



# FORMATIO N

- Management

# Bienvenue sur cette formation Boomerang Consulting

- Guilian GANSTER
- 6 ans en tant que développeur freelance spécialisé dans les startups innovantes
- 4 ans en tant que formateur

Fort de 20 années d'expertises dans la formation professionnelle continue, notamment en informatique, Boomerang Consulting est un organisme qui cherche à vous offrir la qualité optimum pour vos formations avec :

- Un accueil de qualité et de proximité,
- Un service commercial, administratif et pédagogique à votre disposition,
- Un positionnement technologique étendu avec des formateurs experts, qualifiés et certifiés pour certains,
- Une qualité de services et une réponse à vos besoins grâce aux contenus pédagogiques **SUR-MESURE**.

# Centre de formation certifié



- L'engagement pour la qualité en étant certifié Qualiopi.
- Centre de formation spécialisé dans les domaines de l'IT, Ressources humaines et développement personnel.

## Qualité et satisfaction

- Des formateurs certifiés et expérimentés.
- Un taux de satisfaction client supérieur à 93% suite aux évaluations de fin de stage.

## Horaire et convocations

- 9h00- 17h30 (pause 12h15 - 13h30)
- 15 mn de pause le matin
- 1h de pause déjeuner
- 15 mn de pause l'après midi
- Dernier jour à 16h30 (si le plan de cours est terminé uniquement)

# Prérequis

- Sur quels langages avez vous l'habitude de travailler ?
- Avez vous déjà utilisé un langage bas niveau (C, Rust, Go, ...)
- Quelles sont vos attentes/cas d'usage à l'issue de cette formation ?
- Droits admin sur vos PC ?

# Objectifs pédagogiques

A l'issue de cette formation, vous serez capable de :

- Comprendre, lire et écrire des classes en langage C++
- Traduire une conception objet en langage C++
- Développer des applications en langage C++

# Les feuilles d'émargement

Matin et après midi

- Première chose à faire : signer les feuilles d'émargement. Merci de bien vouloir faire des signatures conformes.
- Les croix et initiales ne sont pas autorisées.

# Les évaluations de fin de stage

- Dernier jour de la formation : le centre de formation à l'obligation contractuelle de fournir vos évaluations à votre entreprise avant 15h, donc au retour de la pause déjeuner, 2 ou 3 h avant la fin de la formation, je vous ferai remplir les évaluations formateur.

# Déroulé et structure de la formation

- Équilibre théorie / pratique: 50%
- Chaque notion sera illustré d'un bloc théorique suivi d'un exercice pratique. Une slide avec un code minimale sera fournis pour la syntaxe. Les exercices pratiques reflètent dans la mesure du possible des cas réels.

# Support de cours et outils pédagogiques

- Les énoncés des TP de validation des acquis sont présent sur ces slides.
- Les corrigés et différents codes d'exemple seront sur Github et/ou sur un Drive et accessibles en fin de formation.

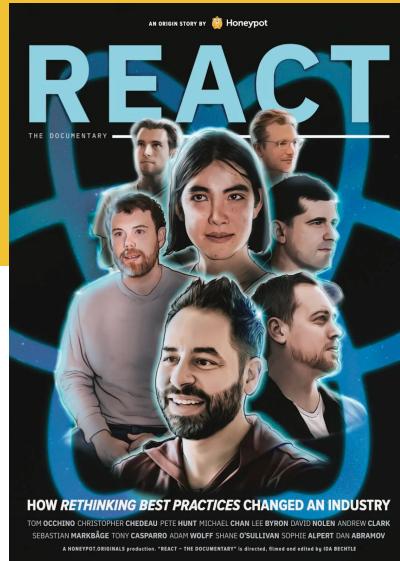
## Suivi quotidien et adaptabilité

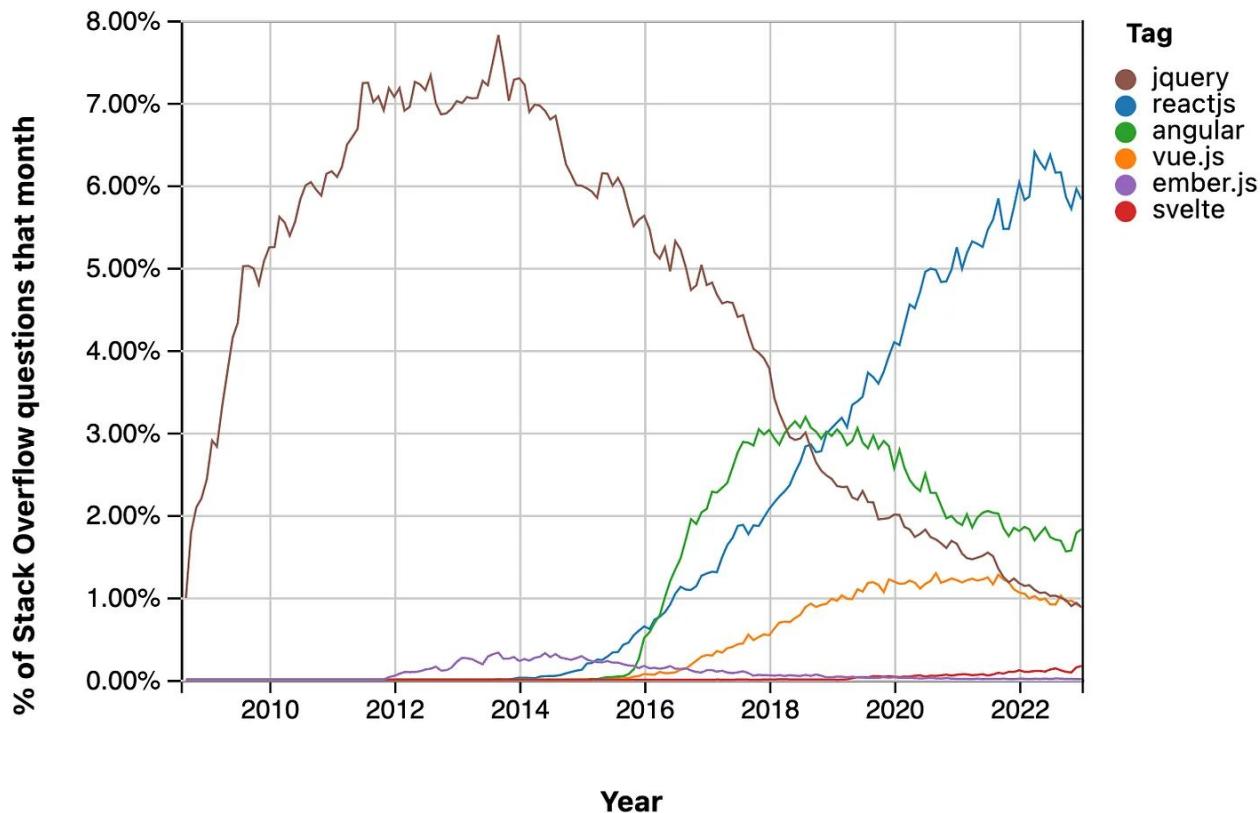
- Google Forms de demi-journée pour la validation des acquis et l'adaptabilité.

A chaque fin de demi journée, les stagiaires devront répondre aux questions suivantes par l'intermédiaire d'un Google Form :

- Le rythme vous convient il ?
- L'équilibre théorie pratique vous convient il ?
- Les notions abordées jusque là sont elles acquises ?
- Y a t'il des notions sur lesquelles revenir avant de poursuivre le cours ?

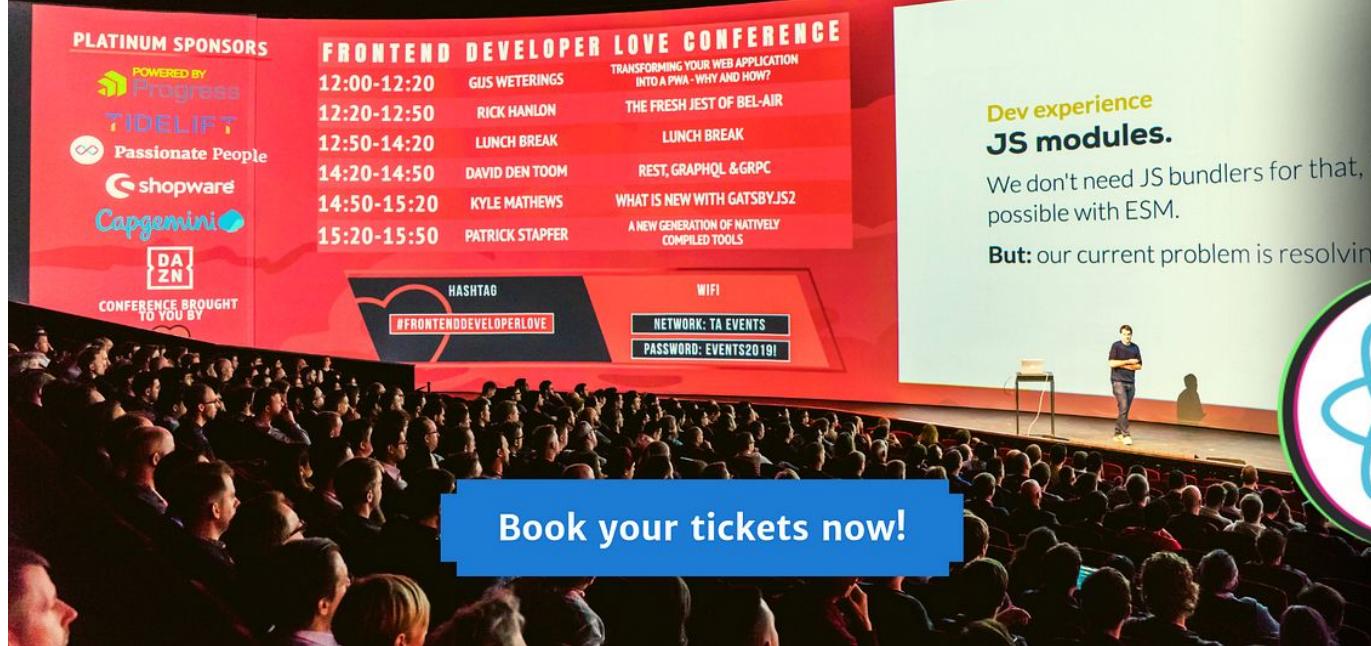
# Chapitre 1 : Histoire et Ecosystème





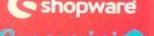
# React Live

13th September | Theater Amsterdam  
[www.reactlive.nl](http://www.reactlive.nl)



**FRONTEND DEVELOPER LOVE CONFERENCE**  
TRANSFORMING YOUR WEB APPLICATION  
INTO A PWA - WHY AND HOW?

**PLATINUM SPONSORS**

- POWERED BY 
-  Tidelift
- Passionate People
-  shopware
-  Capgemini
- 

CONFERENCE BROUGHT TO YOU BY

12:00-12:20 GUS WETERINGS  
12:20-12:50 RICK HANLON  
12:50-14:20 LUNCH BREAK  
14:20-14:50 DAVID DEN TOOM  
14:50-15:20 KYLE MATHEWS  
15:20-15:50 PATRICK STAPFER

LUNCH BREAK  
REST, GRAPHQL & GRPC  
WHAT IS NEW WITH GATSBY.JS2  
A NEW GENERATION OF NATIVELY COMPILED TOOLS

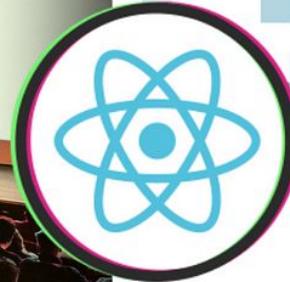
HASHTAG #FRONTENDDVELOPERLOVE  
WIFI  
NETWORK: TA EVENTS  
PASSWORD: EVENTS2019!

**Book your tickets now!**

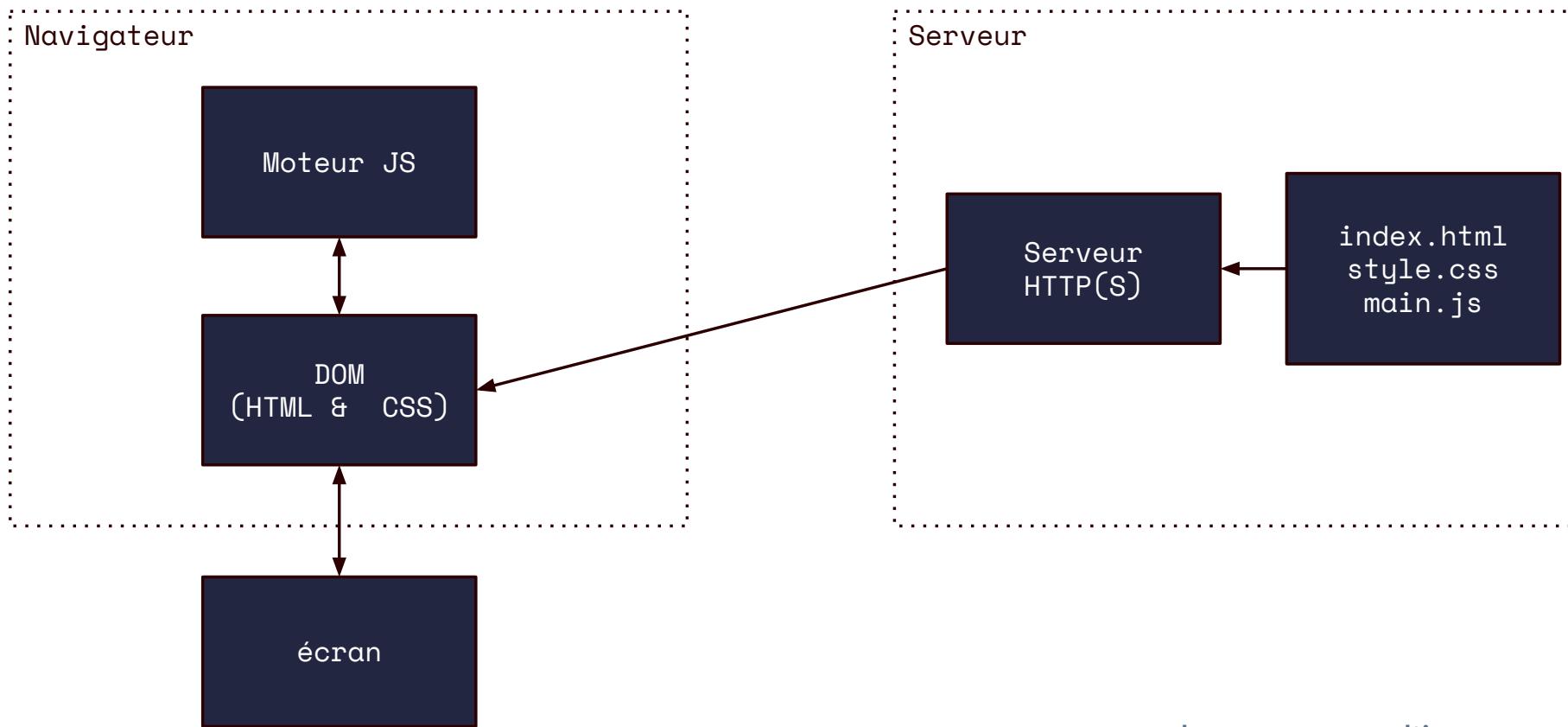
## Dev experience JS modules.

We don't need JS bundlers for that, it's possible with ESM.

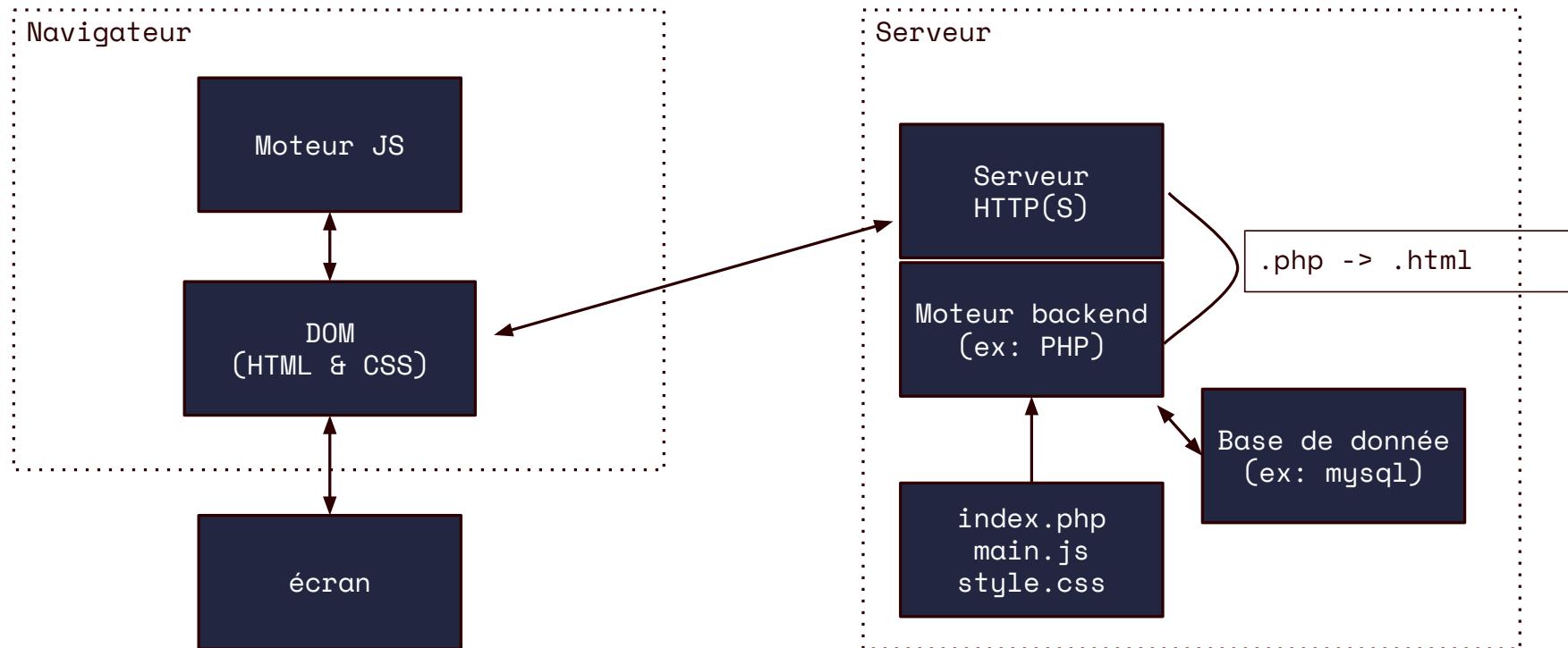
But: our current problem is resolving



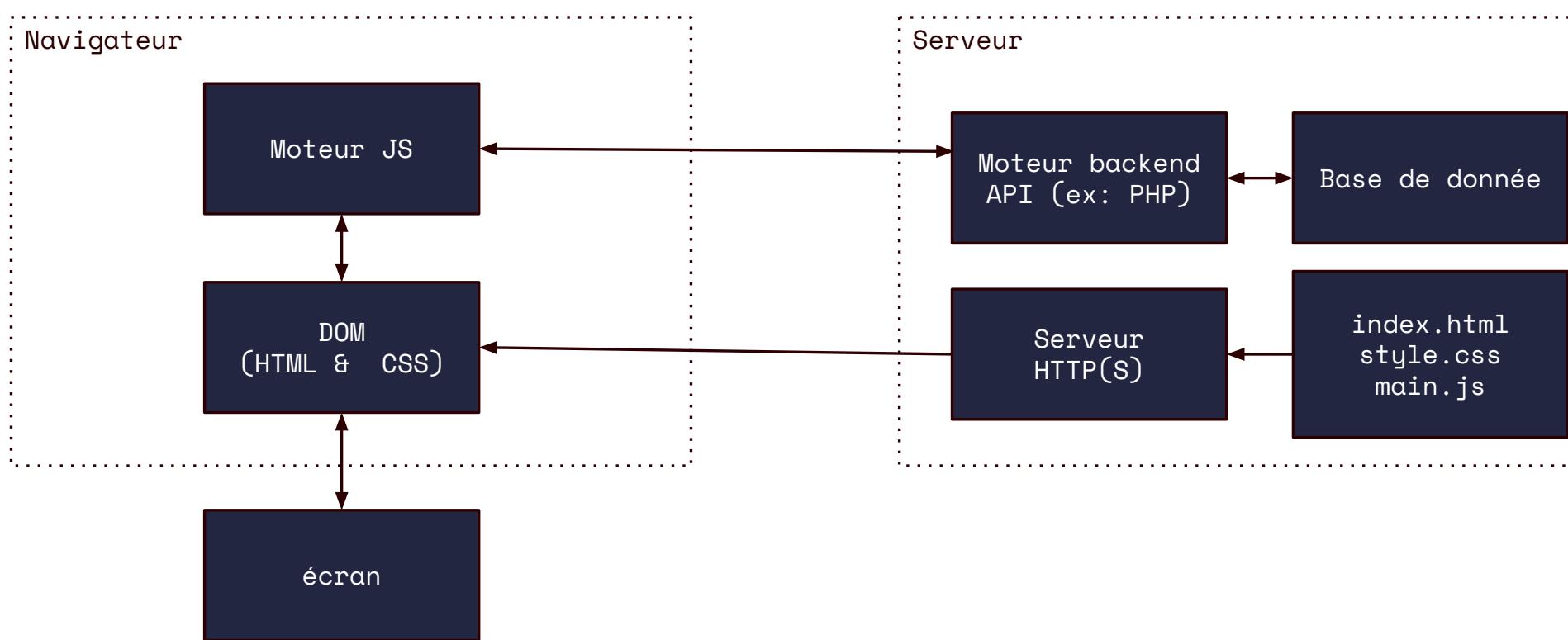
## Site web statique



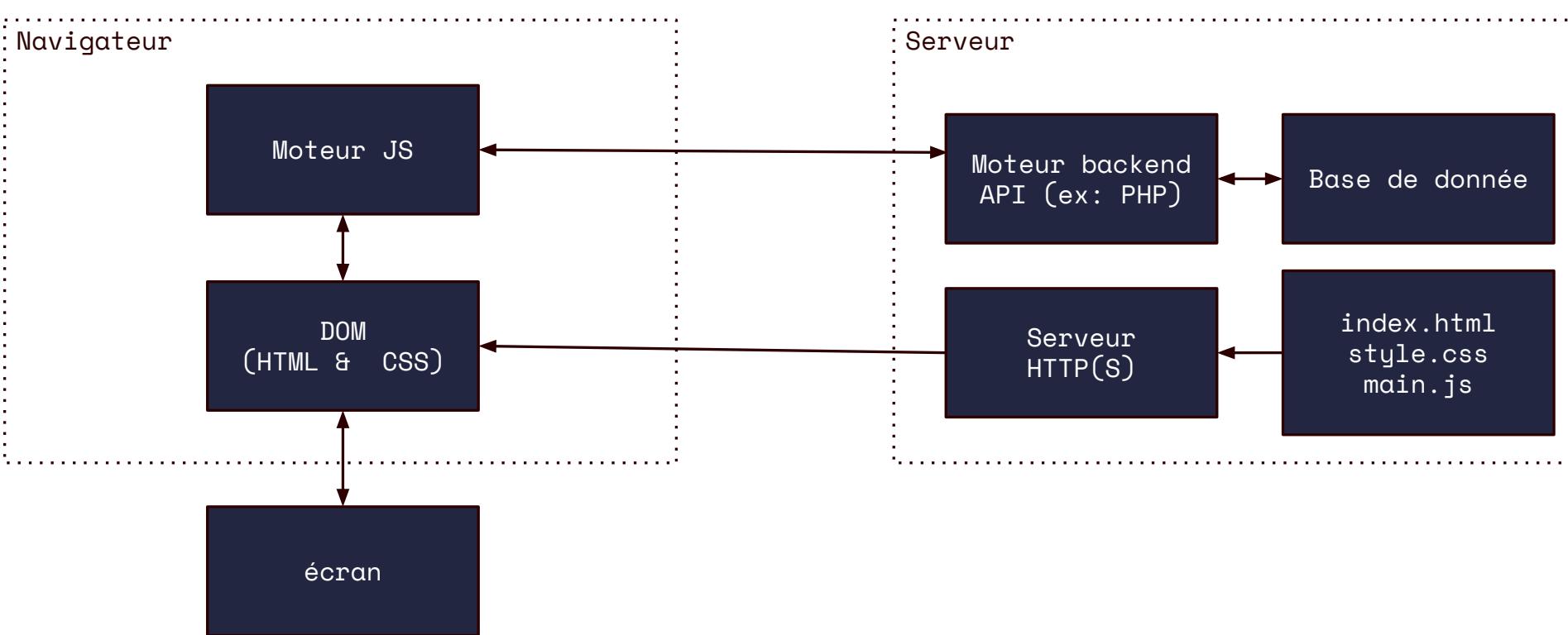
## Site web dynamique (php/ROR/...)



# SPA



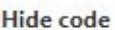
## SSR



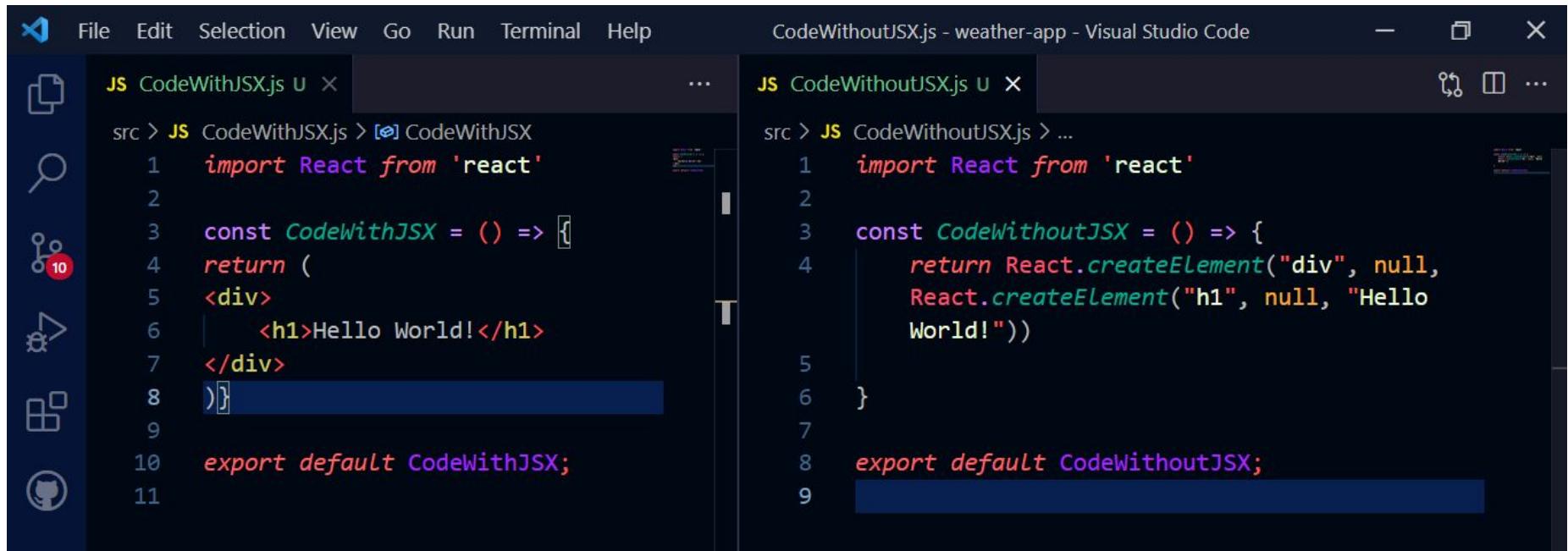
# Philosophie

```
<Progress
  value={50}
/>
```

 Hide code Copy

# JSX



File Edit Selection View Go Run Terminal Help

CodeWithoutJSX.js - weather-app - Visual Studio Code

JS CodeWithJSX.js U X ...

```
src > JS CodeWithJSX.js > [?] CodeWithJSX
1 import React from 'react'
2
3 const CodeWithJSX = () => [
4   return (
5     <div>
6       <h1>Hello World!</h1>
7     </div>
8   )
9
10 export default CodeWithJSX;
```

JS CodeWithoutJSX.js U X ...

```
src > JS CodeWithoutJSX.js > ...
1 import React from 'react'
2
3 const CodeWithoutJSX = () => {
4   return React.createElement("div", null,
5     React.createElement("h1", null, "Hello
6     World!"))
7
8 }
9
9 export default CodeWithoutJSX;
```

# L'environnement

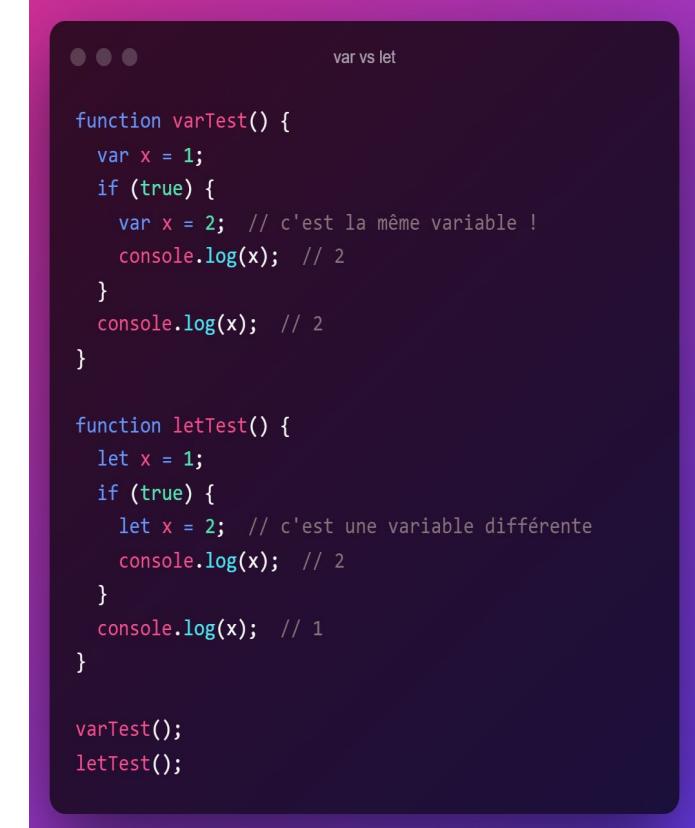




# ES6

{ ES6 JS }

# var vs let



The screenshot shows a mobile browser window with three dots at the top left and the title "var vs let" at the top right. The main content area displays the following JavaScript code:

```
function varTest() {
  var x = 1;
  if (true) {
    var x = 2; // c'est la même variable !
    console.log(x); // 2
  }
  console.log(x); // 2
}

function letTest() {
  let x = 1;
  if (true) {
    let x = 2; // c'est une variable différente
    console.log(x); // 2
  }
  console.log(x); // 1
}

varTest();
letTest();
```

The code illustrates the difference between `var` and `let` declarations. In the first function, both `x` declarations point to the same variable because they are hoisted to the top of their respective scopes. In the second function, each `let` declaration creates a new variable, so both `console.log` statements output `2` for `varTest()` and `1` for `letTest()`.

# ternaires

Uploaded using RayThis Extension

```
let userIsMajor = false;
if (userAge >= 18) {
  userIsMajor = "majeur";
} else {
  userIsMajor = "mineur";
}

//equivalent to
let userIsMajor = userAge >= 18 ? "majeur" : "mineur";
```

Uploaded using RayThis Extension

```
let userIsMajor = false;
if (userAge > 18) {
  userIsMajor = "majeur";
} else if (userAge === 18) {
  userIsMajor = "tout juste majeur";
} else {
  userIsMajor = "mineur";
}

//equivalent to
userIsMajor = userAge > 18 ? true :
  userAge === 18 ? "tout juste majeur"
  : "mineur"
```

# arrow functions

...  
Uploaded using RayThis Extension

```
const a = (param) => param; // paramètre unique, return implicite

const b = param => param; // paramètre unique (parenthèse non requise), return implicite

const c = (param1, param2) => param1 + param2; // paramètres multiples (parenthèses requises), return implicite
```

...  
Uploaded using RayThis Extension

```
const a = () => {
    return "hello"
} // multi-lignes, return explicite

const b = () => "hello" // une seule ligne, return implicite

const c = () => (
    "hello"
) // multi ligne, return implicite
```

# array functions



fonctions fléchées

```
//from this
function foo() {
  console.log("bar");
}

//to this
const foo = () => {
  console.log("bar");
}

//or this
const foo = () => console.log("bar");
```



array functions

```
//from this
const array = [1, 2, 3];

for (let i = 0; i < array.length; i++) {
  console.log(array[i]);
}

//to this
const array = [1, 2, 3];

array.map(i => console.log(i));
```

# Destructuration & spread opérator

• • •

structuration

```
//array structuration
let arrayStructuration = [1, 2]
console.log(arrayStructuration); // [1, 2]

//array copy
let arrayCopy = [...arrayStructuration, 3, 4]
console.log(arrayCopy); // [1, 2, 3, 4]

//array destructuration
let [a, b] = arrayCopy;
console.log(a); // 1
console.log(b); // 2
```

• • •

structuration

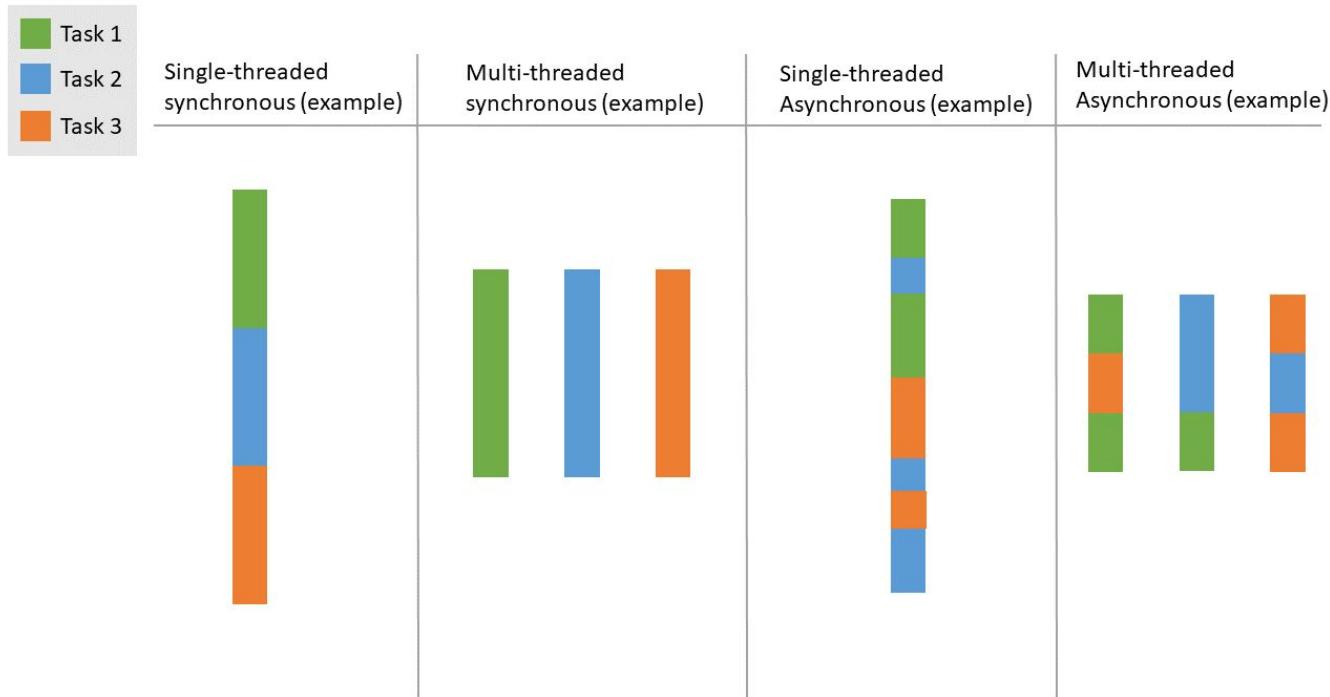
```
//object structuration
let obj = {
  message: "hello world"
}
console.log(obj); //{message: "hello world"}

//object add key
obj.type = "success";
console.log(obj); //{message: "hello world", type: "success"}

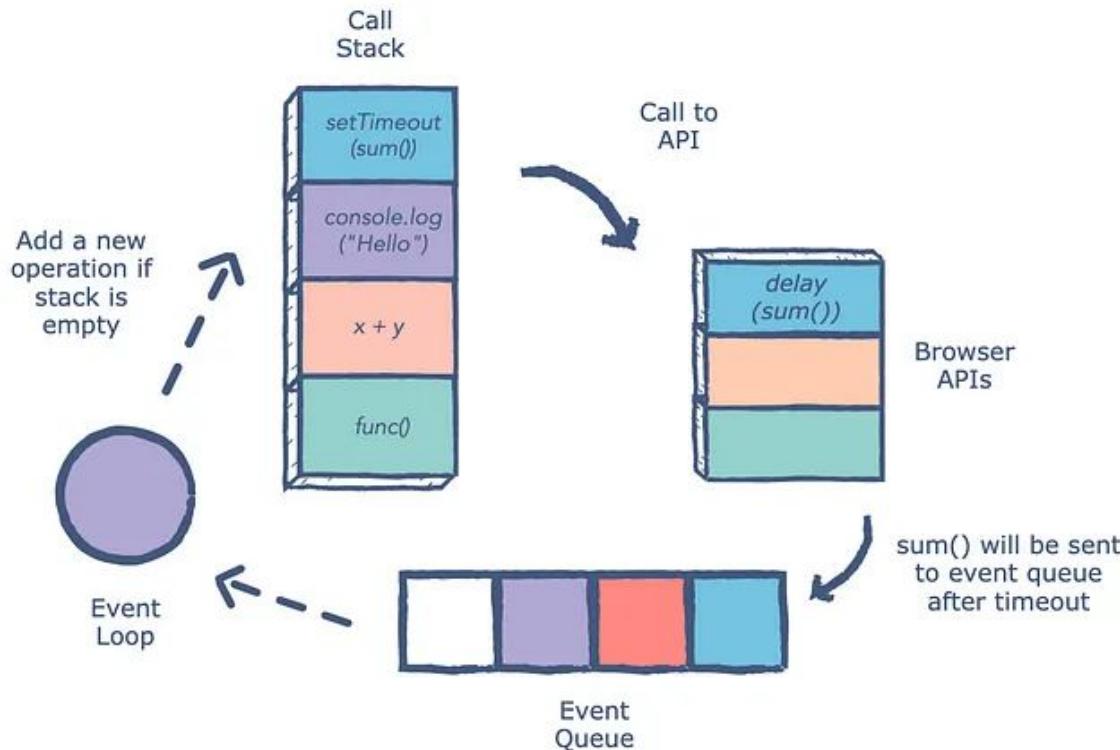
//object copy
let copy = {...obj}

//object destructuration
const {message} = obj;
console.log(message); //"hello world"
```

# async



# async



# Asynchrone

Uploaded using RayThis Extension

```
//retourne une promesse qui se résoudra après ms millisecondes
const sleep = (ms) => new Promise(resolve => setTimeout(resolve, ms));

const synchroneFunction = () => {
    console.log("start");
    sleep(3000);
    console.log("end");// executé immédiatement :-((
}
synchroneFunction();

const asynchroneFunction = async () => {
    console.log("start");
    await sleep(3000);
    console.log("end");// executé après 3 secondes :-D
}
asynchroneFunction();
```

Uploaded using RayThis Extension

```
//retourne une promesse qui se résoudra après ms millisecondes
const sleep = (ms) => new Promise(resolve => setTimeout(resolve, ms));

const synchroneFunction = () => {
    console.log("start");
    sleep(3000).then(() => {
        console.log("end");// executé après 3 secondes
    })
}
synchroneFunction();
```

axios.get(url)

# resolve & reject

```
function asyncFunction() {
    return new Promise((resolve, reject) => {
        // Simulation d'une opération asynchrone
        setTimeout(() => {
            const randomNumber = Math.random();

            if (randomNumber < 0.5) {
                // Résoudre la promesse avec le nombre aléatoire
                resolve(randomNumber);
            } else {
                // Rejeter la promesse avec une erreur
                reject(new Error('Une erreur s\'est produite !'));
            }
        }, 1000);
    });
}
```

```
...  
  
// Utilisation de try/catch pour capturer l'erreur
async function executeAsyncFunction() {
    try {
        const result = await asyncFunction();
        console.log('Résultat :', result);
    } catch (error) {
        console.error('Erreur :', error.message);
    }
}
```

```
...  
  
// Utilisation de .catch pour capturer l'erreur
asyncFunction()
    .then(result => {
        console.log('Résultat :', result);
    })
    .catch(error => {
        console.error('Erreur :', error.message);
    });
}
```

# Parallèlisme

Waterfall

```

async function asyncOperation1() { /* ... */ }
async function asyncOperation2() { /* ... */ }
async function asyncOperation3() { /* ... */ }

async function executeAsyncOperations() {
  try {
    const result1 = await asyncOperation1();
    const result2 = await asyncOperation2(result1);
    const result3 = await asyncOperation3(result2);
    console.log('Résultat final : ', result3);
  } catch (error) {
    console.error('Erreur : ', error);
  }
}

executeAsyncOperations()

```

Parallel

```

async function asyncOperation1() { /* ... */ }
async function asyncOperation2() { /* ... */ }
async function asyncOperation3() { /* ... */ }

async function executeParallelOperations() {
  try {
    const [result1, result2, result3] = await Promise.all([
      asyncOperation1(),
      asyncOperation2(),
      asyncOperation3()
    ]);
    console.log('Résultat 1 : ', result1);
    console.log('Résultat 2 : ', result2);
    console.log('Résultat 3 : ', result3);
  } catch (error) {
    console.error('Erreur : ', error);
  }
}

executeParallelOperations();

```

Parallel

```

const asyncQueue = [];

function addToQueue(asyncTask) {
  asyncQueue.push(asyncTask);
  if (asyncQueue.length === 1) {
    processQueue();
  }
}

function processQueue() {
  const asyncTask = asyncQueue[0];
  asyncTask()
    .then(result => {
      console.log('Résultat de la tâche : ', result);
      asyncQueue.shift();
      if (asyncQueue.length > 0) {
        processQueue();
      }
    })
    .catch(error => {
      console.error('Erreur de la tâche : ', error);
      asyncQueue.shift();
      if (asyncQueue.length > 0) {
        processQueue();
      }
    });
}

// Exemple d'utilisation
addToQueue(asyncOperation1);
addToQueue(asyncOperation2);

```

# Vite

```
PS C:\Users\guili\Desktop> npm create vite@latest
Need to install the following packages:
create-vite@5.1.0
Ok to proceed? (y) y
✓ Project name: ... demo-react
✓ Select a framework: » React
✓ Select a variant: » JavaScript

Scaffolding project in C:\Users\guili\Desktop\demo-react...

Done. Now run:

cd demo-react
npm install
npm run dev

PS C:\Users\guili\Desktop> cd .\demo-react\
PS C:\Users\guili\Desktop\demo-react> npm install
```

# JSX

● ● ●      Uploaded using RayThis Extension

```
import React from "react";

const App = () => {
  return (
    <div>
      <h1>React App</h1>
    </div>
  );
};

export default App;
```



Uploaded using RayThis Extension

```
import { createContext, useContext, useState } from "react";

const ctx = createContext();

const Provider = ({children}) => {
  const [state, setState] = useState(0);

  return (
    <ctx.Provider value={[state, setState]}>
      {children}
    </ctx.Provider>
  )
}

const useCtx = () => {
  const [state, setState] = useContext(ctx);

  return [state, setState];
}

export {Provider};
export default useCtx;
```

Uploaded using RayThis Extension

```
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.jsx'
import {Provider} from './contexts/ctx.jsx';

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <Provider>
      <App />
    </Provider>
  </React.StrictMode>,
)
```

Uploaded using RayThis Extension

```
import Button from "./Button";
import useCtx from "./contexts/ctx";

function App() {
  const [state, setState] = useCtx();

  return (
    <>
      <Button onClick={() => setState(state+1)}>click me: {state}</Button>
    </>
  )
}

export default App
```

# Découpage en composants

...

Uploaded using RayThis Extension

```
import React from "react";
import Component from "./Component";

const App = () => {
  return (
    <div>
      <Component />
    </div>
  );
};

export default App;
```

...

Uploaded using RayThis Extension

```
import React from "react";

const Component = () => {
  return (
    <div>
      <h1>My first component</h1>
    </div>
  );
};

export default Component;
```

# props

Uploaded using RayThis Extension

```
import React from "react";
import Component from "./Component";

const App = () => {
  return (
    <div>
      <Component title="hello world" />
    </div>
  );
};

export default App;
```

Uploaded using RayThis Extension

```
import React from "react";

const Component = (props) => {
  const { title } = props;

  return (
    <div>
      <h1>{title}</h1>
    </div>
  );
};

export default Component;
```

# props with TS

Uploaded using RayThis Extension

```
import { PropsWithChildren } from "react";

type ButtonProps = PropsWithChildren<{
    onClick: () => void;
}>;

/*
  type ButtonProps = {
    children: React.ReactNode;
    onClick: () => void;
  };
*/

export default function Button({ children, onClick }: ButtonProps) {
  return (
    <button onClick={onClick}>
      {children}
    </button>
  );
}
```

# Gestion d'évènements

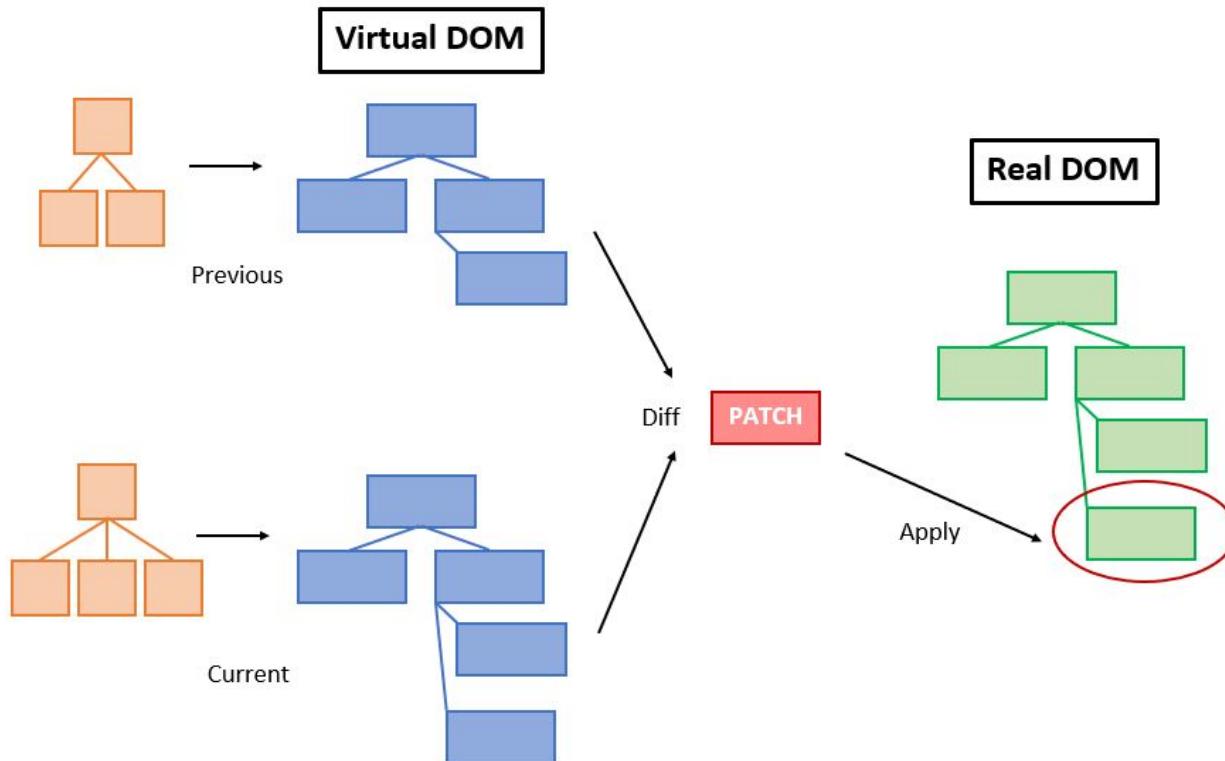


Uploaded using RayThis Extension

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

export default function EventPage() {
  return (
    <div>
      <button onClick={() => alert("hello world")}>Click me</button>
    </div>
  );
}
```

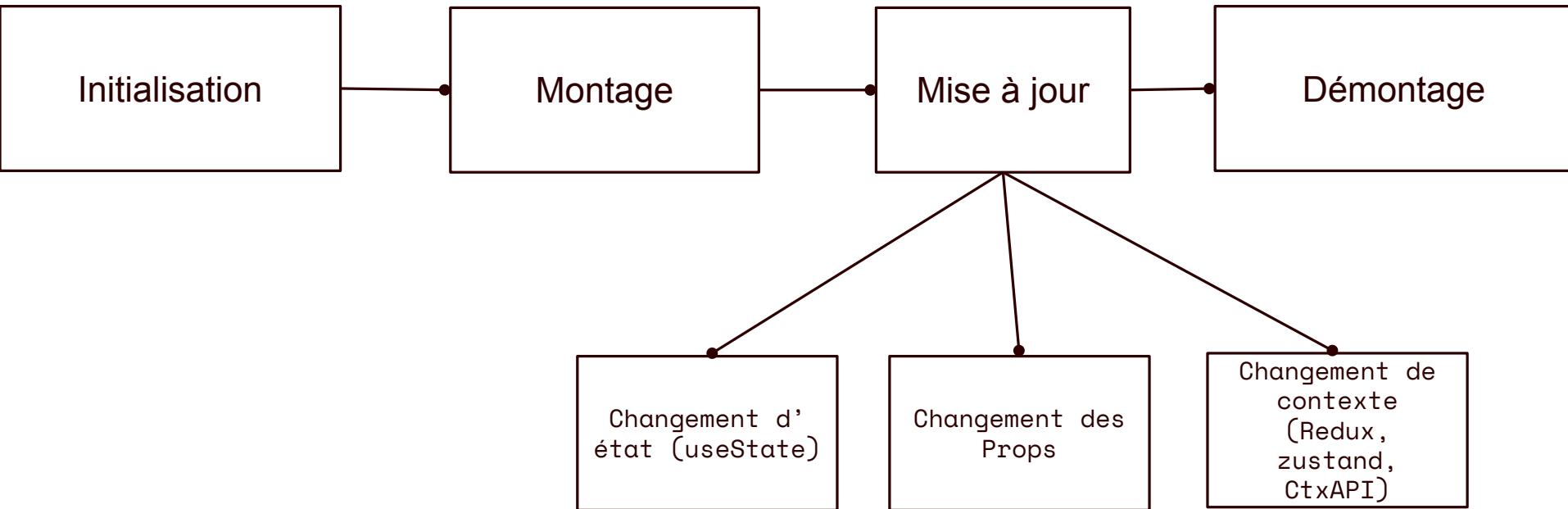
# DOM Virtuel



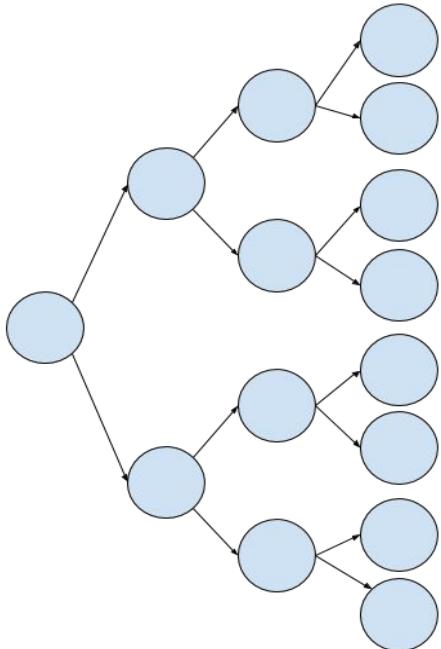
# Gestion d'état

```
• • •  
Uploaded using RayThis Extension  
  
import React, { useState } from 'react';  
  
export default function CounterPage() {  
  const [count, setCount] = useState<number>(0);  
  
  return (  
    <div>  
      <h1>Counter: {count}</h1>  
      <button onClick={() => setCount(count + 1)}>Increment</button>  
      <button onClick={() => setCount(count - 1)}>Decrement</button>  
      <button onClick={() => setCount(0)}>Reset</button>  
    </div>  
  );  
}
```

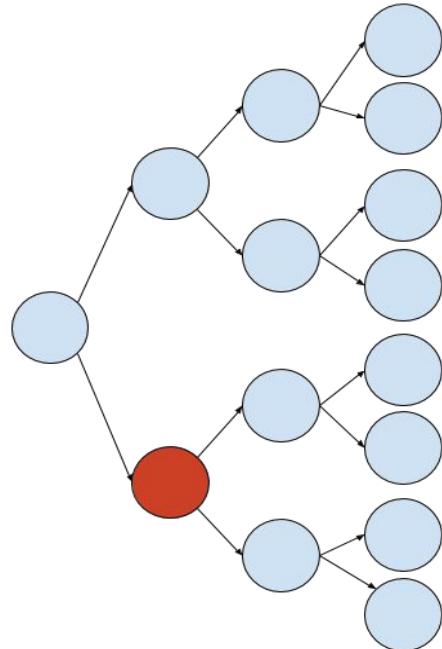
# Cycle de vie



# Cycle de vie



Changement d'état ou props



Propagation aux enfants  
(children)

# Conditional rendering



Uploaded using RayThis Extension

```
const Article = ({item}) => {
  if (!item) return <ActivityIndicator />
  return (
    <View>
      {/* ... */}
    </View>
  )
}
```



Uploaded using RayThis Extension

```
const Article = ({item}) => {
  return (
    <View>
      {item ?
        <View>
          {/* ... */}
        </View>
      :
        <ActivityIndicator />
      }
    </View>
  )
}
```

# Conditional rendering

● ● ● Uploaded using RayThis Extension

```
const Article = ({item}) => {
  return (
    <View>
      {item &&
        <View>
          {/* ... */}
        </View>
      }
    </View>
  );
};
```



# Array rendering

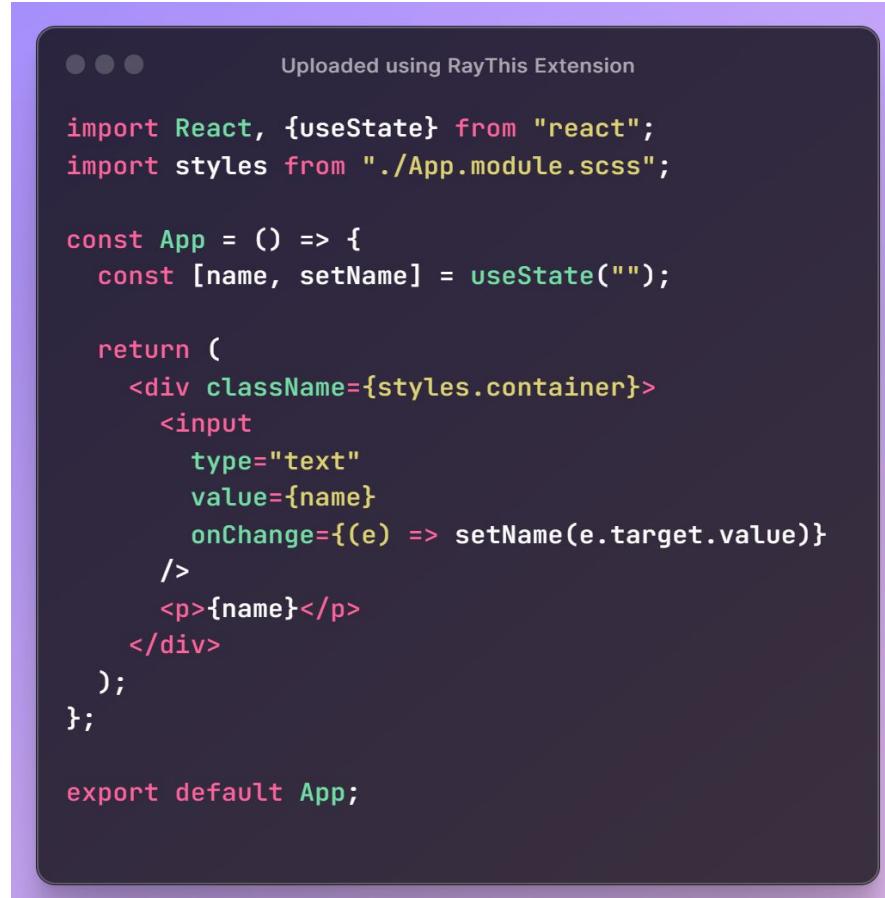


Uploaded using RayThis Extension

```
const Article = ({item}) => {
  console.log(item.comments);
  /*
  [
    {id: 0, content: "commentaire 1"},
    {id: 1, content: "commentaire 2"},
    {id: 2, content: "commentaire 3"},
  ]
  */

  return (
    <View>
      {item.comments.map((comment) => {
        return (
          <View key={comment.id}>
            <Text>{comment.content}</Text>
          </View>
        )
      ))}
    </View>
  )
}
```

# 2way data binding



Uploaded using RayThis Extension

```
import React, {useState} from "react";
import styles from "./App.module.scss";

const App = () => {
  const [name, setName] = useState("");

  return (
    <div className={styles.container}>
      <input
        type="text"
        value={name}
        onChange={(e) => setName(e.target.value)}
      />
      <p>{name}</p>
    </div>
  );
};

export default App;
```

# Effet de bords et abonnements

## Mount

```
useEffect(() => {  
  }, [])
```

Set "[]" empty dependency array to run an effect only on "mount"

## Update

```
useEffect(() => {  
})
```

Do not pass a dependency array at all to run an effect on each component update

```
useEffect(() => {  
  }, [dependencies])
```

Set dependencies array to run an effect only if any dependency change

## Unmount

```
useEffect(() => {  
  return () => {}  
}, [])
```

Set return function to run on component unmount lifecycle.

# Hooks customs

```
● ● ●          Uploaded using RayThis Extension

import React, {useEffect, useState} from "react";

const useCustomHook = () => {
  const [count, setCount] = useState(0);

  useEffect(() => {
    console.log("new count vlaue", count);
  }, [count]);

  const increment = () => setCount(count + 1);
  const decrement = () => setCount(count - 1);

  return {count, increment, decrement};
}

export default useCustomHook;
```

# hook customs

**pas de hook conditionnel**

**commence par “use”**

**préférer les retours cassants**

**Nettoyez vos effets**

**Découper en hook custom**

**Favoriser le code réutilisable**

# Formulaires

```
● ● ● Uploaded using RayThis Extension

const [title, setTitle] = useState("");

const handleSubmit = (e: React.FormEvent<HTMLFormElement>) => {
  e.preventDefault();
  // logic to add item to the list
}

return (
  <div>
    <form onSubmit={handleSubmit} className="flex flex-col gap-4">
      <h1 className="text-2xl font-bold">Form Simple</h1>
      <input
        type="text"
        value={title}
        onChange={(e) => setTitle(e.target.value)}
        placeholder="Enter title"
        className="border rounded p-2"
      />
      <button
        type="submit"
        className="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4 rounded"
      >
        Submit
      </button>
    </form>
  </div>
)
```



# Formulaires

Uploaded using RayThis Extension

```
import { z, ZodType } from "zod"; // Add new import

export const UserSchema: ZodType<FormData> = z
  .object({
    email: z.string().email(),
    githubUrl: z
      .string()
      .url()
      .includes("github.com", { message: "Invalid GitHub URL" }),
    yearsOfExperience: z
      .number({
        required_error: "required field",
        invalid_type_error: "Years of Experience is required",
      })
      .min(1)
      .max(10),
    password: z
      .string()
      .min(8, { message: "Password is too short" })
      .max(20, { message: "Password is too long" }),
    confirmPassword: z.string(),
  })
  .refine((data) => data.password === data.confirmPassword, {
    message: "Passwords do not match",
    path: ["confirmPassword"], // path of error
 });
```

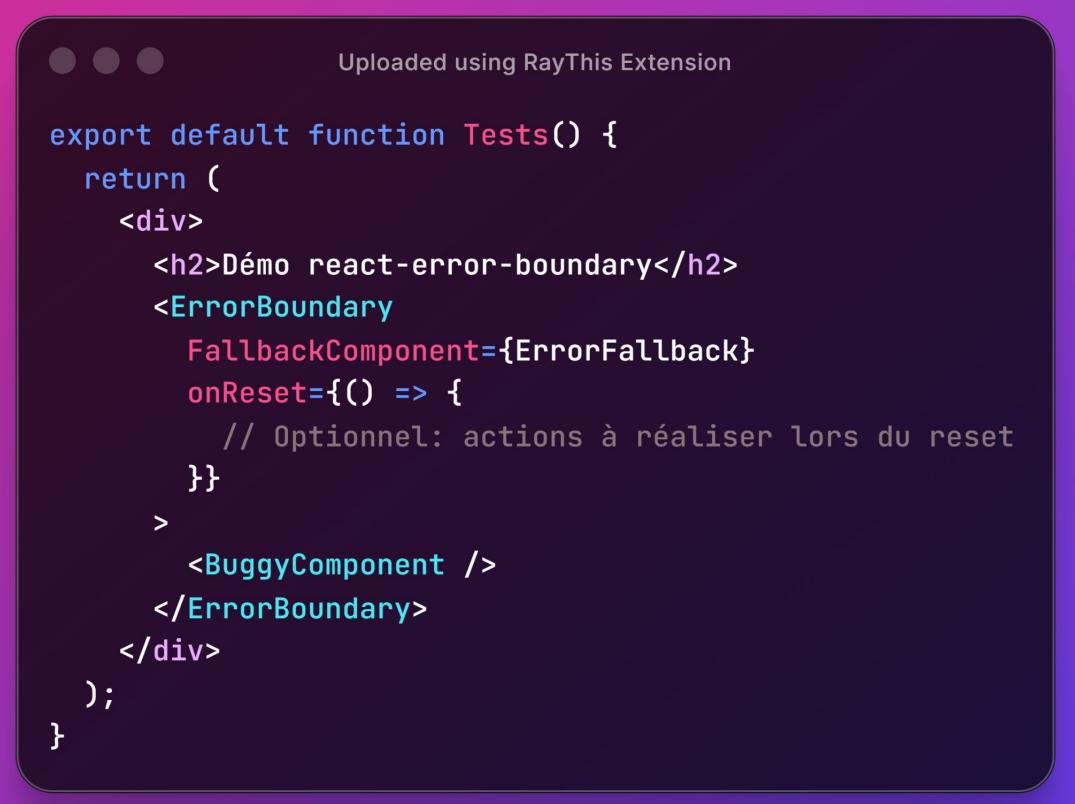
Uploaded using RayThis Extension

```
function Form() {
  const {
    register,
    handleSubmit,
    formState: { errors },
    setError,
  } = useForm<FormData>();

  const onSubmit = async (data: FormData) => {
    console.log("SUCCESS", data);
  }

  return (
    <form onSubmit={handleSubmit(onSubmit)}>
      <div className="grid col-auto">
        <h1 className="text-3xl font-bold mb-4">
          Zod & React-Hook-Form
        </h1>
        <FormField
          type="email"
          placeholder="Email"
          name="email"
          register={register}
          error={errors.email}
        />
      </div>
    </form>
  );
}
```

# ErrorBoundary

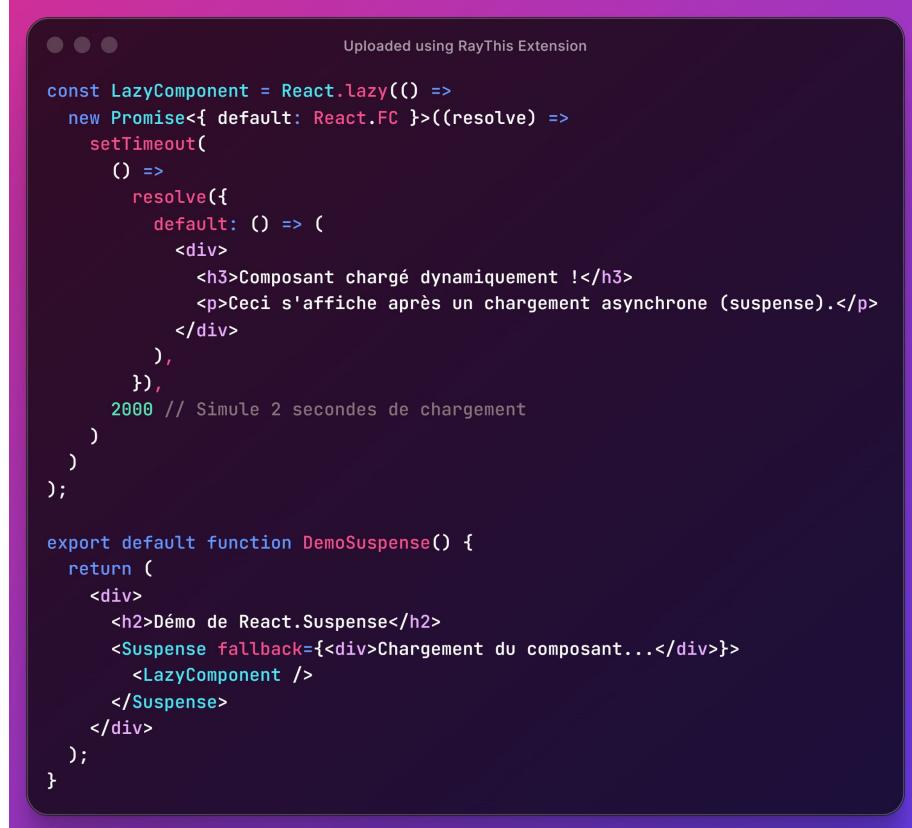


Uploaded using RayThis Extension

```
export default function Tests() {
  return (
    <div>
      <h2>Démo react-error-boundary</h2>
      <ErrorBoundary
        FallbackComponent={ErrorFallback}
        onReset={() => {
          // Optionnel: actions à réaliser lors du reset
        }}
      >
        <BuggyComponent />
      </ErrorBoundary>
    </div>
  );
}
```

npm i  
react-error-boundary

# Suspense

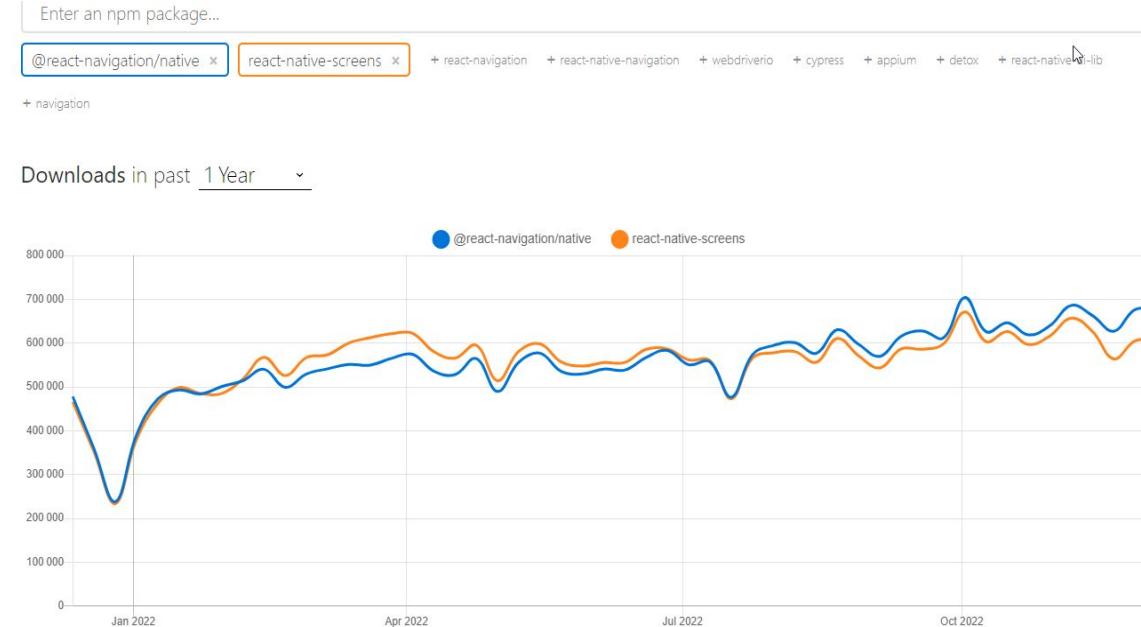


Uploaded using RayThis Extension

```
const LazyComponent = React.lazy(() =>
  new Promise<{ default: React.FC }>((resolve) =>
    setTimeout(
      () =>
        resolve({
          default: () => (
            <div>
              <h3>Composant chargé dynamiquement !</h3>
              <p>Ceci s'affiche après un chargement asynchrone (suspense).</p>
            </div>
          ),
        )),
    2000 // Simule 2 secondes de chargement
  )
);
;

export default function DemoSuspense() {
  return (
    <div>
      <h2>Démo de React.Suspense</h2>
      <Suspense fallback=<div>Chargement du composant...</div>>
        <LazyComponent />
      </Suspense>
    </div>
  );
}
```

# npm & dépendances externes



Stats



# Routing côté client

```
import Link from "next/link";

export default function HomePage() {
  return (
    <Link href="/about">
      | About page
    </Link>
  );
}
```



# React Hook Form + zod

```
...  
Uploaded using RayThis Extension  
  
import { z, ZodType } from "zod"; // Add new import  
  
export const UserSchema: ZodType<FormData> = z  
.object({  
  email: z.string().email(),  
  githubUrl: z  
.string()  
.url()  
.includes("github.com", { message: "Invalid GitHub URL" }),  
  yearsOfExperience: z  
.number({  
    required_error: "required field",  
    invalid_type_error: "Years of Experience is required",  
  })  
.min(1)  
.max(10),  
  password: z  
.string()  
.min(8, { message: "Password is too short" })  
.max(20, { message: "Password is too long" }),  
  confirmPassword: z.string(),  
})  
.refine((data) => data.password === data.confirmPassword, {  
  message: "Passwords do not match",  
  path: ["confirmPassword"], // path of error  
});
```

```
...  
Uploaded using RayThis Extension  
  
function Form() {  
  const {  
    register,  
    handleSubmit,  
    formState: { errors },  
    setError,  
  } = useForm<FormData>();  
  
  const onSubmit = async (data: FormData) => {  
    console.log("SUCCESS", data);  
  }  
  
  return (  
    <form onSubmit={handleSubmit(onSubmit)}>  
      <div className="grid col-auto">  
        <h1 className="text-3xl font-bold mb-4">  
          Zod & React-Hook-Form  
        </h1>  
        <FormField  
          type="email"  
          placeholder="Email"  
          name="email"  
          register={register}  
          error={errors.email}  
        />
```

# État complexe: useReducer

Uploaded using RayThis Extension

```
import { useReducer } from "react";

function cartReducer(state, action) {
  switch (action.type) {
    case "ADD_ITEM":
      return [...state, action.item];
    case "REMOVE_ITEM":
      return state.filter((item) => item.id !== action.id);
    case "CLEAR_CART":
      return [];
    default:
      return state;
  }
}
```

Uploaded using RayThis Extension

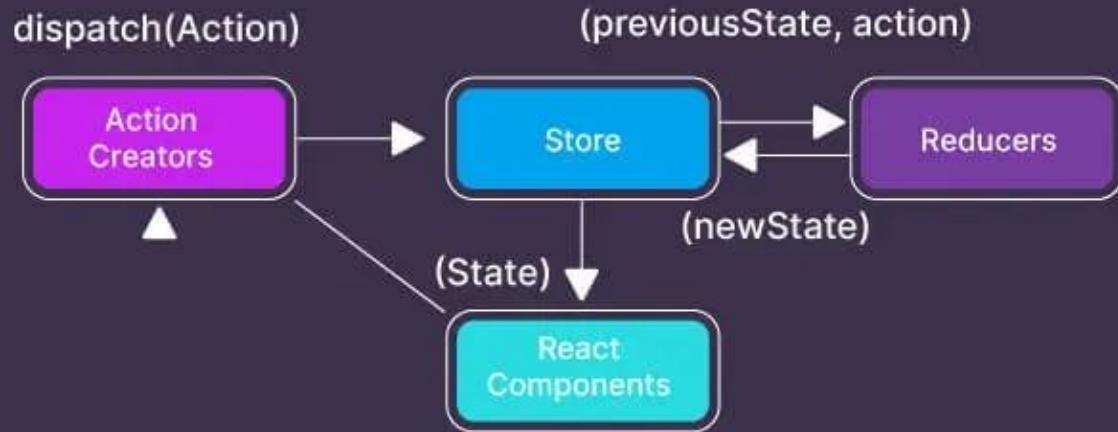
```
export default function Cart() {
  const [cart, dispatch] = useReducer(cartReducer, []);

  return (
    <>
      <button
        onClick={() =>
          dispatch({ type: "ADD_ITEM", item: { id: 1, name: "Parfum" } })
        }
      >
        Ajouter un parfum
      </button>

      <ul>
        {cart.map((item) => (
          <li key={item.id}>
            {item.name}
            <button
              onClick={() => dispatch({ type: "REMOVE_ITEM", id: item.id })}
            >
              Supprimer
            </button>
          </li>
        ))}
      </ul>

      <button onClick={() => dispatch({ type: "CLEAR_CART" })}>Vider</button>
    </>
  );
}
```

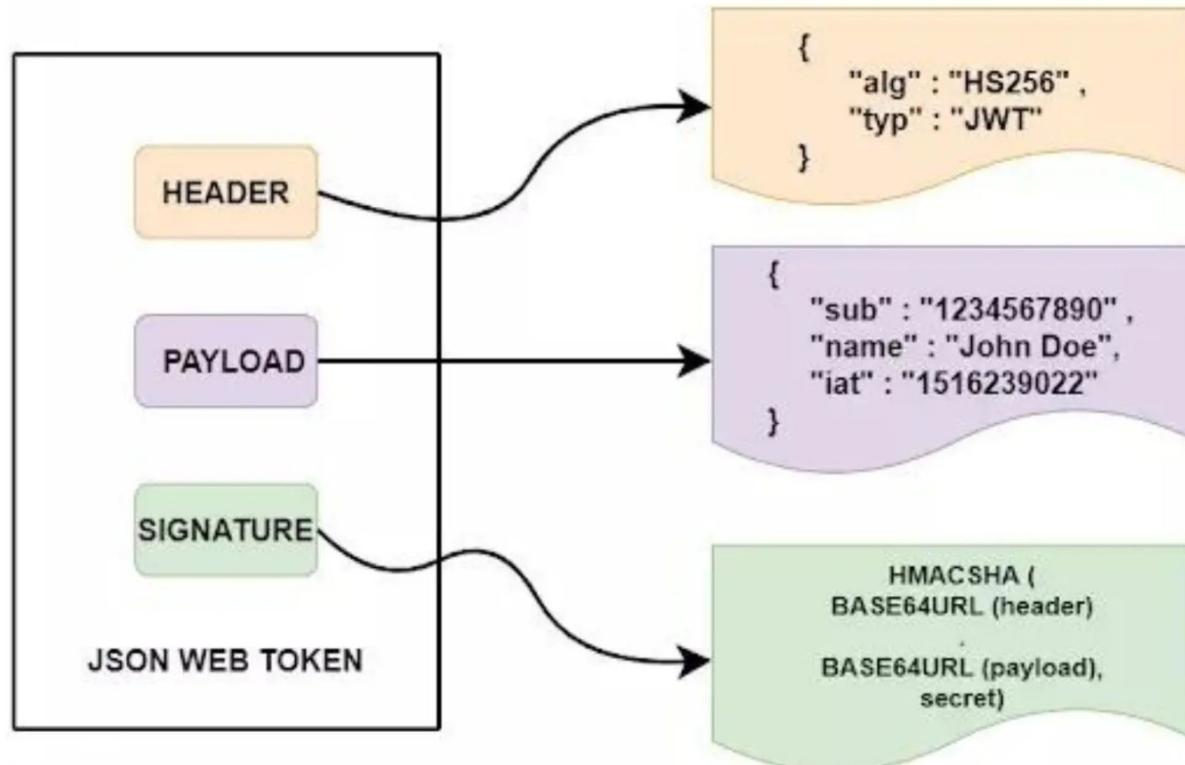
# Redux



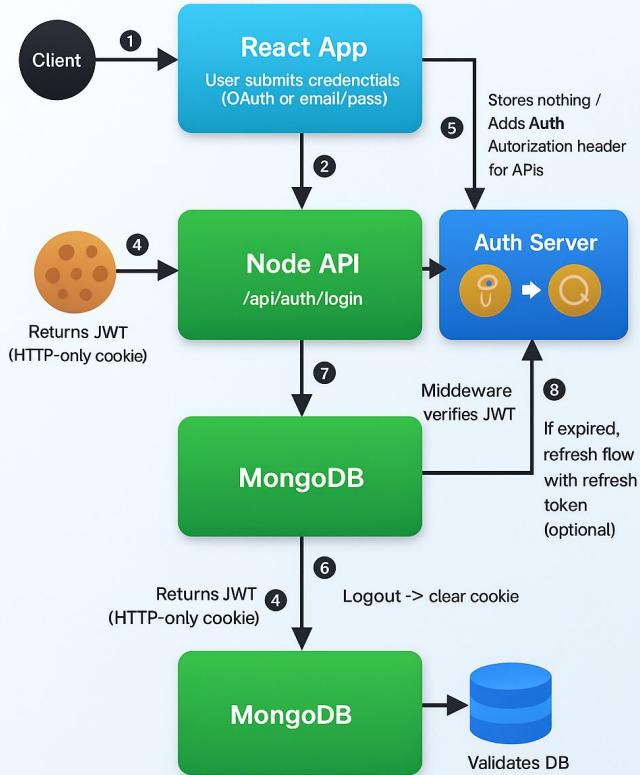
# JWT



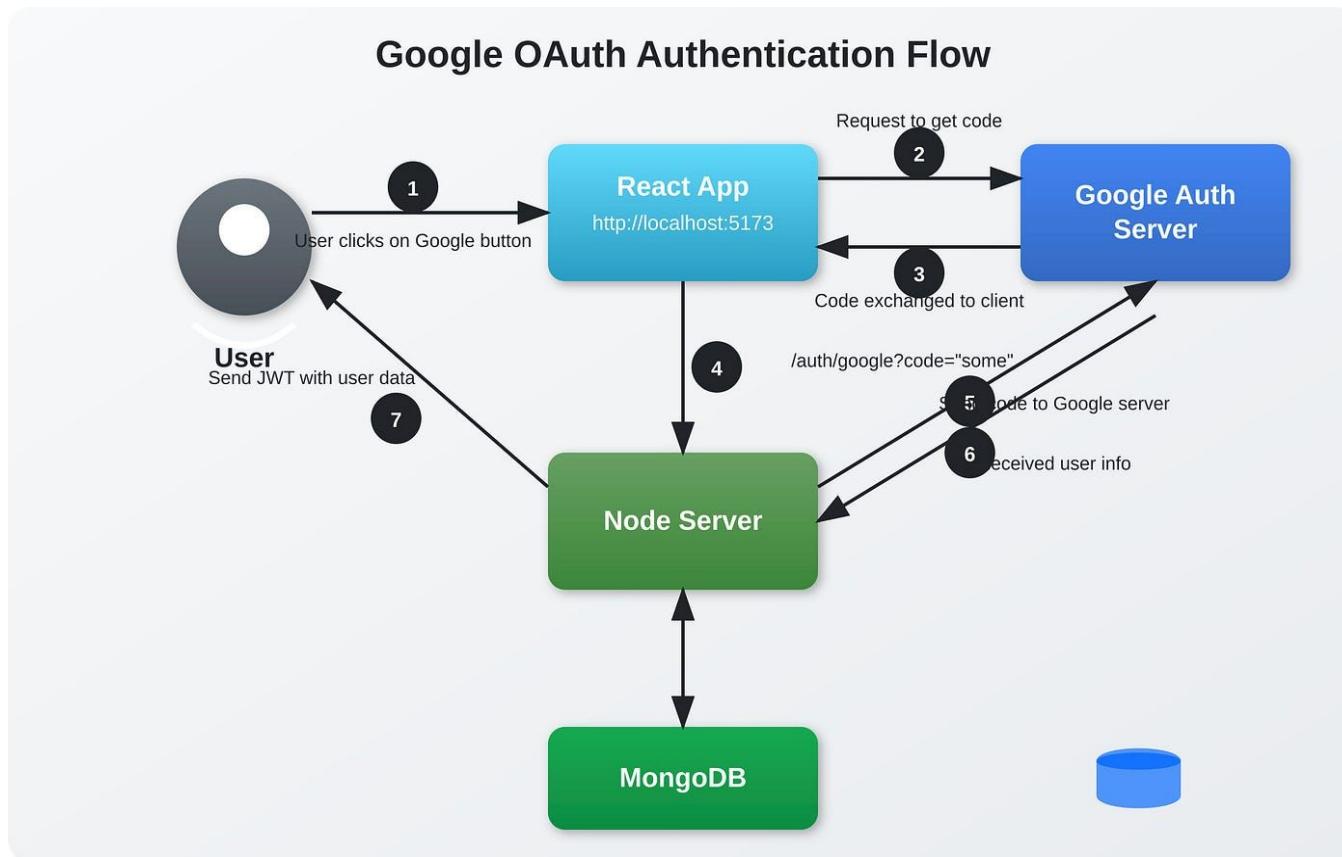
## Structure of JSON Web Token (JWT)



## JWT Authentication Flow with React



# OAUTH







# liste virtuelle (tanstack/virtua l)

```
import { useVirtualizer } from "@tanstack/react-virtual";

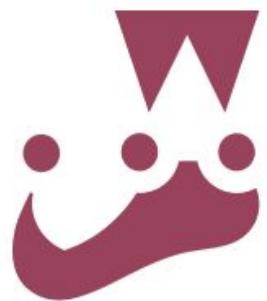
const items = Array.from({ length: 1000 }, (_, i) => `Ligne ${i}`);

function VirtualList() {
  const parentRef = useRef(null);

  const rowVirtualizer = useVirtualizer({
    count: items.length,
    getScrollElement: () => parentRef.current,
    estimateSize: () => 30,
  });

  return (
    <div
      ref={parentRef}
      style={{ height: 300, overflow: "auto", border: "1px solid #ccc", }}
    >
      <div style={{ height: rowVirtualizer.getTotalSize(), position: "relative" }}>
        {rowVirtualizer.getVirtualItems().map((virtualRow) => (
          <div
            key={virtualRow.key}
            style={{
              position: "absolute",
              top: 0,
              left: 0,
              width: "100%",
              height: virtualRow.size,
              transform: `translateY(${virtualRow.start}px)`,
              display: "flex",
              alignItems: "center",
              padding: "0 8px",
            }}
          >
            {items[virtualRow.index]}
          </div>
        )));
      </div>
    </div>
  );
}
```

**tests**



**Jest**



# eslint



# ESLint

```
[--> my-test-proj $ eslint .

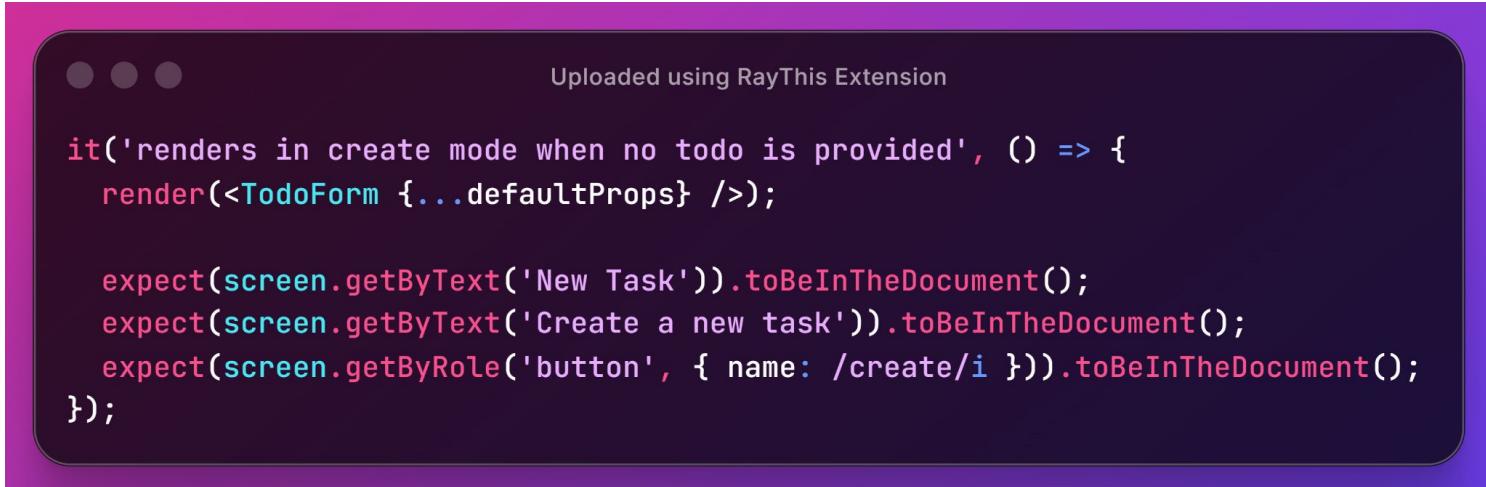
/Users/prasenjitpaul/Documents/rough/my-test-proj/index.js
  1:16  error    'y' is defined but never used
  2:5  error    Expected indentation of 2 spaces but found 4
  2:16  error    'z' is not defined
  5:11  warning  Missing semicolon

✖ 4 problems (3 errors, 1 warning)

-> my-test-proj $ |
```

no-unused-vars  
indent  
no-undef  
semi

# Tests présence



Uploaded using RayThis Extension

```
it('renders in create mode when no todo is provided', () => {
  render(<TodoForm {...defaultProps} />);

  expect(screen.getByText('New Task')).toBeInTheDocument();
  expect(screen.getByText('Create a new task')).toBeInTheDocument();
  expect(screen.getByRole('button', { name: /create/i })).toBeInTheDocument();
});
```

# mocks

```
● ● ●          Uploaded using RayThis Extension

//mock fonction
const fakeFn = jest.fn();
fakeFn('hello');
expect(fakeFn).toHaveBeenCalledWith('hello');

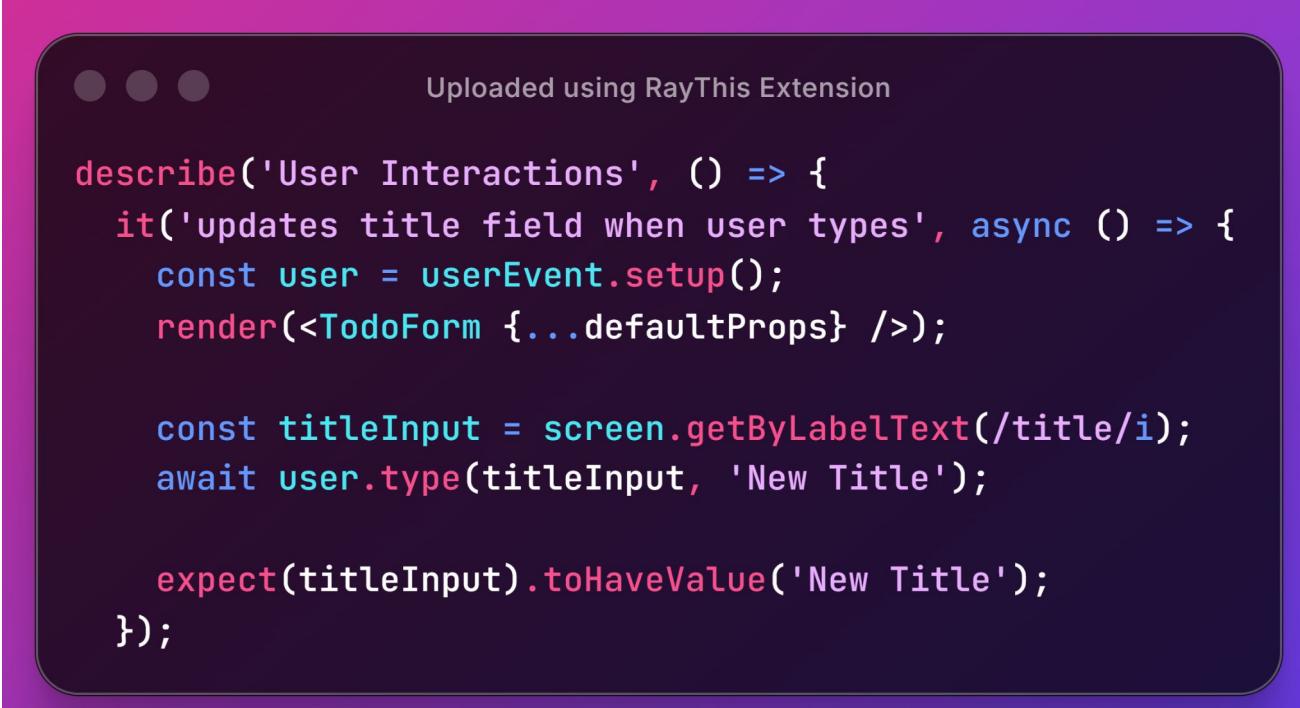
//en version async
const fetchUser = jest.fn().mockResolvedValue({ id: 1 });
const throwError = jest.fn().mockRejectedValue(new Error('fail'));

//mock module (remplacement librairie/fichier)
import axios from 'axios';
jest.mock('axios');
axios.get.mockResolvedValue({ data: { msg: 'ok' } });

//mock hook/composant
jest.mock('../useAuth', () => ({
  useAuth: () => ({ user: { id: 1 } })
}));
jest.mock('../Button', () => () => <div>Fake button</div>);

//
```

# Tests évènements



Uploaded using RayThis Extension

```
describe('User Interactions', () => {
  it('updates title field when user types', async () => {
    const user = userEvent.setup();
    render(<TodoForm {...defaultProps} />);

    const titleInput = screen.getByLabelText(/title/i);
    await user.type(titleInput, 'New Title');

    expect(titleInput).toHaveValue('New Title');
  });
});
```

# Tests asynchrone

```
● ● ● Uploaded using RayThis Extension

it('calls onSubmit with form data when form is submitted', async () => {
  const user = userEvent.setup();
  const onSubmit = jest.fn();
  render(<TodoForm {...defaultProps} onSubmit={onSubmit}>);

  const titleInput = screen.getByLabelText(/title/i);
  const descriptionInput = screen.getByLabelText(/description/i);
  const submitButton = screen.getByRole('button', { name: /create/i });

  await user.type(titleInput, 'New Title');
  await user.type(descriptionInput, 'New Description');
  await user.click(submitButton);

  await waitFor(() => {
    expect(onSubmit).toHaveBeenCalledTimes(1);
    expect(onSubmit).toHaveBeenCalledWith({
      title: 'New Title',
      description: 'New Description',
    });
  });
});
```

# Tests hooks

```
it('adds event listener to media query on mount', () => {
  Object.defineProperty(window, 'innerWidth', {
    writable: true,
    configurable: true,
    value: 1024,
  });

  renderHook(() => useIsMobile());

  expect(matchMediaMock.addEventListener).toHaveBeenCalledWith(
    'change',
    expect.any(Function)
  );
});
```