avionschool

Lesson 1 Introduction to Web Development

BATCH 4

NOVEMBER 9, 2020

Brief History of the World Wide Web or Web

...30 years...

https://appinstitute.com/history-of-the-web-infographic/

Activity: What does it mean to be a...

Software Engineer? Web Developer? Programmer?

Full Stack? Back End? Front End? Senior? Mid-level? Junior? Entry-Level? Associate?

Software Architect? Solutions Architect? Consultant? Tech Lead? Team Lead?

Database? API? Application? Mobile App? Tester? QA? Analyst? Dev Ops? SEO Specialist?

...

C? C#? Objective C? .NET? Java? Spring Boot? Grails? PHP? Symfony? CodeIgniter? Python? Django? Flask? Golang? **Ruby on Rails?** JamStack? MERN? NodeJS? RESTful APIs? GraphQL? Nginx? Redis? Elasticsearch? Memcache? Apache? Docker? Jenkins? MySQL? Oracle PL/SQL Postgres? MongoDB? NoSQL? **HTML? CSS? Javascript? ES6? React?** XML? AJAX? SASS? JQuery? TypeScript? Redux? Relay? Webpack? Gulp? Angular? Vue? Ember? Backbone? Gatsby? NextJS? Nuxt JS? CMS? WordPress? Shopify? Laravel? Android? IOS? React Native? Flutter? Web Designer? Interaction? UI? UX? Product? Figma? CI / CD? SVN? Git? Kafka? Lambda? Agile? Scrum? Kanban? OOP? SOLID? TDD? Unit-testing? Integration? End-to-End? Media Query? Responsive Design? Serverless? Mobile-First? PWA?

...

Rockstar? Ninja? Webmaster?

Software Engineer? Web developer?

SOFTWARE ENGINEER

- A software engineer is someone who writes computer code in order to build or maintain software.

SKILLS THAT MAKE A GOOD SOFTWARE ENGINEER

- Knowledge of computer systems and programming languages
- Ability to work and learn autonomously
- Ability to solve unique unfamiliar problems.
- Ability to stay calm in the face of a complex problem.

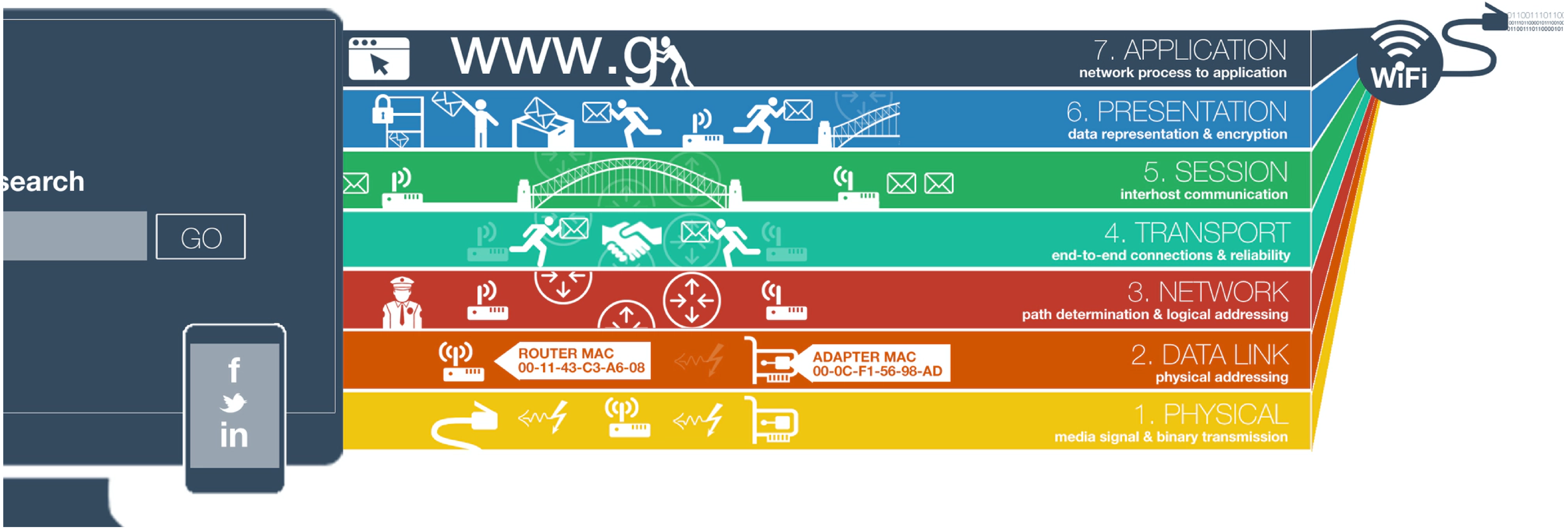
WEB DEVELOPER

- A Web developer is a kind of programmer who specializes in the development of applications relating to the World Wide Web

DOMAINS

- Full-stack, Backend, Frontend, Mobile, DevOps, UI / UX

How the Web works



7. APPLICATION
network process to application

6. PRESENTATION
data representation & encryption

5. SESSION
interhost communication

4. TRANSPORT
end-to-end connections & reliability

3. NETWORK
path determination & logical addressing

2. DATA LINK
physical addressing

1. PHYSICAL
media signal & binary transmission

How the Web works

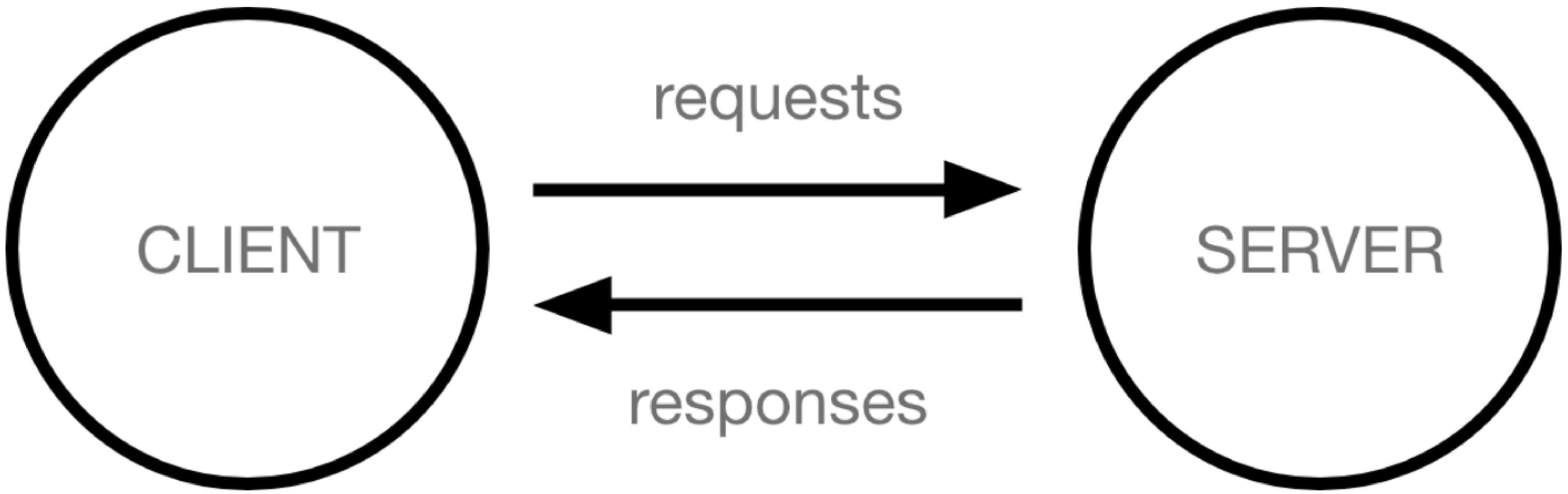
CLIENTS

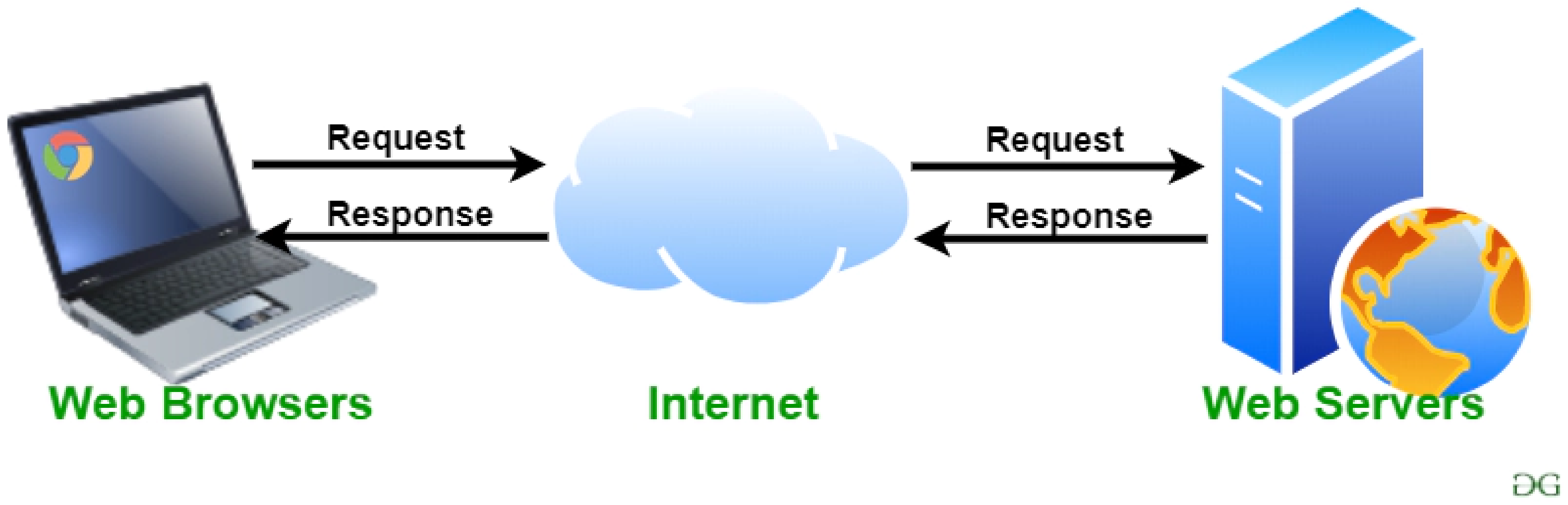
- are the typical web user's internet-connected devices (for example, your computer connected to your Wi-Fi, or your phone connected to your mobile network) and web-accessing software available on those devices (usually a web browser like Firefox or Chrome).

SERVERS

- are computers that store webpages, sites, or apps. When a client device wants to access a webpage, a copy of the webpage is downloaded from the server onto the client machine to be displayed in the user's web browser.

How the Web works





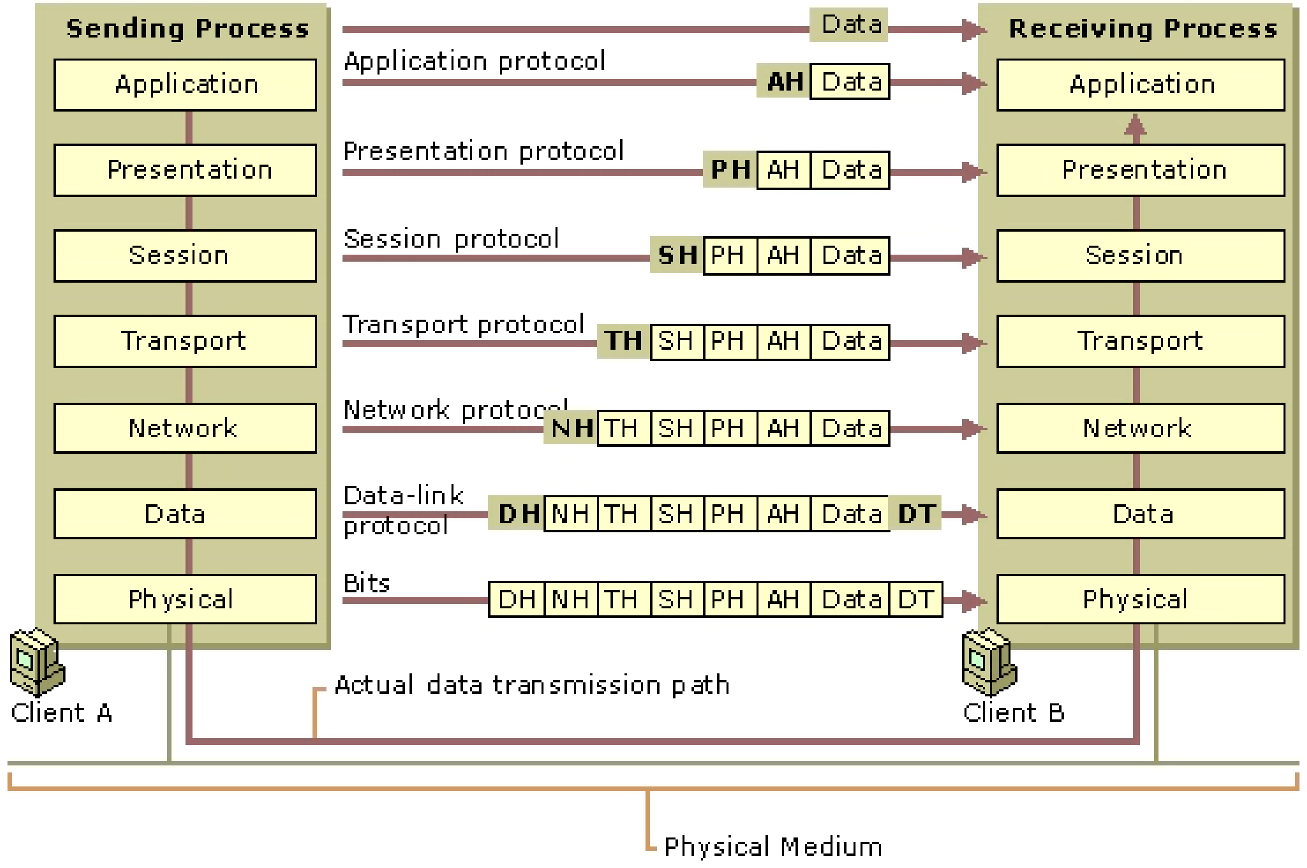
How the Web works

YOUR INTERNET CONNECTION

- Allows you to send and receive data on the web. It's basically like the street between your house and the shop

TCP/IP

- Transmission Control Protocol and Internet Protocol are communication protocols that define how data should travel across the internet. This is like the transport mechanisms that let you place an order, go to the shop, and buy your goods. In our example, this is like a car or a bike (or however else you might get around)



Sending Process
Application
Presentation
Session
Transport
Network
Data
Physical
Client A

Data
Application protocol — AH Data
Presentation protocol — PH AH Data
Session protocol — SH PH AH Data
Transport protocol — TH SH PH AH Data
Network protocol — NH TH SH PH AH Data
Data-link protocol — DH NH TH SH PH AH Data DT
Bits — DH NH TH SH PH AH Data DT

Receiving Process
Application
Presentation
Session
Transport
Network
Data
Physical
Client B

Actual data transmission path
Physical Medium

DNS

- Domain Name Servers are like an address book for websites. When you type a web address in your browser, the browser looks at the DNS to find the website's real address before it can retrieve the website. The browser needs to find out which server the website lives on, so it can send HTTP messages to the right place (see below). This is like looking up the address of the shop so you can access it

How the Web works

HTTP/S

Hypertext Transfer Protocol is an application protocol that defines a language for clients and servers to speak to each other. This is like the language you use to order your goods.

COMPONENT FILES

- A website is made up of many different files, which are like the different parts of the goods you buy from the shop. These files come in two main types:

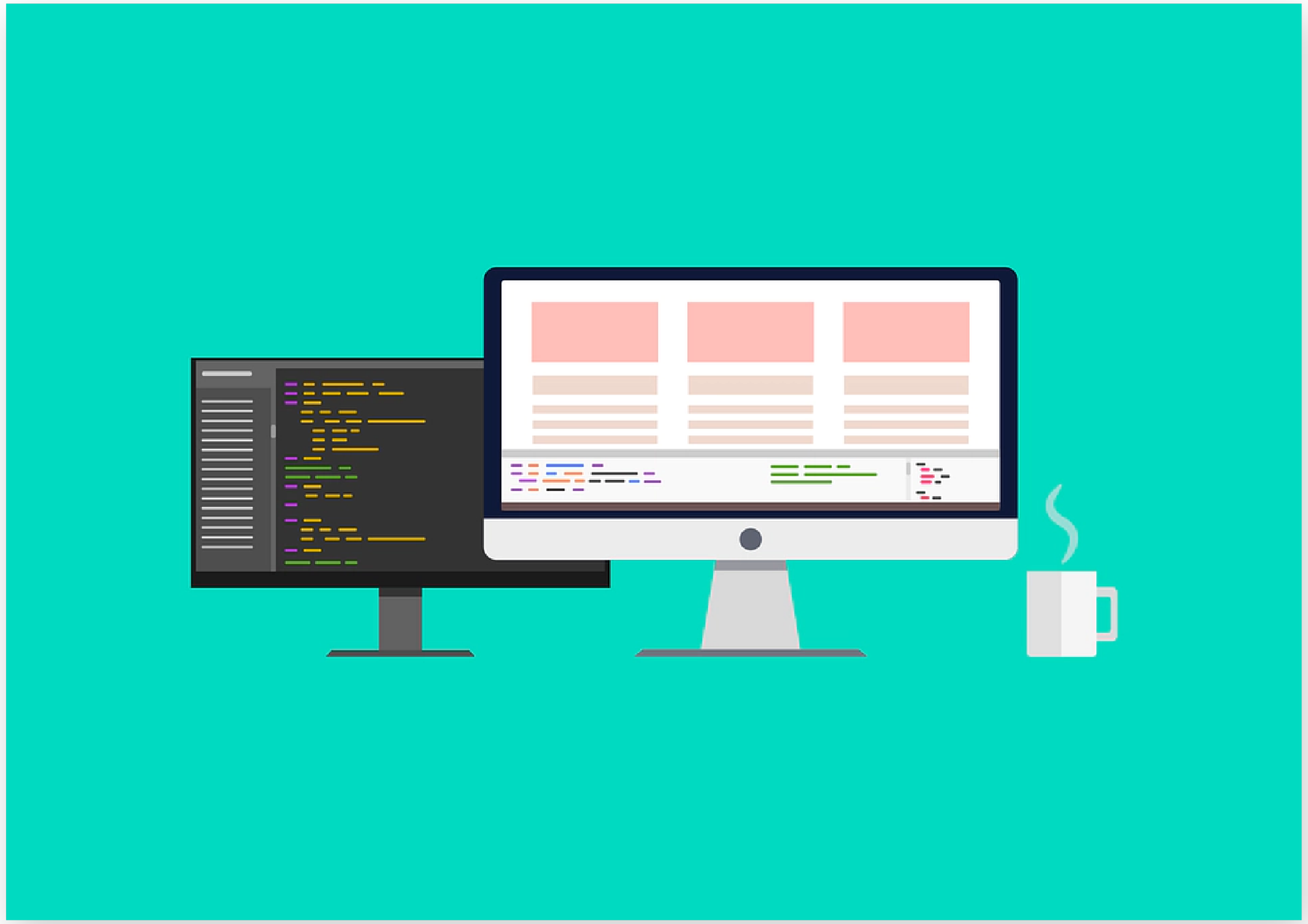
  #### CODE FILES

  - Websites are built primarily from HTML, CSS, and JavaScript, though you'll meet other technologies a bit later.
  - This is a collective name for all the other stuff that makes up a website, such as images, music, video, Word documents, and PDFs.

The Technology Stack

FRONT-END

- This is the interface that a developer creates for interaction with the main application. This can be as simple as a static web page designed in HTML or a complex implementation done using a technology based on virtual dom (such as React.js) for browsers or a mobile application developed natively using android and iOS technologies or a hybrid platform such as React Native.
- HTML, CSS, JS, React.js, Vue.js, React Native



The Technology Stack

BACK-END

- This refers to the server side master program running on single or distributed topology and responsible for catering to the stateful and stateless requests sent from the front-end.
- Node.js, Python, PHP, Ruby, Java, Elixir

Programming

- Just like learning another language: knowing the syntax or grammar, and learning by doing.

HOW TO LEARN

- docs, books, videos
- tools: vscode, git, browsers,
- community: stackoverflow, reddit, avion people

MDN web docs moz://a

stackoverflow

avion

YouTube

Medium

What to learn? Programming & Languages

FRONT-END

- HTML5
- CSS3
- JavaScript

BACK-END

- Ruby on Rails

Assignment

Today I Learn / Best Practice Page

- Find a concept, an idea, or a thought online about a best practice related to web development or software engineering that can be summarized into a statement. Its source can be just be a tweet, or a whole blog post, or from a book, etc.
- Use the statement as a title, then add a description explaining it in your own words, just like how you did in the pre-course's Feyman writing prompts.
- Present this using HTML & CSS on an HTML page.
- Include the link of the source in the HTML page (however you choose to cite it).
- Push your work to a new repo in your Github account, then send me the link in our Slack channel.
- This is your chance to get adventurous and creative so we expect you to immerse yourself in HTML & CSS alongside web development concepts.
- An example: https://react-file-structure.surge.sh/