

// Application e-commerce

// Pizzeria en ligne

MongoDB Express React Node

Projet MERN stack

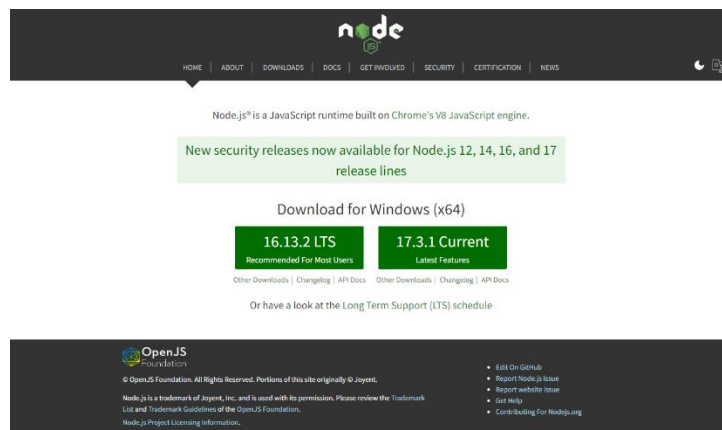
Première Partie

Présentation du projet

Développement d'une application e-commerce en utilisant MongoDB, ExpressJS, React, NodeJS. Cette version est allégée et est surtout faite pour prendre en main React et comprendre le fonctionnement d'une application MERN Stack.

Environnement de Développement

Installer Node.js



<https://nodejs.org/en/>

Installer node.js privilégier la version LTS (Long Term Support). Laisser l'installation par défaut.

Pour vérifier que Node.js est bien installé, ouvrez votre Invite de commande (Touche Windows + R puis rentrez "cmd") et tapez la commande :

Projet MERN Stack : MongoDB ExpressJS ReactJS, NodeJS
Réalisation Stéphane Pontonnier Formateur AFPA

Afpa

```
C:\Users\1701805>node -v
v14.16.0
```

Idem pour npm :

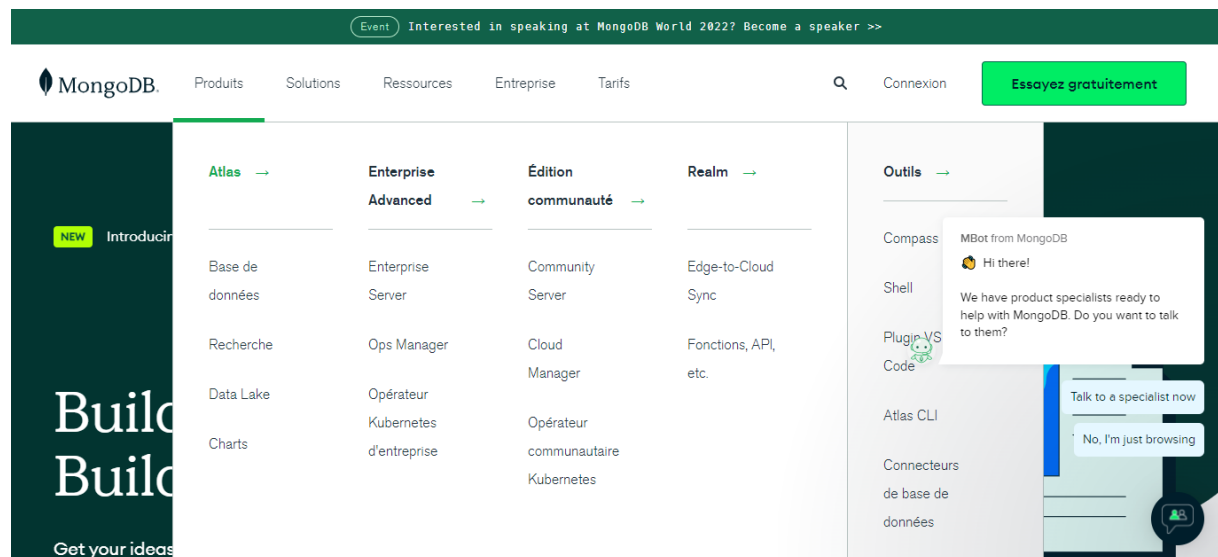
```
C:\Users\1701805>npm -v
6.14.11
```

Configuration de mongoDB Atlas

MongoDB Atlas est une base de données cloud, basée sur le serveur NoSQL open source MongoDB

Allez sur : <https://www.mongodb.com/fr-fr>


Ensuite Produit -> Atlas



Inscrivez-vous :



**Essai gratuit et
sans engagement**

 Se connecter avec Google


ou

Votre société (en option)

Votre e-mail professionnel

Prénom

Nom

Password 

8 caractères minimum

☐ J'accepte les conditions d'utilisation et la politique de confidentialité.

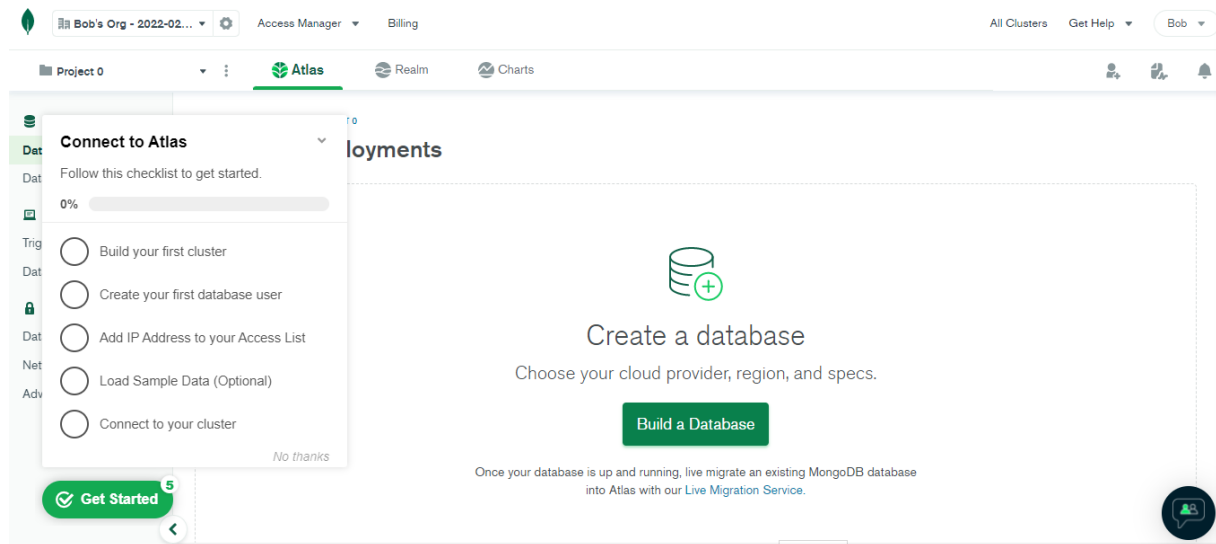
Essai gratuit et

Vous avez déjà un compte ? [Connectez-vous.](#)

Après vérification de votre email, vous pouvez vous connectez à votre espace.

Vous allez créer votre première database

Cliquez sur le bouton « Build a Database »



The screenshot shows the MongoDB Atlas dashboard. On the left, a sidebar menu is open, displaying a checklist titled 'Connect to Atlas' with steps: 'Build your first cluster', 'Create your first database user', 'Add IP Address to your Access List', 'Load Sample Data (Optional)', and 'Connect to your cluster'. A 'Get Started' button is at the bottom of the sidebar. The main content area is titled 'Create a database' and includes the instruction 'Choose your cloud provider, region, and specs.' with a prominent 'Build a Database' button. Below this, a note mentions the 'Live Migration Service'. The top navigation bar shows 'Project 0', 'Atlas', 'Realm', and 'Charts' tabs, along with user account information for 'Bob'.

Sélectionnez le compte « free » à droite en cliquant sur « Create »

PREVIEW

Serverless

For serverless applications that aren't critical with variable traffic. Minimal configuration required.

- ✓ Pay only for the operations you run
- ✓ Resources scale seamlessly to meet your workload
- ✓ Always-on security and backups

Create

Starting at
\$0.30/1M reads

ADVANCED

Dedicated

For production applications with sophisticated workload requirements. Advanced configuration controls.

- ✓ Network isolation and fine-grained access controls
- ✓ On-demand performance advice
- ✓ Multi-region and multi-cloud options available

Create

Starting at
\$0.08/hr*
*estimated cost \$56.94/month

FREE

Shared

For learning and exploring MongoDB in a cloud environment. Basic configuration options.

- ✓ No credit card required to start
- ✓ Explore with sample datasets
- ✓ Upgrade to dedicated clusters for full functionality

Create

Starting at
FREE

Laissez les informations par défaut

aws

Google Cloud

Azure

★ Recommended region ⓘ

🏷️ Paid tier region ⓘ

NORTH AMERICA

EUROPE

AUSTRALIA

🇺🇸 N. Virginia (us-east-1) ★

🇸🇪 Stockholm (eu-north-1) ★

🇦🇺 Sydney (ap-southeast-2) ★

🇺🇸 Oregon (us-west-2) ★

🇩🇪 Frankfurt (eu-central-1) ★

🇦🇸 ASIA

🇺🇸 Ohio (us-east-2) ★🏷️

🇮🇪 Ireland (eu-west-1) ★

🇮🇳 Mumbai (ap-south-1)

Cluster Tier

M0 Sandbox (Shared RAM, 512 MB Storage) Encrypted

Base hourly rate is for a MongoDB replica set with **3 data bearing servers**.

Shared Clusters for development environments and low-traffic applications

Tier	RAM	Storage	vCPU	Base Price
✓ M0 Sandbox	Shared	512 MB	Shared	Free forever
M0 clusters are best for getting started, and are not suitable for production environments.				
500 max connections Low network performance 100 max databases 500 max collections				
M2	Shared	2 GB	Shared	\$9 / MONTH
M5	Shared	5 GB	Shared	\$25 / MONTH

Additional Settings

MongoDB 4.4, No Backup ▾

Turn on Backup (M2 and up)

☐

[See Backup Solutions for Paid Clusters \(M2+\)](#)

Cluster Name

Cluster0 ▾

One time only: once your cluster is created, you won't be able to change its name.

Cluster0

Cluster names can only contain ASCII letters, numbers, and hyphens.

Validez vos choix en cliquant sur le bouton « Create Cluster » en bas de page

FREE

Free forever! Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

[Back](#)

Create Cluster

Bravo vous venez de créer votre premier cluster sous MongoDB Atlas

Maintenant, il va falloir définir les options de connexion


Username and Password

Certificate


Create a database user using a username and password. Users will be given the *read and write to any database privilege* by default. You can update these permissions and/or create additional users later. Ensure these credentials are different to your MongoDB Cloud username and password.

Username

Enter username

Password 

Enter password

 Autogenerate Secure Password


Create User

Laissez par défaut, saisissez un login et un password

Pour la connexion, cliquez sur « Add My Current IP Address »


2 Where would you like to connect from?

Enable access for any network(s) that need to read and write data to your cluster.



My Local Environment

Use this to add network IP addresses to the IP Access List. This can be modified at any time.



Cloud Environment

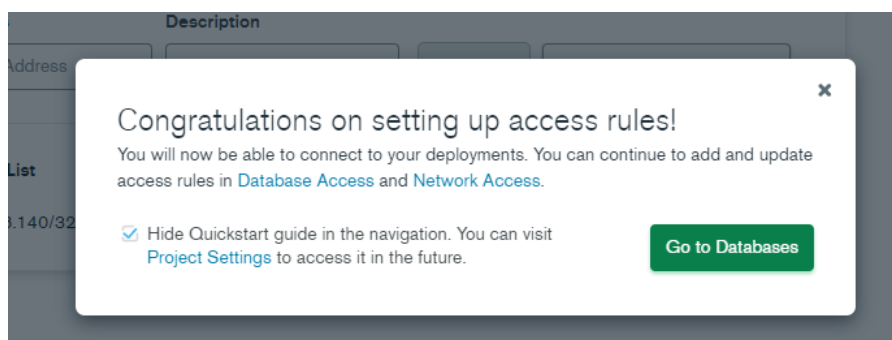
Use this to configure network access between Atlas and your cloud or on-premise environment. Specifically, set up IP Access Lists, Network Peering, and Private Endpoints.

Add entries to your IP Access List

Only an IP address you add to your Access List will be able to connect to your project's clusters.

IP Address	Description		
<input type="text" value="Enter IP Address"/>	<input type="text" value="Enter description"/>	<input type="button" value="Add Entry"/>	<input type="button" value="Add My Current IP Address"/>

Cliquez sur le bouton « Finish and Close » pour terminer



Editeur de code :

Vous pouvez choisir parmi plusieurs (Atom, Sublime Text, notepad++) par souci de simplicité nous utiliserons [Visual studio code](#)

We use cookies to improve your experience on our websites and for advertising. [Privacy Statement](#)

Accept all Manage cookies

Visual Studio Code

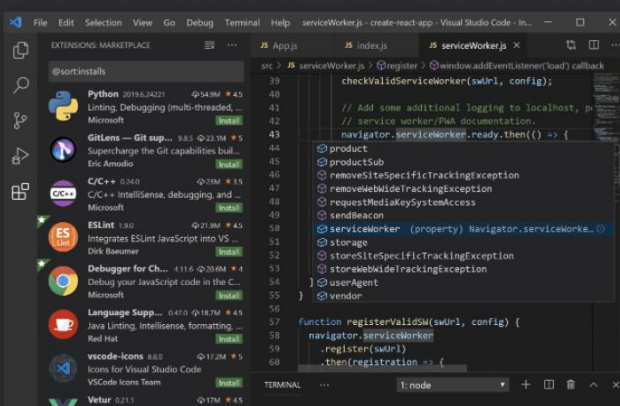
Code editing. Redefined.

Free. Built on open source. Runs everywhere.

[Download for Windows](#)
Stable Build

[Other platforms and Insiders Edition](#)

By using VS Code, you agree to its [license and privacy statement](#).



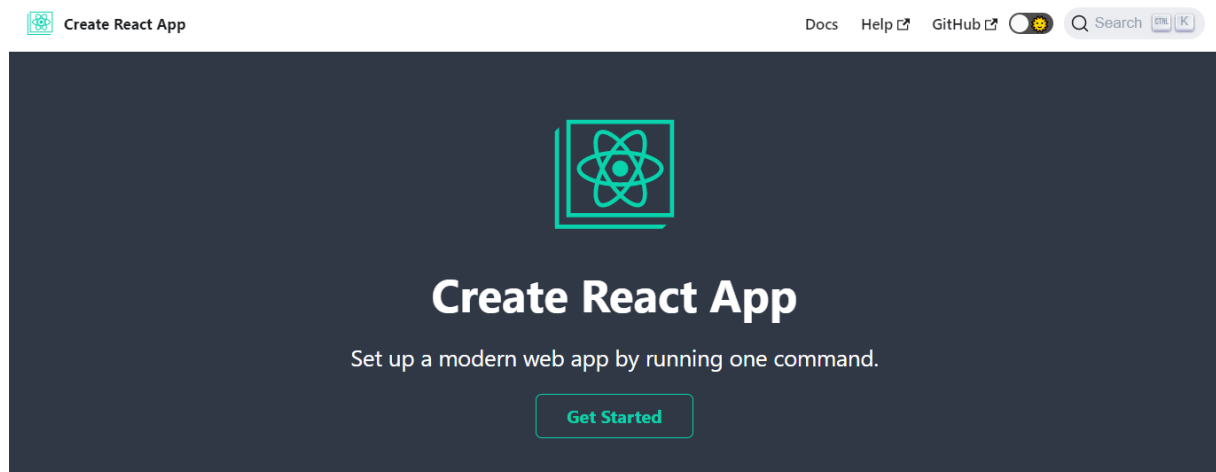
Projet MERN Stack : MongoDB ExpressJS ReactJS, NodeJS
Réalisation Stéphane Pontonnier Formateur AFPA

Afpa

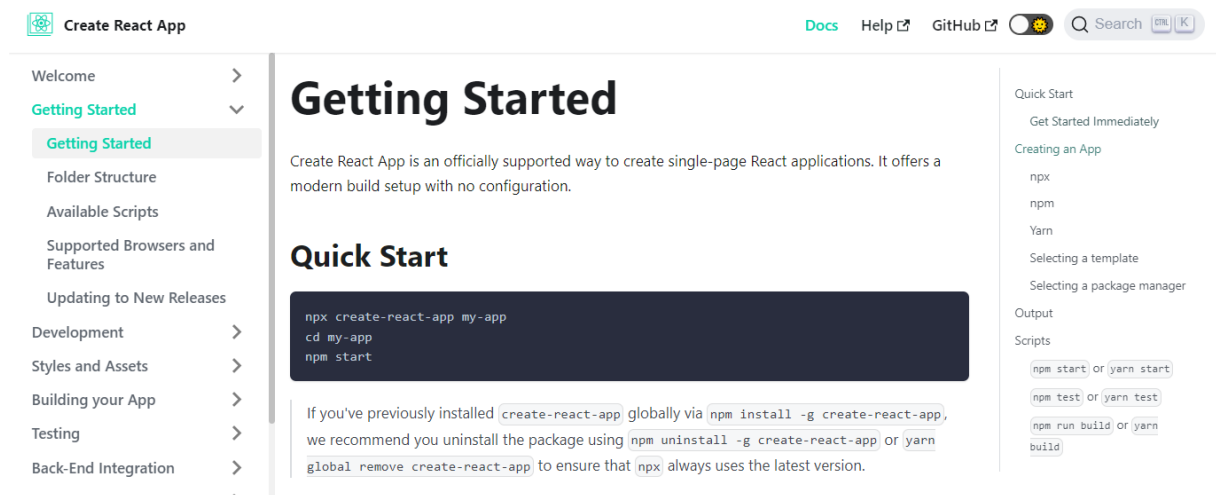
React app :

Vous allez également devoir installer React

<https://create-react-app.dev/>



Cliquez sur « Get Started »



Démarrage du projet (installation de REACT)

Création d'une application avec React app

Sur votre disque dur allez dans le répertoire de votre choix, ouvrez un terminal et ajouter la commande suivante (remplacer my-app par le nom de votre future application) :

```
C:\Users\1701805\OneDrive - AFPA\Documents\
RN-PIZZERIA>npx create-react-app my-app
npx: installed 67 in 6.892s
```

Projet MERN Stack : MongoDB ExpressJS ReactJS, NodeJS
Réalisation Stéphane Pontonnier Formateur AFPA

Afpa

L'installation peut prendre quelques minutes

Ensuite, allez dans le dossier créé avec la commande **cd nomdudossier** et tapez la commande suivante **npm start**

```
We suggest that you begin by typing:

cd my-app
npm start
```

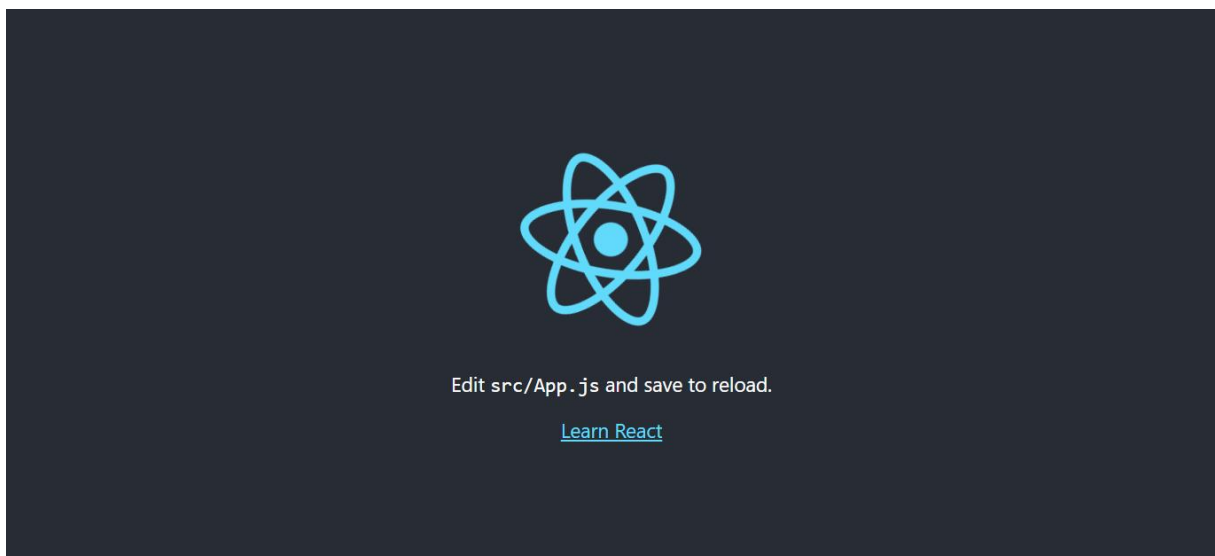
La compilation réussie vous pouvez visualiser la page d'accueil de votre application à l'adresse suivante :

```
Starting the development server...
Compiled successfully!

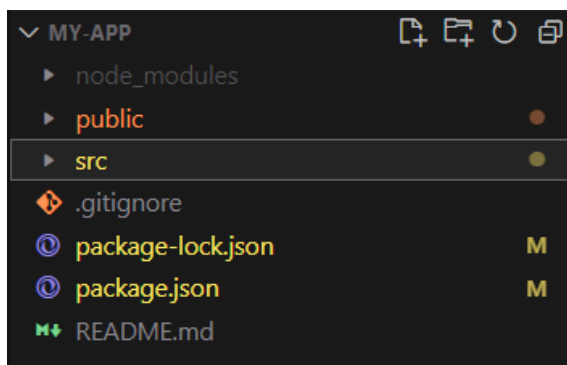
You can now view my-app in the browser.

Local:            http://localhost:3000
On Your Network:  http://172.19.12.47:3000
```

La page d'accueil :



L'arborescence :

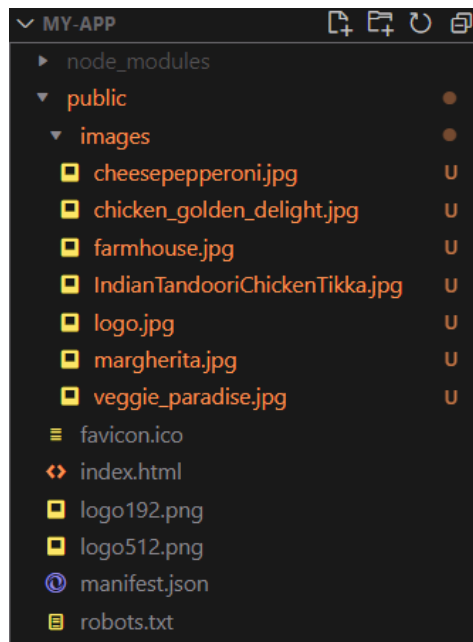


node_modules : les Bibliothèques externes (dont react)

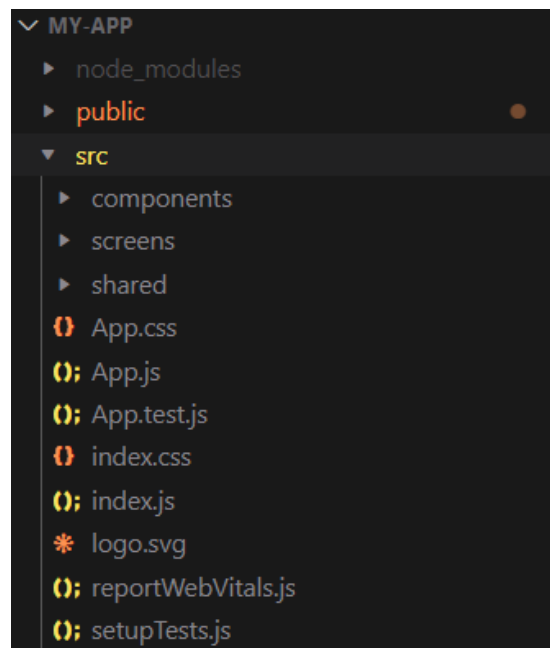
public : contient le fichier principal de l'application afficher par défaut (index.html)

src : contient les fichiers sources .css et .js utilisés par l'application

Dans public créer un répertoire images et mettre les images + le logo



Dans **src** créer les répertoires **components**, **screens** et **shared**



Installation de dépendances au fichier **package.json** :

Dans un terminal ouvert dans votre dossier application, **my-app** dans l'exemple, ligne de code suivante **npm install bootstrap**

```
C:\Users\1701805\OneDrive - AFPA\Documents\
RN-PIZZERIA\my-app>npm install bootstrap
npm WARN @apideck/better-ajv-errors@0.3.2 re
```

Faites la même chose pour :

axios <https://www.npmjs.com/package/axios>

react-bootstrap <https://www.npmjs.com/package/react-bootstrap>

react-router-bootstrap <https://www.npmjs.com/package/react-router-bootstrap>

react-icons <https://www.npmjs.com/package/react-icons>

react-router-dom <https://www.npmjs.com/package/react-router-dom>

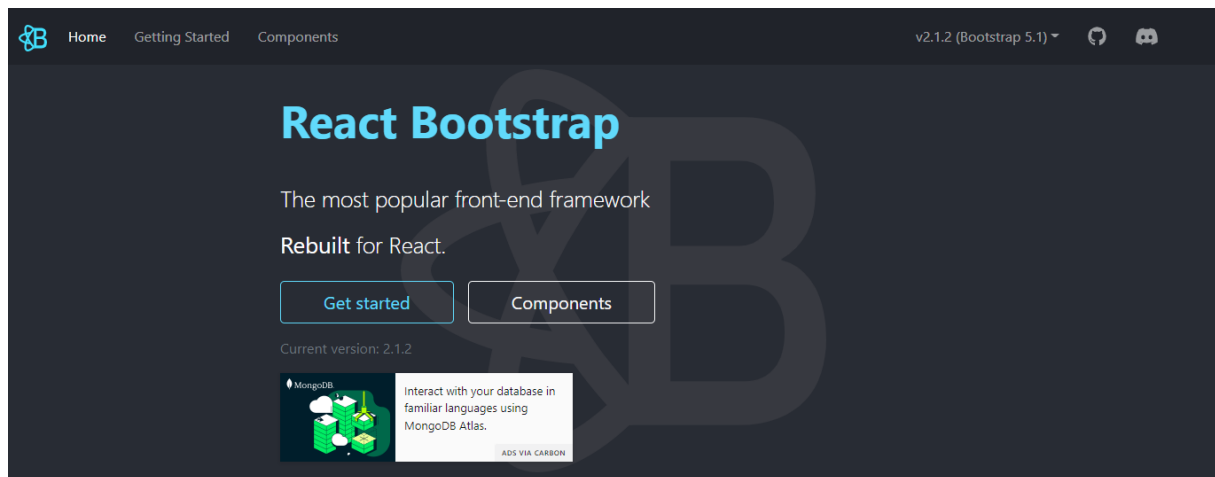
Modifier le fichier src/index.js :

```
index.js M X
src > index.js
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import 'bootstrap/dist/css/bootstrap.min.css';
4 import './index.css';
5 import App from './App';
6 import reportWebVitals from './reportWebVitals';
7
```

Création de la « Topbar » de l'application

Nous allons utiliser le framework Bootstrap pour le css de l'application, allez sur :

<https://react-bootstrap.github.io/>



Rebuilt with React

React-Bootstrap replaces the Bootstrap JavaScript. Each component has been built from scratch as a true React component, without unneeded dependencies like jQuery.

As one of the oldest React libraries, React-Bootstrap has evolved and grown alongside React, making it an excellent choice as your UI foundation.

Bootstrap at its core

Built with compatibility in mind, we embrace our bootstrap core and strive to be compatible with the world's largest UI ecosystem.

By relying entirely on the Bootstrap stylesheet, React-Bootstrap just works with the thousands of Bootstrap themes you already love.

Accessible by default

The React component model gives us more control over form and function of each component.

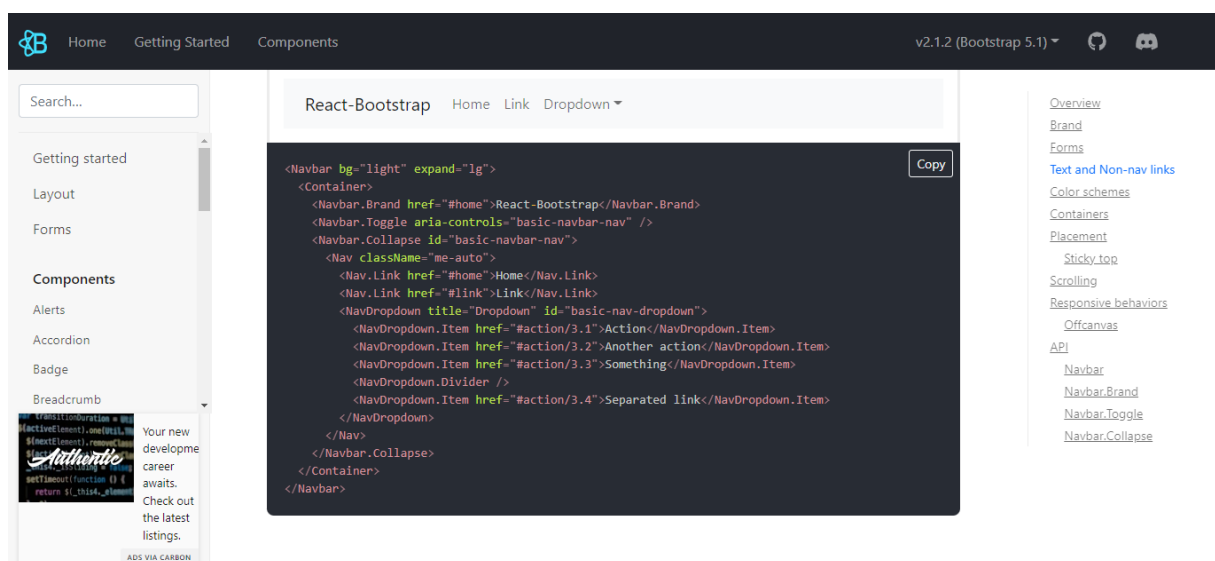
Each component is implemented with accessibility in mind. The result is a set of accessible-by-default components, over what is possible from plain Bootstrap.

Pour créer la navbar, nous utiliserons le composant navbar

Allez dans « Getting Started » -> Components -> Navbar

IMPORTANT

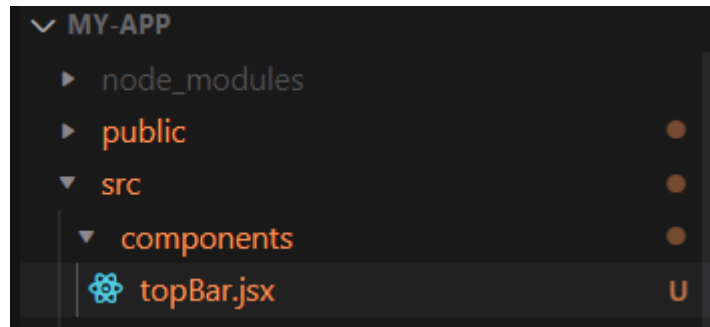
Une erreur sur les visuels, le nom du fichier est **TopBar.jsx**



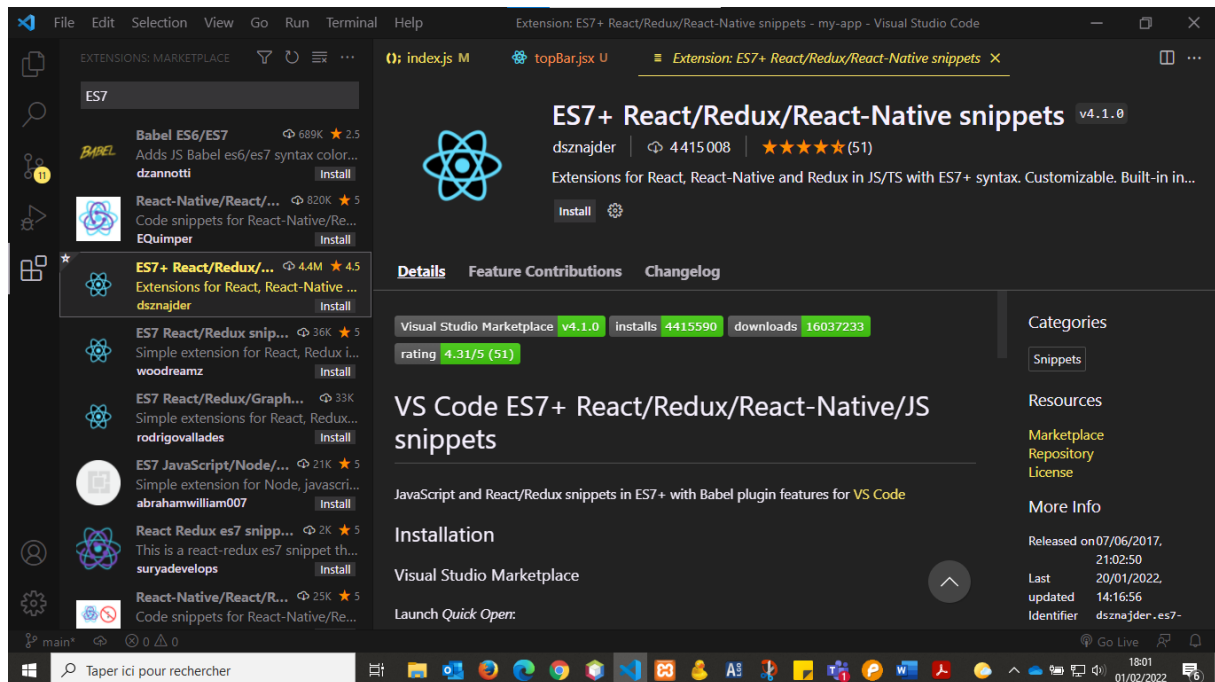
Dans votre application, créez un fichier **TopBar.jsx** dans src/components

Projet MERN Stack : MongoDB ExpressJS ReactJS, NodeJS
Réalisation Stéphane Pontonnier Formateur AFPA

Afpa



Ensuite dans VS Code, installer le plugin suivant :



Dans votre **TopBar.jsx**, tapez **rafce** puis validez. Vous devez avoir ceci :

```
src > components > topBar.jsx > ...
1  import React from 'react';
2
3  const topBar = () => {
4    return <div></div>;
5  };
6
7  export default topBar;
8
```

Vous allez importer les composants dont vous avez besoin pour créer la navigation. Toujours dans le fichier **TopBar.jsx** :

```
src > components > topBar.jsx > topBar
1  import React from "react";
2  import { Navbar, Nav, Container } from "react-bootstrap";
3  import { LinkContainer } from "react-router-bootstrap";
4
```

Ensuite il faut créer la navigation :

```
5  const topBar = () => {
6    return (
7      <
8        <Navbar bg="dark" variant="dark" expand="lg">
9          <Container fluid>
10             <h6>Dummy Data</h6>
11             <Nav className="ms-auto">
12               <LinkContainer to="/">
13                 <Nav.Link>Home</Nav.Link>
14               </LinkContainer>
15               <LinkContainer to="about">
16                 <Nav.Link>About us</Nav.Link>
17               </LinkContainer>
18               <LinkContainer to="contact">
19                 <Nav.Link>Contact</Nav.Link>
20               </LinkContainer>
21               <LinkContainer to="Policy">
22                 <Nav.Link>Terms and Policy</Nav.Link>
23               </LinkContainer>
24             </Nav>
25           </Container>
26         </Navbar>
27       </>
28     );
29   };

```

Dans votre fichier **src/App.js**, il faut maintenant importer le composant TopBar

```

my-app > src > (); App.js > ...
1 | import './App.css';
2 | import { BrowserRouter } from 'react-router-dom';
3 | import TopBar from './components/TopBar';
4 |
5 | function App() {
6 |   return (
7 |     <BrowserRouter>
8 |       <TopBar />
9 |     </BrowserRouter>
10 |   );
11 | }
12 |
13 | export default App;
14 |

```

Le résultat après un npm start dans le terminal :



Vous allez Insérez dans cette entête une icône, allez sur :

<https://react-icons.github.io/react-icons>

React Icons

npm v4.3.1 build passing

Include popular icons in your React projects easily with react-icons, which utilizes ES6 imports that allows you to include only the icons that your project is using.

Installation (for standard modern project)

```
npm install react-icons --save
```

Usage

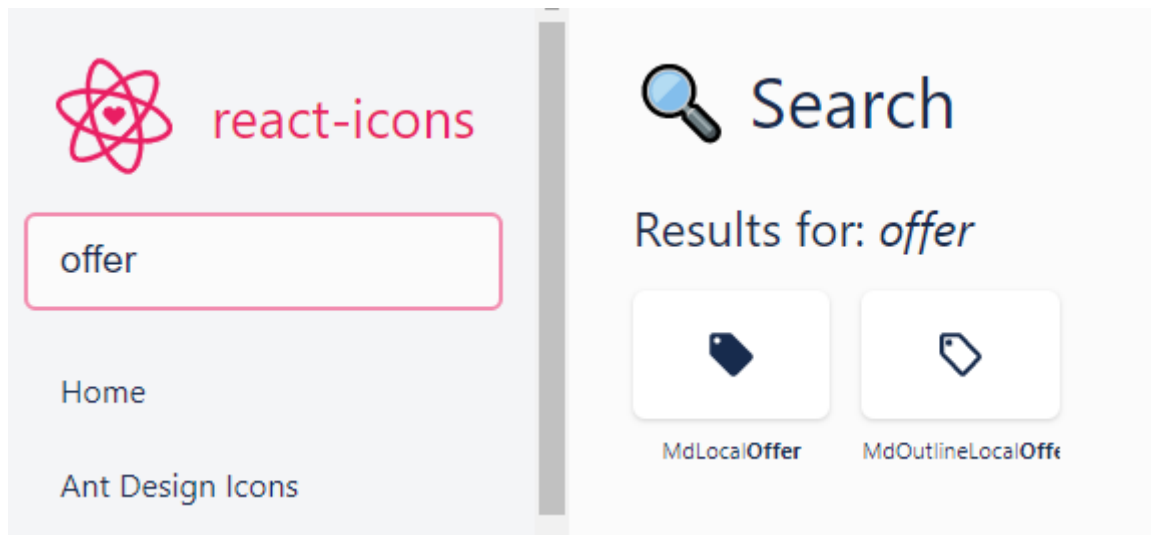
```
import { FaBeer } from 'react-icons/fa';

class Question extends React.Component {
  render() {
    return <h3> Lets go for a <FaBeer />? </h3>
  }
}
```

Installation (for meteorjs, gatsbyjs, etc)

If your project grows in size, this option is available. This method has the trade-off that it takes a long time to install the package. Suitable for

Dans le champ recherche inscrivez **offer** et copiez MdLocal**Offer**



Importez le composant dans **TopBar.jsx**

```
import React from "react";
import { Navbar, Nav, Container } from "react-bootstrap";
import { LinkContainer } from "react-router-bootstrap";
import { MdLocalOffer } from "react-icons/md"
```

Et modifiez le contenu de la balise h6

```
<h6 className="text-light">
  <MdLocalOffer className="text-warning" />
  Livraison gratuite à domicile pour toute commande supérieure à 50 €
</h6>
```

Le résultat sur la page d'accueil dans le navigateur :



Vous pouvez ajouter l'attribut **activeClassName** aux **LinkContainer**

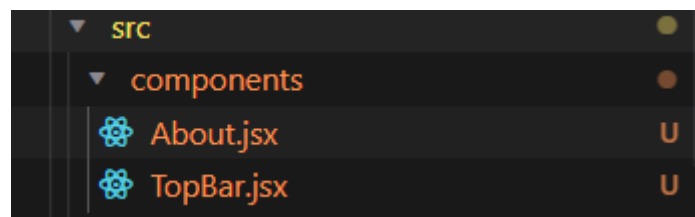
```
<Nav className="ms-auto">
  <LinkContainer to="/" activeClassName>
    <Nav.Link>Accueil</Nav.Link>
  </LinkContainer>
  <LinkContainer to="/about" activeClassName>
    <Nav.Link>A propos</Nav.Link>
  </LinkContainer>
  <LinkContainer to="/contact" activeClassName>
    <Nav.Link>Contact</Nav.Link>
  </LinkContainer>
  <LinkContainer to="/Policy" activeClassName>
    <Nav.Link>CGV</Nav.Link>
  </LinkContainer>
</Nav>
```

activeClassName : La classe à attribuer à l'élément lorsqu'il est actif. Cliquez sur les liens dans votre navigateur.

Vous venez de créer le premier composant de votre application, il va falloir maintenant créer les pages qui correspondent aux différents liens de votre navigation.

Création des pages statiques (components) de l'application

Dans le répertoire **src/components**, créez le fichier **About.jsx**



Dans ce fichier, tapez **rafce** puis validez. Le résultat :

```
my-app > src > components > About.jsx > ...
1  import React from 'react';
2
3  const About = () => {
4    return <div></div>;
5  };
6
7  export default About;
```

Modifiez comme suit :

```
my-app > src > components > About.jsx > ...
1  import React from "react";
2
3  const About = () => {
4    return (
5      <
6
7      </>
8    );
9  };
10
11  export default About;
```


Importez les composants Bootstrap nécessaires

```
my-app > src > components > About.jsx > About
1  import React from "react";
2  import {Container, Row, Col} from 'react-bootstrap';
```

Commencez par créer la structure de votre page à l'aide des composants Bootstrap

```
const About = () => {
  return (
    <Container style={{marginTop: '50px'}}>
    </Container>
  );
};
```

Sur l'utilisation du style inline avec React :

https://www.w3schools.com/react/react_css.asp

Poursuivez en intégrant de nouveaux composants

```
<Container style={{marginTop: '50px'}}>
  <h1>Qui sommes-nous ?</h1>
  <p>Lorem200</p>
  <h1>Notre spécialité</h1>
  <Row>
    <Col md={6}>
      <p>Lorem200</p>
    </Col>
    <Col md={6}>
      <p>200</p>
    </Col>
  </Row>
```

```

    <Row>
      <h1>Le Chef</h1>
      <Col md={3}>
        <p>lorem100</p>
      </Col>
      <Col md={3}>
        <p>Lorem100</p>
      </Col>
      <Col md={3}>
        <p>Lorem100</p>
      </Col>
      <Col md={3}>
        <p>Lorem100</p>
      </Col>
    </Row>
  </Container>

```

A propos de l'utilisation du système de grille de react-bootstrap :
<https://react-bootstrap.github.io/layout/grid/>

Dans le fichier **App.js**

```

src > (🔗) App.js > 📁 App
1  import './App.css';
2  import { BrowserRouter, Routes, Route } from 'react-router-dom';
3  import TopBar from './components/TopBar';
4  import About from './components/About';

```

Importez les composants du paquet react-router-dom

Configurer un routeur

Le paquet a un composant pratique appelé **BrowserRouter**. Nous devons d'abord l'importer depuis react-router-dom afin de pouvoir utiliser le routage dans notre application.

Il doit contenir tous les éléments de notre application où le routage est nécessaire. Cela signifie que si nous avons besoin du routage dans l'ensemble de notre application, nous devons envelopper notre composant le plus haut avec **BrowserRouter**.

Créer une route

Pour créer un itinéraire, nous devons importer **Route** à partir du package du routeur.

Ajoutez-le ensuite là où nous voulons afficher le contenu. Le composant Route a plusieurs propriétés. Mais ici, nous avons juste besoin de path et de element.

path: c'est le chemin à charger lorsque l'itinéraire est atteint. Ici, nous utilisons /About pour accéder à la page « à propos ».

element : permet à notre route de charger un composant précis

Enfin, en enveloppant nos routes avec **Routes**, nous indiquons à React Router de ne charger qu'une seule route à la fois.

```
function App() {  
  return (  
    <BrowserRouter>  
      <TopBar />  
      <Routes>  
        <Route path="/About" element={<About />} />  
      </Routes>  
    </BrowserRouter>  
  );  
}
```

Documentation React Router V6 :

<https://github.com/remix-run/react-router/blob/main/docs/getting-started/tutorial.md>

Le résultat via le navigateur :

🛒 Livraison gratuite à domicile pour toute commande supérieure à 50 €

Accueil A propos Contact CGV

Qui sommes-nous ?

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Libero autem repudiandae officia ab, ad adipisci? Est saepe adipisci voluptatibus totam ex pariat iusto eum accusamus sapiente atque et ipsum non, repellendus soluta consectetur at mollitia officia repudiandae! Dolor, sit labore dicta, excepturi omnis deleniti vitae repudiandae soluta odio debitis tempora nisi veritatis ipsum laudantium consectetur reprehenderit provident animi eum quas at eveniet libero. Omnis quisquam praesentium voluptates, iure saepe unde deleniti accusantium! Pariatur fuga possimus vero modi libero cumque saepe nisi voluptate nulla nemo explicabo est error repellendus deserunt eum aliquam commodi delectus molestias dignissimos voluptatum, quibusdam, minima id nam! Esse soluta dolorum quas facere, in recusandae aspernatur. Consequatur, assumenda laborum similique voluptatem omnis quis optio tenetur possimus iusto distinctio id quas odio doloribus magni quod ipsam saepe deleniti, nemo recusandae minus praesentium perferendis ad. Natus iusto assumenda consequuntur voluptates alias quae porro dolorem voluptatum eveniet, exercitationem, atque est pariat qui neque? Ipsam explicabo similique quasi sit dolore, illum deleniti enim autem soluta quia assumenda deserunt praesentium blanditiis animi in, pariat quae suscipit accusantium accusamus consequuntur aliquid perferendis laborum facilis. Necessitatibus, exercitationem quaerat iste atque ipsum impedit totam! Quis quia error, expedita tempore ab magni voluptatibus. Similique recusandae dignissimos rerum!

Notre spécialité

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Explicabo eos dignissimos quasi assumenda consequatur, cupiditate voluptatem, inventore labore voluptas adipisci, iusto eveniet exercitationem dolores provident? At quia recusandae quos dolore nisi ipsam, error aut suscipit nobis aliquam explicabo dignissimos nihil harum ratione animi est provident sint ullam! Tempora, labore qui!

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Explicabo eos dignissimos quasi assumenda consequatur, cupiditate voluptatem, inventore labore voluptas adipisci, iusto eveniet exercitationem dolores provident? At quia recusandae quos dolore nisi ipsam, error aut suscipit nobis aliquam explicabo dignissimos nihil harum ratione animi est provident sint ullam! Tempora, labore qui!

Le Chef

Lorem ipsum dolor sit amet consectetur, adipisicing elit. Eius molestiae quis, recusandae odit incidunt ullam debitis, reiciendis itaque optio obcaecati minima assumenda quasi natus suscipit. Minima excepturi esse voluptates, numquam reiciendis error iste nihil blanditiis nam explicabo dolorum ratione ipsum culpa facilis porro nostrum quod magni quas magnam optio aut fugiat, id consectetur. In omnis eius dignissimos recusandae debitis nobis dolorum aperiam temporibus sit dolores quibusdam fugiat veniam, aliquid, exercitationem atque beatae voluptatum vero at! Quidem modi voluptatem tempore soluta et voluptatum placeat, at odio, distinctio molestiae animi vero! Repudiandae, impedit. Velit tempore laboriosam labore voluptate, culpa aliquid beatae debitis.

Lorem ipsum, dolor sit amet consectetur adipisicing elit. Lure doloribus porro aliquid debitis quis asperiores repellendus quisquam ea et maiores, fugit veniam nihil recusandae placeat tempore est. Nobis autem iste ullam fugiat animi perferendis sapiente, cumque fugit itaque, tenetur deleniti, laboriosam cum eos incidunt fuga enim numquam sint tempore accusantium minus? Natus, nulla dignissimos? Architecto, ipsam sed repellat nemo assumenda necessitatibus exercitationem impedit, illo magnam eum laudantium libero corporis doloremque. Nisi omnis deserunt voluptates nemo, incidunt quidem sunt autem. Ratione fugit enim odio voluptatibus inventore quia aut voluptate sint veniam dolor voluptatem quibusdam, ab eligendi assumenda ullam, optio amet cupiditate?

Lorem ipsum, dolor sit amet consectetur adipisicing elit. Lure doloribus porro aliquid debitis quis asperiores repellendus quisquam ea et maiores, fugit veniam nihil recusandae placeat tempore est. Nobis autem iste ullam fugiat animi perferendis sapiente, cumque fugit itaque, tenetur deleniti, laboriosam cum eos incidunt fuga enim numquam sint tempore accusantium minus? Natus, nulla dignissimos? Architecto, ipsam sed repellat nemo assumenda necessitatibus exercitationem impedit, illo magnam eum laudantium libero corporis doloremque. Nisi omnis deserunt voluptates nemo, incidunt quidem sunt autem. Ratione fugit enim odio voluptatibus inventore quia aut voluptate sint veniam dolor voluptatem quibusdam, ab eligendi assumenda ullam, optio amet cupiditate?

Lorem ipsum, dolor sit amet consectetur adipisicing elit. Lure doloribus porro aliquid debitis quis asperiores repellendus quisquam ea et maiores, fugit veniam nihil recusandae placeat tempore est. Nobis autem iste ullam fugiat animi perferendis sapiente, cumque fugit itaque, tenetur deleniti, laboriosam cum eos incidunt fuga enim numquam sint tempore accusantium minus? Natus, nulla dignissimos? Architecto, ipsam sed repellat nemo assumenda necessitatibus exercitationem impedit, illo magnam eum laudantium libero corporis doloremque. Nisi omnis deserunt voluptates nemo, incidunt quidem sunt autem. Ratione fugit enim odio voluptatibus inventore quia aut voluptate sint veniam dolor voluptatem quibusdam, ab eligendi assumenda ullam, optio amet cupiditate?


Projet MERN Stack : MongoDB ExpressJS ReactJS, NodeJS
Réalisation Stéphane Pontonnier Formateur AFPA

Afpa




Création du composant « Contact »

Dans le répertoire **src/components**, créez le fichier **Contact.jsx**


Dans ce fichier, tapez **rafce** puis validez. Le résultat :

```
src > components >  Contact.jsx > ...  
1  import React from 'react';  
2  
3  const Contact = () => {  
4    return <div></div>;  
5  };  
6  
7  export default Contact;  
8
```

Importez les composants Bootstrap nécessaires

```
src > components >  Contact.jsx > ...  
1  import React from "react";  
2  import { Container, Row, Col, Table } from "react-bootstrap";  
3  
4  const Contact = () => {  
5    return (  
6        
7        
8        
9    );  
10 };  
11  
12 export default Contact;  
13
```

Créez la structure de votre page en vous inspirant de la page **About.jsx**

```
const Contact = () => {  
  return (  
      
    <Container style={{marginTop: '50px'}}>  
      <Row>  
        <Col md={6}>  
          <h1>Pizza Delicious</h1>  
          <p>lorem100</p>  
        </Col>  
      </Row>  
    </Container>  
  </>  
  );  
};
```

Ajoutez un tableau et réaliser l'intégration de la page

Rendez-vous sur <https://react-bootstrap.netlify.app/components/table/#tables>, et copiez le code du premier tableau de la page. Copiez le dans **Contact.jsx** et modifiez-le, le résultat de la page :

🍷 Livraison gratuite à domicile pour toute commande supérieure à 50 €

Accueil A propos Contact CGV

Pizza Delicious


Notre adresse :

103 , Rue des Olives noires

75021 Paris

POUR VOTRE INFORMATION! Nous offrons un service traiteur complet pour tout événement, grand ou petit. Nous comprenons vos besoins et nous préparons nos plats pour satisfaire les critères les plus importants, à la fois esthétiques et gustatifs.

-- Nos Coordonnées --		
☎	Téléphone	01 23 45 67 89
📱	Portable	01 98 76 54 32
✉	Email	contact@pizza-delicious.com



Les imports de la page **Contact.jsx**

```
src > components > Contact.jsx > Contact
1  import React from "react";
2  import { Container, Row, Col, Table, Image } from "react-bootstrap";
3  import { FiPhoneCall } from "react-icons/fi";
4  import { ImMobile } from "react-icons/im";
5  import { AiOutlineMail } from "react-icons/ai";
6
```

Intégrez la page comme le rendu ci-dessus. Vous pouvez comparer votre intégration avec le fichier à réaliser qui est dans **corrections/Contact.jsx**

Parfait vous venez de réaliser votre second composant, maintenant sur la base des 2 premiers. Veuillez développer le troisième, Policy.jsx. Vous trouverez le fichier dans **corrections/Policy.jsx**

Le résultat dans le navigateur :

Conditions Générales de Ventes

DEFINITIONS

Dans les présentes Conditions Générales de Vente, les mots et expressions suivants auront les significations suivantes :

« Client » désigne le ou les internaute(s) naviguant sur le Site et effectuant une commande par téléphone ou un achat en ligne de Produits.

« CGV » désigne les présentes Conditions Générales de Vente de Produits sur le Site.

« PIZZA DELICIOUS » désigne, selon le cas, soit la société d'exploitation du restaurant sous enseigne PIZZA DELICIOUS sélectionné par le Client et auprès de laquelle la commande sera enregistrée et dont les coordonnées seront indiquées sur l'e-mail confirmant la commande et sur le ticket de caisse délivré par PIZZA DELICIOUS lors du retrait ou de la livraison des Produits, soit, de manière générale, les restaurants sous enseigne PIZZA DELICIOUS.

« Produit(s) » désigne le(s) produit(s) proposé(s) à la vente sur le Site.

PORTEE DES CONDITIONS GENERALES DE VENTE

PIZZA DELICIOUS a pour activité la vente de pizzas, et autres produits d'accompagnement, en livraison, sur place et en vente à emporter.

Les présentes CGV sont accessibles à tout moment sur le Site et s'appliquent à toutes les commandes de Produits passées par le Client sur le Site ou par téléphone auprès de PIZZA DELICIOUS. En passant commande d'un Produit, le Client reconnaît expressément avoir pris connaissance des CGV et les avoir acceptées sans restriction ni réserve, en cochant la case prévue à cet effet, et renonce, notamment, à se prévaloir de toute autre version ou document contradictoire, et notamment les conditions applicables pour les ventes en magasins. L'acceptation des CGV n'est en aucune façon conditionnée par une signature manuscrite de la part du Client.

PIZZA DELICIOUS se réserve le droit de mettre à jour à tout moment les CGV. En cas de modification, il sera appliqué, à chaque commande, les CGV en vigueur au jour de la commande de Produits. Le Client est invité à consulter régulièrement les CGV afin de se tenir informé des évolutions les plus récentes.

Les données enregistrées par le Site constituent la preuve de l'ensemble des transactions conclues avec le Client.

ACCEPTATION DES CGV

Le Client déclare avoir pris connaissance et accepté les présentes CGV avant la passation de sa Commande. Toute Commande d'un Produit sur le Site matérialise donc de la part du Client l'acceptation des présentes CGV

TERRITOIRE – ZONES DE LIVRAISON

Le Client commandant par téléphone ou via le Site et qui souhaite être livré doit fournir une adresse de livraison valide située dans une zone de livraison d'un établissement PIZZA DELICIOUS. La liste actualisée des restaurants PIZZA DELICIOUS et des villes desservies par PIZZA DELICIOUS est disponible sur le Site. Si l'adresse indiquée n'est pas desservie, le Client en est informé avant la commande

Le contenu textuel vient des cgv de la boîte à pizza

Création de la « Navbar » avec le logo

<https://react-bootstrap.netlify.app/components/navbar/#navbars>

The screenshot shows the React-Bootstrap documentation page for the Navbar component. The page is titled "Responsive behaviors" and explains how to use the `expand` prop along with `Navbar.Toggle` and `Navbar.Collapse` to control when content collapses behind a button. It also mentions setting `defaultExpanded` to make the navbar start expanded and `collapseOnSelect` to make it collapse automatically when an item is selected. A yellow warning box states: "Watch out! You need to provide a breakpoint value to `expand` in order for the Navbar to collapse at all."

Below the text is a code block with a "Copy" button, showing the following code:

```
<Navbar collapseOnSelect expand="lg" bg="dark" variant="dark">
  <Container>
    <Navbar.Brand href="#home">React-Bootstrap</Navbar.Brand>
    <Navbar.Toggle aria-controls="responsive-navbar-nav" />
    <Navbar.Collapse id="responsive-navbar-nav">
      <Nav className="me-auto">
        <Nav.Link href="#features">Features</Nav.Link>
```

On the right side of the page, there is a table of contents with links to various sections: Overview, Brand, Forms, Text and Non-nav links, Color schemes, Containers, Placement, Sticky top, Scrolling, Responsive behaviors (which is the current section), Offcanvas, API, and a list of components: Navbar, Navbar.Brand, Navbar.Toggle, and Navbar.Collapse.

Copiez le code ci-dessus (la navbar de la section « Responsive behaviors »)

Dans VS code, créez un nouveau composant **NavBar.jsx**

```
src > components > 🗄️ NavBar.jsx > ...
1  import React from 'react';
2
3  const NavBar = () => {
4    return (
5      <>
6        /* Copiez le code de la NavBar ici */
7      </>
8    );
9  };
10
11 export default NavBar;
```

Et collez le code à l'emplacement indiqué

Importez les composants nécessaires dans **NavBar.jsx**

```
src > components > 🗄️ NavBar.jsx > [🔍] NavBar
1  import React from "react";
2  import { Navbar, Nav, Container, Image } from "react-bootstrap";
3  import { LinkContainer } from "react-router-bootstrap";
```

LinkContainer : Enveloppez votre élément React Bootstrap dans un `<LinkContainer>` pour qu'il se comporte comme un React Router `<Link>`

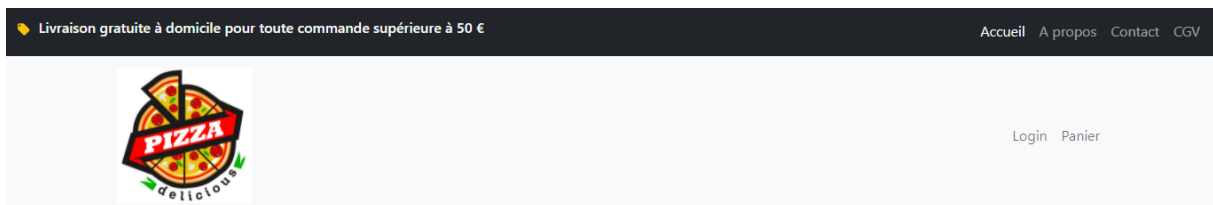
```
src > components > 🗄️ NavBar.jsx > [🔍] default
4
5  const NavBar = () => {
6    return (
7      <>
8        <Navbar collapseOnSelect expand="lg" bg="light" variant="light">
9          <Container>
10             <Navbar.Brand>
11               <Image src="images/logo.jpg" alt="Logo Pizza Delicious" style={{width:'20%'}} />
12             </Navbar.Brand>
13             <Navbar.Toggle aria-controls="responsive-navbar-nav" />
14             <Navbar.Collapse id="responsive-navbar-nav">
15               <Nav className="ms-auto">
16                 <LinkContainer to="/login">
17                   <Nav.Link>Login</Nav.Link>
18                 </LinkContainer>
19                 <LinkContainer to="/panier">
20                   <Nav.Link>Panier</Nav.Link>
21                 </LinkContainer>
22               </Nav>
23             </Navbar.Collapse>
24           </Container>
25         </Navbar>
26       </>
27     );
28  };
```

Intégration du logo et du menu de connexion

Dans **App.jsx**, importez le composant NavBar

```
7 | import NavBar from "../components/NavBar";
8 |
9 | function App() {
10 |   return (
11 |     <BrowserRouter>
12 |       <TopBar />
13 |       <NavBar />
```

Le résultat dans votre navigateur :



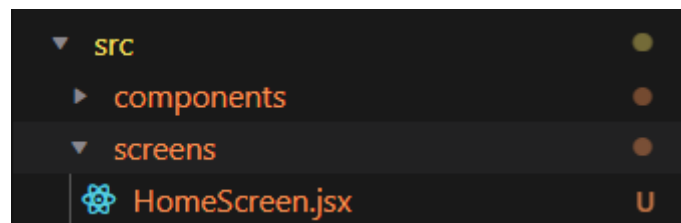
La prochaine étape va être de définir le contenu de la page d'accueil de votre application

Création de la liste des pizzas (page accueil de l'application)

Vous avez à votre disposition un fichier nommé **pizza-data.js** qui liste les différentes pizzas vendues par PIZZA DELICIOUS. Ce fichier contient un tableau d'objets, regardez sa construction.

Copiez **pizza-data.js** dans le répertoire **src**

Dans le répertoire **src/screens**, créez le fichier **HomeScreen.jsx**



Lancez dans ce fichier la commande **rafce** pour créer la structure comme suit :


```
src > screens > HomeScreen.jsx > ...
1  import React from 'react';
2
3  const HomeScreen = () => {
4    return (
5      <
6
7      </>
8    );
9  };
10
11  export default HomeScreen;
```

Il faut ensuite importer les données de **pizza-data.js** ainsi que les composants Bootstrap nécessaires

```
src > screens > HomeScreen.jsx > HomeScreen
1  import React from 'react';
2  import AllPizza from '../pizza-data';
3  import { Container, Row, Col } from 'react-bootstrap';
4
```

Il va falloir ensuite parcourir **pizza-data.js** afin d'afficher chaque pizza :

```
5  const HomeScreen = () => {
6    return (
7      <
8        <Container>
9          <Row>
10             { AllPizza.map( pizza => (
11
12             )) }
13          </Row>
14        </Container>
15      </>
16    );
17  };
```

La méthode `Array.map()` : Retourne un tableau avec les éléments de Array sur lesquels une fonction de callback est appliquée

https://www.w3schools.com/jsref/jsref_map.asp

Dans **src/components**, créez un nouveau fichier **Pizza.jsx**

```
src > components > Pizza.jsx > ...
1  import React from 'react';
2
3  const Pizza = () => {
4    return (
5      <div>
6        <h1>Pizzas titre</h1>
7      </div>
8    );
9  };
10
11 export default Pizza;
```

Et importez le dans **HomeScreen.jsx**

```
src > screens > HomeScreen.jsx > ...
1  import React from 'react';
2  import AllPizza from '../pizza-data';
3  import { Container, Row, Col } from "react-bootstrap";
4  import Pizza from '../components/Pizza';
5
6  const HomeScreen = () => {
7    return (
8      <Container>
9        <Row>
10         { AllPizza.map( pizza => (
11           <Col>
12             <Pizza />
13           </Col>
14         )) }
15       </Row>
16     </Container>
17   );
18 };
19
20 export default HomeScreen;
```

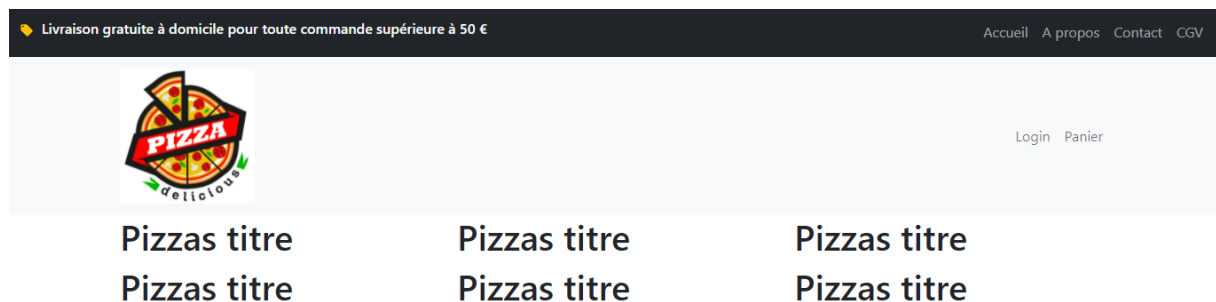
Affichez le composant Pizza sur la page d'accueil

```

6  const HomeScreen = () => {
7    return (
8      <Container>
9        <Row>
10         { AllPizza.map( pizza => (
11           <Col md={4}>
12             <Pizza lapizza={pizza} />
13           </Col>
14         )) }
15        </Row>
16      </Container>
17    </>
18  );
19  };

```

Le composant **Pizza** possède un attribut **lapizza** qui a pour valeur **pizza** (i.e le contenu d'une pizza de pizza-data.js). Le résultat à l'affichage :



Pour afficher le résultat, il faut dans **App.jsx**, importez le composant **HomeScreen** et ajoutez une route

```

import HomeScreen from "../screens/HomeScreen";

function App() {
  return (
    <BrowserRouter>
      <TopBar />
      <NavBar />
      <Routes>
        <Route path="/" element={<HomeScreen />} />
        <Route path="/About" element={<About />} />
        <Route path="/Contact" element={<Contact />} />
        <Route path="/Policy" element={<Policy />} />
      </Routes>
    </BrowserRouter>
  );
}

```

Maintenant, il va falloir afficher les pizzas. Utilisez les composants Bootstrap suivants dans **Pizza.jsx** :

```
src > components > Pizza.jsx > Pizza
1  import React from "react";
2  import { Card, Button, Row, Col } from "react-bootstrap";
3
```

Il va falloir passer à chaque élément Pizza qui va s'afficher, les valeurs stockées dans **pizza-data.jsx**. Pour notre exemple nous avons besoin des **props**.

Une **props** est une propriété que l'on passe à un composant. C'est donc une information qui vient de "l'extérieur" du composant, généralement, de son parent direct (mais pas toujours).

<https://fr.reactjs.org/docs/components-and-props.html>

Notre composant **Pizza** possède une propriété **lapizza** qui contient chaque objet de **pizza-data.js**

```
{ AllPizza.map( pizza => (
  <Col md={4}>
    <Pizza lapizza={pizza} />
  </Col>
)) }
```

Extrait de **HomeScreen.jsx**

Ici, la méthode **Array.map()** permet de lire les données de **pizza-data.js** et d'appeler le composant **Pizza** en lui passant comme valeur de propriété **lapizza** les données de chacune des pizzas.

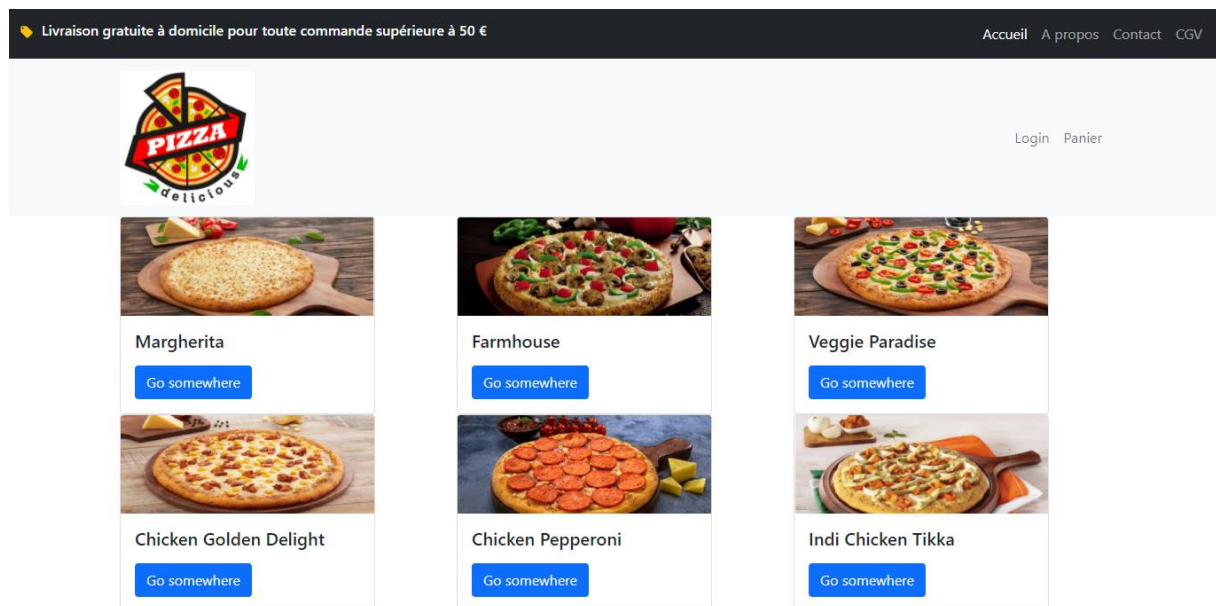
```
const Pizza = (props) => {
  return (
    <Card style={{ width: "18rem" }}>
      <Card.Img variant="top" src={props.lapizza.image} />
      <Card.Body>
        <Card.Title>{props.lapizza.nom}</Card.Title>
        <Card.Text>
          <Row>...
        </Row>
        </Card.Text>
        <Button variant="primary">Go somewhere</Button>
      </Card.Body>
    </Card>
  );
};
```

Dans le composant Pizza, on lui indique qu'il va recevoir une propriété (**props**) et que pour cette propriété nommée **lapizza**, il doit récupérer certaines valeurs stockées dans **pizza-data.js**

Vue du premier objet et de ses attributs

```
const pizzas = [
  {
    nom: "Margherita",
    tailles: ["small", "medium", "large"],
    prix: [
      {
        small: 11,
        medium: 14,
        large: 19,
      },
    ],
    categorie: "veg",
    image: "/images/margherita.jpg",
    description: "Un Délicieux classique avec 100 % de vrai fromage mozzarella",
  },
],
```

Le résultat sur le navigateur :



Pour afficher les différentes tailles des pizzas, dans **Pizza.jsx** :

```

<Card.Text>
  <Row>
    <Col md={6}>
      <h6>Taille :
      <select name="" id="">
        {props.lapizza.tailles.map(taille => (
          <option value={taille}>{taille}</option>
        ))}
      </select>
    </h6>
    </Col>
    <Col md={6}> ...
  </Col>
</Row>
</Card.Text>

```

Pour afficher la quantité commandée, dans **Pizza.jsx** :

```

<Row>
  <Col md={6}>
    <h6>Taille :
    <select name="" id="">
      {props.lapizza.tailles.map(taille => (
        <option value={taille}>{taille}</option>
      ))}
    </select>
  </h6>
  </Col>
  <Col md={6}>
    <h6>Quantité :<br/>
    <select name="" id="">
      {[...Array(10).keys()].map((v,i) => (
        <option value={i+1}>{i+1}</option>
      ))}
    </select>
  </h6>
  </Col>

```

Regardez le code qui permet d'afficher les quantités ; décortiquez-le, comprenez-le afin d'être capable d'expliquer ce qu'il fait.

Vous allez améliorer votre code afin d'ajouter de nouvelles fonctionnalités.



Dans votre fichier **Pizza.jsx** :

```
src > components > Pizza.jsx > Pizza
1  import React, {useState} from "react";
2  import { Card, Button, Row, Col } from "react-bootstrap";
3
```

Commencez par importez le Hook **useState**

Qu'est-ce qu'un Hook ? Un Hook est une fonction qui permet de « se brancher » sur des fonctionnalités React. Par exemple, **useState** est un Hook qui permet d'ajouter l'état local React à des fonctions composants.

Ensuite, déclarez 2 variables d'état

```
const Pizza = (props) => {
  const [taille, setTaille] = useState('small');
  const [quantite, setQuantite] = useState(1);
}
```

Le seul argument à passer au Hook **useState()** est l'état initial.

Pour lire la valeur d'une variable d'état et mettre à jour la nouvelle valeur :

```

<Col md={6}>
  <h6>Taille :
  <select
    // la valeur par défaut est celle de la variable d'état
    value={taille}
    // Pour mettre à jour l'état
    onChange={(e) => setTaille(e.target.value)}
  >
    {props.lapizza.tailles.map(taille => (
      <option>{taille}</option>
    ))}
  </select>
</h6>
</Col>

```

```

<Col md={6}>
  <h6>Quantite :<br/>
  <select
    value={quantite}
    onChange={(e) => setQuantite(e.target.value)}
  >
    {[...Array(10).keys()].map((v,i) => (
      <option>{i+1}</option>
    ))}
  </select>
</h6>
</Col>

```

La documentation React : <https://fr.reactjs.org/docs/hooks-state.html>

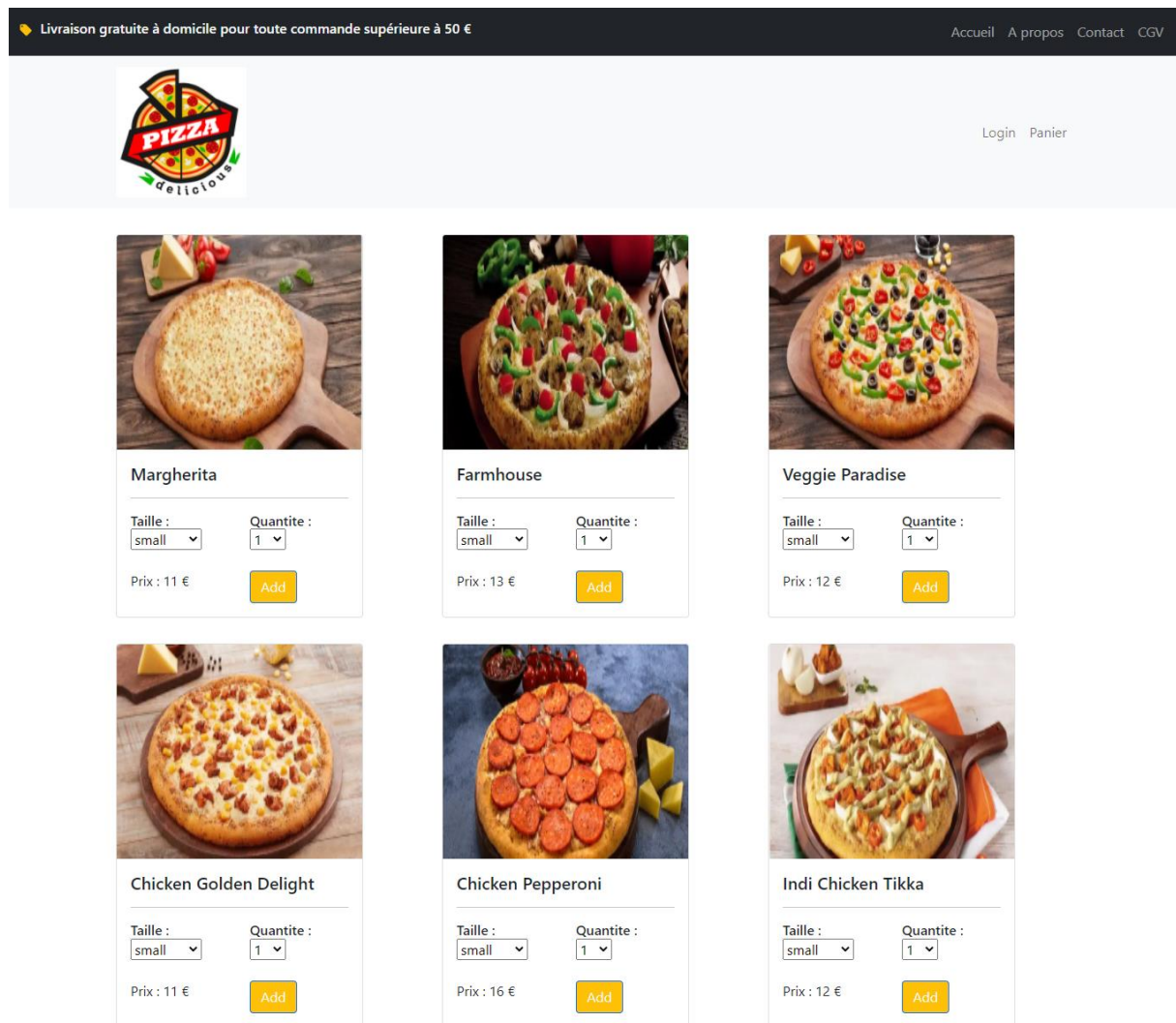
Il ne vous reste plus qu'à gérer l'affichage du prix en fonction de la quantité. A la suite de **<Card.Text/>**

```

<Row>
  <Col md={6}>
    Prix : {props.lapizza.prix[0][taille] * quantite} €
  </Col>
  <Col md={6}>
    <Button className="bg-warning text-light">Add</Button>
  </Col>
</Row>

```


Le résultat sur le navigateur :



Vous venez de créer la page d'accueil de votre application en utilisant les notions de props et de Hook. Un rappel a également été fait sur `array.map()`, il ne vous reste maintenant qu'à présenter chaque pizza grâce à une fenêtre modal.

Création d'une fenêtre modale avec Bootstrap pour afficher les détails d'une pizza

Dans le fichier **Pizza.jsx**, vous allez devoir installer le code nécessaire à la fenêtre modal

Rendez-vous sur <https://react-bootstrap.netlify.app/components/modal/#live-demo>, et regardez attentivement le code présenté. Vous retrouvez des notions que vous venez de découvrir.

Prenez le temps de comprendre ce bout de code.

Dans votre fichier **Pizza.jsx**, commencez par importez le composant Modal

```
src > components > Pizza.jsx > ...
1  import React, {useState} from "react";
2  import { Card, Button, Row, Col, Modal } from "react-bootstrap";
3
```

Ensuite, copiez le code suivant dans votre fonction :

```
const Pizza = (props) => {
  const [taille, setTaille] = useState("small");
  const [quantite, setQuantite] = useState(1);
  const [show, setShow] = useState(false);

  const handleClose = () => setShow(false);
  const handleShow = () => setShow(true);
  // ...
}
```

A la suite du composant **<Card />** insérez le code de la modal en vous aidant de la documentation.

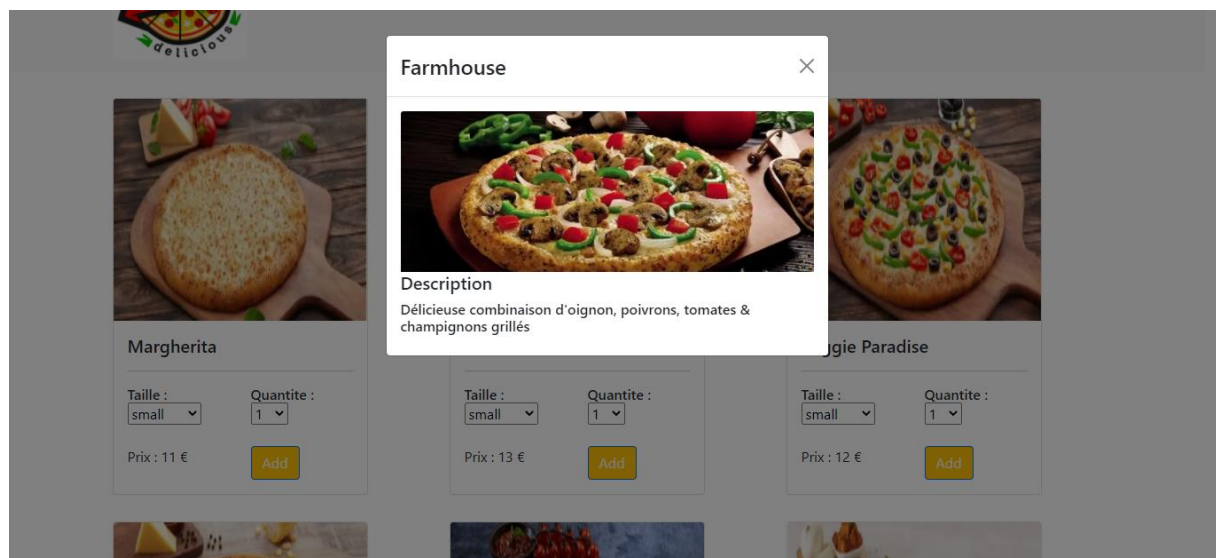
```

    </Card>
    { /* Modal */ }
  <Modal show={show} onHide={handleClose}> ...
  </Modal>
</>
);
};

export default Pizza;

```

Vous devez arriver à ce résultat dans votre navigateur :



Vous pouvez comparer votre code avec celui du fichier corrigé qui se trouve dans ***corrections/Pizza.jsx***

Pensez à mettre votre css dans le fichier ***src/index.css***, si vous avez besoin de styler votre intégration

Cette première partie est terminée. Vous avez pu concevoir rapidement une application React single page grâce à Create React App. Vous avez passé des valeurs à vos composants via les props, utilisé le Hook d'état useState, stylé votre travail avec Bootstrap. Il est temps de mettre en place l'environnement de développement de votre application via Node et MongoDB dans une seconde partie.