# **Arunkumar** Srisailapathi

**Frontend** Engineer - Amazon

https://thearunkumar.github.io/

# Building Functional Webapps

Frontend

# Demo

Open Demo app

# Takeaways

1. HTML overview
2. CSS overview
3. Javascript overview
4. React overview
5. Functional React app (Client side)
6. Deploying to Heroku

# HTML

- A markup language
- Helps to structure the web page
- Browsers recognize HTML documents
    - When they are a **.html** (or) **.htm** files (or)
    - When the response from the server is of type **text/html**
- Follows semantics
- Can help with Accessibility.
- Doesn’t need any special software to program
    - Just a **text editor** to type
    - Any **web browser** to see

# Some common **HTML** tags

## **Form** related

- Text box **<input />**
- Button **<button></button>** (or) **<input type="button" />**
- Textarea **<textarea></textarea>**
- Radio button **<input type="radio" />**
- Checkbox **<input type="checkbox" />**
- Form **<form></form>**

## **Group** related

- div
- article
- aside
- header
- footer
- nav
- ul
- ol
- table
- span

## **Media** related

- img
- video
- audio
- canvas
- svg

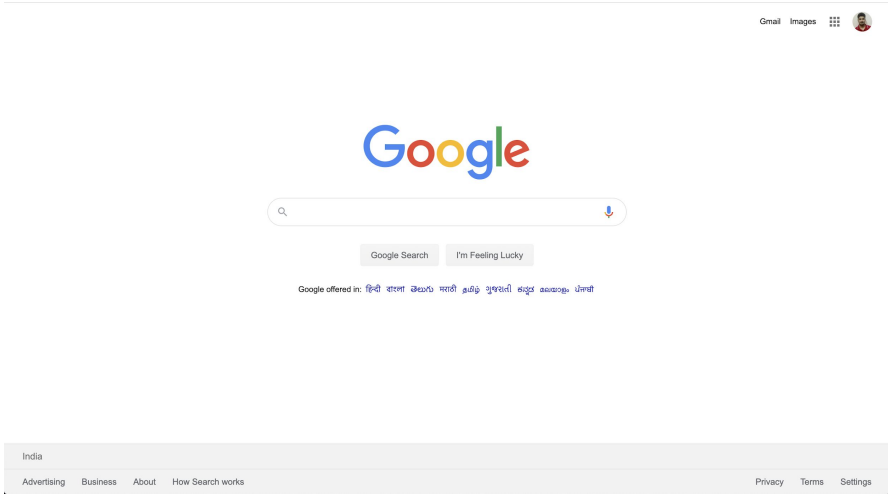
You can know about various tags from here - https://developer.mozilla.org/en-US/docs/Web/HTML/Element

# **Thinking** in HTML

Simply put,

The right side image can be structured as follows:

- **body** (whole page)
  - **section** (top) [can also be **header**]
    - **nav** (right side navigation)
  - **section** (middle)
    - **img** (image)
    - **input** (text box)
    - **button**
    - **button**
  - **section** (bottom) [can also be **footer**]
    - **nav** (some links for navigation)



# **Coding** in HTML

- **body** (whole page)
  - **section** (top) [can also be **header**]
    - **nav** (right side navigation)
  - **section** (middle)
    - **img** (image)
    - **input** (text box)
    - **button**
    - **button**
  - **section** (bottom) [can also be **footer**]
    - **nav** (some links for navigation)

Now, let's write that down in simple **HTML**

```
<body>
   <header>
      <nav>Some links</nav>
   </header>

   <section>
      <img src="path/to/image" alt="displayed when image
can't load" />
      <input type="text" value="" />
      <button>Google Search</button>
      <button>I'm feeling lucky</button>
   </section>

   <footer>
      <nav>
         Some links in footer
      </nav>
   </footer>
</body>
```

# CSS

- A **stylesheet** language for our markup content
- Enables separation of concerns from the actual content
- Browsers recognize CSS documents
    - When they are a **.css** files
    - When the response from the server is of type **text/css**
- Since it deals with only styles, it can scale to different screens, prints, ...
- Can help in **animations** and **transitions**, **responsiveness**
- Doesn't need any special software to program
    - Just a **text editor** to type
    - Any **web browser** to see

Some common **CSS selectors**

***** - Universal selector

**.**class - Class selector

**#**id - Id selector

elementName
(**body**, **span**, **div**, **button**, ...) - Element selector

**[type="text"]**,**[type="button"]**,
[someAttr="someValue"] - Attribute selector.

You can know about basics from here - https://developer.mozilla.org/en-US/docs/Web/CSS

# **Thinking** in CSS

Simply put, the right side content can be styled as follows:

**header**
- **nav**
  - **show on right side**

section**.middle**
- **start with 20% gap on top**
- **img**
  - **put the width as 200px (pixels)**
  - **put the height as 70px**
- **input**
  - **put the width as 500px**
  - **Put the height as 30px**
- **button**
  - **Width as 50px and height as 30px**
  - **show side by side**

**footer**
- **nav**

Our HTML as we wrote it,

- **body** (whole page)
  - **section** (top) [can also be **header**]
    - **nav** (right side navigation)
  - **section** (middle)
    - **img** (image)
    - **input** (text box)
    - **button**
    - **button**
  - **section** (bottom) [can also be **footer**]
    - **nav** (some links for navigation)

# **Coding** in CSS

**header**
- **nav**
  - **show on right side**

section**.middle**
- **start with 20% gap on top**
- **img**
  - **put the width as 200px (pixels)**
  - **put the height as 70px**
- **input**
  - **put the width as 500px**
  - **Put the height as 30px**
- **button**
  - **Width as 50px and height as 30px**
  - **show side by side**

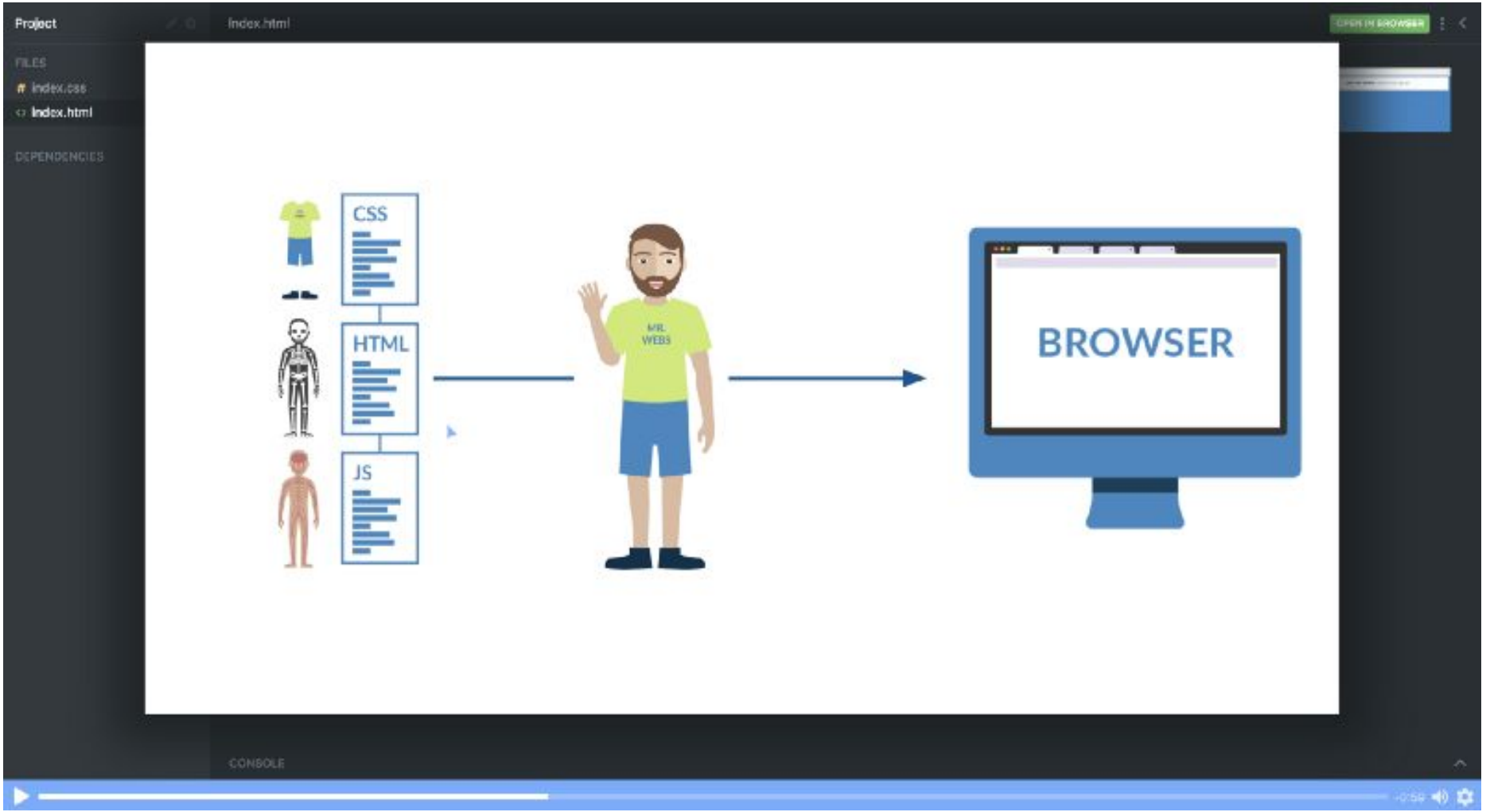
**footer**
- **nav**

```
header nav {
	margin-left: 80%;
}
section.middle {
	margin: 0 auto;
	margin-top: 15%;
	text-align: center;
}
section.middle img, section.middle input {
	display: block;
	margin: 0 auto;
	margin-bottom: 20px;
}
section.middle img {
	width: 200px;
	height: 70px;
}
section.middle input {
	width: 400px;
	height: 30px;
}
section.middle button {
	width: 150px;
	height: 30px;
}
footer {
	position: absolute;
	bottom: 0;
}
```


JS

- A **scripting** language for our static content
- Enables separation of concerns from the actual content
- Browsers recognize CSS documents
    - When they are a **.js** files
    - When the response from the server is of type **text/javascript**
- Basic JS doesn't need any special software to program
    - Just a **text editor** to type
    - Any **web browser** to see

You can know about basics from here - https://developer.mozilla.org/en-US/docs/Web/Javascript


Project
index.html
FILES
index.css
index.html
DEPENDENCIES
CSS
HTML
JS
MR. WEBS
BROWSER
CONSOLE

# JS libraries

- JavaScript is arguably the most widely used computer language in the world.
- The use cases where JavaScript has shown to be replacing traditional platforms are ever increasing.
- As the scope of the usage grows, it becomes extremely important that a method of distribution of code is used, to not only bring simplicity in further development but also allow reuse of code by the masses.
- JavaScript libraries solve this problem really well, and there are dozens if not hundreds of libraries available that cater to various use cases.

- jQuery
- YUI
- Backbone
- Angular
- React
- Redux
- Vue
- ....

npm

- Node package manager
- Packages and publishes libraries online that can be consumed by whole world.
- package.json

> npm install packageName

> npm install packageName --save

> npm install packageName --save-dev

# React

- A **Javascript library** (not a framework)
- Can support both mobile and web apps
- Since it is just a library, it’s main purpose is to render data to DOM
- Can use Redux or some other libraries for state management
- Can use react-router or some other libraries for Routing.
- We can write in **JSX** (or) using Javascript. But JSX is highly preferred and recommended.
- Virtual DOM

- Supports class Components
- Supports Functional Components
- React 16.8.0 and above supports Hooks

# Hands-on

# Let’s get started

Open Readme of Demo app for detailed steps

# Starter app

# Heroku deployment

# Clone Instagram app

# Next steps

We have built various features. Among them,

1. `Search` layout is present but not the capability of searching people.
2. Also, the `Heart` icon is present in the `Header` but it does nothing, you can add functionality to build to show notifications whenever some ones follows / likes / comments just like `Instagram` / `Facebook`

HINT for building Searching,

- API: https://clone-instagram-backend.herokuapp.com/users lists all the users.
- Auto complete: https://material-ui.com/api/autocomplete/

As users type characters, you can show the suggestions and the user can then pick the user based on the suggestions.

You can take reference from actual `Instagram` - https://instagram.com/ on how Search and Notification works.

# Building large scale apps

1. Core Architecture
    a. This ensures scalability
    b. Can extend to support various use cases
2. Performance
    a. Always first MEASURE MEASURE MEASURE.
    b. Only after you measure how your app is performing, you can improve
3. Tests
    a. Unit tests
    b. Functional
    c. End to end tests
4. CI / CD
5. Security

# Code maintenance

1. Have proper **linters**
2. Have auto formatters
3. Have auto fixers
4. Enforce same coding standards via plugins like **prettier**
5. Enforce **git hooks** (not to confuse with react hooks)
6. ...

# Helpful references

# HTML references

Basics:

- https://www.w3schools.com/html/html_basic.asp
- https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics
- https://www.geeksforgeeks.org/html-basics/
- https://html.com/

Advanced:

- https://developer.mozilla.org/en-US/docs/Learn/Accessibility/HTML
- codecademy.com/articles/ready-accessibility
- http://web-accessibility.carnegiemuseums.org/foundations/semantic/

# CSS references

Basics:

- https://www.w3schools.com/css/css_intro.asp
- https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/CSS_basics
- https://www.cssbasics.com/
- https://www.freecodecamp.org/news/get-started-with-css-in-5-minutes-e0804813fc3e/

Advanced:

- https://www.htmldog.com/guides/css/advanced/
- https://www.webfx.com/blog/web-design/3-advanced-css3-techniques-you-should-learn/
- https://marksheet.io/css-advanced.html

# JS references

Basics:

- https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics
- https://www.w3schools.com/js/
- https://javascript.info/first-steps
- http://speakingjs.com/es5/ch01.html

Advanced:

- https://johnresig.com/apps/learn/
- https://www.webfx.com/blog/web-design/6-advanced-javascript-techniques-you-should-know/
- https://htmldog.com/guides/javascript/advanced/

# React references

Basics:

- https://reactjs.org/tutorial/tutorial.html
- https://www.w3schools.com/react/
- https://reactnative.dev/docs/tutorial
- http://speakingjs.com/es5/ch01.html

Advanced:

- https://medium.com/yazanaabed/advanced-react-patterns-7326f5a5ad1b
- https://egghead.io/courses/advanced-react-component-patterns
- https://advancedreact.com/
- https://www.freecodecamp.org/news/everything-you-need-to-know-about-react-eaedf53238c4/

# Thank you!