

---

<Dylan Vidal />

# Web Development

## <101 />





# Workshop Objectives



1. Attendees will understand the purpose of modern web tools by comparing them to their non-modern counterparts.
  2. Attendees will understand the difference between client and server, and how these concepts play into website rendering.
  3. Attendees will get exposed to modern web technologies to reach for them during personal projects.
-



# Table of contents



01

## Building a Site

The skeletal structure of a website.

02

## Concepts

Understand why we do what we do.

03

## Servers

How do we serve a website?

04

## Deployment

Hosting a website server.

---

<Web Development 101 />



# 01

## <// Building a Site //>

Skeletal Structure of a Website

---



# Main Tools (for now)



## JSX

A powerful extension  
on top of HTML.



## Tailwind

CSS without all of the  
headache.



## React

A state management  
system for JS/TS.

---

<The Bones of a Website />

# How is a website structured?



# HTML

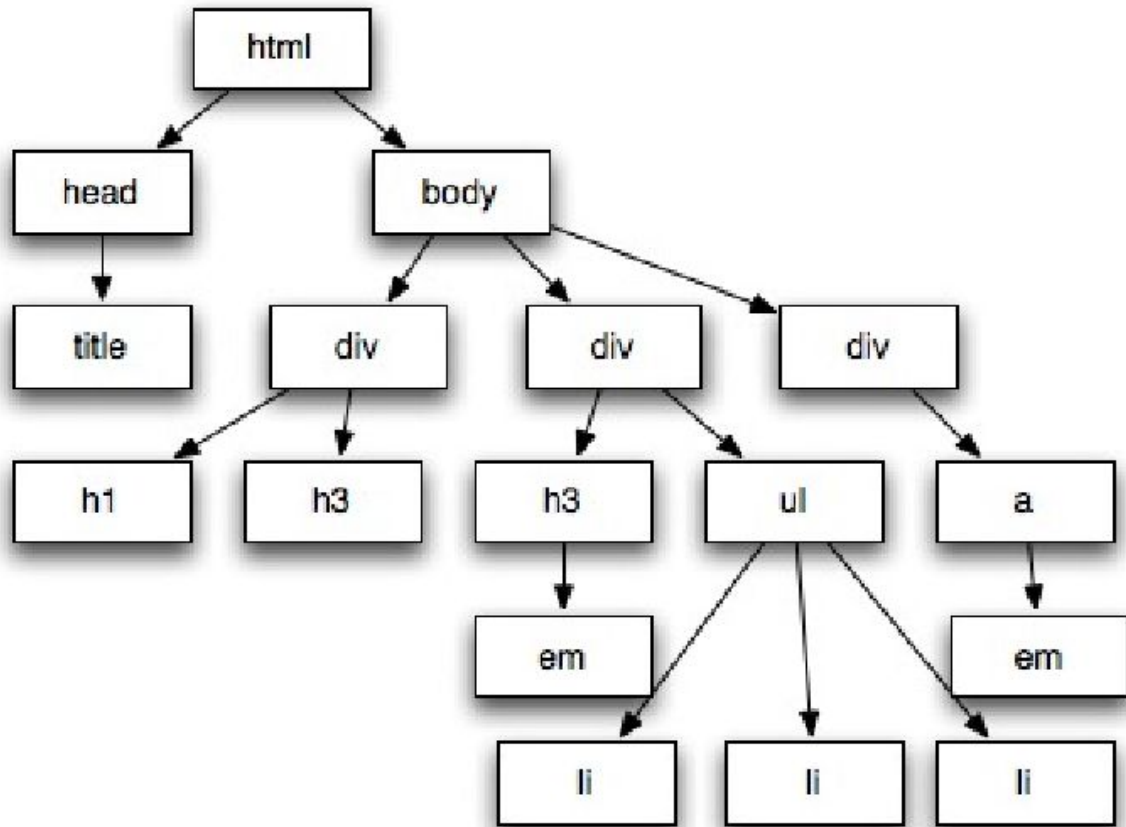


## HTML



### Hypertext Markup Language

- A series of nested elements that represent a website.
- Can be represented as a tree of parents, and children.
- Redundant, and static. No way of serving data or state.







```
HTML Sample

<!-- ***** ABOUT / PROFILE ***** -->
<header>
  <div class="navigation-bar">
    <a href="#" class="logo">My Portfolio</a>

    <nav>
      <a href="#about-me">About Me</a>
      <a href="#projects">Projects</a>
      <a href="#hackathons">Hackathons</a>
      <a href="#experience">Experience</a>
      <a href="#skills">Skills</a>
      <a href="#education">Education</a>
      <a href="#contact">Contact</a>
    </nav>
  </div>

  <section id="about-me" class="item-details main-section">
    <div class="content-wrap hidden profile">
      <h1>Dylan J. Vidal</h1>
      <h2>Aspiring Software Engineer and Computer Science Student</h2>
      
      <p>Recently graduated from a program that allows me to attain an Associate Degree
at 18 with full-time dual-enrollment, I have found a unique passion for software engineering,
machine learning, and higher level mathematics. In the past few semesters, I have taken on
rigorous programming courses and found both talent and joy in the subject. Hoping to become a
back-end developer or machine learning engineer, I continue to strive for academic excellence
and career readiness in the field.</p>
      <p>I am a current Class of 2027 Freshman at the University of Central Florida, and
I am involved in a variety of organizations that help me achieve my mission, including the
Competitive Programming Team and KnightHacks Software Development Club. I also plan on
participating in a handful of Hackathons in the upcoming months, which I am excited to utilize
for professional and personal development.</p>
    </div>
    <div class = "spacer layer1"></div>
  </section>
</header>
```



```
<!-- ***** PROJECTS / PORTFOLIO ***** -->
<section id="projects" class="projects item-details main-section">
  <div class="content-wrap divider">
    <h2>Featured Projects</h2>
    <p>View selected projects below. More information can be found on my <a
href="https://github.com/DVidal1205/Projects">Github</a>.</p>

    <!-- Project WildSpace -->
    <section class="project-item hidden">
      
      <h3>Project WildSpace (Python, Qt Creator, LangChain)</h3>
      <p>A Dungeons and Dragons inspired Generative AI tool built to assist with
worldbuilding. Built using Pyside6, Qt Creator, and LangChain, this GPT-3.5 powered
application reaches into the Wildspace between realms and delivers an entirely unique entity
to enrich your fantasy world. Developed for the KnightHacks 2023 Hackathon held by the
University of Central Florida, and is publicly hosted for use by the D&D community.</p>
      <a class="btn" href="https://www.projectwildspace.tech" target="_blank">Visit
Project WildSpace</a>
    </section>

    <!-- Discord Bot -->
    <section class="project-item hidden">
      
      <h3>UCF Campus Activities Discord Bot (Python, Nextcord, Selenium)</h3>
      <p>Created a Discord Bot using Python that executes a daily web-scraping process
of the Campus
        Event Board website using Selenium and embeds the events in a designated
channel.
        Deployed to an official campus discord server with 1000+ members interacting
with the bot and
        hosted on a Raspberry Pi to ensure 24-hour uptime.
      </p>
      <a class="btn" href="https://github.com/DVidal1205/UCF27-Bot"
target="_blank">Github Repository</a>
    </section>

    <!-- Wordle -->
    <section class="project-item hidden">
      
      <h3>Wordle - The New York Times (C++, Qt Creator)</h3>
      <p>Recreated the Wordle game from The New York Times, coded in C++ using a Qt
Graphical User Interface. The game logic utilizes complex methods such as object-oriented
programming practices, UML diagram construction, two-dimensional arrays, random file access
methods, and dynamic memory allocation.</p>
      <a class="btn" href="https://github.com/DVidal1205/Projects/tree/main/WordleQt"
target="_blank">Github Repository</a>
    </section>
```





# JSX



JSX

## JavaScript XML / JavaScript Syntax Extension

- The prodigal love child of HTML and JS.
- Functions the same way as HTML does, but allows for JavaScript IN the markup.

```
const skills = [
  { name: "C", svg: "/c.svg" },
  { name: "C++", svg: "/cpp.svg" },
  { name: "Qt", svg: "/qt.svg" },
  { name: "Python", svg: "/python.svg" },
  { name: "Selenium", svg: "/selenium.svg" },
  { name: "OpenAI API", svg: "/openai.svg" },
  { name: "Discord API", svg: "/discord.svg" },
  { name: "HTML", svg: "/html.svg" },
  { name: "CSS", svg: "/css.svg" },
  { name: "JavaScript", svg: "/javascript.svg" },
  { name: "TypeScript", svg: "/typescript.svg" },
  { name: "React", svg: "/react.svg" },
  { name: "Tailwind CSS", svg: "/tailwind.svg" },
  { name: "Node", svg: "/node.svg" },
  { name: "Flask", svg: "/flask.svg" },
  { name: "Next.js", svg: "/next.svg" },
  { name: "Vercel", svg: "/vercel.svg" },
  { name: "Git", svg: "/git.svg" },
  { name: "GitHub", svg: "/github-mark.svg" },
  { name: "VS Code", svg: "/vscode.svg" },
];
```

```
<div className="mb-20 px-6" id="skills">
  <div className="text-3xl font-bold pt-24">Skills</div>
  <p className="my-2 mb-8">
    A collection of some of the technologies I have worked with.
    Hover for more details.
  </p>
  <div className="grid grid-cols-2 sm:grid-cols-3 md:grid-cols-4 lg:grid-cols-5 gap-
8 mt-6">
    {skills.map((skill, index) => (
      <div
        key={index}
        className="relative flex justify-center items-center"
        onMouseEnter={() => setHoveredSkill(skill.name)}
        onMouseLeave={() => setHoveredSkill(null)}
      >
        <Image
          width={100}
          height={100}
          src={skill.svg}
          alt={skill.name}
          className="transition-transform duration-300 ease-in-out transform
hover:scale-110"
        />
        <div
          className={`absolute inset-0 flex items-center justify-center
bg-gradient-to-tl from-purple-950 via-purple-800 to-violet-950
bg-transparent backdrop-blur-xl mshadow-md drop-shadow-2xl
border-2 border-violet-900 rounded-2xl
transition-opacity duration-300 ease-linear`}
          ${
            hoveredSkill === skill.name
              ? "opacity-100"
              : "opacity-0"
          }
          cursor-default`
        >
          {hoveredSkill === skill.name && (
            <span className="text-xl">{skill.name}</span>
          )}
        </div>
      </div>
    )})
  </div>
</div>
```





#### Conditional HTML

```
<CardFooter className="flex flex-col items-start space-y-2 md:flex-row md:justify-between md:space-x-0">
  {subscriptionPlan.isSubscribed ? (
    <Button type="submit">
      Manage Subscription
    </Button>
  ) : (
    <Link
      href="/pricing"
      className={buttonVariants({ size: "lg" })}
    >
      Upgrade
    </Link>
  )}
</CardFooter>
```

---

<The Skin of a Website />

# How is a website styled?





# CSS



CSS



## Cascading Style Sheets

- Primary purpose of styling elements in HTML.
- Create styled classes, which you then apply to HTML.
- Has media queries to change styles based on screen size (responsiveness).





```
CSS File

/* Education
-----*/

.education {
  background-image: none;
  background-color: #0b090a;
  color: #F7FFF7;
  padding-bottom: 100px;
  margin-bottom: -50px;
}

.education p {
  width: 60%;
}
```



```
HTML File

<!-- ***** EDUCATION & CERTIFICATIONS ***** -->
<section id="education" class="education item-details main-section">
  <div class="content-wrap">
    <h2>Education</h2>
    <p>For more details on clubs, activities, or accolades, visit my <a
href="https://www.linkedin.com/in/dylanvidal1204/">LinkedIn</a>.</p>

    <section class="hidden">
      <h3>University of Central Florida - Orlando, FL</h3>
      <p>August 2023 - May 2027</p>
      <p>Bachelor's Degree - Computer Science</p>
      <p>At UCF, I am pursuing a Bachelor's Degree in Computer Science. I am
a proud Honors Scholar of the Burnett Honors College, and strive for academic
excellence in everything I do. I am heavily involved with the university's
Competitive Programming Team, frequently attending lectures, practices, and mock
contests. I am also a proud member of the KnightHacks software development club,
which hosted our schools success story of a hackathon. </p>
    </section>

    <section class="hidden">
      <h3>Broward College - Davie, FL</h3>
      <p>August 2021 - May 2023</p>
      <p>Associate in Arts Degree - Computer Science</p>
      <p>Full-time dual enrollment focused on computer science and
mathematics.</p>
    </section>

    <section class="hidden">
      <h3>College Academy at Broward College - Davie, FL</h3>
      <p>August 2021 - May 2023</p>
      <p>High School Diploma - Computer Science</p>
      <p>At CA@BC, I started college 2 years early while simultaneously
continuing my high-school education.</p>
    </section>

  </div>
</section>
```





```
Responsiveness

/* Responsive
-----*/
@media screen and (min-width: 750px) {
  header, footer {
    text-align: center;
  }

  .project-item img {
    float: left;
    margin-right: 20px;
  }

  .job-item {
    display: grid;
    grid-template-columns: 1fr 2fr;
    column-gap: 20px;
  }

  .skill-item {
    display: grid;
    grid-template-columns: 1fr 2fr;
    column-gap: 20px;
  }

  .contact-list {
    display: flex;
    justify-content: center;
  }

  .contact-list a {
    padding: 15px;
  }
}
```



# Tailwind CSS



## Cascading Style Sheets

- In-Line styling solution, with live refresh (EXCELLENT FOR LEARNING).
- Opinionated Design System and Color Palette.
- Utility Classes.
- Bundled and Compiled for Optimization.





<https://tailwindcss.com/>



```
<Image
  src="/logo.png"
  height={1080}
  width={1080}
  alt="Project Wildspace Logo"
  className="aspect-auto h-[112] w-auto"
></Image>

<div className="mb-32 max-w-5xl sm:mt-24"
  <div className="mb-12">
    <h2 className="mt-2 font-bold text-4xl text-foreground sm:text-5xl">
      Start world building in minutes
    </h2>
    <p className="mt-4 text-lg text-foreground">
      Creating a world is hard. We&apos;ve made it easy.
    </p>
    <div className="my-12 border border-primary rounded-lg drop">
      <Image
        src="/demo.png"
        height={1080}
        width={1920}
        alt="Wildspace Demo"
        className="rounded-lg"
      />
    </div>
  </div>
```

```
.text 4xl {
  font-size: 2.25rem/* 36px */;
  line-height: 2.5rem/* 40px */;
}
```

```
<Image
  src="/logo.png"
  height={1080}
  width={1080}
  alt="Project Wildspace Logo"
  className="aspect-auto h-[112] w-auto"
></Image>

<div className="mb-32 max-w-5xl sm:mt-24">
  <div className="mb-12">
    <h2 className="mt-2 font-bold text-4xl text-foreground sm:text-5xl">
      Start world building in minutes
    </h2>
    <p className="mt-4 text-lg text-foreground">
      Creating a world is hard. We&apos;ve made it easy.
    </p>
    <div className="my-12 border border-primary rounded-lg drop">
      <Image
        src="/demo.png"
        height={1080}
        width={1920}
        alt="Wildspace Demo"
        className="rounded-lg"
      />
    </div>
  </div>
```

```
@media (min-width: 640px) {
  .sm\:text-5xl {
    font-size: 3rem/* 48px */;
    line-height: 1;
  }
}
```

```
height={1024}
width={1024}
src={
  image
  ? `data:image/png;base64,${image}`
  : ""
}
alt="factio
className={
  isImage
  ? "h-[85vw] md:h-[85vh] w-auto"
  : ""
}
></Image>
</div>
```

```
.h-[85vw] {
  height: 85vw;
}
```



# Learning CSS



## Things to Learn

- Box Model for Margin and Padding
  - Layouts
    - Flex
    - Grid
  - Positioning
    - Absolute, Relative, etc.
  - Colors
-

# CSS FLEXBOX CHEATSHEET

BY @Hasanstack

## Flex Container

Row

1 2 3

Row - reverse

3 2 1

1

2

3

Column

1

2

3

Reverse

## Flex Items

Cross axis

Main axis

Flex Item

Flex Item

Flex Item

Flex Items Order

3 2 1

1 2 3

## Justify Content

Flex-start

Space - between

Flex-end

Space - around

Center

Space - evenly

## Flex Grow

how each child grow in remaining space

1

2

3

## Flex Shrink

set amount of flex item shrink relative to others

1 2 3

1 2 3

## Flex Basis

default size of a flex item

First item 20%

Second item 40%

Align Self

allows the default alignment

Flex-start

Flex-end

center

stretch

## Align-Content

Flex-start

Flex-end

center

normal

Margin Left

Border Left

Padding Left

Margin Top

Border Top

Padding Top

Content

Padding Right

Border Right

Margin Right

Padding Bottom

Border Bottom

Margin Bottom

---

<The Brain of a Website />

# How is a website reactive?



# JS



## JavaScript

- How a website “does” things.
- Depends on a client (browser).
- Works by targeting HTML elements, and updating them programmatically.
- Event Driven / Event Bound





#### JavaScript Animation Observer

```
const observer = new IntersectionObserver((entries => {
  entries.forEach((entry) => {
    console.log(entry)
    if (entry.isIntersecting) {
      entry.target.classList.add('show');
    }
  });
}));

const hiddenElements = document.querySelectorAll('.hidden');
hiddenElements.forEach((el) => observer.observe(el));
```



```
Adaptive Navbar

let sections = document.querySelectorAll('.main-section');
let navLinks = document.querySelectorAll('header nav a');

window.onscroll = () => {
  let scrollTop = window.innerHeight + window.pageYOffset;
  let pageHeight = document.documentElement.scrollHeight;

  if (scrollTop >= pageHeight - 2) {
    navLinks.forEach(link => {
      link.classList.remove('active');
    });
    document.querySelector('header nav
a[href="#contact"]').classList.add('active');
  } else {
    let activeSection = null;

    sections.forEach(sec => {
      let top = window.scrollY;
      let offset = sec.offsetTop - 75;
      let height = sec.offsetHeight;
      let id = sec.getAttribute('id');

      if (top >= offset && top < offset + height) {
        activeSection = id;
      }
    });

    navLinks.forEach(link => {
      link.classList.remove('active');
      if (link.getAttribute('href') === '#' + activeSection) {
        link.classList.add('active');
      }
    });
  }
};
```


---

<Web Development 101 />

# Not so fast



# JavaScript / TypeScript



JS

```
type EntityItem = {  
  value:  
    | Character  
    | City  
    | Faction  
    | Quest  
    | Building;  
  label: string;  
};
```



TS



# Why Care?



Types are IMPORTANT

- JavaScript is dynamically typed. Imagine trying to divide an array by a string by accident.
- Better Development Experience
  - Linting
  - Auto Complete

```
// Define the structure of a menu item
type MenuItem = {
  id: string;
  name: string;
  description: string;
  price: number;
  category: string;
};
```

```
const [filteredItems, setFilteredItems] = useState<MenuItem[]>(menuItems);
```

filteredItems.

```
You, 1 second ago • Uncommitted changes
// State variable
const [customerName, setCustomerName] = useState<string>('');
const [customerPhone, setCustomerPhone] = useState<string>('');
const [customerEmail, setCustomerEmail] = useState<string>('');
const [special, setSpecial] = useState<boolean>(false);
const [pickupTime, setPickupTime] = useState<string>('');
const [timeError, setTimeError] = useState<string>('');
const [filteredItems, setFilteredItems] = useState<MenuItem[]>(menuItems);
const [lastIndexOf, setLastIndexOf] = useState<number>(0);
```

(method) Array<MenuItem>.map<U>(callbackfn: (value: MenuItem, index: number, array: MenuItem[]) => U, thisArg?: any): U[]

Calls a defined callback function on each element of an array, and returns an array that contains the results.

@param callbackfn - A function that accepts up to three arguments. The map method calls the callbackfn function

```
const [filteredItems, setFilteredItems] = useState<MenuItem[]>(menuItems);
```

filteredItems[0].

```
You, 1 second ago • Uncommitted changes
// State variable
const [customerName, setCustomerName] = useState<string>('');
const [customerPhone, setCustomerPhone] = useState<string>('');
const [customerEmail, setCustomerEmail] = useState<string>('');
const [special, setSpecial] = useState<boolean>(false);
const [pickupTime, setPickupTime] = useState<string>('');
const [timeError, setTimeError] = useState<string>('');
const [filteredItems, setFilteredItems] = useState<MenuItem[]>(menuItems);
const [lastIndexOf, setLastIndexOf] = useState<number>(0);
```

(property) category: string

(property) description

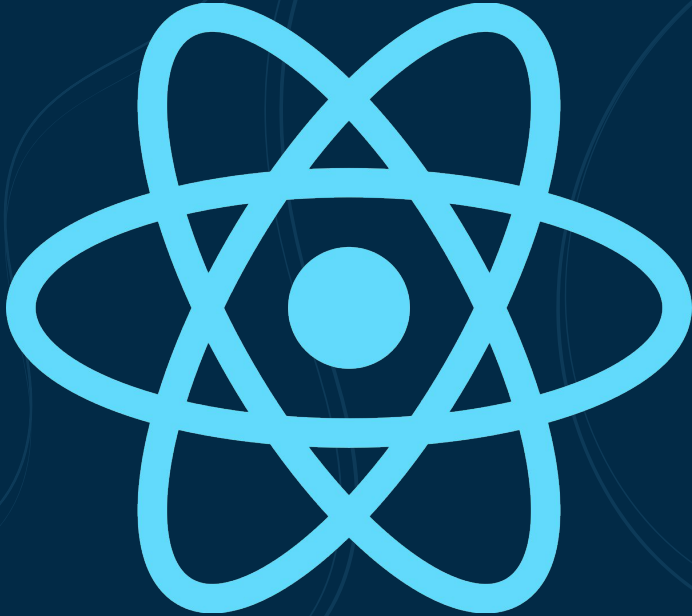
(property) id

(property) name

(property) price



# React

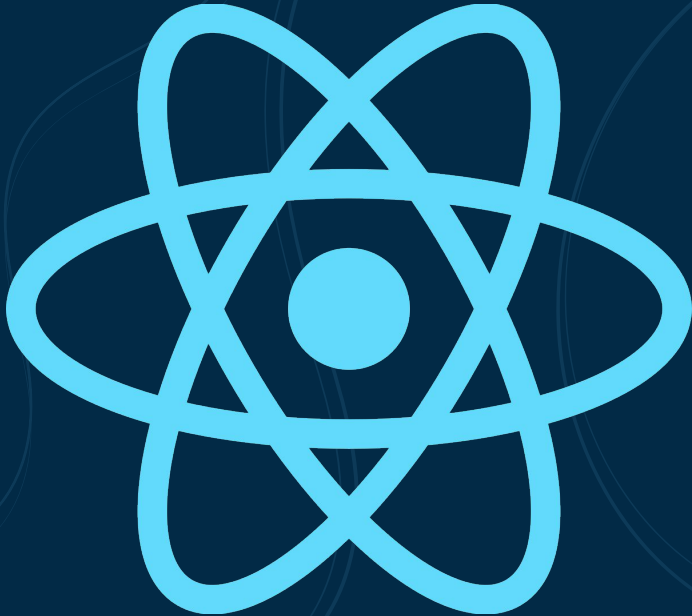


## JavaScript UI Framework

- COMPONENT BASED ARCHITECTURE.
  - Client Side Only.
  - State Management System.
  - Optimized Interactions on the DOM.
  - Hook Based Bindings.
-



# Component Based Architecture



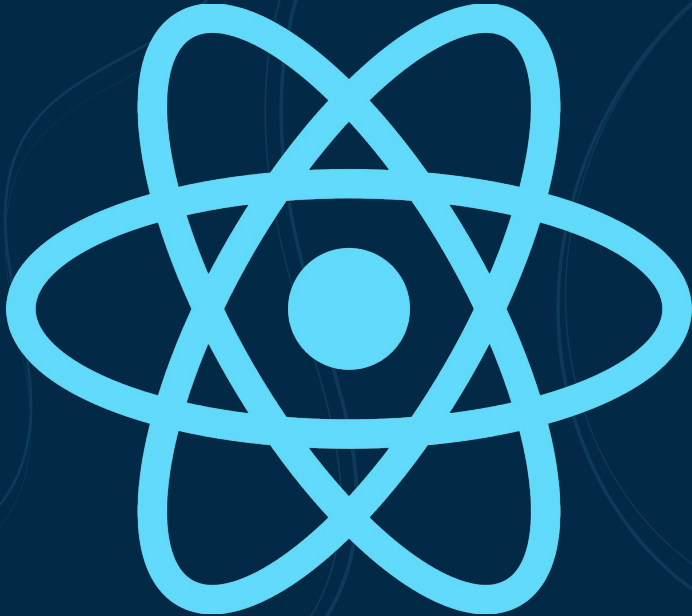
## Reusable Components

- Contains the Styles, Structure, and State logic.
  - Imagine a function, but it is for a UI component and not just logic.
-





# What is State?



The Current Condition of the Application

- Data that is runtime dependent and not stored
- If a dropdown is opened or closed
- If a modal is visible or not
- The current text inside of an input field
- A visual representation of actions



```

About Me Component

import React from "react";

function About() {
  return (
    <div className="mb-20 px-6" id="about">
      <div className="text-3xl font-bold pt-24">About Me</div>
      <div className="max-w-xl">
        <div className="text-xl font-bold pt-8 pb-2">
          Hi, I&apos;m Dylan Vidal
        </div>
        <div className="text-xl">
          I am a Computer Science Student at the University of Central Florida, and
an aspiring Software Engineer.
          I began my journey in Computer Science in 2021 during my Junior year of
high school after taking dual-enrollment courses in Python and C++.
          Since then I have learned my love for programming by attending hackathons,
participating in coding competitions, and working on personal projects.
          I am currently looking for an internship for Summer 2024, and I am always
ready and excited to learn!
        </div>
        <div className="text-xl font-bold pt-8 pb-2">
          My Interests
        </div>
        <div className="text-xl">
          In the industry, I am interested in Web Development, Artificial
Intelligence, and Application Engineering.
          Beyond my professional life, I am an avid hobbyist, and enjoy lifting and
tabletop roleplaying games!
        </div>
        <div className="text-xl font-bold pt-8 pb-2">
          What I&apos;m up to
        </div>
        <div className="text-xl">
          Currently, I have been exploring the web development industry, and
learning new tools and frameworks in the ecosystem like React, Next.js, and Tailwind CSS,
which I used to build this website!
        </div>
      </div>
    </div>
  )
}

export default About;
```

```
Component + Props

import Image from "next/image";

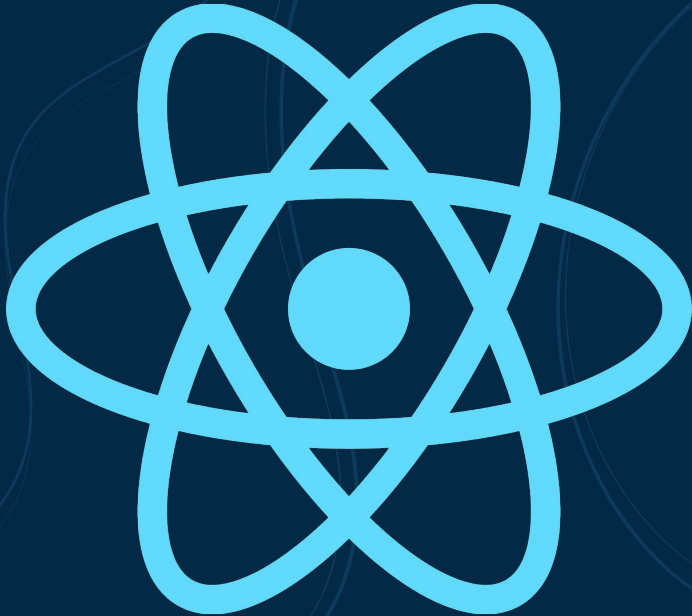
interface FreelanceProps {
  name: string;
  shortDescription: string;
  image: string;
  link: string;
}

function Freelance({ name, shortDescription, image, link }: FreelanceProps) {
  const handleClick = () => {
    window.open(link, "_blank");
  };
  return (
    <>
      <li
        className="cursor-pointer p-4 bg-gradient-to-tl from-purple-950 via-purple-800 to-violet-950 transition-transform hover:-translate-y-2 bg-transparent backdrop-blur-xl mshadow-md drop-shadow-2xl border-2 border-violet-900 rounded-2xl my-3"
        onClick={handleClick}
      >
        <div>
          <div className="flex">
            <p className="text-2xl font-semibold">{name}</p>
            <div className="relative top-0 right-0 flex">
              <a target="_blank" className="hover:text-violet-400" href={link}>
                <LinkImage />
              </a>
            </div>
          </div>
          <div>
            <p>{shortDescription}</p>
            <Image
              src={image}
              width={1920}
              height={1080}
              alt={` ${name} Image`}
              className="rounded-md drop-shadow-xl my-4 h-44 lg:h-96 xl:h-96 w-auto mx-auto aspect-auto"
            />
          </div>
        </div>
      </li>
    </>
  );
}

export default Freelance;
```

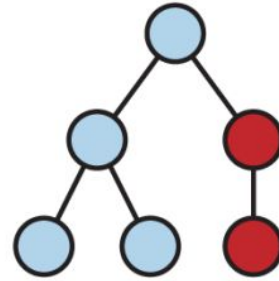
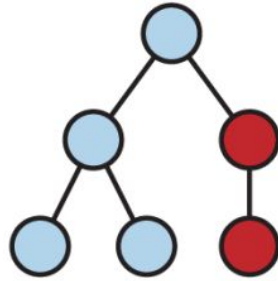
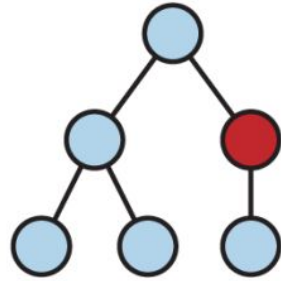


# Hook 1 - useState



Creates STATEFUL Data

- Stateful Data is different from normal data.
- Changes in Stateful Data will cause a re-rendering of the appropriate JSX.
- Returns the stateful data, and the data setter.



**Virtual  
DOM**

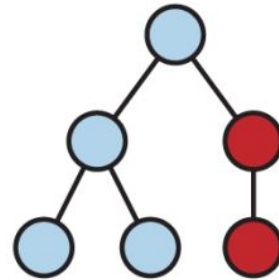
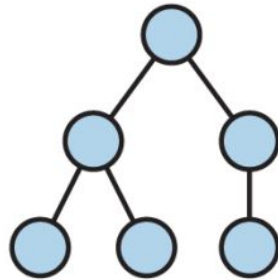
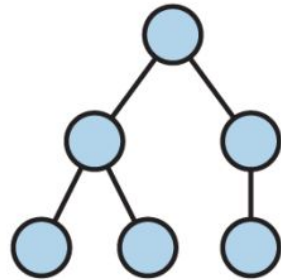
State Change



Compute Diff



Re-render



**Browser  
DOM**



Stateful Data!

```
const [name, setName] = useState<string>("");

<Input
  id="name"
  autoComplete="off"
  onChange={(e) => setName(e.target.value)}
/>

<div> {name} </div>
```



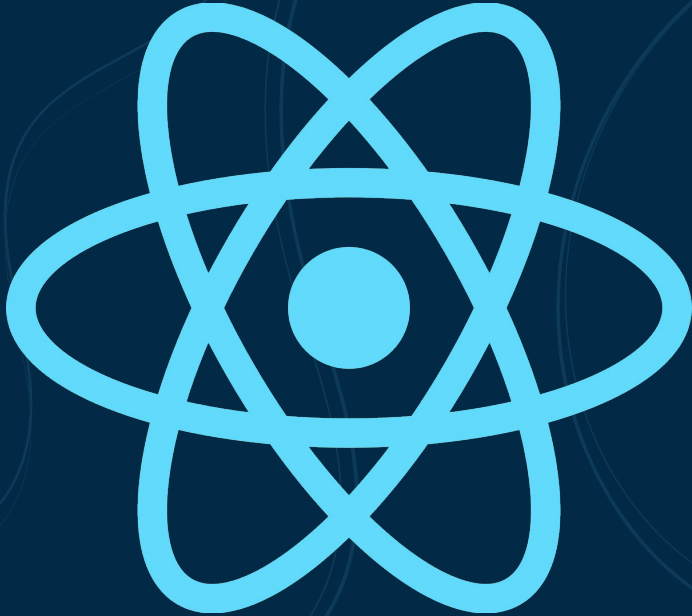
```
import React, { useState } from 'react';

function Example() {
  // Declare a new state variable, which we'll call "count"
  const [count, setCount] = useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
```



## Hook 2 - useEffect



Reacts to changes in data

- Provides UI interactions to non-stateful data.
- Allows react to react.
- References data in a dependency array.





```
useEffect()

//TRPC... Coming Soon
const { data: city } = trpc.getCity.useQuery({ id: entityid });

useEffect(() => {
  if (city) {
    setName(city.name);
    setPopulation(city.population);
    setSprawl(city.sprawl);
    setArchitecture(city.architecture);
    setIndustries(city.industries);
    setClimate(city.climate);
    setSafety(city.safety);
    setEducation(city.education);
    setModernity(city.modernity);
    setWealth(city.wealth);
    setDescription(city.description);
    setLore(city.lore);
    setGovernance(city.governance);
    setQuests(city.quests);
    setImage(city.imageUrl);
    setCityData(city);
  }
}, [city] //Dependency Array...);
```



```
// Save cart to localStorage whenever it changes
useEffect(() => {
  if (typeof window !== "undefined" && cart !== null) {
    localStorage.setItem("cart", JSON.stringify(cart));
  }
}, [cart]);
```



```
// Get the language of the web client
useEffect(() => {
  startLang = navigator.language;
  if (startLang.includes("en")) {
    setLang("en");
  } else {
    setLang("es");
  }
}, []);
```



On Mount

```
//Empty Dependency Array  
useEffect(() => {  
    setEntity("");  
    setResponseData("");  
}, []);
```

---

<Web Development 101 />



# 02

## <// Concepts //>

What the is the difference between client and server!?

---



# Server and Client

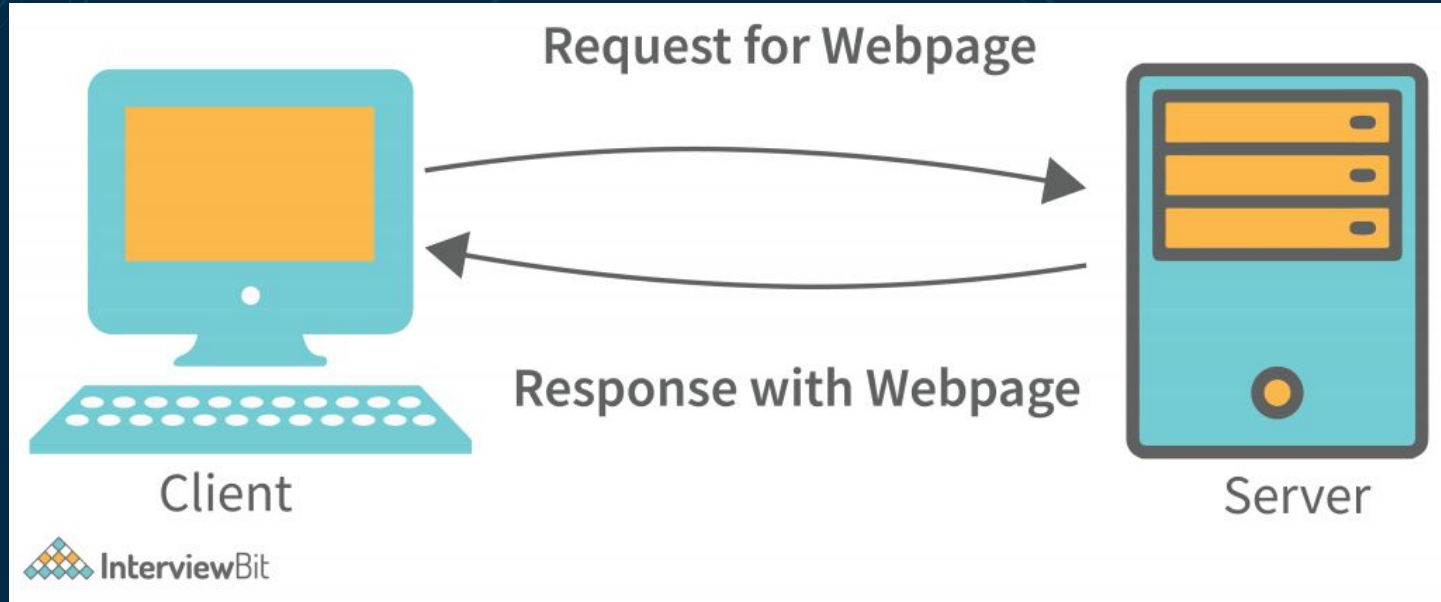


## <Server />

- The machine the site is hosted on.
- Where API servers are hosted.
- Communicates with databases.
- No User Interface, and therefore no intrinsic state.

## <Client />

- The machine the site is being viewed on.
  - Most commonly a web browser like Google or Safari.
  - Has access to Client data (language, local time, state)
-





# How does a React App actually Work?



## <Server Sends />

- Receives a request from the Client when URL is typed.
- Packages and sends over the React App to the browser.

## <Client Receives />

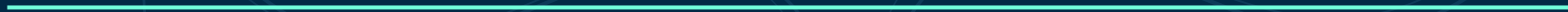
- Receives all of the code in its packaged form.
  - Unpacks, and renders in the Data.
-



---

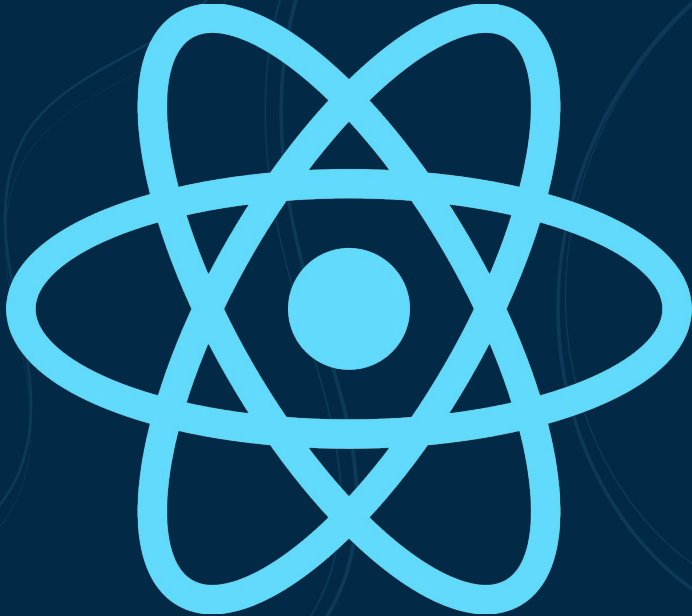
<Web Development 101 />

# Client Side Rendering



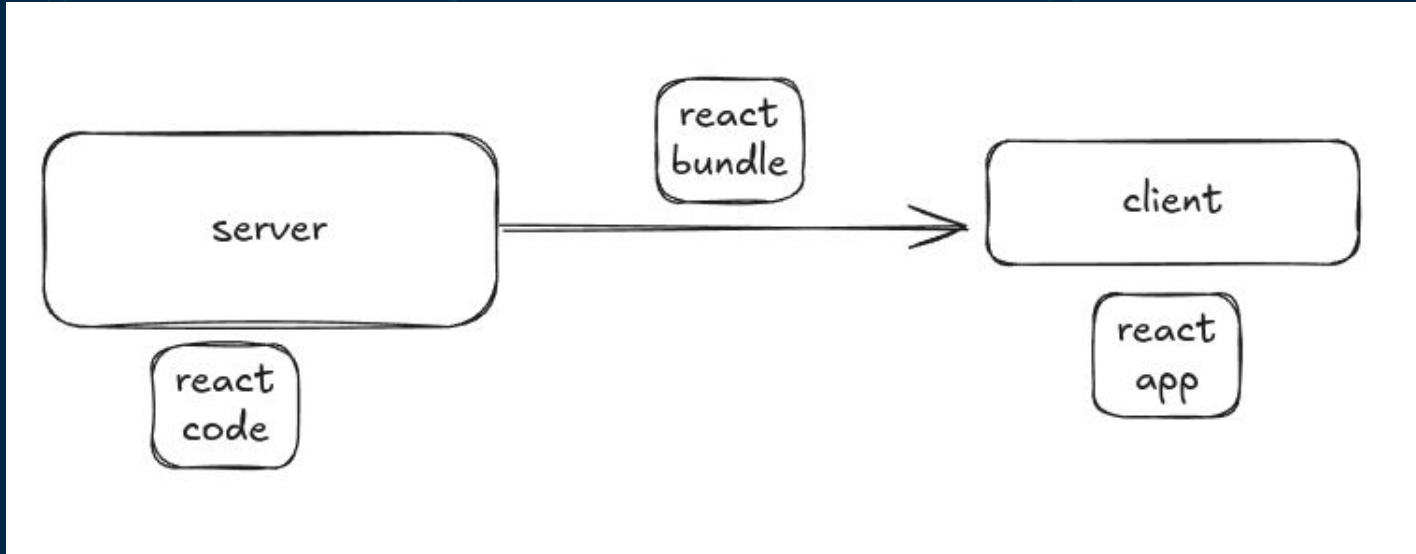


# Client Side Rendering



Recall React is ran in the Web Browser

- It is not until the Browser receives the website bundle that it is rendered.
- On the server, the only HTML visible is things that are NOT React.
- Most React apps are all React by standards.



---

<Web Development 101 />

# How do Websites appear on Google?

---



# SEO



## Search Engine Optimization

- Google Indexes websites by performing routine web scraping.
- Web scraping agents are not a web browser, and thus, will NOT render CSR content.
- Having less information on the server means less data Google can index you by.
- No Server Data = No Search Results.

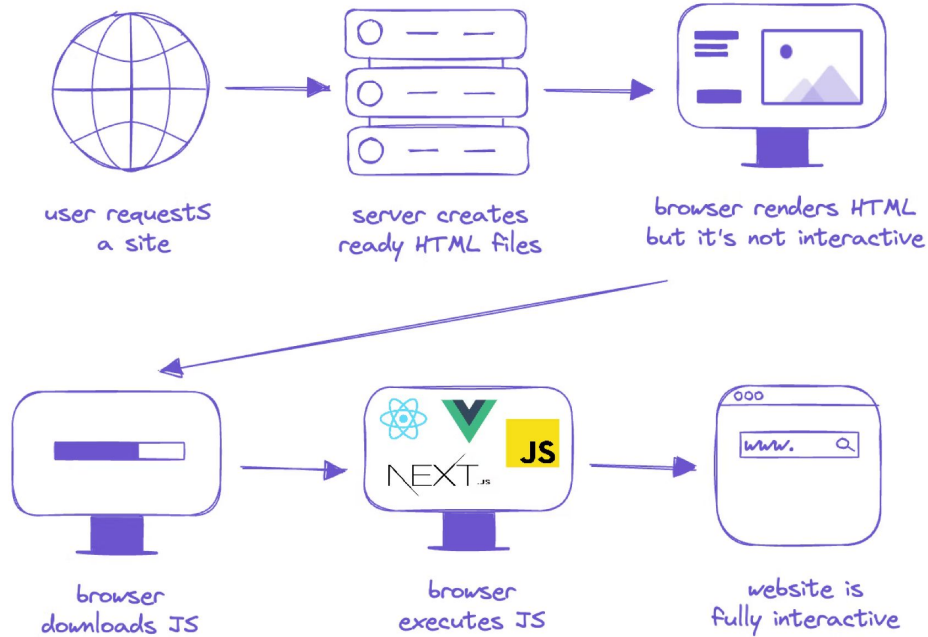
---

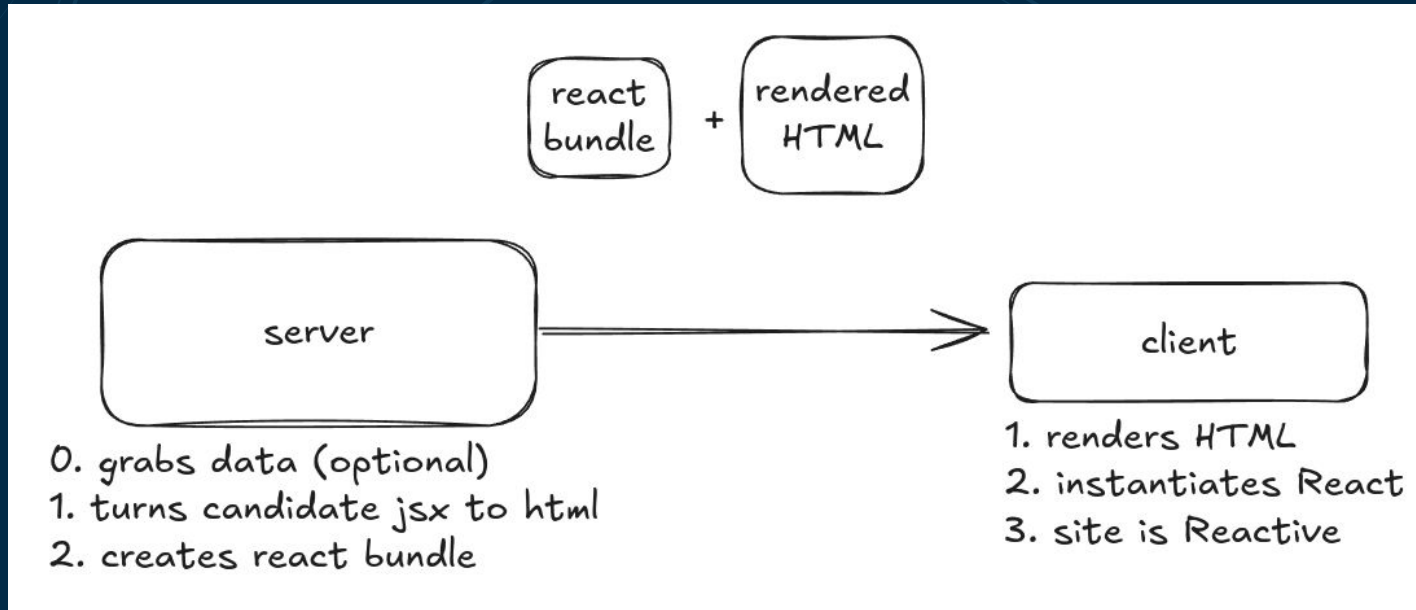
<Web Development 101 />

# Server Side Rendering



# Server-side Rendering (SSR)







**That's Complex.  
How can I do that?**

---



# 03

## <// Servers //>

How do we serve a website effectively? What about SSR?



# NextJS



## Server Side Meta Framework

- Allows you to write React Server Components.
- Handles Static and Dynamic Routing for you.
- Handles Metadata Optimizations for SEO.
- Optimizes Images and Links
- Middleware



Dylan Vidal

<https://www.dvidal.dev>



## Dylan Vidal

**Dylan Vidal** is a undergraduate student at the University of Central Florida and an aspiring software engineer.



LinkedIn · Dylan Vidal

820+ followers

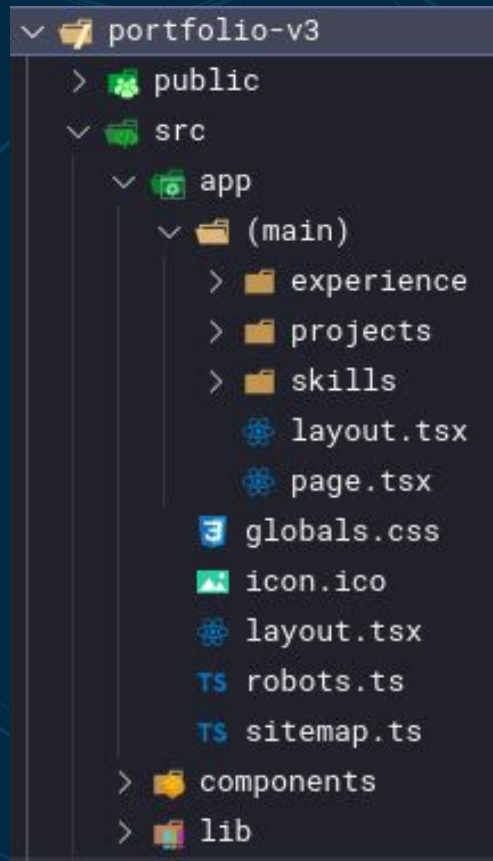


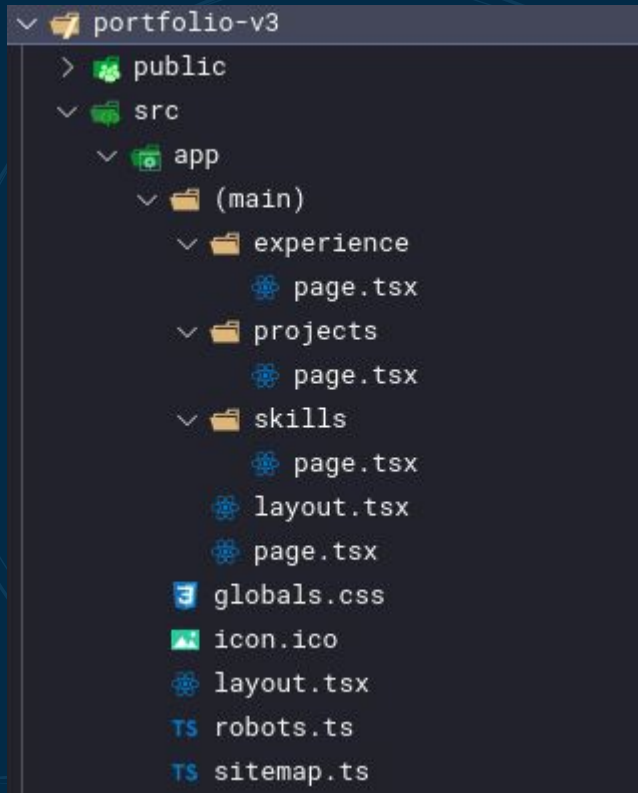
## Dylan Vidal - President - Knight Hacks

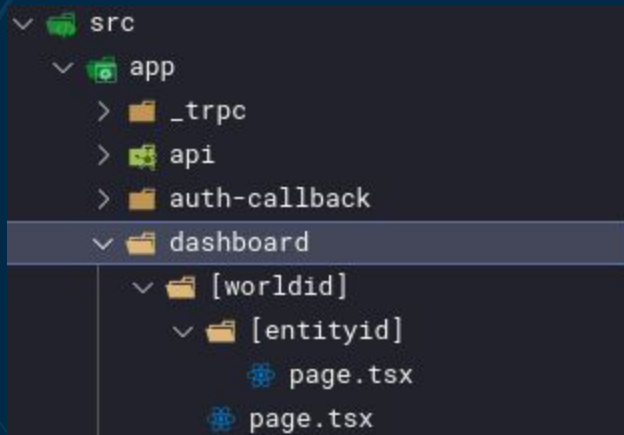
I am a software engineer and undergraduate student at the University of Central Florida. I am currently the Secretary of Knight Hacks, UCF's Software ...



```
1  "use client";  
2  import Image from "next/image";  
3  import React, { useState, useEffect } from "react";  
4
```











```
interface PageProps {  
  params: {  
    worldid: string;  
  };  
}  
  
const World = async ({ params }: PageProps) => {  
  const { worldid } = params;
```

```
interface PageProps {  
  params: {  
    worldid: string;  
    entityid: string;  
  };  
}  
  
const Gallery = async ({ params }: PageProps) => {  
  const { worldid, entityid } = params;
```



```
import Navbar from "@components/navbar";
import { Metadata } from "next";

export const metadata: Metadata = {
  title: "Dylan Vidal",
  description:
    "Dylan Vidal is a undergraduate student at the University of Central Florida and an aspiring software engineer.",
  keywords: [
    "Dylan Vidal",
    "Software Engineer",
    "UCF",
    "University of Central Florida",
    "Knight Hacks",
    "Web Development",
    "Full Stack",
  ],
  openGraph: {
    type: "website",
    title: "Dylan Vidal",
    description:
      "Dylan Vidal is a undergraduate student at the University of Central Florida and an aspiring software engineer.",
    url: "https://dvidal.dev",
  },
};

export default function Layout({ children }: { children: React.ReactNode }) {
  return (
    <>
      <Navbar />
      <main className="min-h-screen p-4 xl:p-24 bg-purple-950 w-full">
        {children}
      </main>
    </>
  );
}
```





```
import { MetadataRoute } from "next";

export default function robots(): MetadataRoute.Robots {
  return {
    rules: {
      userAgent: "*",
      allow: "/",
      disallow: "/private/",
    },
    sitemap: "https://www.dvidal.dev/sitemap.xml",
  };
}
```



```
import { MetadataRoute } from "next";

export default function sitemap(): MetadataRoute.Sitemap {
  return [
    {
      url: "https://www.dvidal.dev",
      lastModified: new Date(),
      priority: 1,
    },
    {
      url: "https://www.dvidal.dev/experience",
      lastModified: new Date(),
      priority: 0.9,
    },
    {
      url: "https://www.dvidal.dev/projects",
      lastModified: new Date(),
      priority: 0.9,
    },
    {
      url: "https://www.dvidal.dev/skills",
      lastModified: new Date(),
      priority: 0.7,
    },
  ];
}
```





```
const isAuth = middleware(async (opts) => {
  const { getUser } = getKindeServerSession();
  const user = await getUser();

  if (!user?.id || !user?.email) {
    throw new TRPCError({ code: "UNAUTHORIZED", message: "Unauthorized" });
  }

  return opts.next({
    ctx: {
      userId: user.id,
      user,
    },
  });
});
```

---

<Web Development 101 />



# 04

## <// Deployment //>

This one will be pretty fast...

---

---

<Web Development 101 />

# **npx create-next-app**

---



# Vercel



Owner of NextJS

- One-Click deployment for Next apps.
- Free for most use cases.
- Continuous Updates and Deployments.





---

<Web Development 101 />



# 05

**<// What now? //>**

Some final key takeaways

---

---

<Web Development 101 />

# Portfolio





# Personal Portfolio



Hey there, I'm Dylan 🙋

Projects

Resume



Some good challenges

- Make it extensible by using JS Mapping (slide 12).
- Make a good sitemap, robots, and Metadata for pages to maximize SEO.
- Add some sort of interactivity with React animations, and obviously make it look pretty.

---

<Web Development 101 />



joshtriedcoding



fireship



theo browne

---

---

<Web Development 101 />

# What's next?

tRPC, ORMs, Databases, Authentication, Component Libraries

---

---

<Web Development 101 />

**Thank You :D**

---