GitLab

Lesson 16: Frontend Engineering

Lecture

Topic

1. What is Frontend engineering
2. Features of a typical roles
    a. Collaboration + Communication
    b. Accessibility
    c. Responsive web design
    d. Application architecture
3. Key tools and technologies
4. Common Challenges
5. Emerging trends

Instructor Introduction (5 minutes)

What is my story (first time instructors only)?

- Where am I from?
  - Born in Nigeria, grew up in Australia
  - Currently living in Melbourne
- Timeline
  - Grew up interested in video games, robotics and computers
  - Studied computer science at university
  - Worked in a range of different software engineering roles, across education, government, fintech and now enjoying life at gitlab
- What is my job at GitLab?
  - Sr. Frontend engineer for the Manage optimize team
  - Working in a cross functional team to help customers draw insights from and better understand how they’re using gitlab
  - Frontend Maintainer
- What are my roles and responsibilities at GitLab
  - Implementing and maintaining features our team are directly responsible for
  - Working on features that cut across other domains and teams
  - Contributing to discussions about feature proposals and helping to shape the final proposal
  - As a maintainer
    - I spend a lot of time reviewing code contributions from colleagues or community members
    - Making improvements to the overall health and performance of the gitlab project

GitLab

Part 1: What is Frontend Engineering?

What is Frontend engineering?

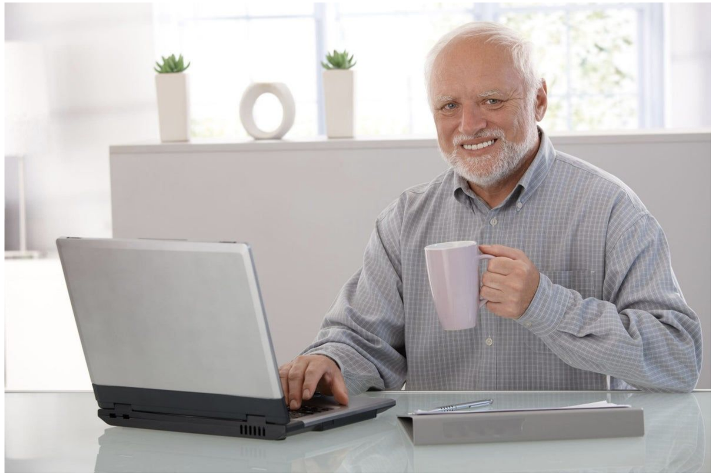
Why?

- The internet has become an integral part of the lives of many people around the world
- Bridging the gap between users and a product or service, informed by design decisions
- Typically a frontend engineer is perceived to be the person(s) that build the visual part of a website or application

What is Frontend engineering?

Understanding users

- Understanding how users use typical web sites and applications is very beneficial
- User expectations
  - Web design has become more advanced over the years
  - Users have much higher expectations and opinions of how web applications should look, feel and operate
  - It should "just work"
  - Catering for the various conditions users use web applications, on their phone on a busy train vs in an office with dual 4K monitors



What is Frontend engineering?

Understanding the business

- Understanding the company’s goals, business processes, user objectives and technical debt
    - The goal is to tie these processes and objectives together with the best outcome for users
    - Working within the constraints of the available data

What is Frontend engineering?

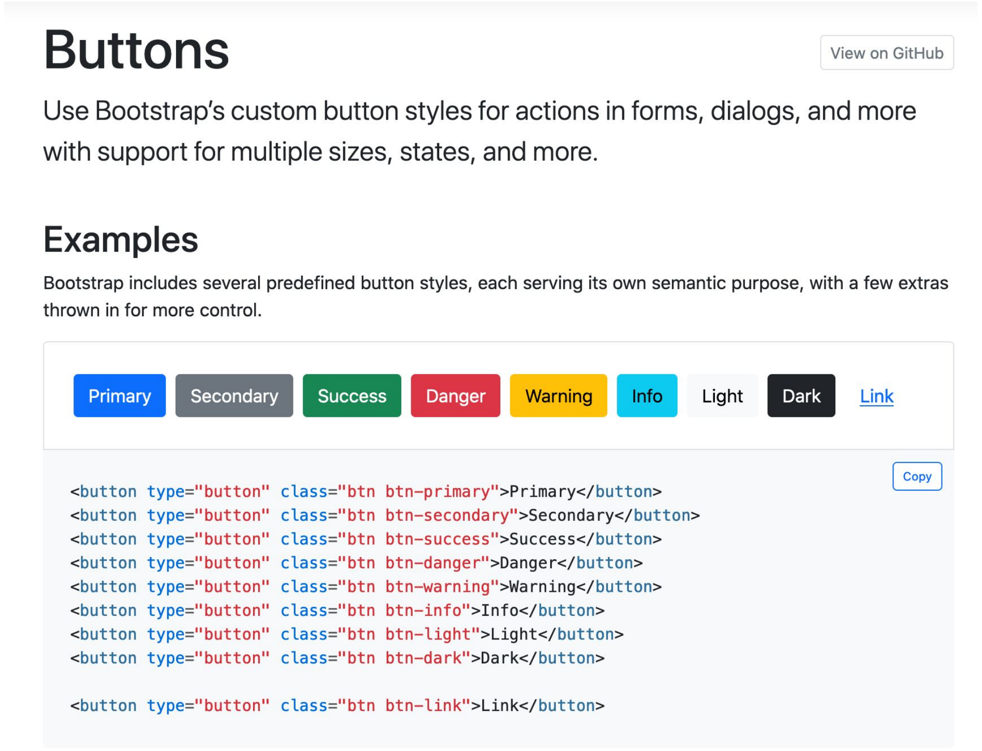
But what *is* a Frontend engineer?

Generally we can think of Frontend engineers falling into one or more groups:

- UI focused
- Application focused
- Frontend architect

What is Frontend engineering?

UI Focused (UI Engineer)



Buttons

View on GitHub

Use Bootstrap's custom button styles for actions in forms, dialogs, and more with support for multiple sizes, states, and more.

Examples

Bootstrap includes several predefined button styles, each serving its own semantic purpose, with a few extras thrown in for more control.

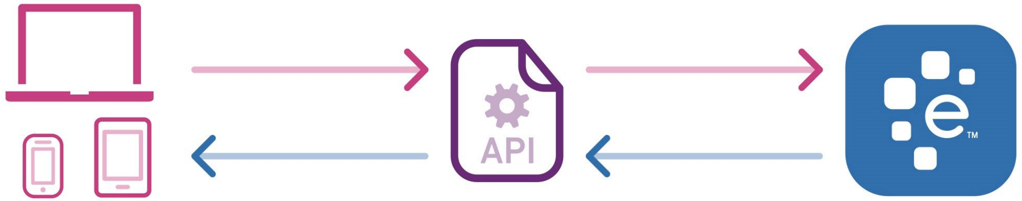
```
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-dark">Dark</button>

<button type="button" class="btn btn-link">Link</button>
```

- The common perception of Frontend engineering
- Work closer with UX researchers and designers
- In larger teams might develop a shared library of components for other engineers to use

What is Frontend engineering?

Application Focused (JavaScript Engineer)

- Focused on connecting the user interface to business processes
- Make heavy use of internal or external APIs to execute tasks
- Sometimes called "JavaScript engineers"



What is Frontend engineering?

Frontend architect

- Focused on connecting the user interface to business processes
- Make heavy use of internal or external APIs to execute tasks
- Most Frontend roles will incorporate different aspects for frontend architecture

GitLab

Part 2: Features of a typical role

Key skills

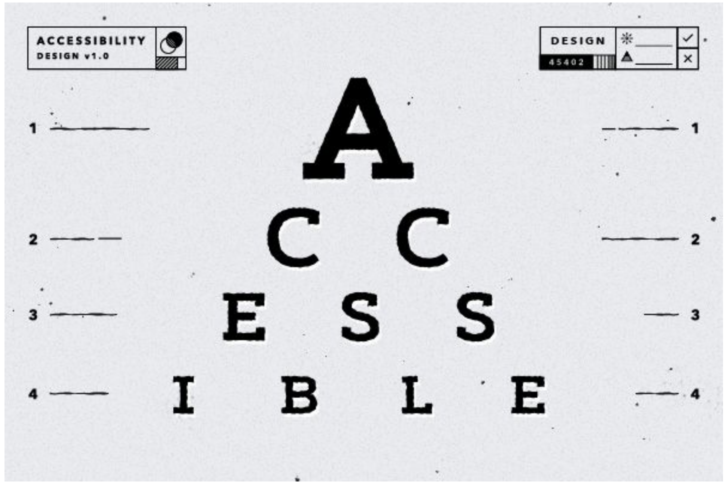
Collaboration + communication

- Often overlooked as a key skill
- Most Frontend roles will require some collaboration with a wide range of peers, UXers, designers, product managers, backend engineers and even customers
- The ability to understand requirements and effectively articulate constraints will always help avoid misunderstandings

Key skills

Accessibility and inclusive design

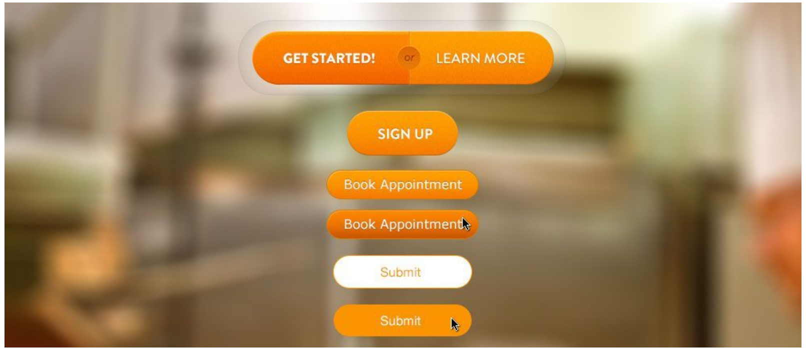
- Increasingly, applications and pages need to be accessible and performant on multiple devices with different capabilities
- Inclusive design means designing websites, applications, browsers, tools, and every other digital product minding everyone who is permanently or temporarily disabled.



Key skills

Responsive web design

- Use of media queries allows targeting specific device capabilities
- Not all users are on a modern phone with a fast connection
- Often harder to retrofit into a project



Application architecture

- Data fetching
- Asynchronous (async) communication / events
- Managing application state
- Single Page Applications (SPAs)

Application architecture

Data fetching

AJAX

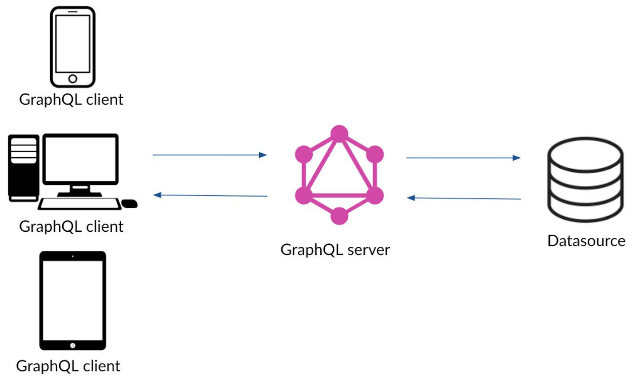
Asynchronous JavaScript and XML

JS

eXtensible Markup Language

{JSON}

GraphQL



GraphQL for BE devs | @engfragui

Application architecture

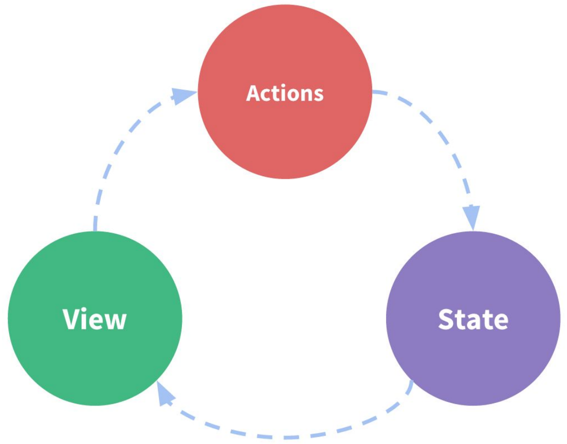
Asynchronous events

- User interactions: clicks, mouse over, touch events, scrolling
- Device events: screen orientation change, switching to full screen, network disconnects
- Content related events: all images have loaded, media starts playing

Application architecture

Managing application state

- Application state might refer to the available data + elements visible in the user interface
- Many many approaches have been developed to help engineers deal with complex states and transitions
- State management can often be a contentious issue in teams, each approach has its own pros and cons



Application architecture

Single page applications (SPAs)

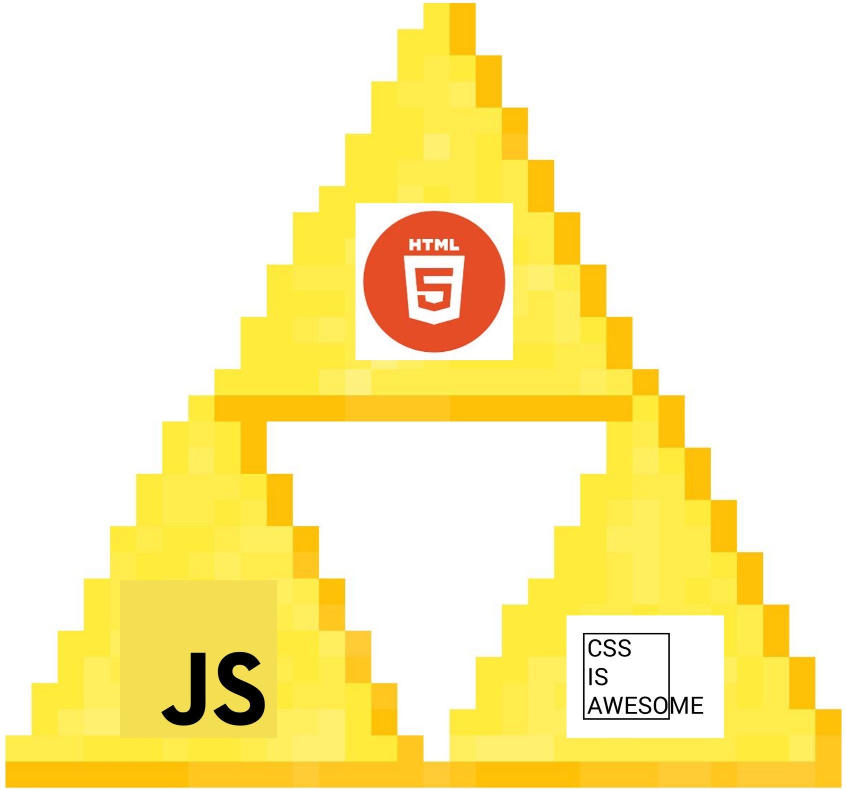
- The application loads at once, with dynamic updates made to segments of the page as the user interacts with it
- Provides a more seamless interaction for the users and can appear to be faster than multi page applications
- Common examples: Spotify, Google maps, Facebook
- Some cons:
  - Harder to "deep link" into,
  - Harder to optimize for SEO
  - Often add an additional layer of complexity

GitLab

Part 3: Key tools and technologies

Tools and technologies

HTML + CSS + JavaScript



Tools and technologies

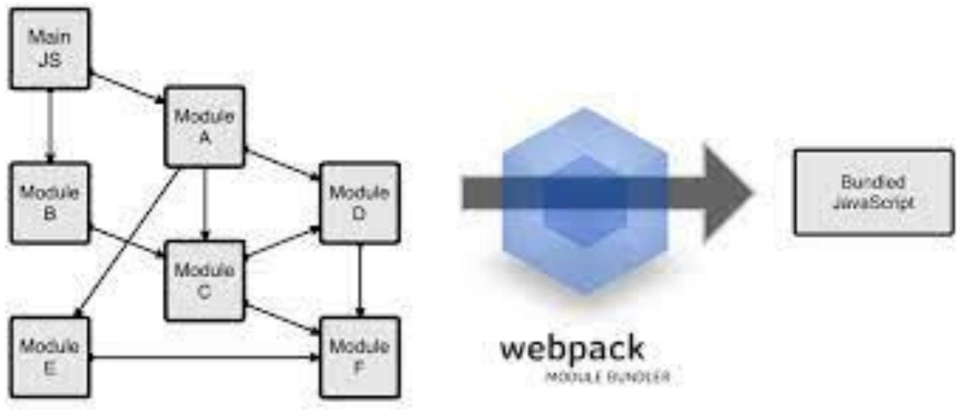
Frameworks and libraries

- Often great choice for teams and collaborative projects
  - Provide a clear and consistent structure to the application
  - Help to abstract away some of the common boilerplate for web applications
- Popular frameworks: React, VueJS and Angular

Tools and technologies

Build tools / Bundlers / Task runners

- Concatenate + minify
- Transpiling + polyfilling
- Linting and static analysis
- Webpack, rollup and parceljs



GitLab

Common challenges and emerging trends

Common challenges

- Performance
- Framework / language churn
- Reliability and safety
- Cross browser compatibility

Emerging trends

- Progressive web applications (PWAs)
- Compile to JS tools + Webassembly
- JAMStack + SSG
  - JAMStack: Javascript APIs and Markup
  - SSG: static site generators
  - In the javascript world GatsbyJS and NuxtJs are leading the way
  - There is also Hugo, Jekyll, Pelican and many others

GitLab

Hands on demonstration

Hands-on Learning (classroom activity)

Exploring responsive design

- Testing how a website works under different conditions can be tricky there are lots of tools online to help
- We can use these tools to debug and diagnose layout issues with our website or application
- Luckily most modern web browsers bundle some of these tools internally, specifically for responsive design we have
  - Chrome: Device mode simulator
  - Firefox: Responsive design mode

GitLab

Chrome: Device mode simulator

GitLab

Firefox: Responsive design mode

Homework Assignment

1. Read through the information and accessibility checklist from a11yproject, to get a general idea of some of the areas websites might fail to be accessible

2. Read through Responsive web design from, A List Apart

3. Spend some time looking through the example web pages on Mediaqueri.es and compare some of the tradeoffs that have been made at different sizes

4. Think of 3 websites you visit regularly, explore how each one performs on different browsers, at different sizes and maybe even on different devices (if you have another device available)
   - For firefox users: use responsive design mode to test
   - For chrome users: - use the device mode simulator
   - Did the websites meet your expectations on different devices? (Performance, layout and functionality)
   - Were there any areas you think could be improved?

5. TodoMVC compares different frontend frameworks by building the same simple TODO app
   - Try the VueJS version, now try the React version
   - Browse the source code for the VueJS implementation and the React implementation
   - Compare the 2 implementations, are there any things that stand out to you? Anything you find interesting?

Additional reading

- Spotify engineering - building spotifys new web player
- PWAs introduction
- SurviveJS - comparison of build tool
- Accessible by design
- Inclusive design

GitLab

Thank you!