Marvel - Version 0.4.0

Alexandre GIRARD - Conseiller en Nouvelles Technologies - alexandre.girard@maif.fr

Gestion de la navigation dans l'application

- installation et configuration de **React Router**, librairie tierce permettant de gérer la navigation
- Mise en place de la navigation dans l'application Marvel
  - définir les routes
  - définir les composants à afficher en fonction de la route
- Définir le layout de l'application
  - header, footer, menu, sidebar, contenu principal...

git

A vous de faire le nécessaire pour travailler correctement avec git et générer la version 0.4.0 de l'application.

- Rappel:
  - On ne travaille pas directement sur la branche principale.
  - On utilise des branches de fonctionnalités pour travailler sur des fonctionnalités spécifiques.
  - On commit régulièrement de manière atomique et avec des messages de commit explicites.
  - On ne commit pas du code rendant l'application inutilisable.
  - On ne merge pas une branche sans avoir testé le code.

git

Nous verrons plus tard comment sécuriser nos branches de développement et de production. Pour l'instant c'est à nous de faire attention à ne pas commettre d'erreurs.

Création des pages de l'application

- Création des pages de l'application
  - Home
  - Characters
  - Contact
  - About

Création des pages de l'application (suite)

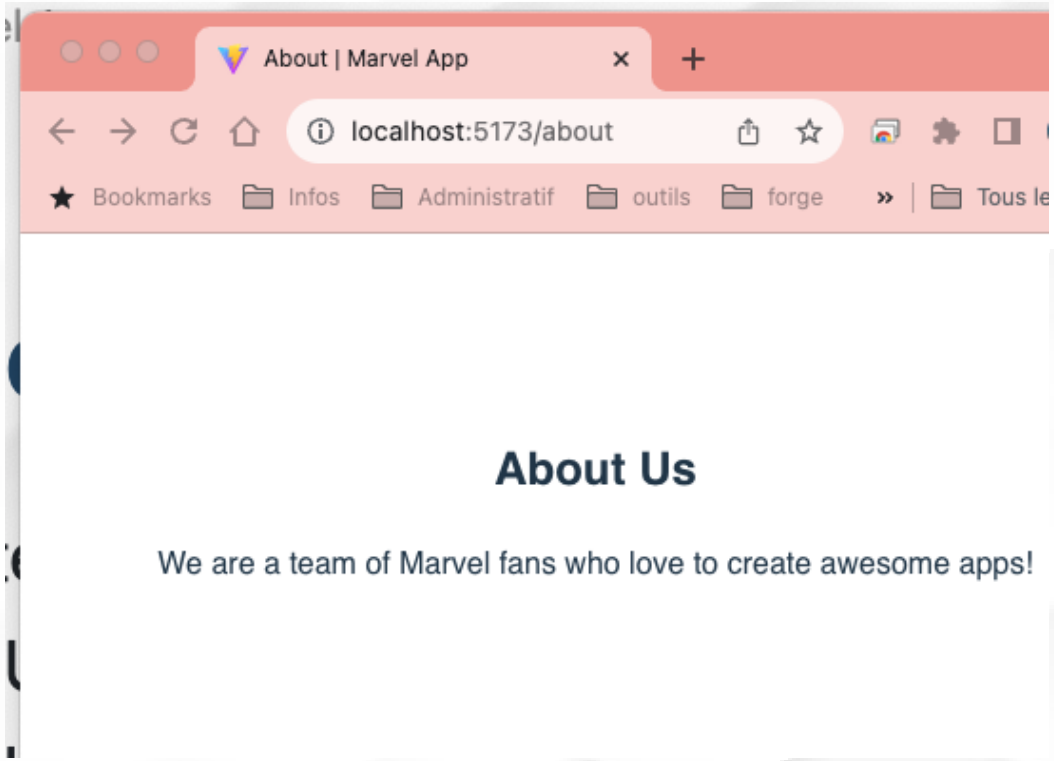
- Les pages sont des composants React
  - définis dans le dossier `src/pages`
  - importés dans le composant principal `App`

Page About

Créez le composant `AboutPage` dans le dossier `src/pages` et ajoutez le contenu suivant:

- Un système permettant de modifier le titre de la page (document.title)
- Un en-tête h2
- Un paragraphe

Pour tester cette page, modifier le fichier App.jsx pour utiliser le composant **AboutPage** (et supprimer l'affichage actuel).

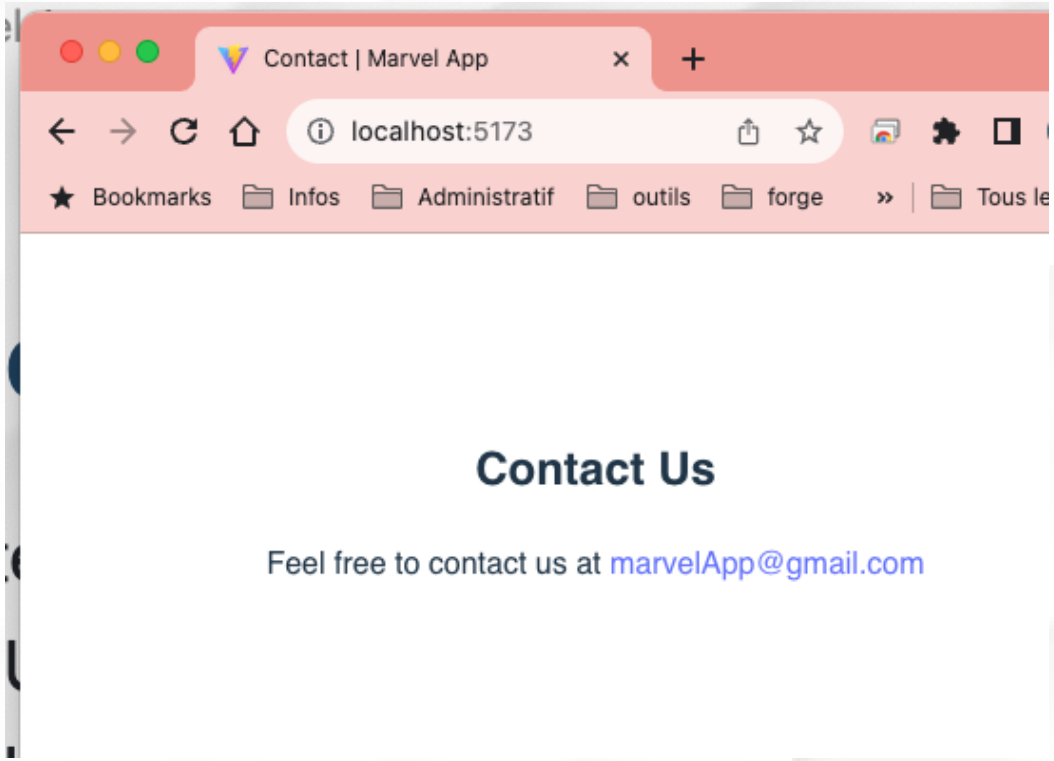


Page Contact

Créez le composant `ContactPage` dans le dossier `src/pages` et ajoutez le contenu suivant:

- Un système permettant de modifier le titre de la page (document.title)
- Un en-tête h2
- Un message de contact avec un lien de type mailto

Pour tester cette page, modifier le fichier App.jsx pour afficher la page **Contact**.

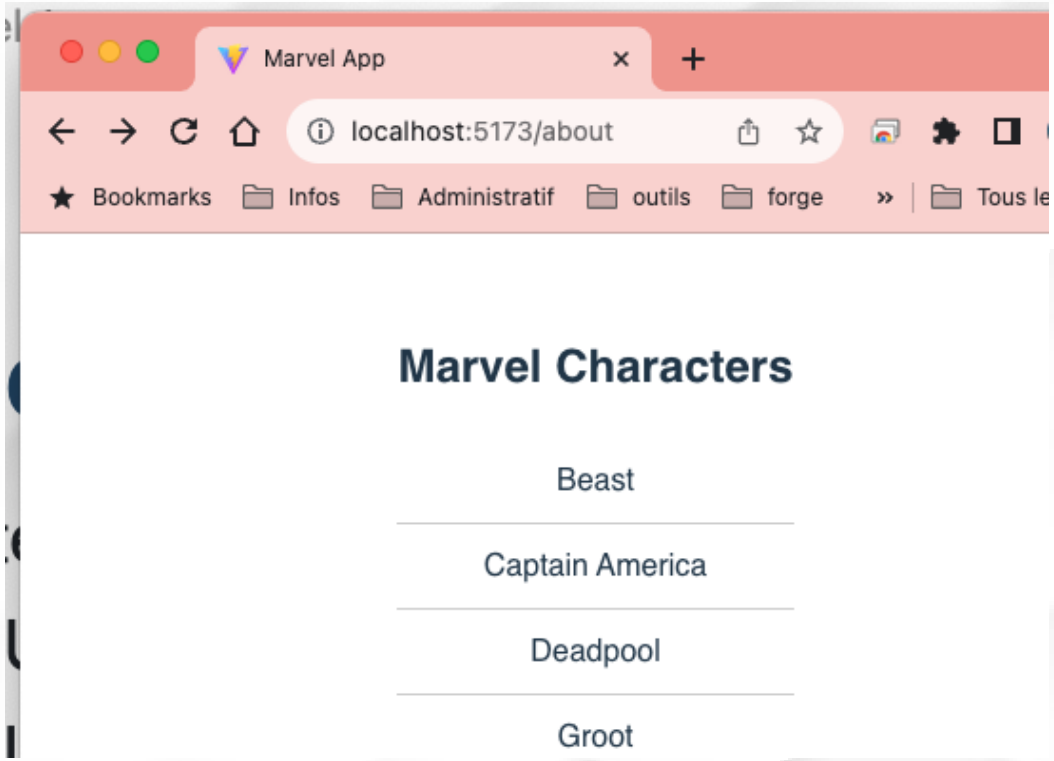


Page Characters

Créez le composant `CharactersPage` dans le dossier `src/pages` et ajoutez le contenu suivant:

- Un système permettant de modifier le titre de la page (document.title)
- Le contenu qui était affiché sur la page d'accueil, attention au chemin d'accès aux imports

Pour tester cette page, modifier le fichier App.jsx pour afficher la page **Characters**.



Page Home

Nous n'avons pas de contenu spécifique pour la page Home, nous afficherons le contenu de la page Characters. Nous traiterons la page Home plus tard.

Problématique - Gestions des routes

- Comment gérer la navigation dans une application React ?
- Comment définir les routes de l'application ?
- Comment afficher les composants en fonction de la route ?

React Router

- **React Router** est une librairie tierce permettant de gérer la navigation dans une application React
- Quelques explications sur react-router :
  - Guide React
  - React Router

React Router - Layout

Le layout de l'application permet de définir la structure de l'application, c'est-à-dire les éléments qui seront affichés sur toutes les pages de l'application.

- **Header** : en-tête de l'application
  - navigation entre les pages
- **Main** : contenu principal de l'application
- **Footer** : pied de page de l'application

React Router - Layout (suite)

- Création du layout de l'application
  - Création du composant `Layout` dans le dossier `src`
  - Importation du composant `Layout` dans le composant `App`
  - Utilisation du composant `Layout` dans le composant `App`

React Router - Layout (suite)

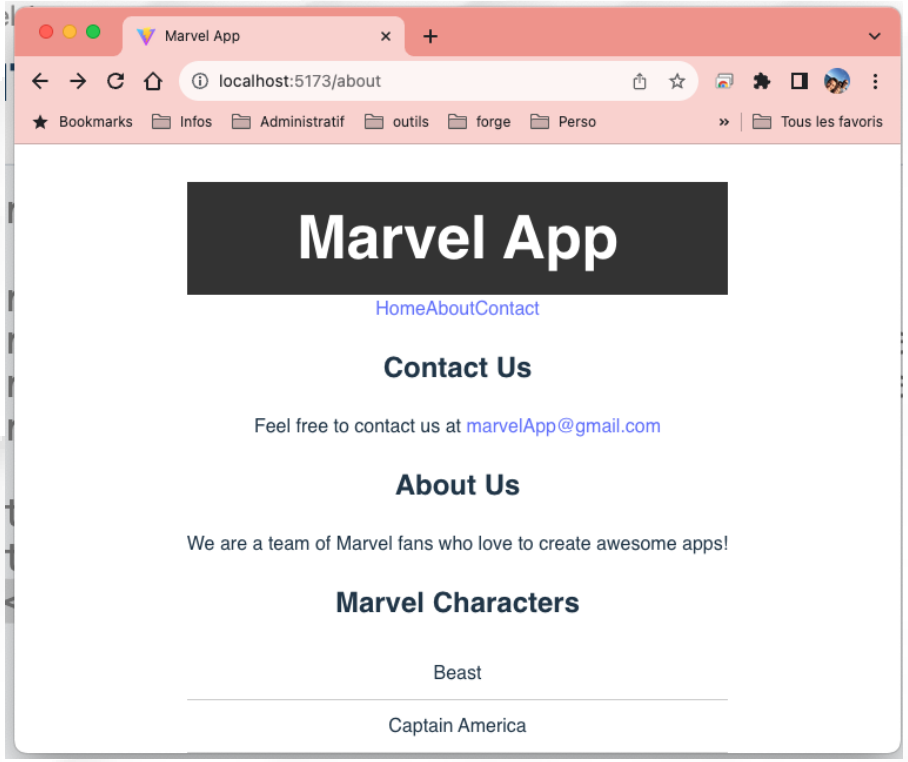
```
import React from 'react';

const Layout = ({ children }) => {
    return (
        <>
            <header>
                <h1>Marvel App</h1>
                <nav>
                    <a href="/">Home</a>
                    <a href="/about">About</a>
                    <a href="/contact">Contact</a>
                </nav>
            </header>
            <main>
                {children}
            </main>
            <footer>
                <p>Marvel App - 2023</p>
            </footer>
        </>
    );
};

export default Layout;
```

Router - Layout (suite)

```
import './App.css'

import AboutPage from './pages/AboutPage'
import ContactPage from './pages/ContactPage'
import CharactersPage from './pages/CharactersPage'
import Layout from './Layout'

function App() {
  return (
    <Layout>
      <AboutPage />
      <ContactPage />
      <CharactersPage />
    </Layout>
  )
}

export default App
```

Router - Layout (suite)

Cela n'est toujours pas la bonne solution, mais nous avançons étape par étape, nous allons maintenant utiliser **React Router** pour gérer la navigation entre les pages.



React Router - routes

- Création du composant `routes` dans le dossier `src`
  - Importation du composant `routes` dans le composant `App`
  - Utilisation du composant `routes` dans le composant `App`

React Router - routes (suite)

```
import Layout from "./Layout";
import AboutPage from "./pages/AboutPage";
import CharactersPage from "./pages/CharactersPage";
import ContactPage from "./pages/ContactPage";

const routes = [
    {
        path: "/",
        element: <Layout />,
        children: [
            { path: "/", element: <CharactersPage /> },
            { path: "/about", element: <AboutPage /> },
            { path: "/contact", element: <ContactPage /> },
        ],
    },
];

export default routes;
```

React Router - routes - explications

path : chemin de la route

element : composant à afficher

children : routes enfants

- Nous avons défini les routes de l'application
  - `/` : page d'accueil, affiche la liste des personnages
  - `/about` : page About
  - `/contact` : page Contact
- Les trois pages sont affichées dans le composant `Layout` qui contient le header, le main et le footer de l'application

React Router - App

```
import './App.css'

import { RouterProvider, createBrowserRouter } from 'react-router-dom';
import routes from './routes'

// Create a router that uses the client side history strategy for
const router = createBrowserRouter(routes)

function App() {
  return (
    <RouterProvider router={router} />
  )
}

export default App
```

React Router - App - explications

- Nous avons importé les composants `RouterProvider` et `createBrowserRouter` de `react-router-dom`
- Nous avons importé le composant `routes` que nous avons créé précédemment
- Nous avons créé un router avec la fonction `createBrowserRouter` en lui passant les routes de l'application
- Nous avons enveloppé l'application avec le composant `RouterProvider` en lui passant le router que nous avons créé

React Router - Layout

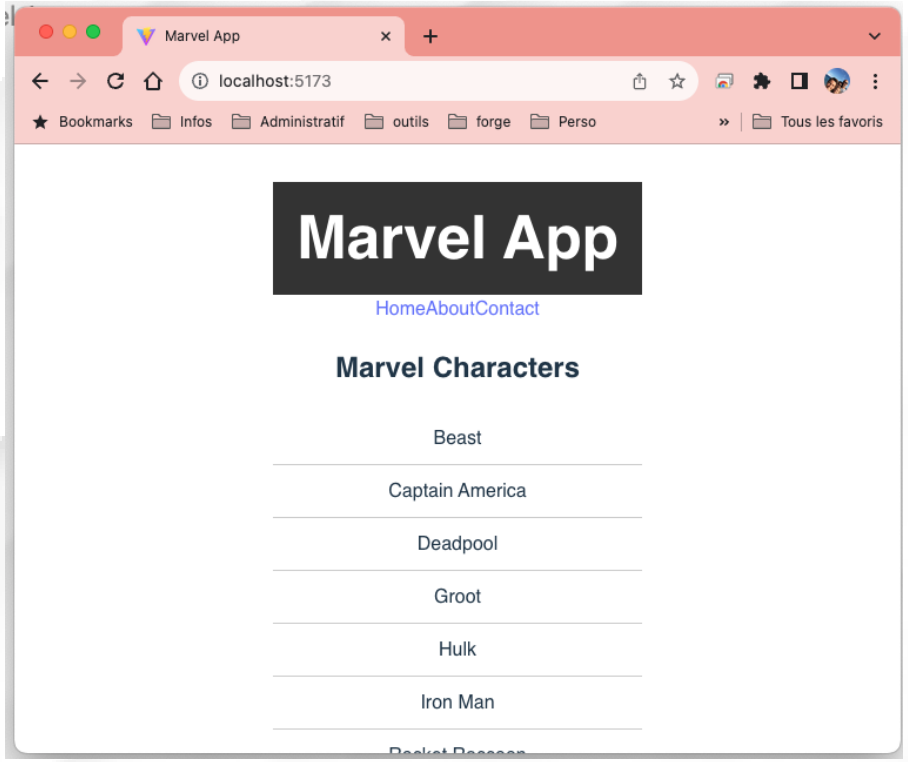
```
import React from 'react';
import { Outlet } from 'react-router';

const Layout = () => {
    return (
        <>
            <header>
                <h1>Marvel App</h1>
                <nav>
                    <a href="/">Home</a>
                    <a href="/about">About</a>
                    <a href="/contact">Contact</a>
                </nav>
            </header>
            <main>
                <Outlet />
            </main>
            <footer>
                <p>Marvel App - 2023</p>
            </footer>
        </>
    );
};

export default Layout;
```

React Router - Layout - explications

- Nous avons importé le composant `Outlet` de `react-router`
- Nous avons remplacé le contenu principal de l'application qui était géré grâce à la prop `children` par le composant `Outlet`
- Le composant `Outlet` permet d'afficher le composant correspondant à la route

React Router

Nous avons maintenant une application avec une gestion de la navigation grâce à **React Router**.



React Router - NavLinks

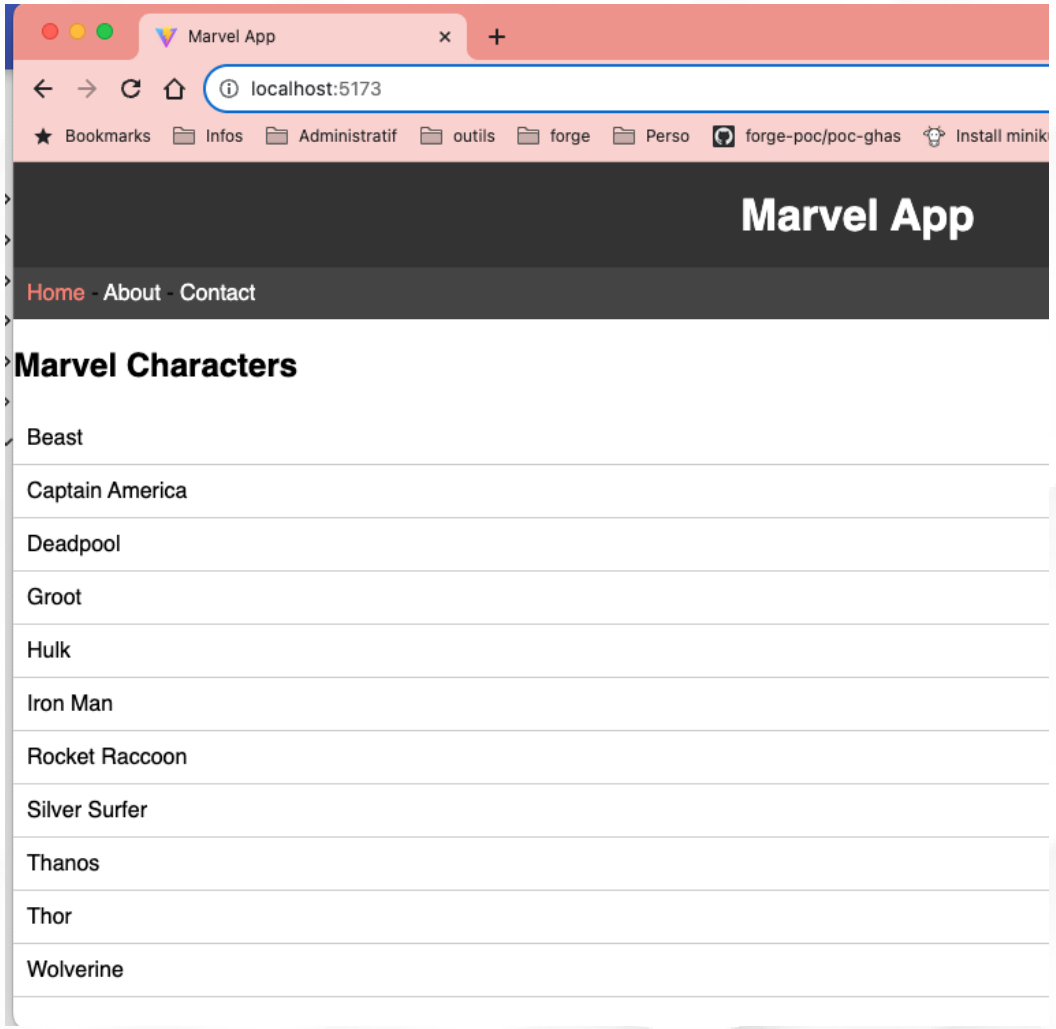
- Nous avons utilisé des balises `a` pour les liens de navigation
- Nous allons remplacer ces balises par des composants `NavLink` de **React Router**
  - Permet de gérer la navigation sans recharger la page
  - Permet de gérer les classes CSS pour les liens actifs

React Router - NavLinks (suite)

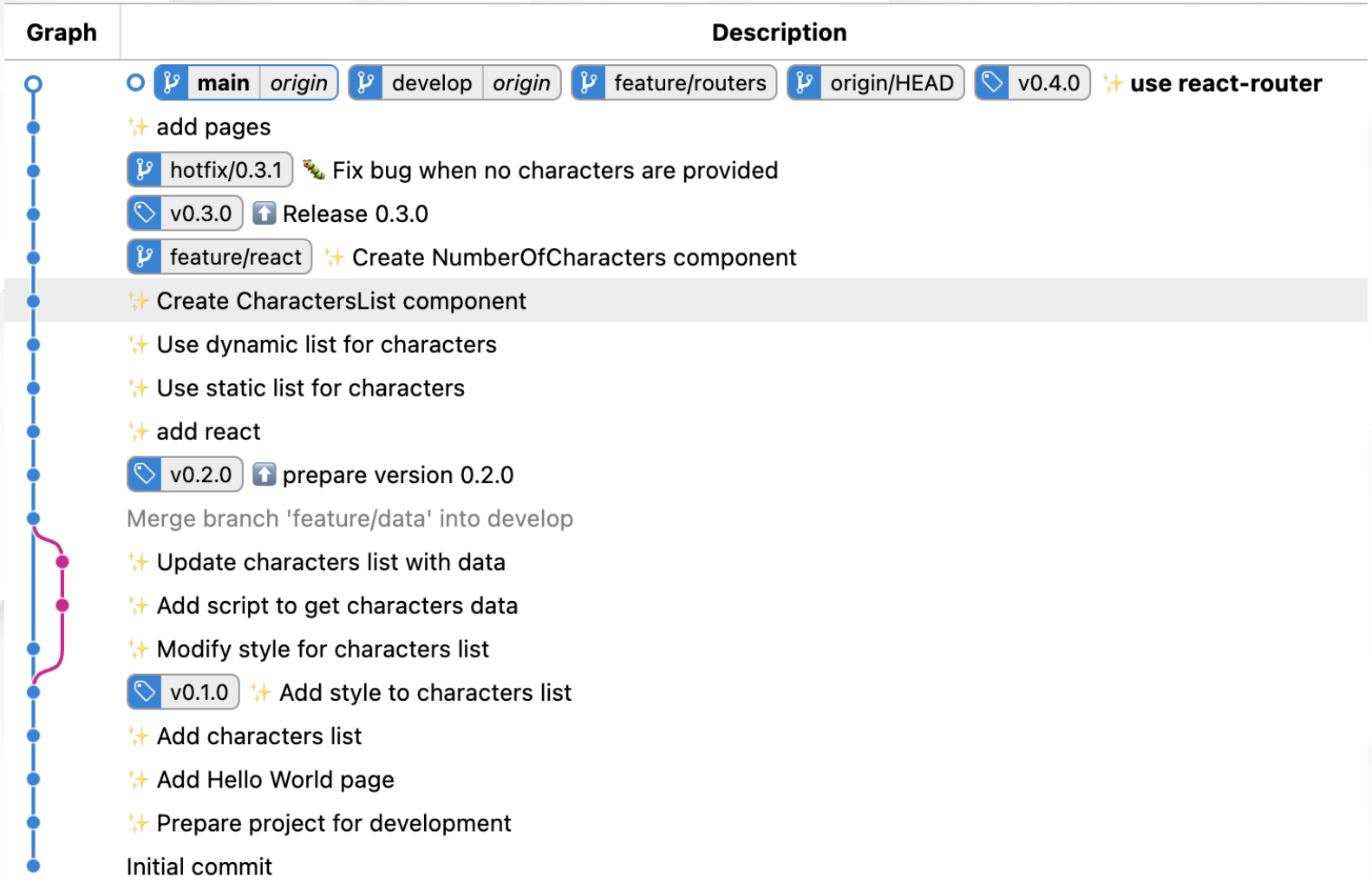
```
import React from 'react';
import { Outlet } from 'react-router';
import { NavLink } from 'react-router-dom';

const Layout = () => {
    return (
        <>
            <header>
                <h1>Marvel App</h1>
                <nav>
                    <NavLink to="/">Home</NavLink> - <NavLink to="/about">About</NavLink> - <NavLink to="/contact">Contact</NavLink>
                </nav>
            </header>
            <main>
                <Outlet />
            </main>
            <footer>
                <p>Marvel App - 2023</p>
            </footer>
        </>
    );
};

export default Layout;
```

Ajout de css pour améliorer l'affichage

- Remplacer le contenu du fichier `App.css` afin d'améliorer l'affichage de l'application
- Commenter la ligne `import './index.css'` dans le fichier `main.jsx`
- Télécharger le fichier App.css



git - Etat final



Alexandre GIRARD - Conseiller en Nouvelles Technologies - alexandre.girard@maif.fr