**Soutenance du Site Kasa**

**Introduction :**

* Présentation du site Kasa, une plateforme de location d'appartements.
* Objectif : Examiner l'architecture, les fonctionnalités et les choix techniques du site.

**Fichier HTML (index.html):**

* **Doctype et Balise Racine :**
  + **<!DOCTYPE html>** : Indique la version de HTML utilisée.
  + **<html lang="fr">** : Définit la langue de la page comme le français.
* **En-tête (<head>) :**
  + **charset** : Définit l'encodage des caractères de la page.
  + **viewport** : Spécifie comment le navigateur doit contrôler le dimensionnement et l'échelle de la page sur différents appareils.
  + **description** : Fournit une description du site pour les moteurs de recherche.
  + **favicon** : Définit l'icône du site.
  + **title** : Définit le titre de la page.
* **Corps (<body>) :**
  + **<noscript>** : Affiche un message si JavaScript est désactivé.
  + **<div id="root"></div>** : Point d'entrée où l'application React sera rendue.
  + **<script src="./index.js"></script>** : Lie le fichier JavaScript principal de l'application.

**Fichier JavaScript (index.js):**

* **Création du Point d'Entrée :**
  + **ReactDOM.createRoot(document.getElementById("root"))** : Crée un point d'entrée racine pour l'application React.
* **Rendu Initial :**
  + **root.render(<React.StrictMode><App /></React.StrictMode>)** : Rend le composant **App** dans le point d'entrée racine.

**Fichier JavaScript (App.js):**

* **Importations :**
  + **React**, **BrowserRouter**, **Routes**, **Route** : Importe les composants React nécessaires pour gérer la navigation.
* **Définition du Composant App :**
  + Structure de base de l'application, incluant la navigation entre différentes vues.

### Autres Fichiers JavaScript (par exemple Header.js, Navigation.js, etc.):

* **import React from "react";** : Importe la bibliothèque React.
* Définit différents composants, tels que **Header**, **Navigation**, etc., qui sont des parties réutilisables de l'interface utilisateur.

Chaque fichier JavaScript définit des composants React qui encapsulent différentes parties de l'application, favorisant ainsi la modularité et la réutilisabilité du code.

**Architecture :**

* **Header, Navigation, Footer :** Ces composants fournissent une navigation cohérente et une expérience utilisateur unifiée à travers toutes les pages de l'application.
* **Carroussel, Collapse, Card :** Ces composants sont utilisés pour afficher des informations détaillées sur les logements, notamment des images supplémentaires, des évaluations et des descriptions.
* **Routes :** Nous utilisons React Router pour définir les routes et gérer la navigation entre les différentes vues de l'application (front-end).

**Utilisation de React :** Le choix de React comme bibliothèque JavaScript pour le développement front-end offre de nombreux avantages. React permet une gestion efficace de l'état de l'application, ce qui est crucial pour une expérience utilisateur fluide. De plus, la composition des composants et la syntaxe déclarative de React rendent le code plus lisible et plus facile à comprendre, ce qui accélère le processus de développement et de maintenance.

**React Router pour la Navigation :** La navigation entre les différentes vues du site est gérée de manière transparente grâce à React Router, Le composant **NavLink** est utilisé pour créer des liens de navigation dans l'application. Il fonctionne de manière similaire à un lien (**<a>**), mais il ajoute automatiquement une classe active lorsque l'URL correspond à la route spécifiée. Cette bibliothèque permet la gestion des routes dans une application React, ce qui signifie que les vues peuvent être chargées de manière dynamique en fonction de l'URL de la page. Cela garantit une expérience utilisateur fluide et sans heurts lors de la navigation sur le site.

**Fonctionnalités Implémentées :**

1. **Page d'Accueil (Home) :**

**map :** La méthode **map** est utilisée pour parcourir la liste des appartements disponibles et générer une carte pour chaque appartement. Cela permet d'afficher dynamiquement une galerie d'appartements sur la page d'accueil.

* + Affichage des appartements disponibles avec leurs titres et images.
  + Utilisation d'une boucle pour générer dynamiquement les cartes d'appartement.
  + Intégration d'une bannière attractive avec un titre évocateur.

1. **Page de Détails d'un Logement (Logement) :**

**useParams (React Router) :** La fonction **useParams** est utilisée pour extraire les paramètres de l'URL. Dans le composant **Logement**, elle est utilisée pour obtenir l'identifiant unique (**id**) du logement à afficher.

**find (Array) :** La méthode **find** est utilisée pour rechercher un élément spécifique dans un tableau en fonction d'un critère donné. Dans le composant **Logement**, elle est utilisée pour trouver les détails du logement correspondant à l'identifiant extrait de l'URL.

* + Affichage des détails d'un logement spécifique : images, informations sur l'hôte, évaluations, etc.
  + Utilisation d'un carrousel pour naviguer à travers les images.
  + Description et équipements du logement affichés dans des sections déroulantes.

1. **Page A Propos (About) :**
   * Le composant **AboutBanner** est utilisé pour afficher une bannière ou une image représentant l'application. Cette bannière peut être une image fixe ou un élément visuel attrayant pour attirer l'attention des utilisateurs.
   * Le composant **Collapse** est utilisé pour afficher le contenu de la page "À Propos". Il est utilisé pour afficher des sections déroulantes avec des informations supplémentaires de l’application.
2. **Gestion des Erreurs (NotFound) :**

Le composant **NotFound** est une fonction React qui retourne du JSX décrivant l'apparence de la page d'erreur.

Lorsque l'utilisateur accède à une URL invalide, le composant **NotFound** est rendu à la place de la page demandée. Il affiche un message d'erreur (page404) indiquant que la page demandée n'existe pas et guide les utilisateurs vers la page d'accueil de l'application à l'aide du composant **Link** de React Router. Cela permet à l'utilisateur de retourner à une page fonctionnelle de l'application..

**Choix Technologiques :**

* **React :** Nous avons choisi d'utiliser la bibliothèque React pour développer l'interface utilisateur de notre application en raison de sa modularité, de sa performance et de sa popularité dans le développement web moderne.
* **React Router :** Pour gérer la navigation entre les différentes vues du site, nous avons utilisé React Router. Cette bibliothèque nous permet de définir des routes pour chaque page de notre application, facilitant ainsi la gestion de la navigation côté client.
* **FontAwesome et Polices Google :** Nous avons intégré FontAwesome pour les icônes et des polices Google pour le design afin d'offrir une expérience utilisateur visuellement attrayante et cohérente sur l'ensemble du site.

**Conclusion :**

* Le site Kasa offre une expérience utilisateur fluide et intuitive grâce à son architecture modulaire et ses fonctionnalités bien pensées.
* Les choix technologiques et l'organisation du code contribuent à la maintenabilité et à l'évolutivité du site.

**Carroussel :**

La fonction **useState** est utilisée pour gérer l'état local du composant. Dans le cas du carrousel, elle est utilisée pour suivre l'index de l'image actuellement affichée.

**goToNextSlide :** Cette fonction est utilisée pour passer à l'image suivante dans le carrousel. Elle est déclenchée lorsqu'un utilisateur clique sur la flèche de droite ou utilise une autre méthode de navigation.

**goToPrevSlide :** Cette fonction est utilisée pour revenir à l'image précédente dans le carrousel. Elle est déclenchée lorsqu'un utilisateur clique sur la flèche de gauche ou utilise une autre méthode de navigation.

**Collapse :**

La fonction **useState** est utilisée pour gérer l'état local du composant. Elle est utilisée pour suivre l'état d'ouverture ou de fermeture de la section déroulante.

**toggleCollapse :** Cette fonction est utilisée pour basculer l'état d'ouverture ou de fermeture de la section déroulante lorsqu'un utilisateur clique sur l'en-tête de la section.

**Affichage Conditionnel du Contenu :** Le contenu de la section déroulante est affiché ou masqué en fonction de l'état de la variable **isOpen**. Si la variable est **true**, le contenu est affiché, sinon il est masqué.

**ficheInfos :**

La fonction **ratedStars** est utilisée pour calculer et afficher le classement d'un logement sous forme d'étoiles. Le nombre d'étoiles pleines est déterminé en fonction du classement donné, et les étoiles restantes sont affichées comme vides.

La boucle **map** est utilisée dans le composant **FicheInfos** pour rendre dynamiquement les tags associés à un logement. Elle crée un élément **<div>** distinct pour chaque tag dans le tableau **tags**, affichant ainsi tous les tags dans la fiche d'informations du logement.

**Navigation**:

le composant **Navigation** utilise principalement des opérations de rendu pour afficher les liens de navigation. Il crée des liens réactifs à l'aide du composant **NavLink** de React Router et utilise l'attribut **isActive** pour déterminer si un lien est actif, lui permettant ainsi d'appliquer une classe CSS spéciale pour le styliser en conséquence.

**Logement** :

Le composant utilise **useParams** de React Router pour extraire l'identifiant unique du logement à partir de l'URL.

La méthode **find** est utilisée pour rechercher le logement correspondant à l'identifiant extrait de l'URL dans les données disponibles.

**Explication des Fonctions Clés**

1. **React Components :** Les composants React sont des fonctions qui retournent du JSX décrivant la structure et l'apparence d'une partie de l'interface utilisateur. Par exemple, le composant **Header** définit l'en-tête du site, le composant **Card** représente une carte d'appartement sur la page d'accueil, etc.
2. **React Router :** React Router est une bibliothèque qui permet de gérer la navigation entre les différentes vues de l'application. Les routes sont définies à l'aide des composants **Routes** et **Route**. Chaque route est associée à un composant spécifique à afficher lorsque l'URL correspond à cette route. Par exemple, la route "/" est associée à la page d'accueil, la route "/logement/:id" est associée à la page de détails du logement, etc.
3. **Importations (import) :** Les importations permettent d'inclure des bibliothèques et des composants dans les fichiers JavaScript. Par exemple, dans le fichier **App.js**, nous importons React et les composants de gestion de la navigation à l'aide de la syntaxe **import**.
4. **Rendu Initial :** Le point d'entrée de l'application est créé à l'aide de **ReactDOM.createRoot**, où le composant racine est rendu à l'aide de **root.render**. Cela se produit dans le fichier **index.js**, où nous importons également notre composant **App** pour le rendre dans le DOM.
5. **Handlers d'événements :** Ces fonctions sont utilisées pour gérer les interactions de l'utilisateur avec l'interface utilisateur, telles que les clics, les survols, etc. Par exemple, un gestionnaire d'événements peut être attaché à un bouton pour déclencher une action lorsque l'utilisateur clique dessus.
6. **Fonctions auxiliaires :** Ces fonctions sont utilisées pour effectuer des tâches spécifiques à différents endroits de l'application. Par exemple, une fonction auxiliaire peut être utilisée pour formater des données, effectuer des calculs ou manipuler des chaînes de caractères. Ces fonctions contribuent à rendre le code plus modulaire et plus facile à comprendre.

Ces fonctions clés constituent le cœur du fonctionnement de l'application Kasa, permettant de créer une expérience utilisateur fluide et intuitive