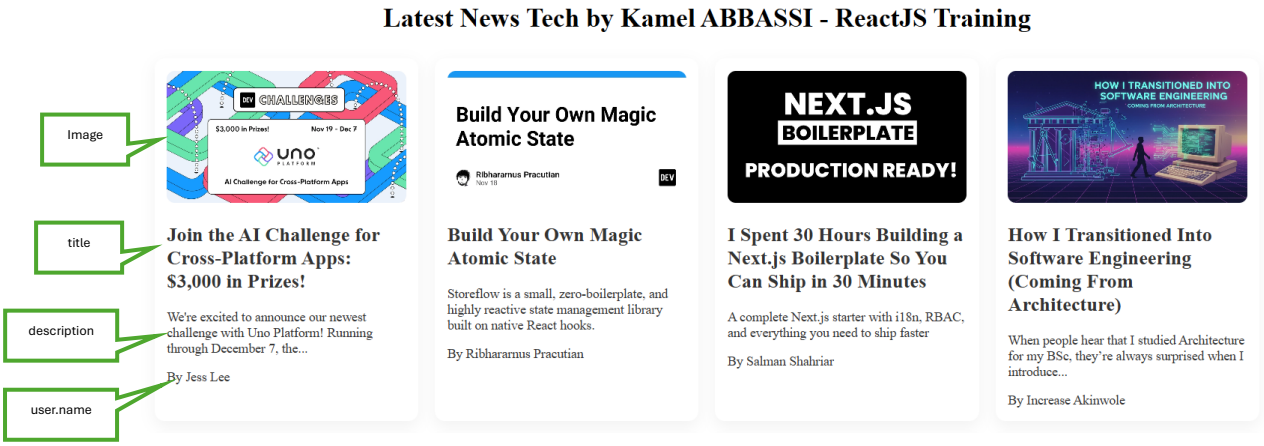
Atelier 07 NewsTech



Aperçu : https://newstech-lime.vercel.app/

Étape 1 : Préparer l'environnement React

Objectif pédagogique :
Savoir créer un projet React fonctionnel et structurer les fichiers.

Instructions :

1. Dans le dossier Ressources de cet atelier, copier le dossier **NewsTechApp** dans le dossier **C:\ReactProjects**
2. Ouvrir le dossier **C:\ReactProjects\NewsTechApp** avec votre éditeur de code
3. Démarrer docker et vérifier que les containers ne sont pas en exécution
4. Avec l'invite de commande, entre dans le dossier C:\ReactProjects\NewsTechApp et lancer la commande suivante :

   docker-compose up -d --build

5. Créez un projet React avec Vite ou Create React App.
   1. Accéder au NewsTechApp_container avec la commande suivante :

      docker exec -it NewsTechApp_container sh

   2. npm create vite@latest **.** --template

Deux traits avant template

```
/usr/src/app # npm create vite@latest . --template

> npx
> create-vite .

|
◇  Select a framework:
|  React
|
◇  Select a variant:
|  JavaScript
|
◇  Use rolldown-vite (Experimental)?:
|  No
|
◆  Install with npm and start now?
|  ● Yes / ○ No
└
```

3. Si le serveur Vite lancé, arrête-le (CTR + C)
4. Quitter le container

```
/usr/src/app # exit
```

5. Modifier le fichier vite.config.js

```
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'

// https://vitejs.dev/config/
export default defineConfig({
  plugins: [react()],
  server: {
    host: true,           // équivaut à --host
    watch: {
      usePolling: true,   // <== active le mode "polling"
      interval: 1000,     // <== vérifie les changements toutes les 1s
    },
  },
})
```

6. Puis arrêter et démarrer le container, directement avec le docker ou bien exécuter ces deux commandes

```
 docker-compose down
docker-compose up -d --build
```

7. Entre dans le container

   docker exec -it NewsTechApp_container sh

8. Lancer cette commande :

   npm run dev -- --host

9. Lester l'application sur l'url http://localhost:5173/

Étape 1 — Installer axios + router

Dans ton projet React vierge :

npm install axios react-router-dom

Étape 2 — Créer l'arborescence

cd src

mkdir components

mkdir pages

mkdir services

mkdir styles

Étape 3 — Créer App.jsx (version minimale)

src/App.jsx

```jsx
import { BrowserRouter, Routes, Route } from "react-router-dom";
import PostList from "./pages/PostList";
//import PostDetails from "./pages/PostDetails";

function App() {
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element={<PostList />} />
        {/* <Route path="/post/:id" element={<PostDetails />} /> */}
      </Routes>
    </BrowserRouter>
  );
}

export default App;
```

Étape 4 : src/pages/PostList.jsx

```jsx
export default function PostList() {
  return (
    <div>
      <h1>Articles</h1>
      <p>La liste sera affichée ici.</p>
    </div>
  );
}
```

👉 Teste maintenant dans le navigateur :
http://localhost:5173/ (ou http://localhost:3000)

Étape 5 PostList.jsx (mise à jour)

```
import { useState, useEffect } from "react";
import axios from "axios";

export default function PostList() {
  const [posts, setPosts] = useState([]);

  useEffect(() => {
    axios.get("https://dev.to/api/articles")
      .then(res => setPosts(res.data))
      .catch(err => console.log(err));
  }, []);

  return (
    <div>
      <h1>Articles</h1>

      {posts.length === 0 && <p>Chargement...</p>}

      {posts.map((p) => (
        <div key={p.id}>
          <h3>{p.title}</h3>
        </div>
      ))}
    </div>
  );
}
```

👉 À ce stade :

Tu vois les titres → C'est volontairement simple.

Étape 6 : PostList.jsx (card + image)

```
import { useState, useEffect } from "react";
import axios from "axios";
import { Link } from "react-router-dom";
//import "../styles/PostList.css";

export default function PostList() {
  const [posts, setPosts] = useState([]);

  useEffect(() => {
    axios.get("https://dev.to/api/articles")
      .then(res => setPosts(res.data))
      .catch(err => console.log(err));
  }, []);

  return (
    <div className="post-list-container">
      <h1>Articles</h1>

      <div className="post-grid">
        {posts.map((p) => (
          <Link to={`/post/${p.id}`} key={p.id} className="post-card">
            <img src={p.social_image} alt={p.title} />
            <h2>{p.title}</h2>
            <p>{p.description}</p>
            <span>By {p.user.name}</span>
          </Link>
        ))}
      </div>
    </div>
  );
}
```

Étape 7 : src/pages/PostDetails.jsx

```
export default function PostDetails() {
  return (
    <div>
      <h1>Détails de l'article</h1>
    </div>
  );
}
```

Étape 8 : **PostDetails.jsx** (mise à jour) **Récupérer ID + fetch API**

```
import { useParams } from "react-router-dom";
import axios from "axios";
import { useEffect, useState } from "react";

export default function PostDetails() {
  const { id } = useParams();
  const [post, setPost] = useState(null);

  useEffect(() => {
    axios.get(`https://dev.to/api/articles/${id}`)
      .then(res => setPost(res.data))
      .catch(err => console.log(err));
  }, [id]);

  if (!post) return <p>Chargement...</p>;

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

Étape 9 :PostDetails.jsx

```
import { useParams } from "react-router-dom";
import axios from "axios";
import { useEffect, useState } from "react";
import "../styles/PostDetails.css";

export default function PostDetails() {
  const { id } = useParams();
  const [post, setPost] = useState(null);

  useEffect(() => {
    axios.get(`https://dev.to/api/articles/${id}`)
      .then(res => setPost(res.data))
      .catch(err => console.log(err));
  }, [id]);

  if (!post) return <p>Loading...</p>;

  return (
    <div className="details-container">
      <h1>{post.title}</h1>

      <img src={post.social_image} alt={post.title} />

      <h3>By {post.user.name}</h3>
```

```
      <div
        className="details-content"
        dangerouslySetInnerHTML={{ __html: post.body_html }}
      />
    </div>
  );
}
```

Étape 9 : src/styles/PostList.css

```
.post-list-container {
  width: 90%;
  margin: auto;
  padding: 20px;
}

h1 {
  text-align: center;
  margin-bottom: 30px;
}

.post-grid {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));
  gap: 20px;
}

.post-card {
  background: #fff;
  padding: 15px;
  border-radius: 12px;
  box-shadow: 0px 6px 18px rgba(0,0,0,0.05);
  text-decoration: none;
  color: #333;
  transition: .30s ease;
}

.post-card:hover {
  transform: scale(1.05);
}

.post-card img {
  width: 100%;
  height: 160px;
  border-radius: 10px;
  object-fit: cover;
}
```

Étape 10 src/styles/PostDetails.css

```
.details-container {
  width: 85%;
  margin: auto;
  padding: 20px;
}

.details-container h1 {
  text-align: center;
  margin-bottom: 20px;
}

.details-container img {
  width: 100%;
```

```
  border-radius: 12px;
  margin-bottom: 20px;
}

.details-content {
  background: white;
  padding: 20px;
  border-radius: 12px;
  box-shadow: 0px 6px 18px rgba(0,0,0,0.05);
}
```