

# Introduction to Web Development

# Table of content

- Recap web development
- Today of Web development
- Web Framework
  - Frontend
  - Backend
- Look outside Javascript

# Web Development

# Basic Web



The HyperText Markup Language or HTML is the standard markup language for documents designed to be **displayed** in a web browser



Cascading Style Sheets (CSS) is a stylesheet language used for **describing the presentation** (eg. color, size) of a document written in a markup language such as HTML



JavaScript is the dominant client-side scripting language of the Web. Scripts are embedded in or included from HTML documents and **interact** with the DOM (Document Object Model)

# Basic Web



Display



Interaction



Styling

# Web 1.0, 2.0, 3.0



## Web 1.0

Read only, static website.  
Eg, company website

## Web 2.0

Participation,  
Community focus.  
Eg, social media

## Web 3.0

Decentralized,  
Intelligent web-base  
Eg, uniswap,  
Killswitch

# Web 1.0, 2.0, 3.0



## Web 1.0

HTML,CSS,  
(a little bit of)  
Javascrip

## Web 2.0

HTML, CSS,  
(a lot of)  
Javascrip  
(included javascript  
framework)

## Web 3.0

HTML,CSS,Javascript  
+ Blockchain, AI and  
other...

# Web 1.0, 2.0, 3.0



## Web 1.0

What you've learn.

## Web 2.0

What you are  
going to learn

## Web 3.0

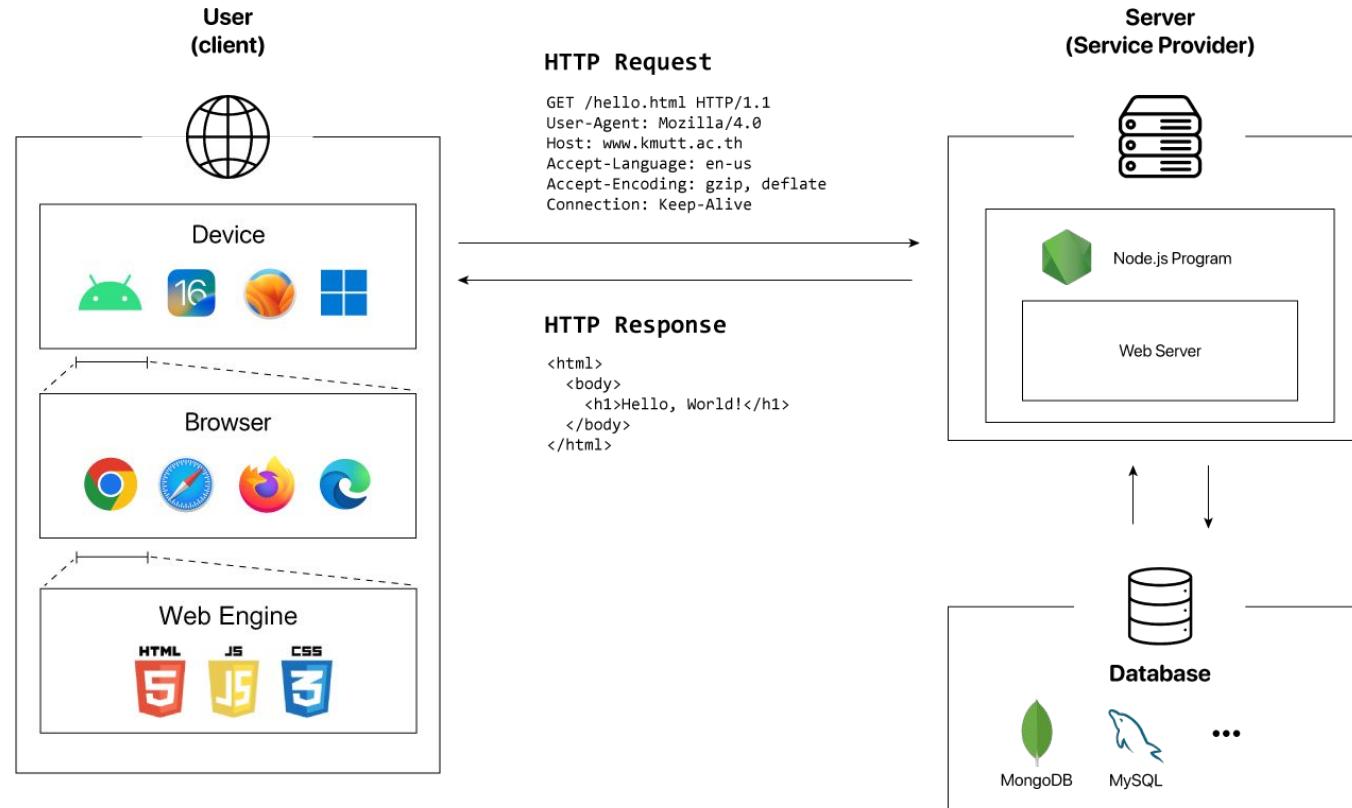
They doesn't  
even figure it all  
out yet...

# Web 2.0

Today of web development

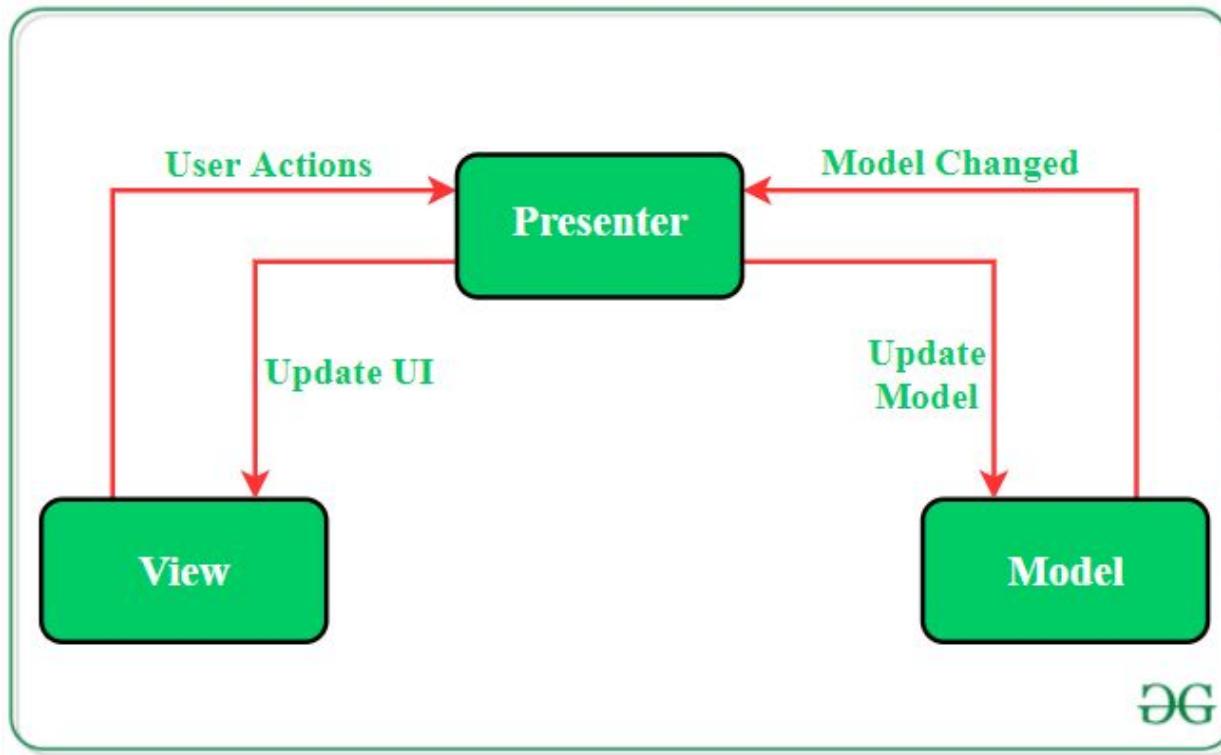
# Web & Server Architecture

REST API concept, used in Facebook, Instagram, and common web applications



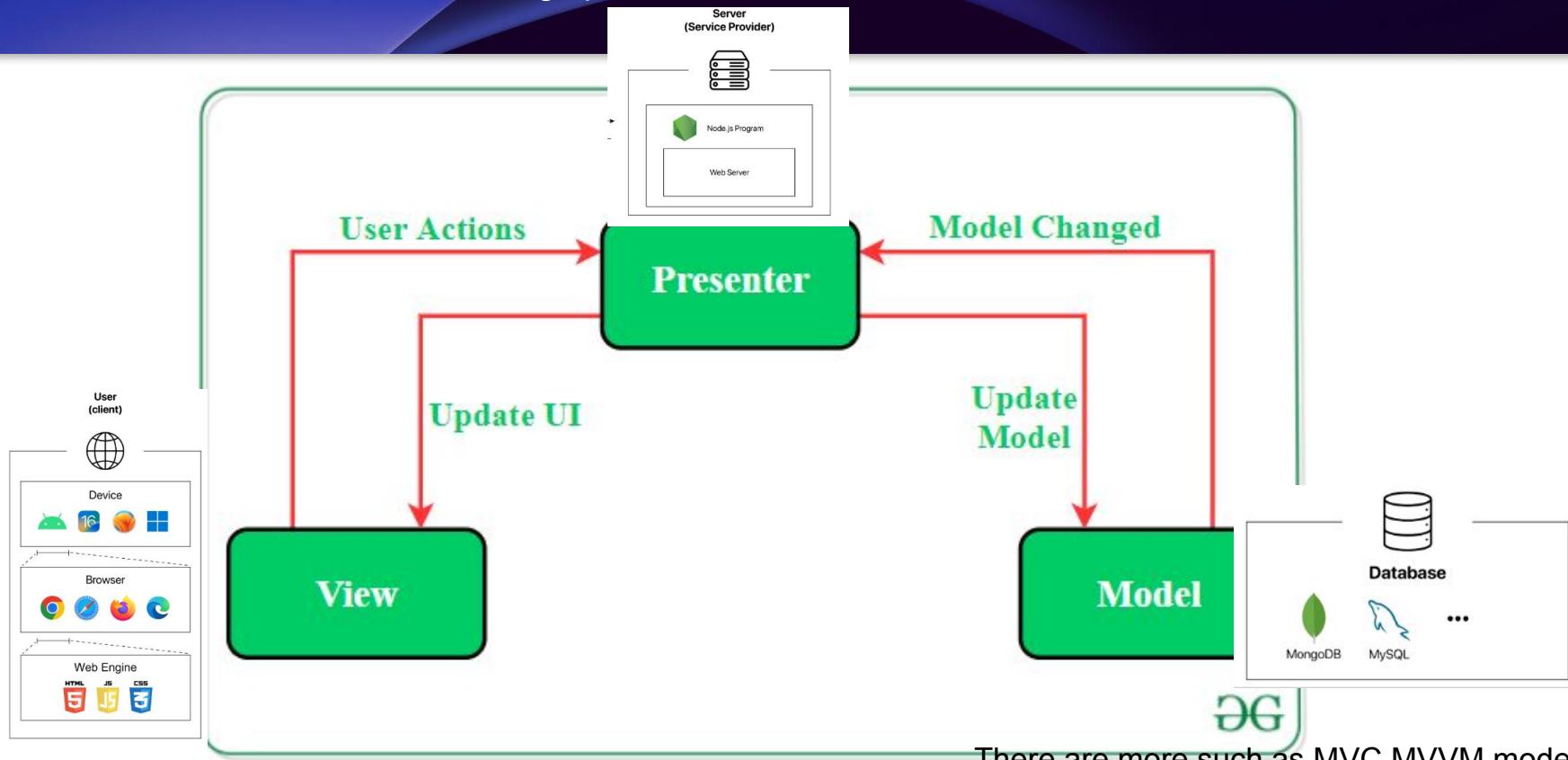
# MVP

Model View Presenter Software design pattern

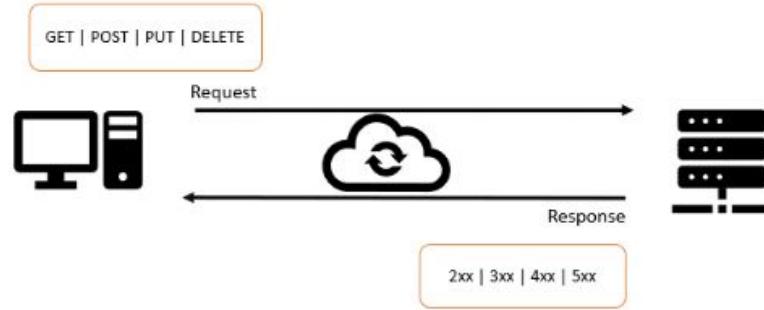


# MVP

## Model View Presenter Software design pattern

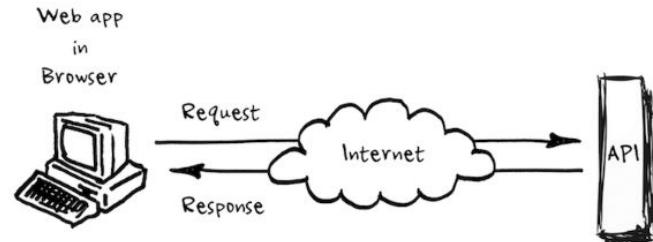


# API ... ?



An **application programming interface (API)**  
is a way for two or more computer programs to communicate with each other

# API Example: Twitter's Tweet Detail



# API Example: Twitter's Tweet Detail

```
X Headers Payload Preview Response Initiator Timing Cookies
▼ {data: {threaded_conversation_with_injections_v2: {...}}}
  ▼ data: {threaded_conversation_with_injections_v2: {...}}
    ▼ threaded_conversation_with_injections_v2: {...}
      instructions: [{type: "TimelineAddEntries",...}, {type: "TimelineTerminateTimeline", direction: "Top"}]
        ▼ 0: {type: "TimelineAddEntries",...}
          ▶ entries: [{entryId: "tweet-1622208705331884033", sortIndex: "7601163331522891774",...},...]
            ▶ 0: {entryId: "tweet-1622208705331884033", sortIndex: "7601163331522891774",...}
              ▶ content: {entryType: "TimelineTimelineItem", __typename: "TimelineTimelineItem",...}
                entryType: "TimelineTimelineItem"
              ▶ itemContent: {itemType: "TimelineTweet", __typename: "TimelineTweet",...}
                hasModeratedReplies: false
                itemType: "TimelineTweet"
                tweetDisplayType: "Tweet"
              ▶ tweet_results: {result: {__typename: "Tweet", rest_id: "1622208705331884033", has_birdwatch_notes: false,...}}
                ▶ result: {__typename: "Tweet", rest_id: "1622208705331884033", has_birdwatch_notes: false,...}
                  ▶ core: {user_results: {...}}
                  ▶ edit_control: {edit_tweet_ids: ["1622208705331884033"], editable_until_msecs: "1675601469000",...}
                  ▶ edit_perspective: {favorited: false, retweeted: false}
                    has_birdwatch_notes: false
                    is_translatable: true
                ▶ legacy: {created_at: "Sun Feb 05 12:21:09 +0000 2023", conversation_id_str: "1622208705331884033",...}
                  conversation_id_str: "1622208705331884033"
                  created_at: "Sun Feb 05 12:21:09 +0000 2023"
                ▶ display_text_range: [0, 211]
                ▶ entities: {user_mentions: [], urls: [], hashtags: [], symbols: []}
                  favorite_count: 131
                  favorited: false
                  full_text: "กินข้าว นั่งพื้น แล้วจ่ายป่า/ggn กว่า \"ชูเปอร์โพลเทวต์คนไทยมีความหวังถูกใจไปต่อ\" กพร้อมกับข่าวดีป้อนไว้วางสวนลุมตอนเจ็ดโมง แนะนำข้าวแกง/ก/กุ้งส้มห่มดหร่วงมาก ไม่
id_str: "1622208705331884033"
is_quote_status: false
lang: "th"
```

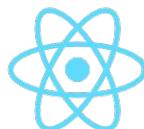


npm

TS

BABEL

JS<sup>®</sup>



npm

TS



# Web Framework.



Why we don't want  
to write only  
HTML, CSS, JS ?

Because it hard as hell.

Creating web application without  
library would take us too much  
time.

# Let's create the Todo app using pure HTML, CSS and JS

```
<!DOCTYPE html>
<html>
  <head>
    <title>Parcel Sandbox</title>
    <meta charset="UTF-8" />
  </head>

  <body>
    <div id="app">
      <h1>Todo List !</h1>
      <div
        id="list"
        style="display: flex; gap: 20px; flex-direction: column;">
        <form onsubmit="return handleSubmit()">
          <input placeholder="Type Here" id="input" />
          <button type="submit">Add</button>
        </form>
      </div>
    </div>

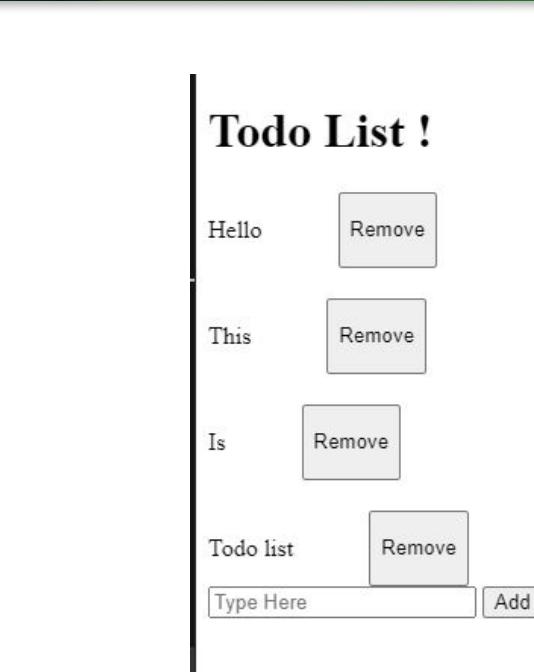
    <script>
      let todos = [];

      function deleteTodo(key) {
        todos.pop(key);
        renderTodos();
      }

      function handleSubmit() {
        let input = document.getElementById("input");
        const inputValue = input.value;
        input.value = "";
        todos.push(inputValue);
        renderTodos();
        return false;
      }

      function renderTodos() {
        const listBox = document.getElementById("list");
        listBox.innerHTML = "";
        for (let todoIndex in todos) {
          const todo = todos[todoIndex];
          listBox.appendChild(renderTodoElement(todoIndex, todo));
        }
      }

      function renderTodoElement(key, value) {
        const todoBox = document.createElement("div");
        todoBox.style = "display:flex; gap:50px;";
        todoBox.innerHTML = `
          <p>${value}</p>
          <button onclick="deleteTodo(${key})"> Remove </button>
        `;
        return todoBox;
      }
    </script>
  </body>
</html>
```



<https://codesandbox.io/s/simple-todolist-in-vanila-qd4h4c>

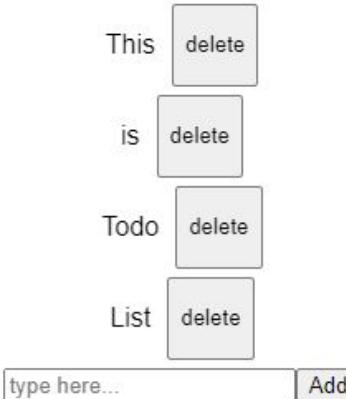
# Let's create the Todo app using React

```
import './styles.css';
import { useRef, useState } from "react";

export default function App() {
  const [todos, setTodos] = useState([]);
  const inputRef = useRef();
  const onSubmit = (e) => {
    e.preventDefault();
    const inputText = inputRef.current.value;
    setTodos((d) => [...d, inputText]);
    inputRef.current.value = "";
  };
  const deleteTodo = (key) => {
    setTodos((v) => v.filter((todo, index) => index !== key));
  };
  return (
    <div className="App">
      <h1> Todo List ! </h1>

      <div style={{ display: "flex", gap: 5, flexDirection: "column", alignItems: "center" }}>
        {todos.map((todo, key) => (
          <div key={key} style={{ display: "flex", gap: 10 }}>
            <p>{todo}</p>
            <button onClick={() => deleteTodo(key)}>Delete</button>
          </div>
        ))}
        <form onSubmit={onSubmit}>
          <input placeholder="type here..." ref={inputRef} />
          <button>Add</button>
        </form>
      </div>
    </div>
  );
}
```

## Todo List !



<https://codesandbox.io/s/todo-react-e7jwc7>

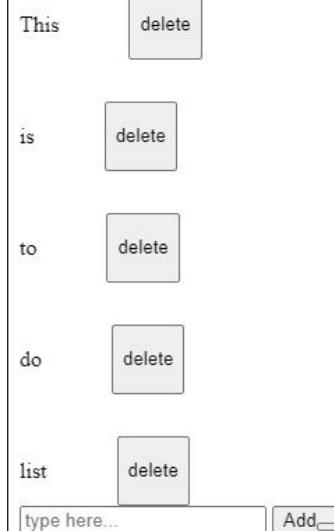
# Let's create the Todo app using Svelte

```
<script>
  let todos = [];
  let todoText = "";
  const addTodos = () => {
    todos = [...todos, todoText];
    todoText = "";
  };

  const deleteTodo = key => {
    todos = todos.filter((data, index) => index !== key);
  };
</script>

<main>
  <h1>Todo List ! </h1>
  <div style="display:flex;gap:30px;flex-direction:column" >
    {#each todos as todo,key}
    <div style="display:flex;gap:50px">
      <p>{todo}</p>
      <button on:click={()=>deleteTodo(key)}>delete</button>
    </div>
    {/each}
  </div>
  <form on:submit|preventDefault={addTodos}>
    <input bind:value={todoText} placeholder="type here..." />
    <button type="submit" >Add</button>
  </form>
</main>
```

## Todo List !

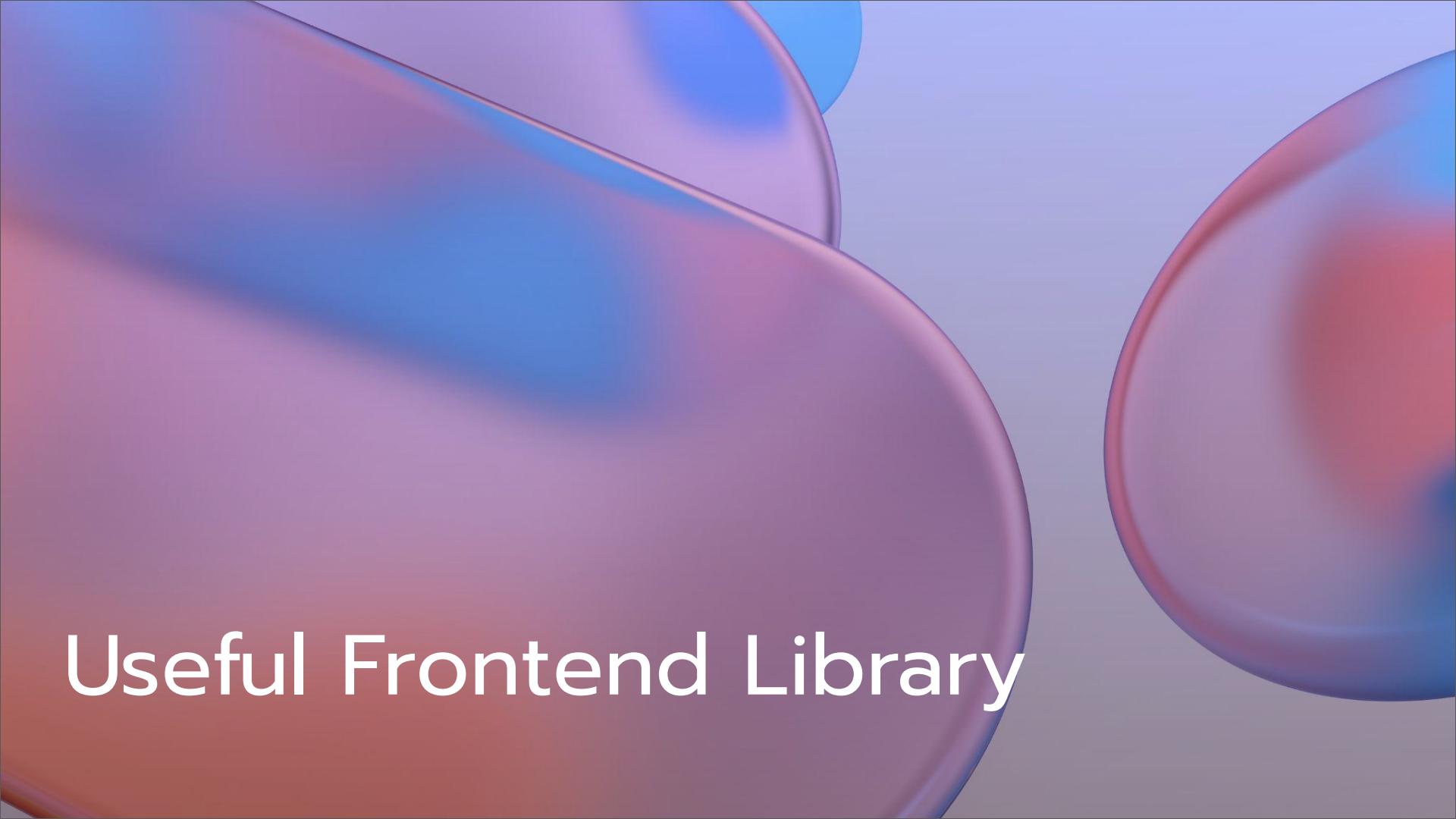


<https://codesandbox.io/s/todo-svelte-jcljn>



# Advantage of using web framework.

- Easy to learn.
- Easy to set up.
- There are a lot of libraries to use.

The background of the slide features a minimalist abstract design. It consists of several large, semi-transparent circles in shades of blue, purple, and red, which overlap each other. These circles are set against a light gray background. The overall effect is clean and modern, suggesting a focus on technology or design.

# Useful Frontend Library

# MUI

[MUI Core](#)

React components library that use Material design principle.

# Move faster with intuitive React UI tools

MUI offers a comprehensive suite of UI tools to help you ship new features faster. Start with Material UI, our fully-loaded component library, or bring your own design system to our production-ready components.

[Get started >](#)

npm install @mui/material @emotion...



🕒 March 25th



Check the docs for getting every component API



Assigned to  
Michael Scott

60%



Ultraviolet

Basement • Beside Myself



React

TypeScript

CSS



React

JavaScript

February 2023 ▾

S	M	T	W
			1
5	6	7	8
12	13	14	15
19	20	21	22
26	27	28	

Yesterday

Today

50°C

25°C

Fonts

# TanStack Query <sup>v4</sup>

**Powerful asynchronous state management for TS/JS, React, Solid, Vue and Svelte**

Toss out that granular state management, manual refetching and all the bowls of async-spaghetti code. TanStack Query gives you fine-grained control, always-up-to-date auto-managed queries and tools that **directly improve both your developer and user experiences**.

## TanStack Query

[TanStack Query](#)

A state management library for managing data sent by an API.

[GET STARTED](#)

Want to skip the docs? Take the official React Query course



## Getting Started

[Feature Overview](#)

[Tutorial](#)

[Examples](#)

[FAQs](#)

[Main Concepts](#)

## Upgrading

[Upgrading from v5](#)

[Migrating from @reach/router](#)

# React Router

[React Router](#)

A client side routing library using for routing through your web application.

[createMemoryRouter](#) NEW

[RouterProvider](#) NEW

## What's New in 6.4?

v6.4 is our most exciting release yet with new data abstractions for reads, writes, and navigation hooks to easily keep your UI in sync with your data. The new feature overview will catch you up.



## I'm New

Start with the tutorial. It will quickly introduce you to the core features of React Router: from configuration to reading and mutating data, to pending and optimistic rendering.



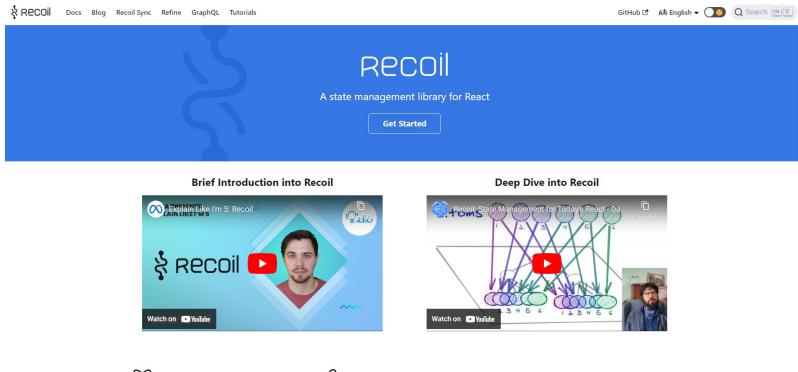
## I'm on v5

This migration guide will help you migrate incrementally and keep your code clean along the way. Or, do it all in one yolo commit! Either way, we've got you covered to start using the new features right away.

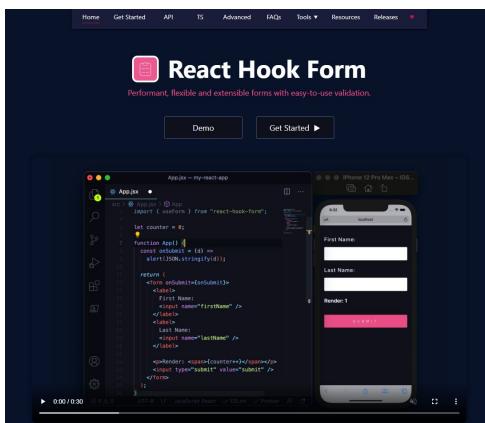
## I'm Stuck!

Running into a problem? Chances are you're not alone. Check out the common questions about React Router.

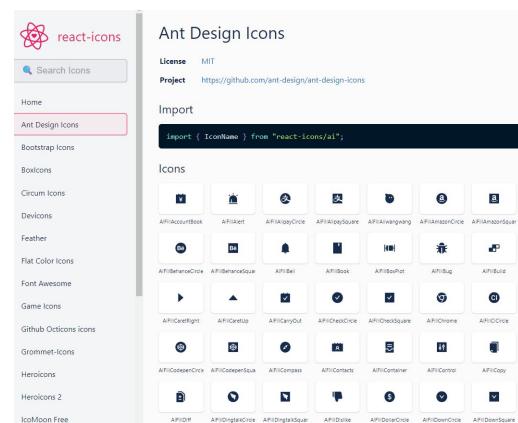
# Other Libraries you might be interested.



The Recoil library homepage features a blue header with the Recoil logo and navigation links for Docs, Blog, Recoil Sync, Refine, GraphQL, and Tutorials. Below the header is a large "Recoil" title and a subtitle "A state management library for React". There are two video thumbnails: "Brief Introduction into Recoil" and "Deep Dive into Recoil", each with a "Watch on YouTube" button. A "Get Started" button is located at the bottom left.



The React Hook Form homepage has a dark theme. It features a "React Hook Form" logo and a subtitle "Performant, flexible and extensible forms with easy-to-use validation.". Below this are "Demo" and "Get Started" buttons. To the right is a screenshot of a code editor showing a form component and its corresponding mobile preview. The code includes imports for React, useState, and the form hook, and defines a function that handles form submission.



The react-icons library homepage has a light theme. It features a logo and a search bar labeled "Search Icons". Below the search bar is a sidebar with categories like Home, Ant Design Icons (which is highlighted), Bootstrap Icons, Boxicons, Circum Icons, Devicons, Feather, Flat Color Icons, Font Awesome, Game Icons, Github Octicons Icons, Grommet Icons, Heroicons, Heroicons 2, and IcoMoon Free. The main content area shows a grid of icons from the Ant Design Icons category.

## A X I O S

Promise based HTTP client for the browser and node.js

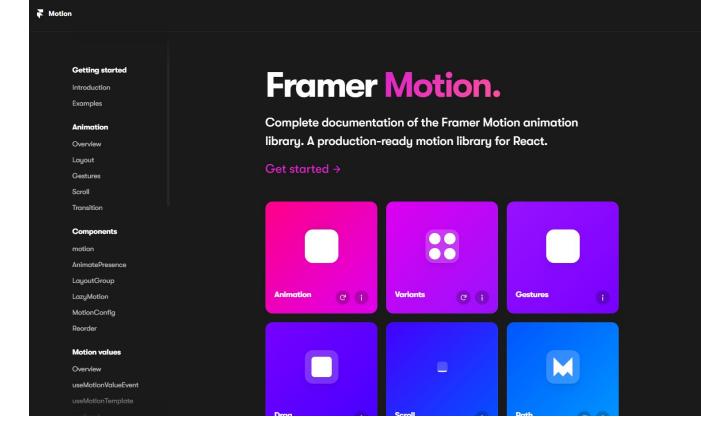
Axios is a simple promise based HTTP client for the browser and node.js. Axios provides a simple to use library in a small package with a very extensible interface.

[Get Started](#) [View on GitHub](#)

```
import axios from "axios";  
axios -|
```

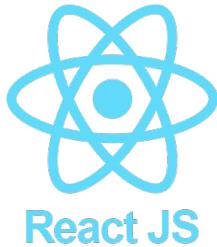
Sponsors:

SerpApi CasinoREVIEWS.net

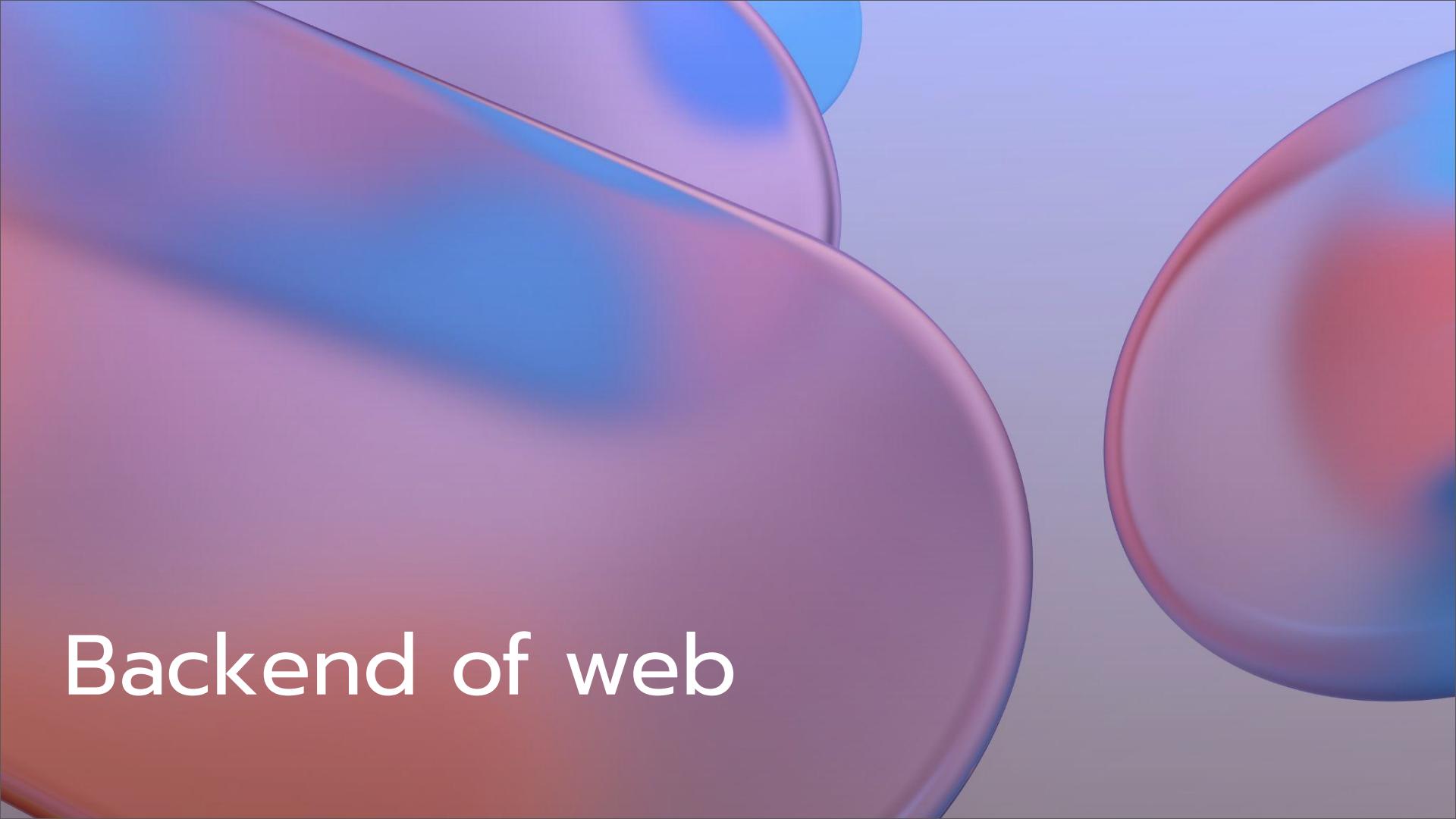


The Framer Motion library homepage has a dark theme. It features a "Motion" logo and a "Getting started" section with links to Introduction, Examples, Animation, Overview, Layout, Gestures, Scroll, Transition, Components, and Motion values. Below this is a "Get started →" button. The main content area displays four cards: "Animation" (pink background), "Variants" (purple background), "Gestures" (blue background), "Drop" (dark blue background), "Scroll" (light blue background), and "Path" (dark blue background).

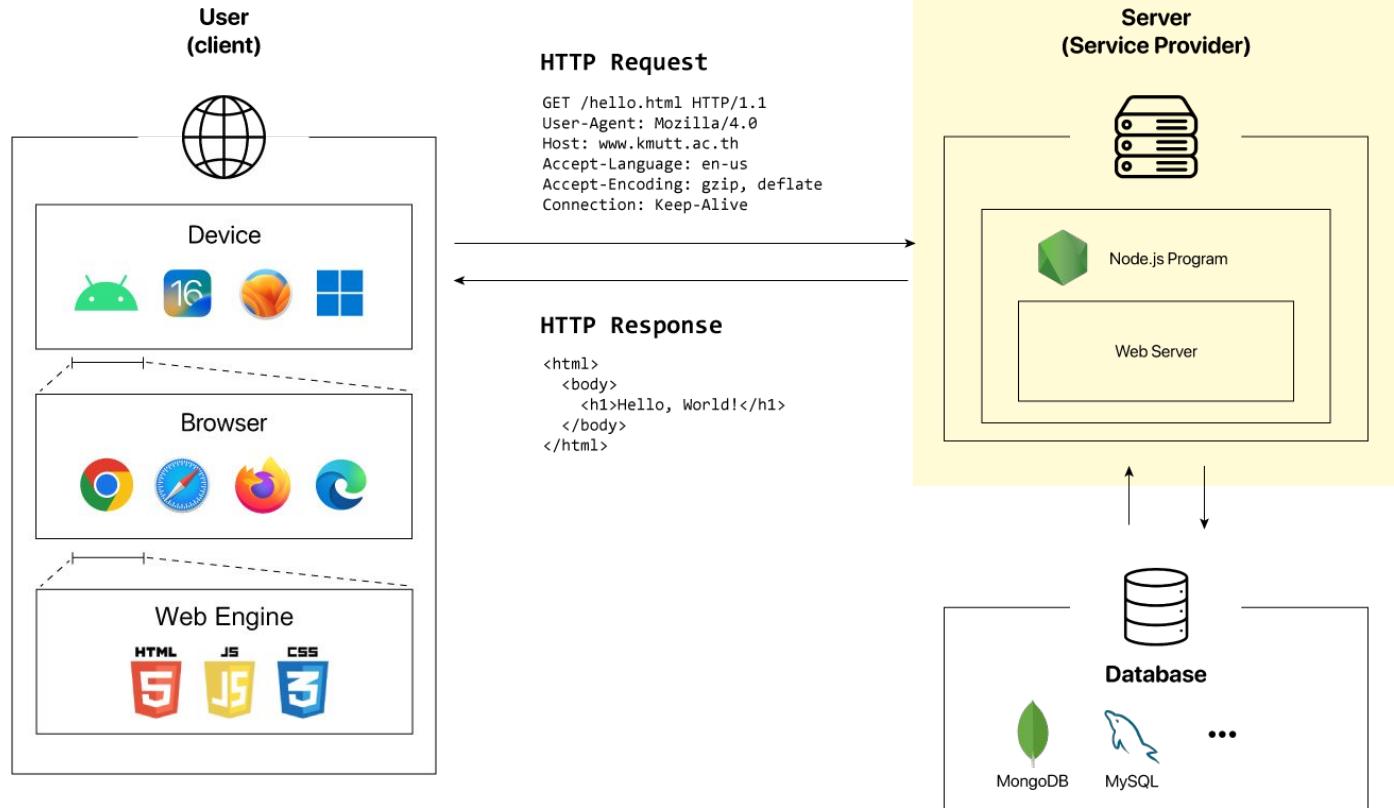
Here is some of web framework (Frontend) you can take some look.

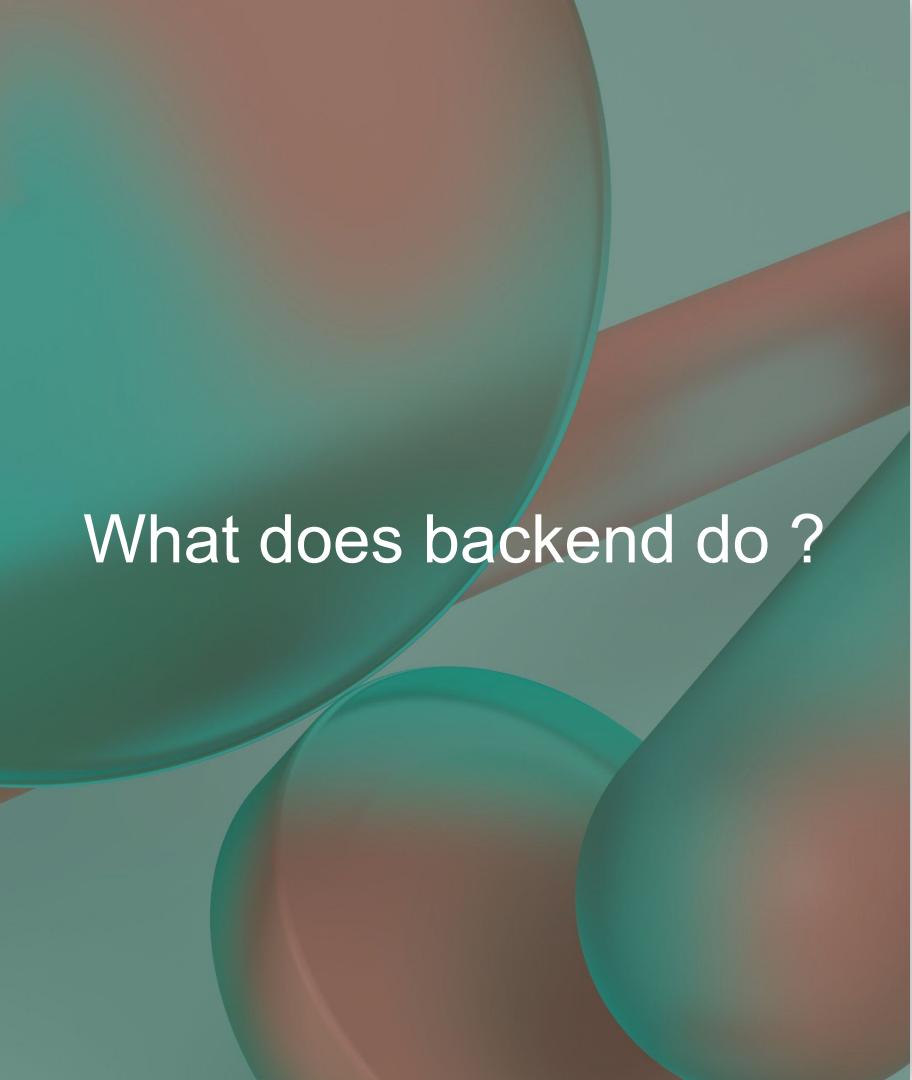


Each web framework has their own style and community, so you can choose what you like.

The background of the slide features a minimalist design with three large, semi-transparent circles. One circle is positioned in the lower-left foreground, another is in the upper-left background, and a third is on the right side. These circles overlap each other and the slide's surface, creating a sense of depth. The colors used are soft purples, blues, and reds, which blend together.

Backend of web





## What does backend do ?

- Receive, process and response data for client to use.
- Communicate to database and make CRUD (create, read, update, delete) operation.
- Process user credential for authentication and authorization.

# How can I write a backend ?

There are a lot of **tools**, **languages** and **libraries** you can choose.

Here is what you might want to learn:



**mongoose**

elegant mongodb object modeling for node.js

**Express**



# Node.js

Node.js is a JavaScript runtime built on Chrome's V8 JavaScript engine.

Basically, The thing that make your Javascript can be run outside browser just like Java or python.

We use Node.js to create a server to listen requests from client.

# Express

## Express.js

Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.

Basically, A library for nodejs to provide simpler server development in Node.js.

We use Express to create server, provide middleware or even do a server side rendering.

# mongoose

elegant `mongodb` object modeling for `node.js`

## Mongoose

Mongoose is a JavaScript object-oriented programming library that creates a connection between MongoDB and the Node.js JavaScript runtime environment

We use mongoose to communicate with database (Mongodb). It provided type structure that help you developing.

So, here is some question that you might need to ask to yourself first.

- What kind of **client** are we communicating with ?
- What kind of **Api** are we building ?
- What kind of **database** are we using ?

But that's a topic of the backend session.....

# But backend might be a little complicated to virtualized.

So why don't we see it from the real project?

package.json    README.md    index.js    Sandbox Info

```
1 import express from "express";
2
3 const app = express();
4
5 app.get("/", (req, res) => {
6   return res.send("Hello This is our webpage !");
7 });
8
9 app.get("/cat", (req, res) => {
10   return res.send({ name: "Meow", breed: "scottish fold", age: 35 });
11 });
12
13 app.post("/cat", (req, res) => {
14   return res.send({
15     name: "Post Meow",
16     description: "You won't be able to access this on normal webpage",
17     breed: "scottish fold",
18     age: 35,
19   });
20 });
21
22 app.listen(3000, () => {
23   console.log("This app is listening on port 3000 !");
24 });
25 |
```

<https://codesandbox.io/p/sandbox/eloquent-frost-s2e5vx>

# Look outside Javascript.

So.... you don't like Javascript but you want to be a web developer ?

Don't worry, we can develop website without using any of Javascript !

The background features a minimalist design with three large, semi-transparent circles. One circle is positioned in the upper left, another in the lower right, and a third smaller one is located near the top center. The circles overlap, creating a soft, blurred effect where their colors—shades of blue, purple, and red—merge. The overall aesthetic is clean and modern.

Frontend without Javascript.

# Flutter - Frontend Alternative

Flutter use Dart instead of HTML,CSS and Javascript. If you write your web application in Flutter, you can convert it into Mobile Application or a computer program!

## Build apps for any screen



[Flutter - Build apps for any screen](#)

# WebAssembly

Alternative of Javascript  
that's even faster than  
Javascript !

Web assembly allows you to run binary instruction in web browser ! you can write an application in Rust, Python, Go and other languages then run it (almost) natively in browser.



[WebAssembly](#)



# Sycamore - Writing web application in Rust !

Sycamore is Rust library that let you write webAssembly instead of javascript.

The screenshot shows the official Sycamore website homepage. It features a large orange leaf logo at the top right. Below the logo, the word "Sycamore" is written in a bold, orange, sans-serif font. A subtext below it reads: "A reactive library for creating web apps in Rust and WebAssembly". There are several callout boxes: one red box on the left says "Lightning speed" and describes how Sycamore harnesses the full power of Rust via WebAssembly; another orange box in the middle says "Ergonomic and intuitive" and describes how it allows writing natural Rust code without a virtual DOM; and a yellow box on the right says "No JavaScript" and simply states "Had enough of JavaScript? So have we."

**Sycamore**

A reactive library for creating web apps in Rust and WebAssembly

Stars 2k crates.io v0.8.2 docs.rs passing contributors 50 discord 98 online

Get Started

**Lightning speed**

Sycamore harnesses the full power of Rust via WebAssembly, giving you full control over performance.

**Ergonomic and intuitive**

Write code that feels natural. Everything is built on reactive primitives without a cumbersome virtual DOM.

**No JavaScript**

Had enough of JavaScript? So have we.

[Sycamore \(sycamore-rs.netlify.app\)](https://sycamore-rs.netlify.app)

The background features a minimalist design with three large, semi-transparent circles. One circle is positioned in the lower-left foreground, another in the upper-right background, and a third smaller one is visible near the top center. The circles have a soft, blurred gradient from blue at the top to red at the bottom, creating a sense of depth and motion.

Backend without Javascript.

# Gin, Fiber: Go's alternative web framework.

Gin and Fiber is a web framework that let you write Go instead of Javascript in Node.js. These two are very popular in modern web development and provide more performance than Nodejs!

The screenshot shows the official website for Gin Web Framework. At the top right, there are links for Documentation and Blog. The main title "Gin Web Framework" is displayed prominently. Below it, a sub-section titled "Fiber" describes it as "An Express-inspired web framework written in Go." It highlights that Fiber is built on Fasthttp, the fastest HTTP engine for Go, and is designed for fast development with zero memory allocation and performance in mind. A code snippet demonstrates a simple Go application using the Fiber library:

```
package main

import (
    "log"
    "github.com/gofiber/fiber/v2"
)

func main() {
    app := fiber.New()

    app.Get("/", func(c *fiber.Ctx) error {
        return c.SendString("Hello, World!")
    })

    log.Fatal(app.Listen(":3000"))
}
```

Below the code, a screenshot of a browser window shows the output: "Hello, World!" at <http://localhost:3000>. A large blue button labeled "Get Started →" is present, along with the text "...or scroll to learn more." At the bottom, a section titled "Robust Routing" is partially visible.

[Gin Web Framework \(gin-gonic.com\)](http://gin-gonic.com)  
[Fiber \(gofiber.io\)](http://gofiber.io)

# Rust - Actix a web framework in Rust

Gin and Fiber is a web framework that let you write Rust instead of Javascript in Nodejs. It even give more performance than Go !.

[Actix](#)



Actix

Actix Web is a powerful, pragmatic, and extremely fast web framework for Rust

[Get Started](#)

## 🛡 Type Safe

Forget about stringly typed objects, from request to response, everything has types.

## 💻 Feature Rich

Actix provides a lot of features out of box. HTTP/2, logging, etc.

## 🔌 Extensible

Easily create your own libraries that any Actix application can use.

## ⚡ Blazingly Fast

Actix is blazingly fast. Don't take our word for it -- [see for yourself!](#)

```
use actix_web::{get, web, App, HttpServer, Responder};

#[get("/")]
async fn index() -> impl Responder {
    "Hello, World!"
}

#[get("/{name}")]
async fn hello(name: web::Path<String>) -> impl Responder {
    format!("Hello {}!", &name)
}

#[actix_web::main]
async fn main() -> std::io::Result<()> {
    HttpServer::new(|| App::new().service(index))
        .bind(("127.0.0.1", 8080))?
        .run()
}
```

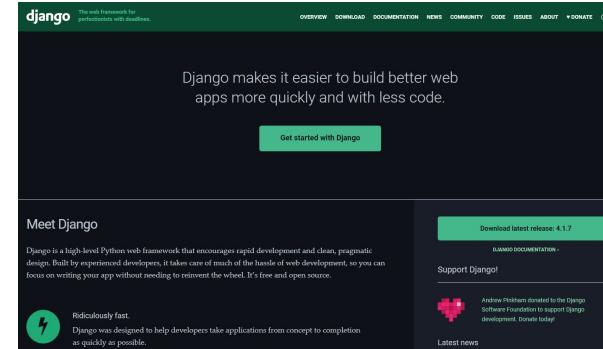
Hello World!

Getting started with Actix is easy. An Actix app comes with a URL routing system that lets you match on URLs and invoke individual handlers.

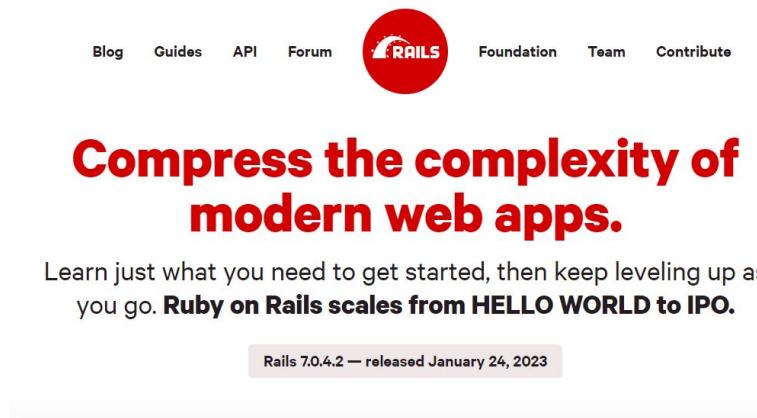
# And Other....



The Phoenix Framework homepage features a red header with the logo and title. Below it, a large section highlights "Peace of mind from prototype to production" with a screenshot of a live application and some code. A "GET STARTED" button is at the bottom.



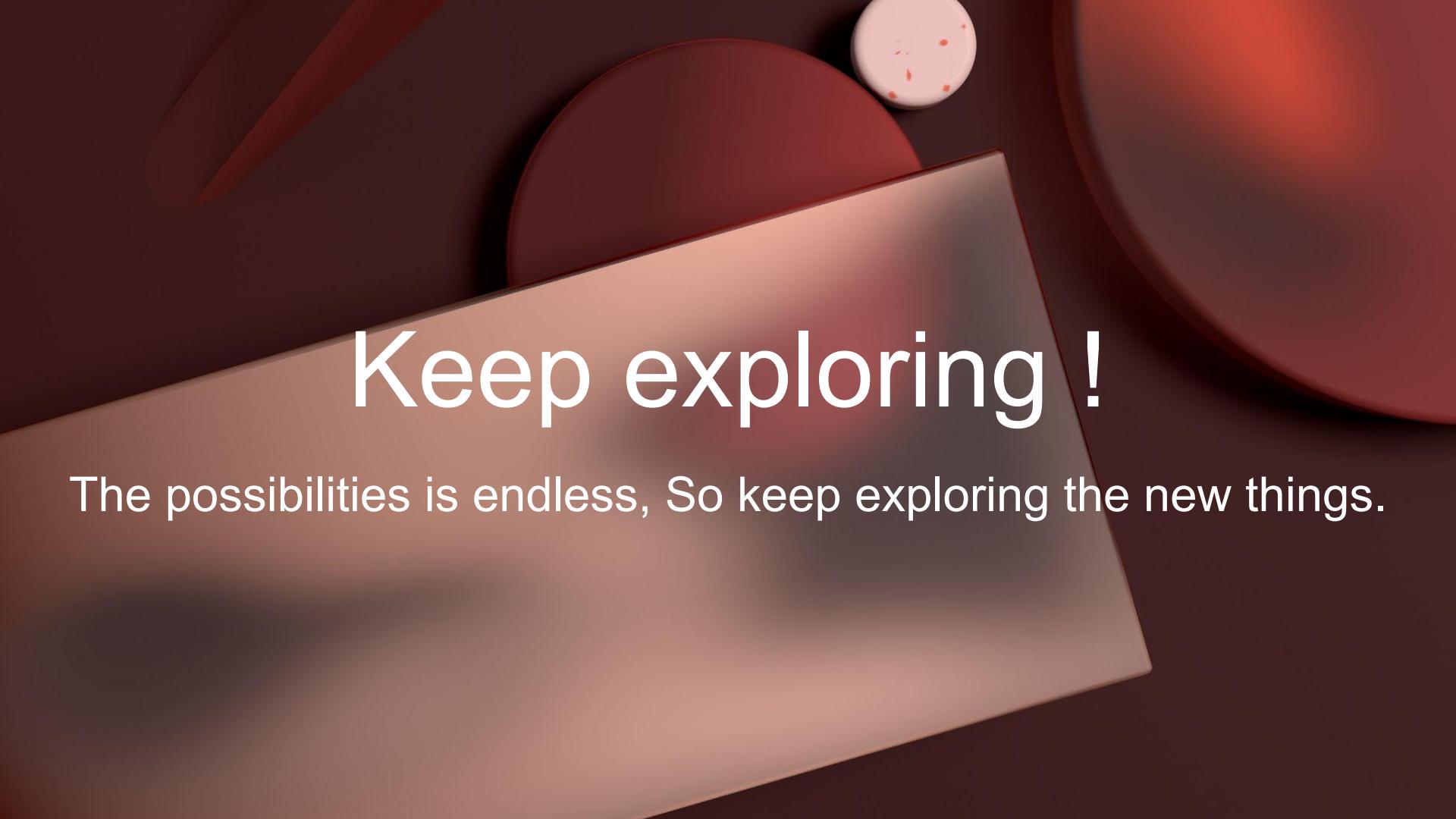
The Django homepage has a dark header with navigation links. It features a main section with a "Get started with Django" button and a "Meet Django" section with a video thumbnail and text about the framework's history and philosophy.



The Rails homepage includes a red circular logo with the word "RAILS". It features a large red headline: "Compress the complexity of modern web apps." Below it, a call-to-action says "Learn just what you need to get started, then keep leveling up as you go. Ruby on Rails scales from HELLO WORLD to IPO." A "Rails 7.0.4.2 — released January 24, 2023" badge is at the bottom.



The Spring Boot homepage has a dark header with navigation links. On the left is a sidebar with links like "Spring Boot", "Spring Framework", "Spring Data", etc. The main content area features a "Spring Boot 3.0.3" section with tabs for "OVERVIEW", "LEARN", "SUPPORT", and "SAMPLES". It describes Spring Boot's purpose and provides links for more information.



# Keep exploring !

The possibilities is endless, So keep exploring the new things.