CMPSC 104 – Document Engineering

Prof. Hang Zhao

ALLEGHENY COLLEGE

Overview

What is JAMstack?

- JAMstack stands for JavaScript, APIs, and Markup.
- It's a modern web architecture focused on decoupling the frontend from backend services.
- The concept promotes building websites and apps using pre-rendered assets served from a Content Delivery Network (CDN), interacting with APIs for dynamic features.

Key Components of JAMstack

- JavaScript: Handles the dynamic aspects of the site (e.g., validation, interactions).
- APIs: Interact with server-side data and services without needing a full backend.
- Markup: Static HTML that can be pre-built, improving load speed and reducing server load.

Why JAMstack

- **Pre-rendering**:
  - JAMstack sites are pre-rendered, meaning they are generated in advance rather than on-demand, which leads to faster load times and reduced server complexity.
  - This approach improves Time to First Byte (TTFB), and allows deployment on a Content Delivery Network (CDN), optimizing global performance by delivering content from the nearest server.
- **Decoupling**:
  - JAMstack separates frontend and backend, making it easy to maintain and update each independently.
  - Developers can integrate trusted tools and services (e.g., headless CMS, search providers, media storage) without being tied to a single backend, allowing for flexible and scalable applications.

JAMstack Tools & Services

Jamstack.org lists over 300 static site generators!

- **Building a JAMstack Site**:
  - Choose a framework (e.g., static site generator),
  - Select a content management system,
  - Decide on a deployment service.

CMS

- CMS stands for Content Management System. It provides a user-friendly interface to manage website content without directly editing code.
- Common tasks include creating and updating blog posts, managing media, and organizing pages or sections.

Types of CMS in JAMstack

- Traditional CMS: Combines backend and frontend (e.g., WordPress).
- Headless CMS: Separates the backend (content management) from the frontend, ideal for JAMstack sites. Examples include Contentful, Sanity, and Netlify CMS.

Static Site Generators

- **Definition**: Unlike traditional CMSes, static site generators create pre-rendered HTML pages, reducing the need for client-side JavaScript.
- **Examples**: Gatsby, Jekyll, Next.js, and Hugo.
- **Benefit**: Pages are bundled with data at build time and served from a CDN for fast, scalable performance.

APIs

APIs standards for Application Programming Interfaces, are mechanisms that enable two software components to communicate with each other using a set of definitions and protocols.

For example, the weather bureau's software system contains daily weather data. The weather app on your phone "talks" to this system via APIs and shows you daily weather updates on your phone.

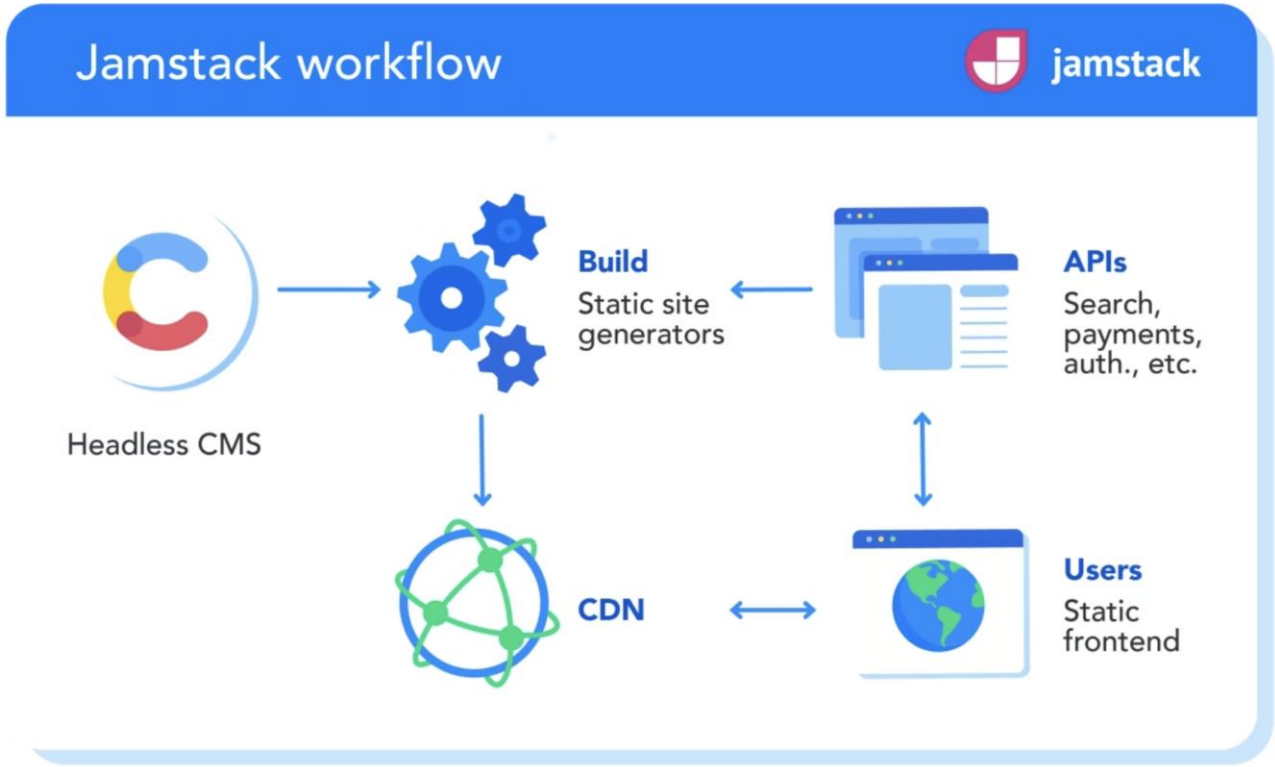
APIs Are Important in JAMstack

- They allow static site generators (SSGs) to integrate with external services for features like search, payment processing, and data retrieval from external sources
- For example: search, authentication, media management, and more.
- Popular API Integrations: Shopify (ecommerce), Auth0 (authentication).

Deployment

- **Deploy in Minutes**:
  - Services like **Netlify**, **Vercel**, and **GitHub** connect to Git and streamline the deployment process.

JAMstack Workflow



Popular Examples of JAMstack Projects

- GatsbyJS: A React-based framework optimized for static sites, often used for blogs and e-commerce.
- Netlify: A hosting platform specifically designed for JAMstack, with built-in CI/CD and serverless functions.
- Netlify CMS: A headless CMS that provides a backend-as-a-service for content management in JAMstack sites.

How to Get Started with JAMstack

- Choose a Static Site Generator: Popular choices are Gatsby, Next.js, Hugo, or Jekyll.
- Pick a Headless CMS: Options like Contentful, Sanity, and Netlify CMS work well with JAMstack.
- Deploy and Host: Platforms like Netlify and Vercel provide easy hosting and deployment for JAMstack projects.

Gatsby - Installation

https://nodejs.org/en/download/package-manager

```
node –version
npm – version

npm install --global gatsby-cli
gatsby --version
```

Gatsby – Creating a New Site

```
cd path\to\gatsby

gatsby new ga_site https://github.com/gatsbyjs/gatsby-starter-hello-world

cd ga_site

gatsby develop
```

Gatsby – Adding Content

```
import * as React from "react"

export default function Home() {
  return <div>Hello world!</div>
}
```

Add additional paragraphs and modify text colors:

```
import * as React from "react"

export default function Home() {
  return <div style={{color:'tomato', backgroundColor: 'lightgray'}}>
            <h1>Hello world!</h1>
            <p>This is my first Gatsby Site</p>
          </div>
}
```

Gatsby – Adding Images

```
import * as React from "react"

export default function Home() {
  return <div style={{color:'tomato', backgroundColor: 'lightgray'}}>
            <h1>Hello world!</h1>
            <p>This is my first Gatsby Site</p>
             <img src="https://raw.githubusercontent.com/allegheny-
college-cmpsc-104-Fall-2024/lab01/main/graphics/DocEng_logo.png"
alt="Logo"  width="150" />
         </div>
}
```

Gatsby – Linking Pages

```
import * as React from "react“
import { Link } from "gatsby"


export default function Home() {
  return <div style={{color:'tomato', backgroundColor: 'lightgray'}}>
            <h1>Hello world!</h1>
            <p>This is my first Gatsby Site</p>
            <img src="https://raw.githubusercontent.com/allegheny-
college-cmpsc-104-Fall-2024/lab01/main/graphics/DocEng_logo.png"
alt="Logo"  width="150" />
          <br />
            <Link to="/page-2">Page 2</Link>
          </div>
}
```

Gatsby – Linking Pages

```
import * as React from "react“
import { Link } from "gatsby"


export default function Home() {
  return <div>
            <h1>Page 2</h1>
            <Link to="/">Home</Link>
         </div>
}
```

Gatsby – Interactive Pages

Add a new file “counter.js” under src/pages:

```
import * as React from "react"

class Counter extends React.Component {
    render() {
    return (
        <div>
            Hello Class Component
        </div>
        )
    }
}

export default Counter
```

Gatsby – Interactive Pages

```
import * as React from "react"

class Counter extends React.Component {
    render() {
    return (
        <div>
         <h1>Counter</h1>
         <p>current count: 0</p>
         <button>plus</button>
         <button>minus</button>
         </div>
        )
    }
}

export default Counter
```

```
import * as React from "react"

class Counter extends React.Component {
    constructor() {
        super()
        this.state = {
        count: 0
        }
    }
    render() {
     return (
        <div>
         <h1>Counter</h1>
         <p>current count: {this.state.count}</p>
         <button onClick = {() =>
            this.setState({
            count: this.state.count + 1
        })
        }>plus</button>
         <button onClick = {() =>
            this.setState({
            count: this.state.count - 1
        })
        }>minus</button>
         </div>
        )
    }
}

export default Counter
```

Gatsby – Components

```
import * as React from "react“
import { Link } from "gatsby“
import Counter from "./counter"


export default function Home() {
  return <div style={{color:'tomato', backgroundColor: 'lightgray'}}>
            <h1>Hello world!</h1>
            <p>This is my first Gatsby Site</p>
             <img src="https://raw.githubusercontent.com/allegheny-
college-cmpsc-104-Fall-2024/lab01/main/graphics/DocEng_logo.png"
alt="Logo"  width="150" />
          <br />
             <Link to="/page-2">Page 2</Link>
          <br />
          <Counter />
          </div>
}
```

Gatsby – Components

```
import * as React from "react“
import { Link } from "gatsby“
import Counter from "./counter"


export default function Home() {
  return <div style={{color:'tomato', backgroundColor: 'lightgray'}}>
            <h1>Hello world!</h1>
            <p>This is my first Gatsby Site</p>
             <img src="https://raw.githubusercontent.com/allegheny-
college-cmpsc-104-Fall-2024/lab01/main/graphics/DocEng_logo.png"
alt="Logo"  width="150" />
          <br />
             <Link to="/page-2">Page 2</Link>
          <br />
          <Counter />
          </div>
}
```

Gatsby – Plugins

https://www.gatsbyjs.com/docs/plugins/

```
npm install gatsby-plugin-typography react-typography typography
```

(https://www.gatsbyjs.com/plugins/gatsby-plugin-typography/?=typography)

```
module.exports = {
  plugins: [`gatsby-plugin-typography`]
}
```

Check by inspecting (right-click) the page

Working with Markdown

```
mkdir pages/myFirstPost.md

mkdir pages/mySecondtPost.md

npm install gatsby-source-filesystem gatsby-transformer-remark
```

Update gatsby-config.js

Working with Markdown

```
module.exports = {
  plugins: [
    'gatsby-plugin-typography',
    {
       resolve: 'gatsby-source-filesystem',
       options: {
         name: 'src',
         path: `${__dirname}/src/pages`,
       }
    },
    'gatsby-transformer-remark',
  ]
}
```

Working with Markdown

Set up a **gatsby-node.js** to generate pages.

Create a basic template for rendering Markdown content.

(`src/templates/markdownTemplate.js`)

restart the Gatsby development server: `gatsby develop`

Gatsby – Building Your Site

```
cd ga_site

npm run build

or

gatsby build
```

Deploying to Netlify

1. Create a GitHub repository under your account.
2. Log in to Netlify using your GitHub account.
3. Click the “Add New Site” button, then select “Import an existing project.”
4. Choose GitHub and select your site repository.
5. Click the deploy button at the bottom.

Reference and Resources

- Contentful's JAMstack CMS (https://www.contentful.com/help/jamstack-cms/)
- What is an API? - Application Programming Interface Explained - AWS (amazon.com)