

*MSc. Enterprise Software Systems*

# **Module: Enterprise Web Development**

Website: <https://tutors.dev/course/enterpwebdev-2024-setu-deise>

## Overview

**Diarmuid O' Connor ([doconnor@wit.ie](mailto:doconnor@wit.ie))**

**Dr. Frank Walsh ([fwwalsh@wit.ie](mailto:fwwalsh@wit.ie))**

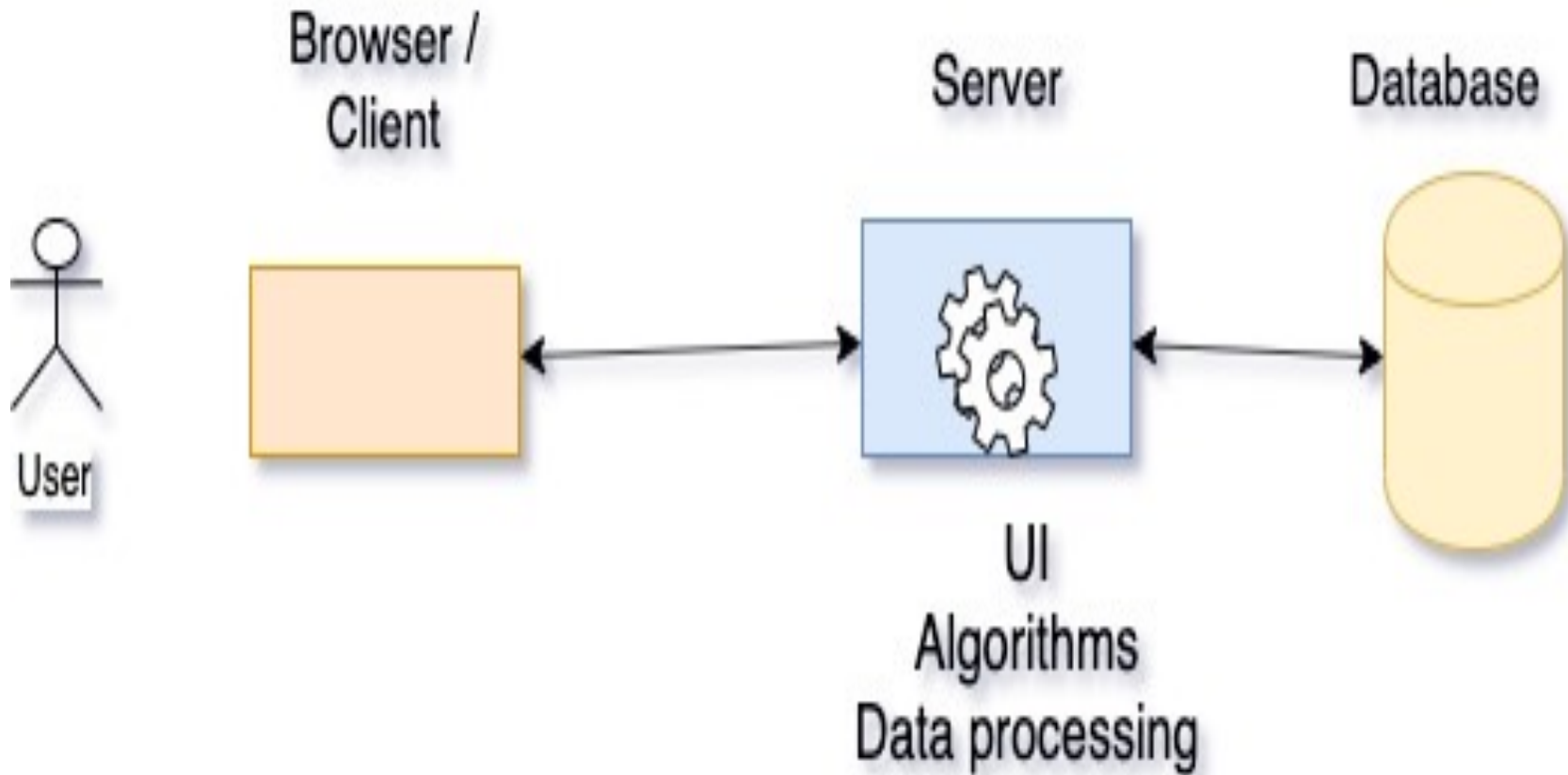
TL;DR

***Design and implement client-side rendering web apps and deploy them on a serverless platform using automated provisioning of the required resources.***

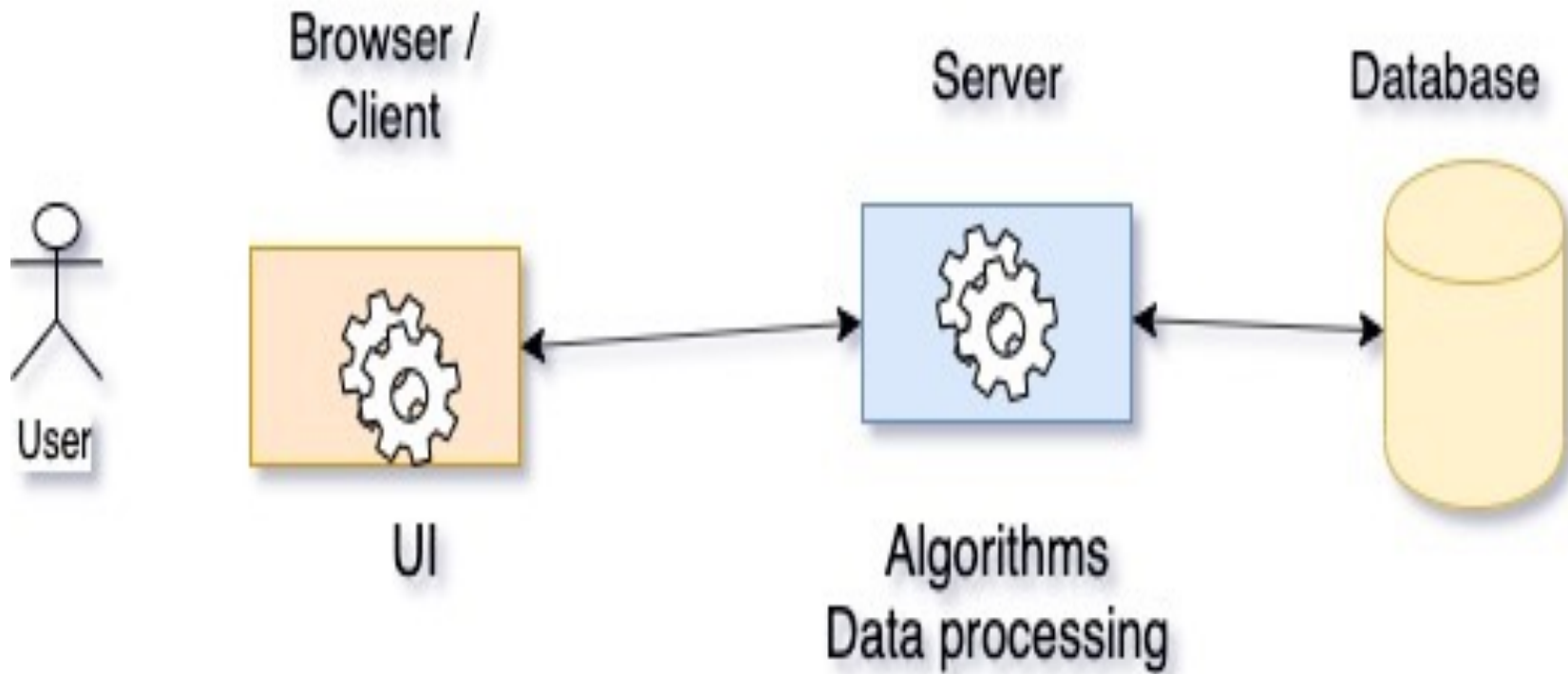
# Agenda

- Context.
- This module's focus
- Software installation requirements

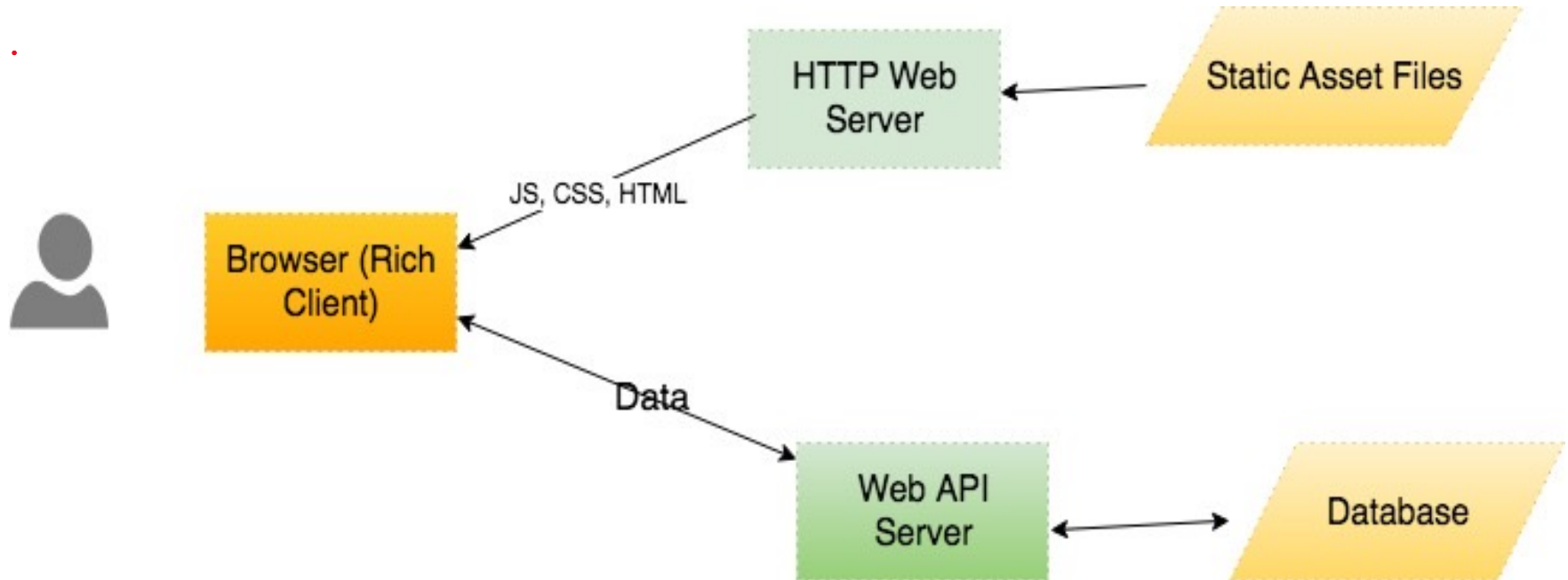
# Web Apps Architecture – Server-Side Rendering (SSR)



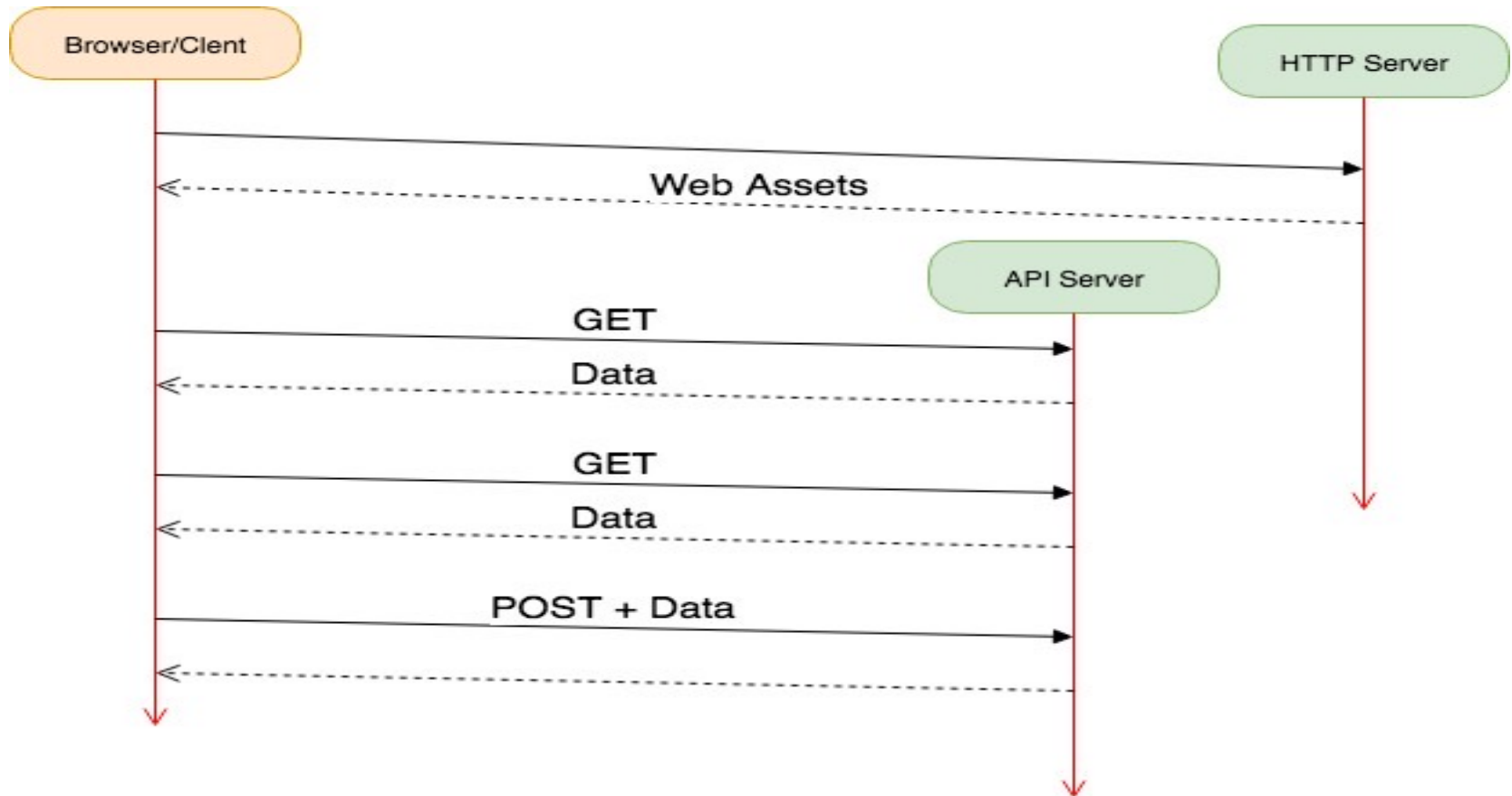
# Web Apps Architecture – Client-Side Rendering (CSR)



# CSR Web Apps - Architecture



# CSR Web Apps – HTTP Communications flow



# CSR Web Apps - Evolution.

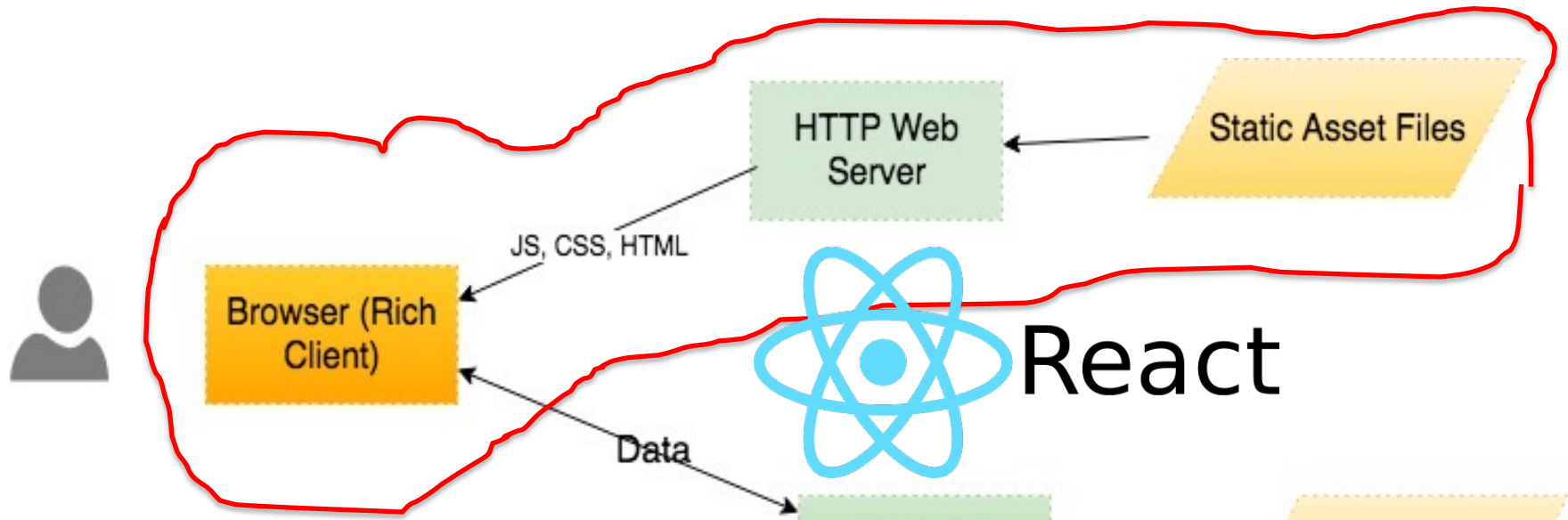
1. **Early Examples: Gmail (2004), YouTube (1<sup>st</sup> generation)**
  - **Disadvantage:**
    - **Poor developer experience (DX) – code bloat; poor maintainability; browser inconsistencies.**
    - **Lacked addressability (Key principle of the web).**
2. **Jquery – A cross-platform JavaScript library to simplify the client-side scripting of HTML pages. (2006)**
  - **Adv.: Better developer productivity; reduced code bloat**
  - **Disadvantages:**
    - **Lacked Addressability.**
    - **Poor code maintainability (spaghetti code).**
3. **Single Page App (SPA) *frameworks*. (2010 ish)**
  - **Support addressability; Improved DX; Improved maintainability.**



# Agenda

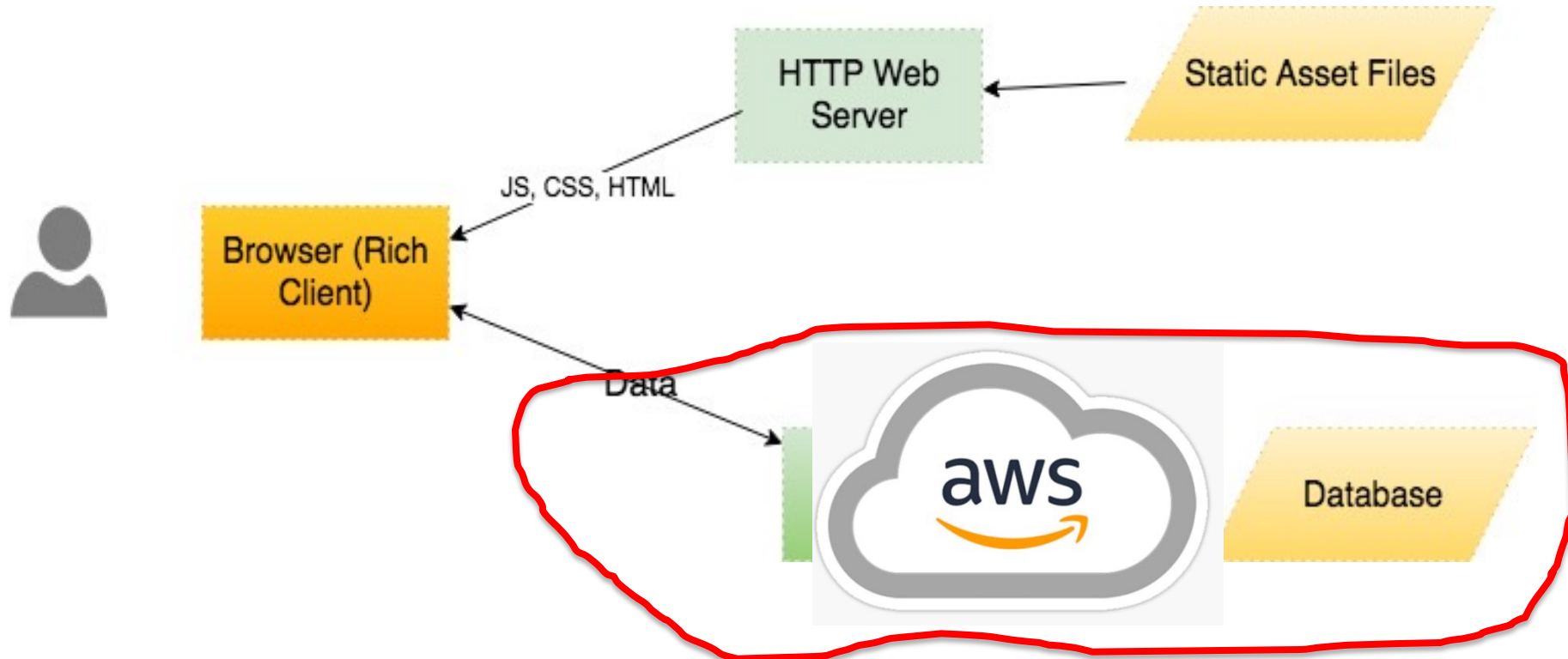
- Context. ☒
- This module's focus

# Our Technology stack - Client.



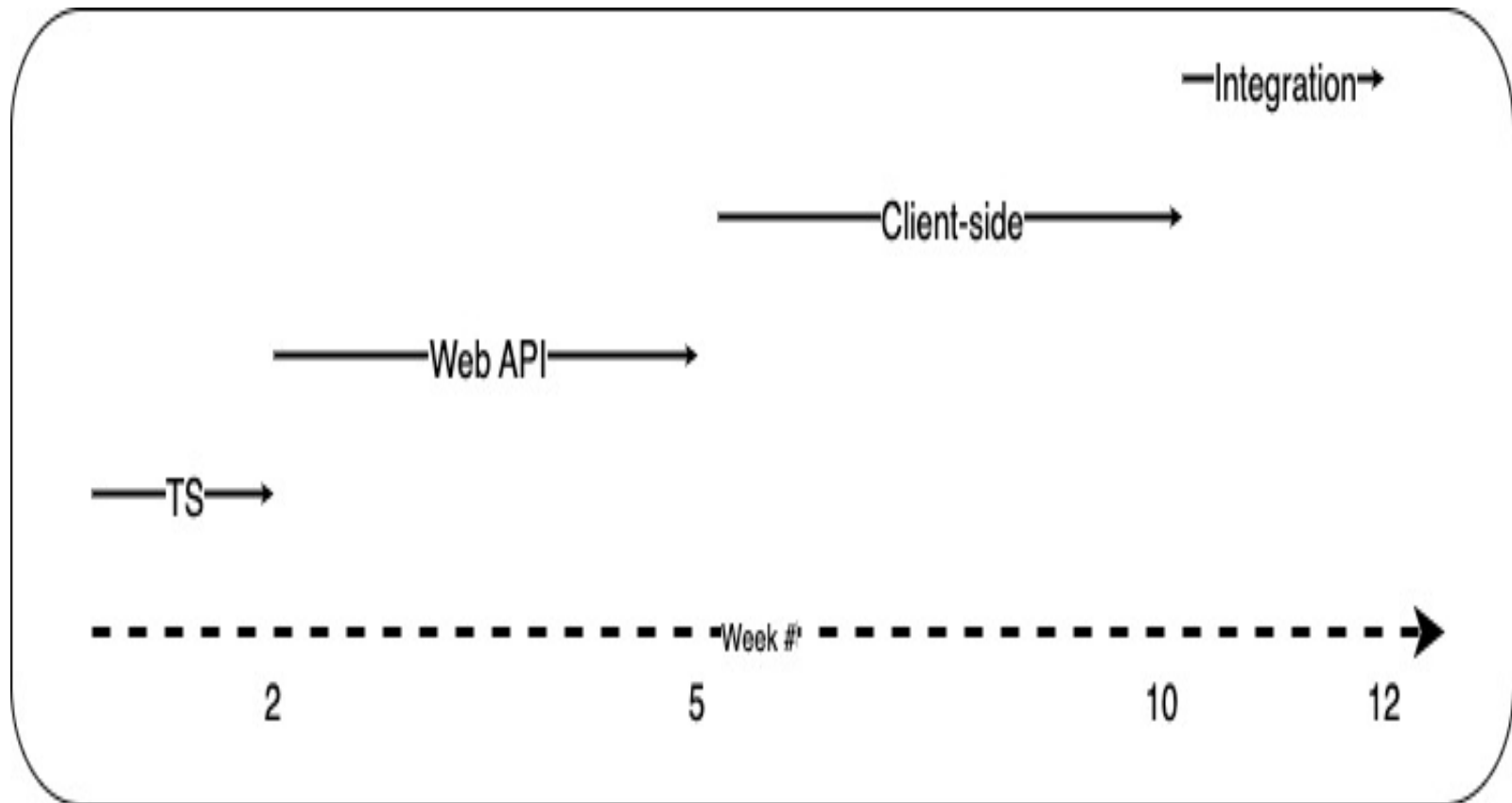
A JavaScript library for building user interfaces

# Our Technology stack - Server.



The AWS platform provides a range of serverless services and frameworks for automated resource provisioning

# Outline schedule

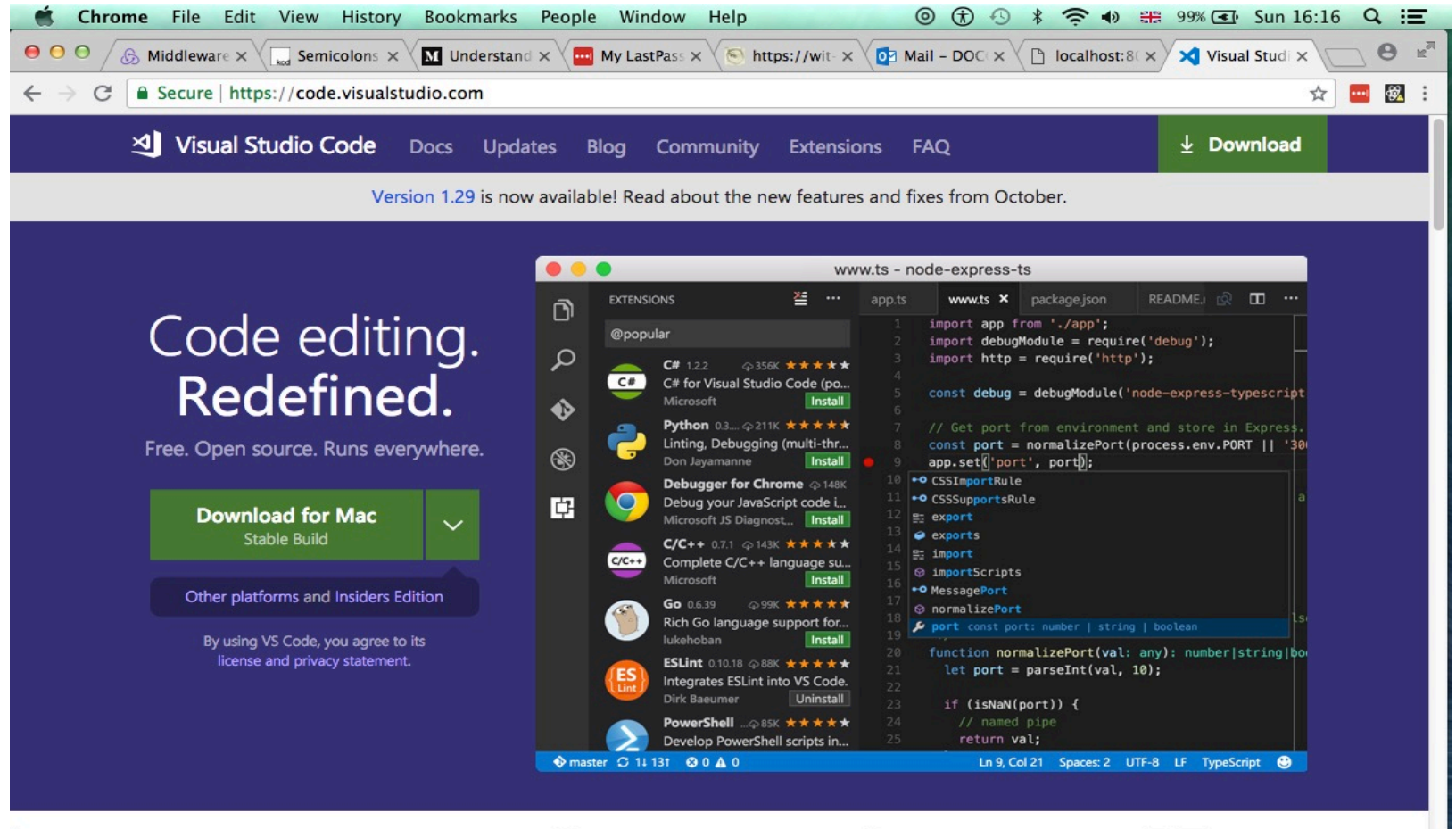


# Agenda

- Context. ☒
- This module's focus
  - Single Page Apps, using React ☒
  - (Serverless) Web APIs, using AWS services ☒
- Software Installation Requirements

# Text Editor

(Not a heavyweight IDE, thankfully)



# Node.js

(A JavaScript runtime platform)

Node.js® is an open-source, cross-platform JavaScript runtime environment.

## Download Node.js®

**20.10.0 LTS**  
Recommended For Most Users

**21.5.0 Current**  
Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#)   [Other Downloads](#) | [Changelog](#) | [API Docs](#)

For information about supported releases, see the [release schedule](#).

Copyright OpenJS Foundation and Node.js contributors. All rights reserved. The OpenJS Foundation has registered trademarks and uses trademarks. For a list of trademarks of the OpenJS Foundation, please see our [Trademark Policy](#) and [Trademark List](#). Trademarks and logos not indicated on the list of OpenJS Foundation trademarks are trademarks™ or registered® trademarks of their respective holders. Use of them does not imply any affiliation with or endorsement by them.

[The OpenJS Foundation](#) | [Trademark Policy](#) | [Privacy Policy](#) | [Code of Conduct](#) | [Security Reporting](#)




# Git

**git** --distributed-even-if-your-workflow-isnt

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.

 **Learn Git in your browser for free with Try Git.**

**About**  
The advantages of Git compared to other source control systems

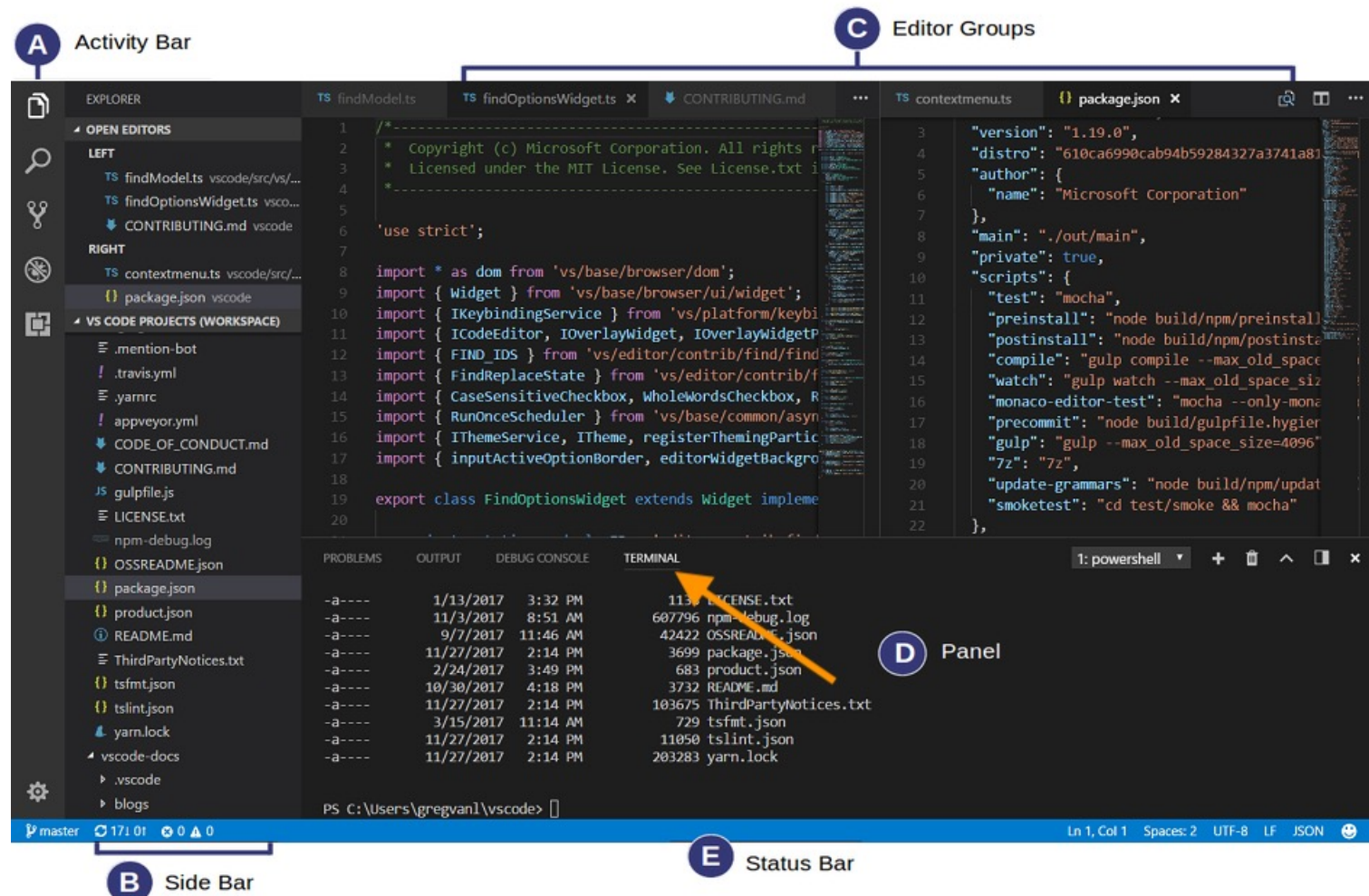
**Documentation**  
Command reference pages, Pro Git book content, videos and more

**Latest source Release 2.15.1**

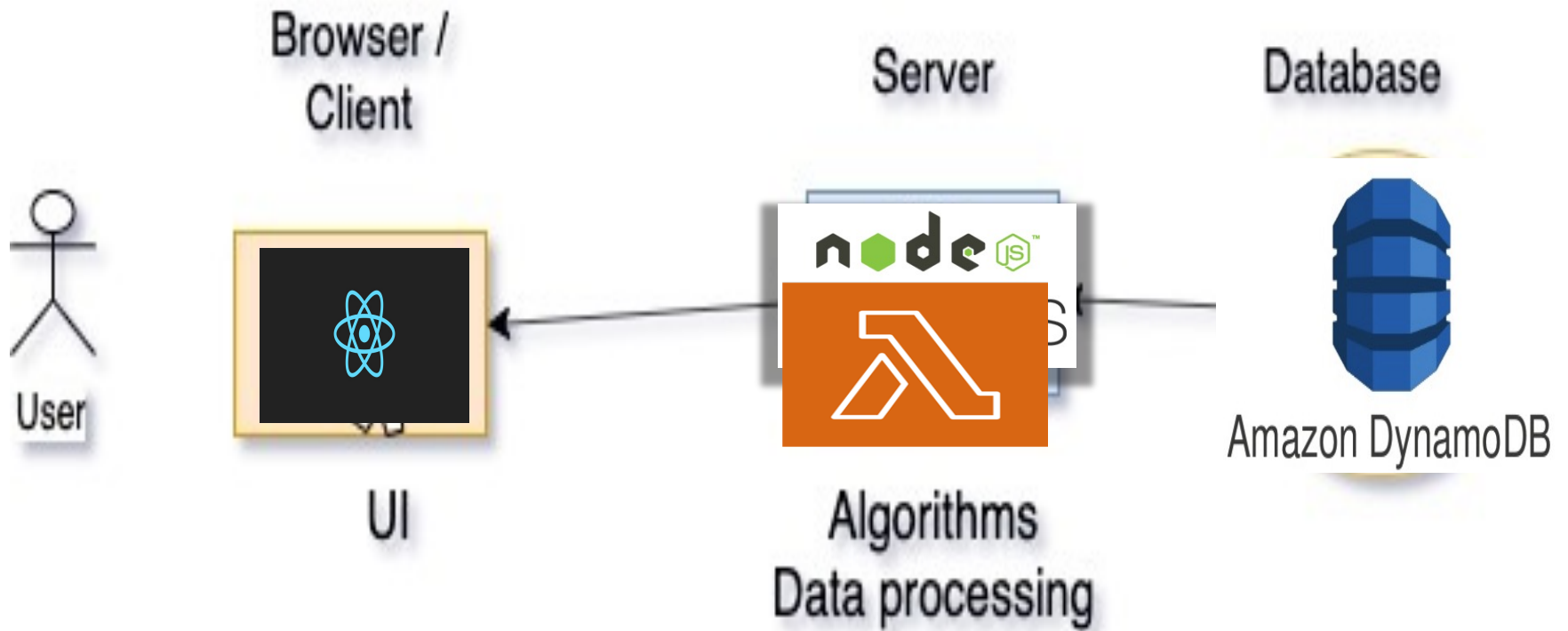
35 misspelled words   Spaces: 4   Markdown



# VS Code's Integrated Terminal.



# Full-Stack Developer



# Assessment.

- **100% continuous assessment**
- **Two assignments:**
  - **Web API – 40%**
  - **Front end & Integration – 60%**

# Summary.

- **Two models of web apps:**
  - **Server Side Rendering (SSR)**
  - **Client Side Rendering (CSR)**
- **CSR model has evolved:**
  - **Unmaintainable JS codebase to SPA framework based codebase**
  - **No addressability to addressable**
- **Server-side.**
  - **From SSR web applications to Web APIs.**
- **NoSQL databases.**

