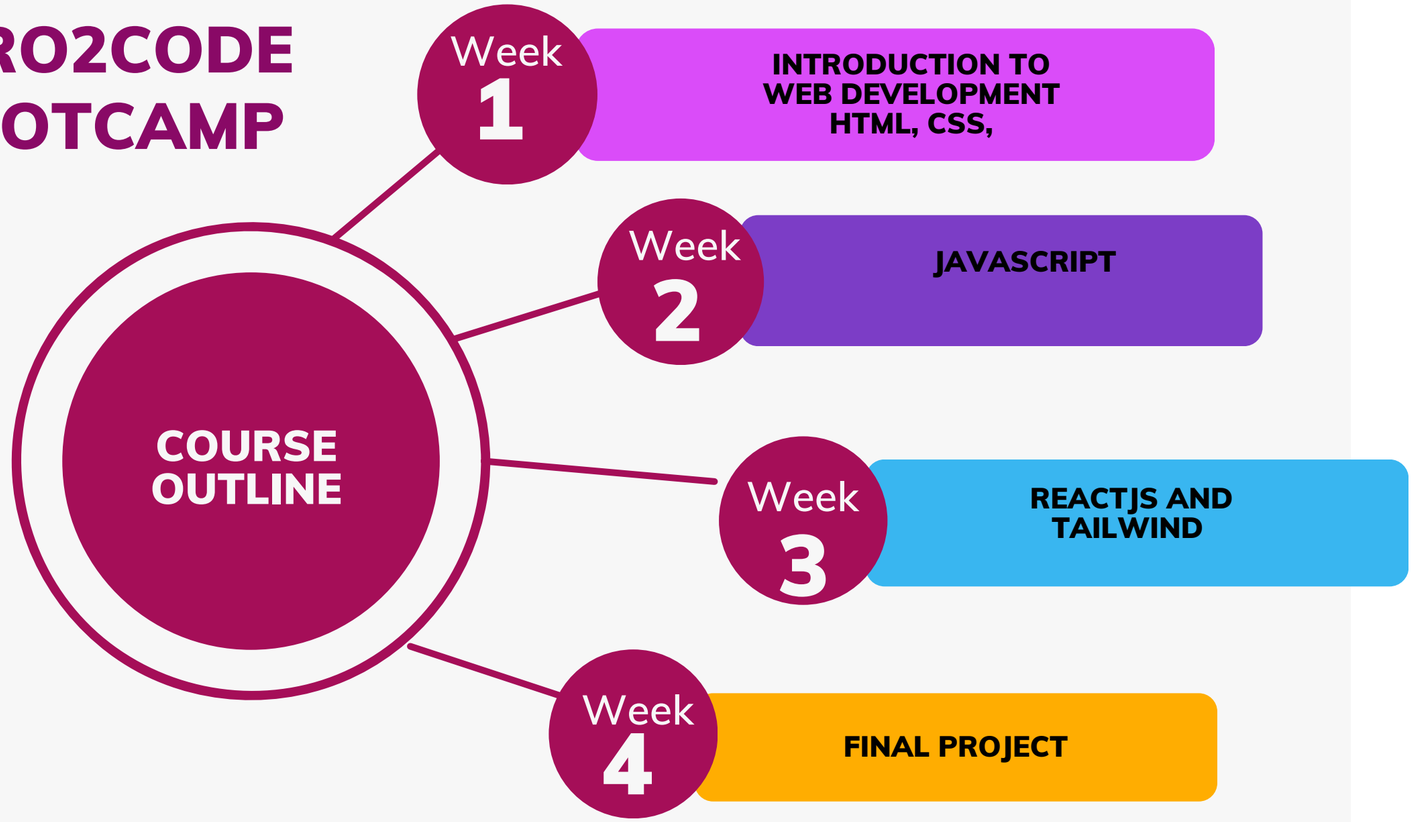
ZERO2CODE BOOTCAMP



PREREQUISITES

- Computer
- Internet Connection
- Github Account

WEEK 1 : INTRODUCTION TO WEB DEVELOPMENT

- The interenet
- Web architecture
- Front-end and Backend
- Computer Programming Language
- Frameworks
- Databases
- Environment Setup
- HTML & CSS

WEEK 2 : JAVASCRIPT

- Introduction to JavaScript Basics
- Basic Syntax and Data Types
- Operators and Expressions
- Conditional Statements
- Loops
- Functions
- Arrays
- DOM Manipulation

WEEK 3 : REACTJS

- Introduction to React
- React Component Basics
- State and Props
- Events
- Conditional Rendering
- Lists and Keys
- Forms in React
- Router Components
- Fetching Data

WEEK 4 : FINAL PROJECT

- Project

CLASS DAY1

PREREQUISITES

- Computer / Smart Phone
- Internet Connection
- Github Account

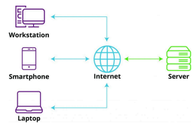
The Internet

- The internet is a global network of interconnected computers and servers that communicate with each other to share data, resources, and information. It allows users to access and exchange information, engage in communication, and utilize various online services.
- At its core, the internet functions through the use of protocols and technologies that enable data transfer between devices. Some of the key components of the internet include:
- Websites and Webpages: Websites are hosted on servers and accessed through browsers (like Chrome, Firefox, etc.) using URLs. Each website is made up of pages that display text, images, videos, and other content.
- Protocols: The internet operates using protocols like the Transmission Control Protocol (TCP) and the Internet Protocol (IP), which govern how data is transmitted and routed across networks.
- IP Addresses: Every device connected to the internet is assigned a unique IP address, which allows other devices to find and communicate with it.
- World Wide Web (WWW): The World Wide Web is a system of interlinked hypertext documents (web pages) that are accessed via the internet. The web is just one part of the internet, alongside services like email, file sharing, and instant messaging.
- Services: The internet supports a wide range of services, including social media, online banking, cloud computing, gaming, video streaming, and much more.
- The internet has revolutionized communication, commerce, education, entertainment, and many other aspects of daily life. It's the infrastructure that enables vast online ecosystems, connecting billions of people and devices across the world.

Web Browser

- A web browser is a software application that allows users to access, view, and interact with content on the World Wide Web. It retrieves web pages from servers and displays them on your device. Web browsers also allow you to navigate between different websites and services using hyperlinks and search engines.

Web Architecture



Front-end and Back-end

Front-End
The front-end is everything that users interact with directly in their web browser. It's the user interface (UI) and the user experience (UX) of a website or application. Essentially, the front-end is what users see and engage with.
Key Technologies:
1. HTML (HyperText Markup Language): The structure of web pages (e.g., headings, paragraphs, links, etc.).
2. CSS (Cascading Style Sheets): Controls the layout, design, and appearance of the web pages (e.g., colors, fonts, spacing).
3. JavaScript: Adds interactivity and dynamic behavior to web pages (e.g., animations, form validation, data updates without reloading the page).
Front-End Frameworks and Libraries:
- React: A JavaScript library for building user interfaces, particularly single-page applications (SPAs).
- Vue.js: A progressive framework for building UIs and single-page applications.
- Angular: A comprehensive front-end framework for building dynamic web applications.
- Bootstrap: A popular framework that provides pre-designed UI components and layout structures.

Responsibilities:
- Building and designing the visual elements of a site (buttons, menus, images, etc.).
- Ensuring responsiveness (making sure websites work on different devices and screen sizes).
- Handling user interactions like clicks, form submissions, and navigation.

Back-End
The back-end refers to the server side of a web application. It's responsible for managing and processing data, handling requests from the front-end, and sending responses back to the user. The back-end is not visible to users, but it powers the functionality behind the scenes.
Key Technologies:
1. Server-Side Languages:
   - Node.js (JavaScript runtime environment).
   - Python (using frameworks like Django or Flask).
   - Ruby (using the Ruby on Rails framework).
   - PHP (widely used for web development).
   - Java (used with Spring framework for large-scale applications).
2. Databases:
   - SQL Databases (e.g., MySQL, PostgreSQL, SQLite): Store structured data in tables.
   - NoSQL Databases (e.g., MongoDB): Store unstructured or semi-structured data.
3. Web Servers:
   - Apache and Nginx: Serve content from the server to the front-end and handle HTTP requests.
4. APIs (Application Programming Interfaces): Allow communication between the front-end and back-end or between different systems (e.g., RESTful APIs, GraphQL).

Responsibilities:
- Database Management: Storing and retrieving data from databases (e.g., user accounts, products, posts).
- Server Logic: Handling business logic, processing requests, and ensuring that the application functions correctly.
- Authentication and Authorization: Verifying user identities (login systems) and controlling access to data.
- API Creation: Exposing data to the front-end or other services via APIs for integration (e.g., sending user information to the browser).

How Front-End and Back-End Work Together
- Front-End: Sends requests (such as user actions like submitting a form) to the Back-End.
- Back-End: Processes those requests, interacts with databases or other services, and sends a response (such as data or confirmation) back to the Front-End.
- Front-End: Updates the user interface based on the response (e.g., displaying new information or confirming an action).

For example, when you log into a website, the front-end collects your username and password, then sends them to the back-end for verification. The back-end checks the credentials in a database, and if they match, it sends a response to the front-end to grant access.
Full-Stack Development
Full-stack developers work on both the front-end and back-end, creating and maintaining all layers of a web application. Full-stack developers have a broad understanding of how the front-end and back-end work together to provide a seamless user experience.

Resources

1. What is the Internet?
- What is the Internet? (by Webopedia): A comprehensive guide to understanding the Internet. Webopedia: What is the Internet?
- How the Internet Works (by HowStuffWorks): An easy-to-understand explanation of the Internet's inner workings. HowStuffWorks: How the Internet Works

2. What is a Web Browser?
- What is a Web Browser? (by Mozilla): A detailed guide explaining the basic functions and popular browsers. Mozilla: What is a Web Browser?
- Web Browser Types & Features (by TechRadar): An overview of the most common browsers and their features. TechRadar: Web Browser Comparison

3. Front-End Development
- Front-End Development Overview (by MDN Web Docs): A comprehensive introduction to front-end technologies. MDN Web Docs: Front-End Web Development
- Front-End Development Resources (by freeCodeCamp): Tutorials and learning paths for becoming a front-end developer. freeCodeCamp: Front-End Development
- Popular Front-End Frameworks and Libraries:
  - React Official Docs: React
  - Vue.js Official Docs: Vue.js
  - Angular Official Docs: Angular

4. Back-End Development
- Back-End Web Development Guide (by MDN Web Docs): A deep dive into back-end technologies and practices. MDN Web Docs: Back-End Web Development
- Learn Back-End Development (by freeCodeCamp): A learning path for back-end web development, including APIs, databases, and more. freeCodeCamp: Back-End Development
- Back-End Technologies:
  - Node.js Official Docs: Node.js
  - Django Official Docs (Python): Django
  - Ruby on Rails Docs: Ruby on Rails

5. Full-Stack Development
- What is Full-Stack Development? (by MDN Web Docs): A complete guide to full-stack development. MDN Web Docs: Full-Stack Web Development
- Full-Stack Development Resources (by freeCodeCamp): Learn how to build both the front-end and back-end of applications. freeCodeCamp: Full-Stack Development

6. Web Development (Front-End + Back-End)
- Web Development for Beginners (by W3Schools): An easy-to-follow guide for starting web development, covering both front-end and back-end. W3Schools: Web Development
- Full-Stack Web Development Path (by The Odin Project): A free, comprehensive curriculum for learning full-stack web development. The Odin Project: Full-Stack