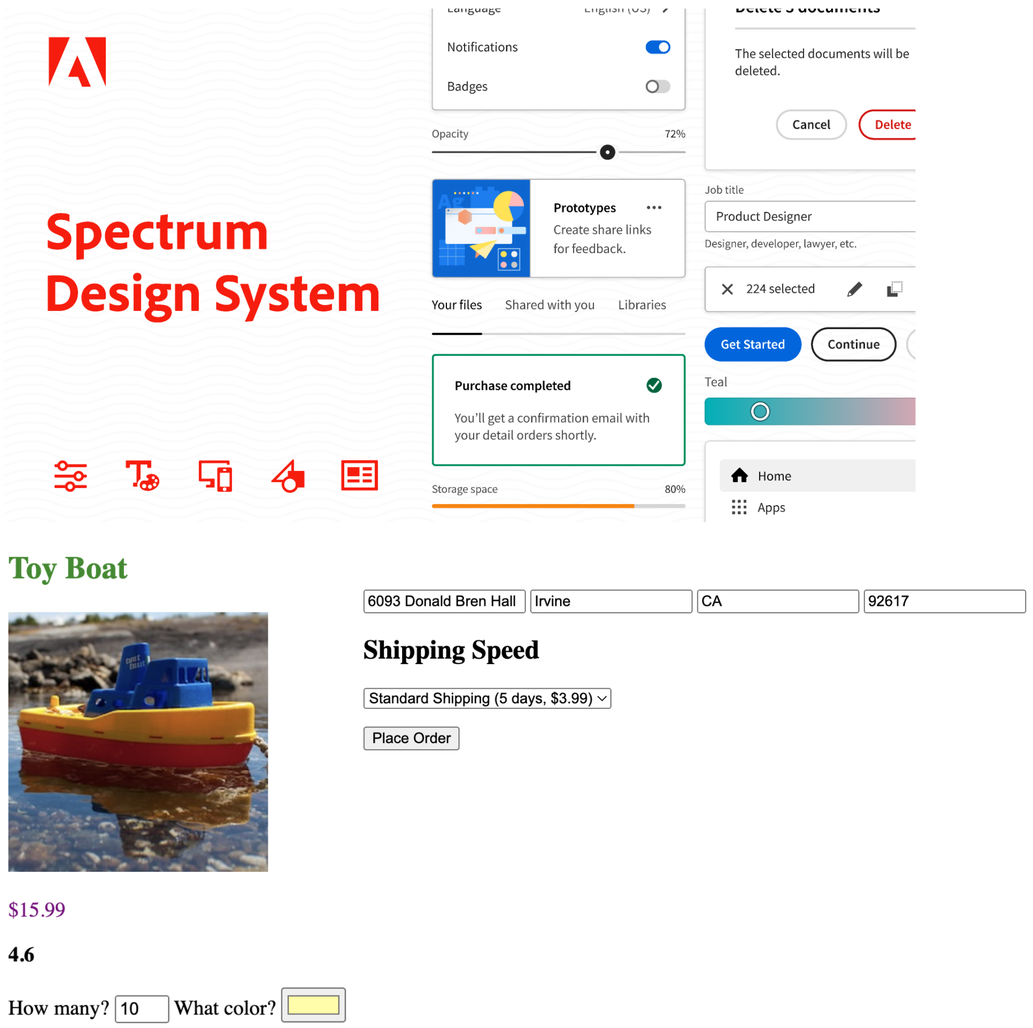
Learning objectives

By the end of this course, you should be able to:

- Implement basic interactive websites in HTML, CSS, and JavaScript
- Be conversant with developers on more complex technical concepts
  - Frontend vs. Backend development
  - Storage and authentication
- Follow principles for working effectively with developers
  - Version control
  - Code libraries

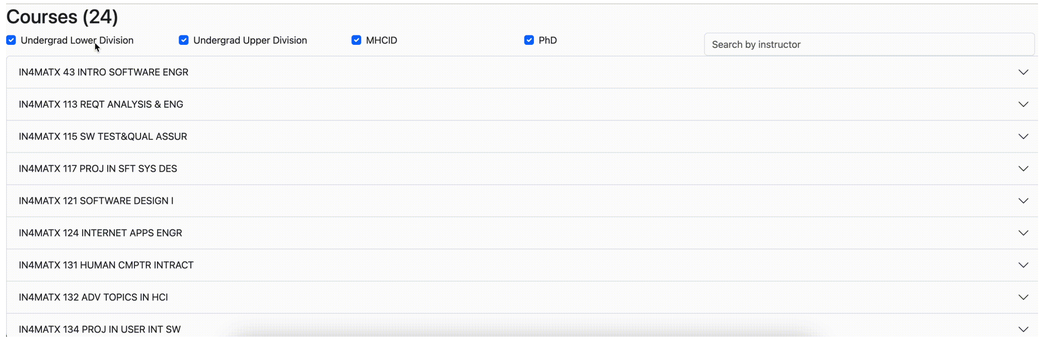
A1

- Hands-on experience implementing a website following a Design System
  - Applied practice with HTML and CSS
- Translation from mockup to implementation



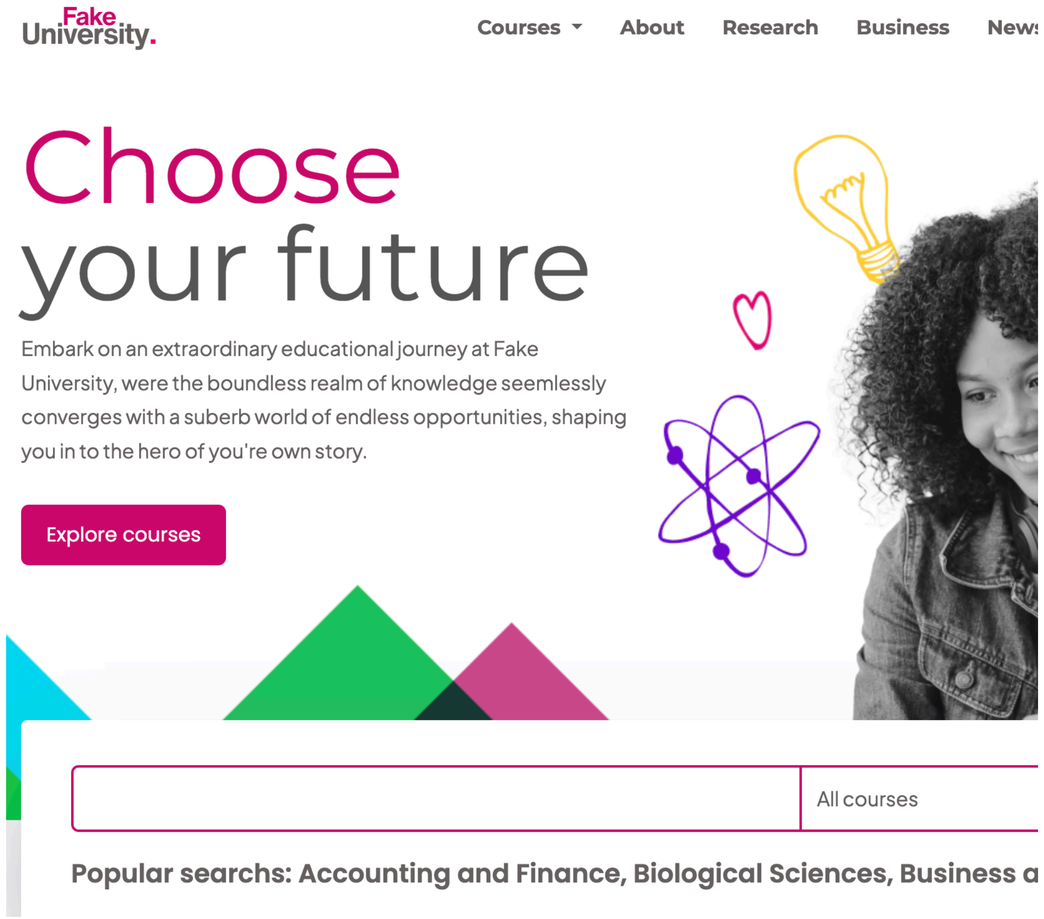
A2

- Introduction to data structures, functions, conditionals, and other programming logic
  - Practice with JavaScript
- Practice implementing responsive design and responding to user input



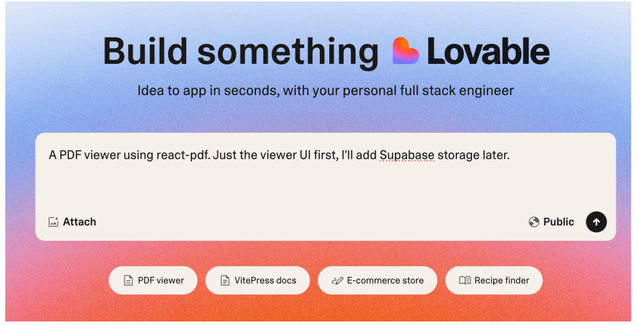
A3

- Practice modifying existing sites programmatically
  - Identification of common accessibility issues which arise in implementation



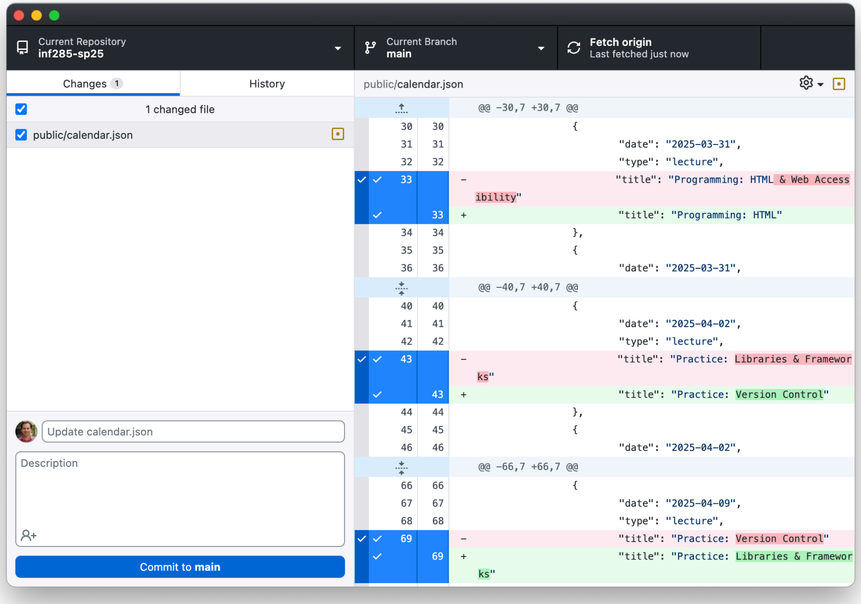
Practice lectures

- Software structures: version control, libraries
- Architectures and tools: databases, logins, generative AI
- Development paradigms: frontend, backend, cross-platform, responsive, beyond web and mobile
- Communication: handoffs



IONIC

OpenID



You did it!

- This class threw you in the deep end of programming, and you all swam
- There were inevitably bumps along the way, but we hope you found it useful

Keep in touch, I look forward to seeing what you do next!

Today’s goals

By the end of today, you should be able to…

- Follow a few principles for handing off mockups to a developer
  - Technical principles
  - Social principles
- Reflect back on the course learning objectives

IN4MATX 285:
Interactive Technology Studio

Practice: Developer Handoffs