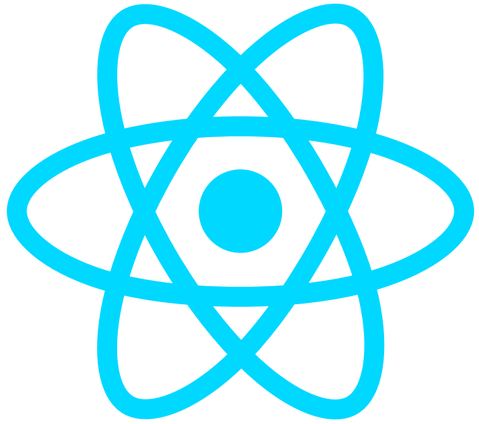
Soutenance Projet 7 OpenClassrooms

Kasa

Créez une application web de location immobilière avec React



Présenter par Rhani

PROJET 6

Kasa

01

Présentation du projet

02

React.js

03

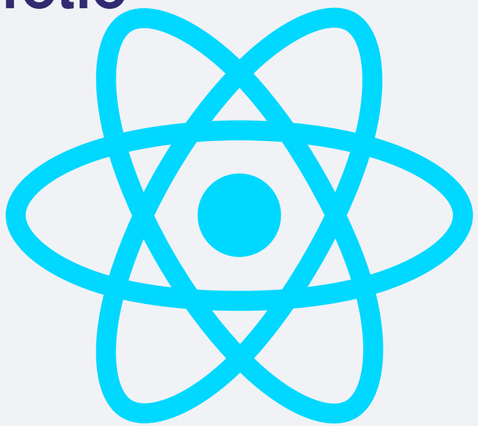
Présentation de l'application Kasa

Kasa nous recrute en tant que développeur front-end en freelance pour développer sa nouvelle plateforme web.

Kasa est dans le métier de la location d’appartements entre particuliers depuis près de 10 ans maintenant.

Avec plus de 500 annonces postées chaque jour, Kasa fait partie des leaders de la location d’appartements entre particuliers en France.

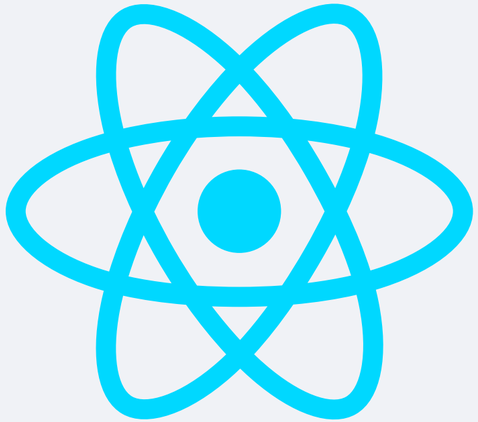
L'occasion parfaite pour nous d'ajouter une belle référence à notre portfolio de freelance !



Le site de Kasa a été codé il y a maintenant plus de 10 ans en ASP.NET avec un code legacy important.

Laura, la CTO, a donc lancé une refonte totale pour passer à une stack complète en JavaScript avec NodeJS côté back-end, et React côté front-end.

Kasa en a également profité pour commander de nouvelles maquettes auprès de son designer habituel, qui est en freelance. Un gros chantier pour cette année !



Présentation du projet

Un point est prévu avec la CTO plus tard dans la semaine, mais pour que vous puissiez vous familiariser avec les enjeux de votre mission, vous trouvez le récapitulatif suivant dans votre boîte mail :

Hello,

Bienvenue pour ton premier jour !

On est vraiment ravis de te compter parmi nous !

Tu vas pouvoir nous aider à donner vie à un chantier sur lequel on travaille depuis plusieurs mois.

Je préfère te mettre par écrit quelques éléments de contexte qui t'aideront à te projeter dans ta mission avant notre point de jeudi.

Ton objectif : Démarrer le projet React et développer l'ensemble de l'application, les composants React, les routes React Router, en suivant les maquettes Figma (responsives !) et toutes les infos que je te donne ci-dessous. Et ce avec un code de qualité !

Back-end / data : Le recrutement de la personne en charge du back-end n'est pas terminé et va prendre plus de temps que prévu. Du coup, il va falloir que tu fasses sans pour le moment. Je t'ai extrait les 20 dernières annonces de logements dans ce fichier JSON pour que tu puisses construire le Front qui correspond.

Contraintes techniques : Tu trouveras les coding guidelines de Kasa ici.

Voilà, j'espère que ce petit récapitulatif t'aura donné suffisamment de pistes pour aborder sereinement tes premiers jours chez nous.

Très bonne journée à toi !
Laura

Présentation du projet

Plus tard dans la journée, vous recevez un e-mail de Paul, le designer, qui vous donne plus d'informations sur le design et les fonctionnalités du site.

Objet : Fonctionnalité et design

De : Paul
À : Moi

Salut,

Bienvenue parmi nous ! Laura m'a dit de te briefer sur le design de la nouvelle version du site, alors voici les infos clés.

Design

Voici les maquettes sur Figma pour le design d'interface. J'ai l'habitude de travailler avec la logique de composants sur Figma ; Sandra m'a dit que ça te faciliterait le travail sur React. Tu trouveras toutes les ressources dont tu as besoin directement dans la maquette (logo, icônes pour les composants, etc.). Pour cela, il suffit de cliquer sur la ressource souhaitée et de faire "Exporter" au format voulu.

Contraintes fonctionnelles

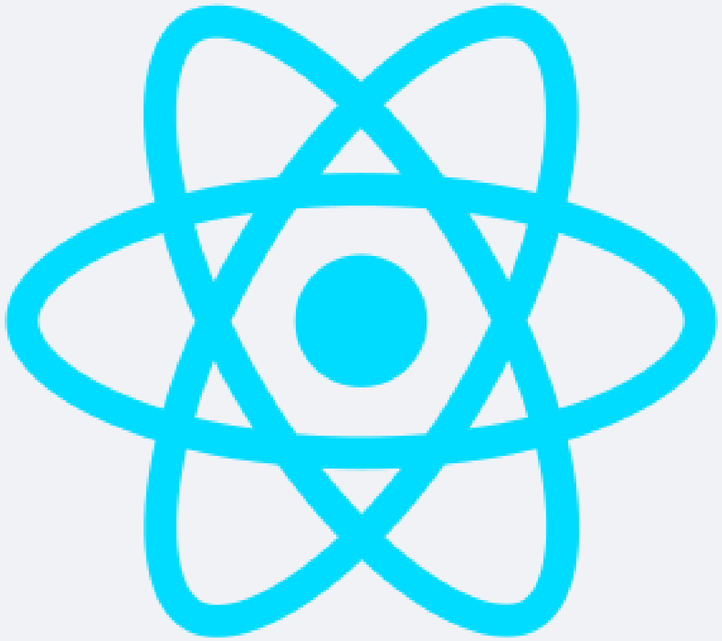
Quelques précisions sur les fonctionnalités du site :

- Pour le défilement des photos dans la galerie (composant Gallery) :
  - Si l'utilisateur se trouve à la première image et qu'il clique sur "Image précédente", la galerie affiche la dernière image.
  - Inversement, quand l'image affichée est la dernière de la galerie, si l'utilisateur clique sur "Image suivante", la galerie affiche la première image.
  - S'il n'y a qu'une seule image, les boutons "Suivant" et "Précédent" n'apparaissent pas.
- La galerie doit toujours rester de la même hauteur, celle indiquée sur la maquette Figma. Les images seront donc coupées et centrées dans le cadre de l'image.
- Collapse : Par défaut, les Collapses sont fermés à l'initialisation de la page.
- Si le Collapse est ouvert, le clic de l'utilisateur permet de le fermer.
  - Inversement, si le Collapse est fermé, un clic permet de l'ouvrir.

Bon courage pour le développement, j'ai hâte de voir ce que ça va donner !
Paul

REACT

React est un framework orienté composant.
On n'a qu'une seule page HTML, au moment on l'on va concevoir notre application.
C’est cette page qui va contenir l’application REACT. Cette application est lancée à partir du fichier index.js. Cette page HTML peut bien évidemment contenir des liens vers des fichiers CSS pour ajouter du style.



La première chose qui va être exécutée au moment où ce script va être chargé va être ce qu'on retrouve dans le fichier index.js c'est le point d'entrée de notre application.
Ici le premier élément qui va nous intéresser c'est la ligne const root = ReactDOM.creat.Root (document.GetElementById('root'));

```
import React from "react";
import ReactDOM from "react-dom/client";
import App from "./App";
import "./styles/index.scss";

const root = ReactDOM.createRoot(document.getElementById("root"))
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Cette constante root c'est la racine de notre application.
Cet élément fait référence à un élément de la page HTML (élément avec l'id root) et servira de point d'ancrage au composant App de l'application. App est le premier composant de l'application à être généré lorsque celle-ci est lancée.
Le composant va se convertir en Élément React (React.element) cet élément va d'écrire notre partie HTML

```
import React from "react";
import ReactDOM from "react-dom/client";
import App from "./App";
import "./styles/index.scss";

const root = ReactDOM.createRoot(document.getElementById("root"))
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

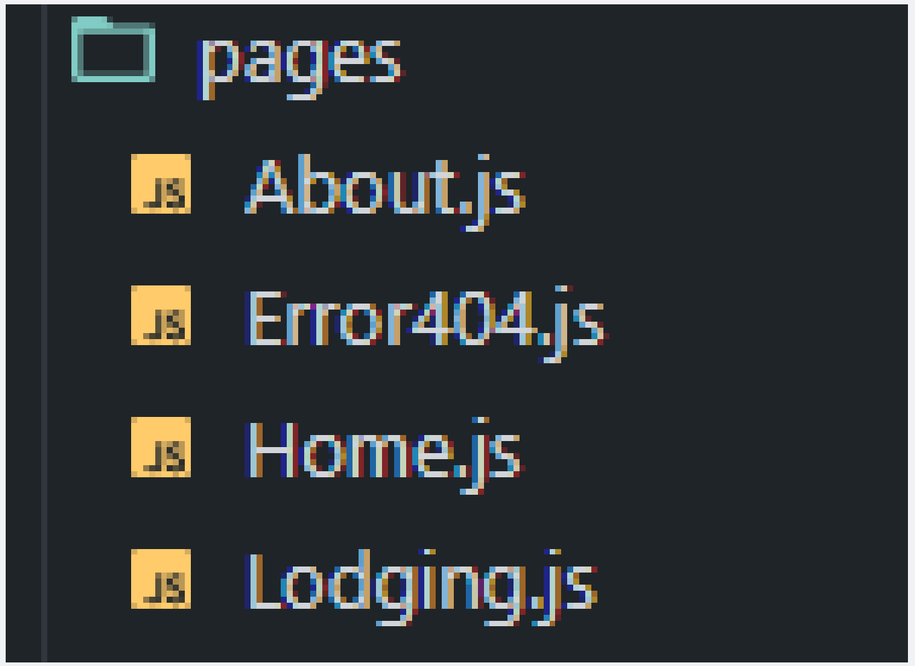
À partir du moment où cela est fait, la méthode root.render va instancier notre composant App, il va le convertir en un Élément React, quand il va générer tous les composants qui seront situés dans la propriété children, ces composants vont ensuite former notre DOM virtuel.

Les composants REACT sont écrits avec du langage JSX on peut voir ça comme du JavaScript + HTML ça va nous permettre pour la création de nos composants d'utiliser un seul langage.
JSX sera converti par Webpack en JavaScript, HTML et CSS.

```
import React from "react";
import ReactDOM from "react-dom/client";
import App from "./App";
import "./styles/index.scss";

const root = ReactDOM.createRoot(document.getElementById("root"))
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

L’application est composée de différents composants formant les pages (About, Error404, Home et Lodging.
Chacune de ces pages est liée à une URL par le routeur de React qui est appelé dans le composant App.
Chaque composant de page est lui-même constitué de HTML ainsi que de composants.



Par exemple la page Home est constituée des composants Header, Banner, Card et Footer ainsi que d'une partie en HTML. Cette partie en HTML utilise la fonction Javascript map pour générer autant de blocs HTML qu'il y a de logements à louer. On génèrera donc autant de composants Card qu'il y a de logement, on passe deux propriétés à chaque composant Card : une clé (nécessaire pour la fonction map) et les informations du logement correspondant.

```jsx
import React, { useContext } from "react";
import Banner from "../components/Banner";
import Card from "../components/Card";
import Header from "../components/Header";
import { NavLink } from "react-router-dom";
import { AppContext } from "../AppContext";
import Footer from "../components/Footer";

const Home = () => {
  // Context import (datas)
  const data = useContext(AppContext);

  return (
    <div>
      <Header />
      <main>
        <Banner />
        <section>
          <div className="card_container">
            {/* Creation of a card + NavLink for each data index*/}
            {data.map((lodging, index) => (
              <NavLink
                key={index}
                to={`/lodging/${lodging.id}`}
              >
                <Card key={index} lodging={lodging} />
              </NavLink>
            ))}
          </div>
        </section>
      </main>
      <Footer />
    </div>
  );
};

export default Home;
```

Les données du fichier JSON contenant les informations des logements sont chargées dans le composant App et chargées en contexte via l'instruction <AppContext.Provider value={data}>

Cette instruction permet de mettre les données des logements en contexte, ce qui signifie que ces données sont en mémoire quelque soit le composant que l'on utilise.

Dès lors que l'on a besoin de ces données il suffira d'utiliser l'instruction const data = useContext(AppContext);

```
const data = useContext(AppContext);
```

Le routeur de React permet de définir différentes URL qui seront utilisées par l'application.
Pour chaque URL on va indiquer quel composant on génère à l'emplacement définit dans le fichier HTML (emplacement avec l'id root).
La dernière URL (*) permet d'indiquer que pour toute URL différente de celles du dessus on génèrera le composant Error404 indiquant donc à l'utilisateur qu'il n'existe pas de page à cette URL.
404 est le code retour HTML signifie Not Found (ressource non trouvée).

```
    <Routes>
      <Route path="/" element={<Home />} />
      <Route path="/lodging/:id" element={<Lodging />} />
      <Route path="/about" element={<About />} />
      {/* path = "*" is for all other url adresses that non exist */}
      <Route path="*" element={<Error404 />} />
    </Routes>
  );
};
```

Front-end

Nous extrait les 20 dernières annonces de logements dans un fichier JSON pour que nous puissions construire le Front qui correspond.


```
"id": "c67ab8a7",
"title": "Appartement cosy",
"cover": "https://s3-eu-west-1.amazonaws.com/course.oc-static.com/projects/front-end-kasa-project/accommodation-20-1.jpg",
"pictures": [
        "https://s3-eu-west-1.amazonaws.com/course.oc-static.com/projects/front-end-kasa-project/accommodation-20-1.jpg",
        "https://s3-eu-west-1.amazonaws.com/course.oc-static.com/projects/front-end-kasa-project/accommodation-20-2.jpg",
        "https://s3-eu-west-1.amazonaws.com/course.oc-static.com/projects/front-end-kasa-project/accommodation-20-3.jpg",
        "https://s3-eu-west-1.amazonaws.com/course.oc-static.com/projects/front-end-kasa-project/accommodation-20-4.jpg",
        "https://s3-eu-west-1.amazonaws.com/course.oc-static.com/projects/front-end-kasa-project/accommodation-20-5.jpg"
],
"description": "Votre maison loin de chez vous. Que vous veniez de l'autre bout du monde, ou juste de quelques stations de RER, vous vous sentirez chez vous dans notre appartement.",
"host": {
        "name": "Nathalie Jean",
        "picture": "https://s3-eu-west-1.amazonaws.com/course.oc-static.com/projects/front-end-kasa-project/profile-picture-12.jpg"
},
"rating": "5",
"location": "Ile de France - Paris 17e",
"equipments": [
        "Équipements de base",
        "Micro-Ondes",
        "Douche italienne",
        "Frigo",
        "WIFI"
],
"tags": [
        "Batignolle",
        "Montmartre"
]
```


Screen du fichier Json

Présentation de l'application web

Fin de la Soutenance Projet OpenClassrooms

Créez une application web de location immobilière avec React

Présenter par Rhani