



# ARGENT BANK



Utiliser une API pour un compte utilisateur bancaire avec React

# CONTEXTE

## Nouvelle banque qui démarre et essaie de percer dans le secteur

### OBJECTIF

Créer application WEB complète

Ce que doit faire l'application

**ARGENTBANK**

L'utilisateur :

consulte la page d'accueil,

se connecte au système,

se déconnecte du système,

consulte les informations de son profil une fois connecté,

peut modifier son profil et les modifications sont conservées dans la base de données.

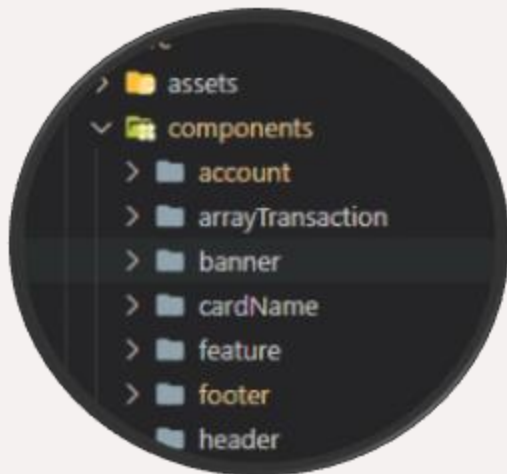


# Technologies - Outils utilisés pour développer l'application Argent Bank

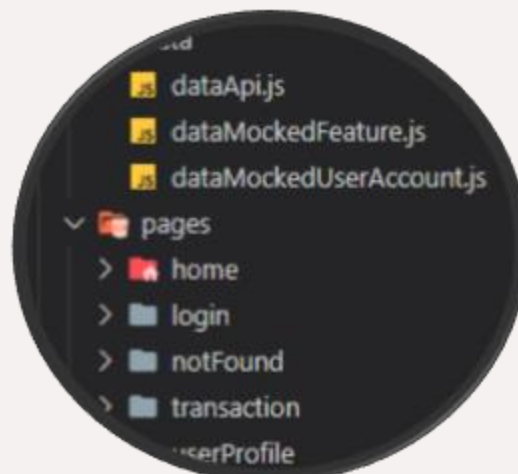
React version **18.2.0**,  
Redux/toolkit version **1.9.5**,  
sass version **1.62.1**,  
react-router-dom version **18.2.0**, et  
react-redux version **8.0.5**.

## Architectures . Sources = src

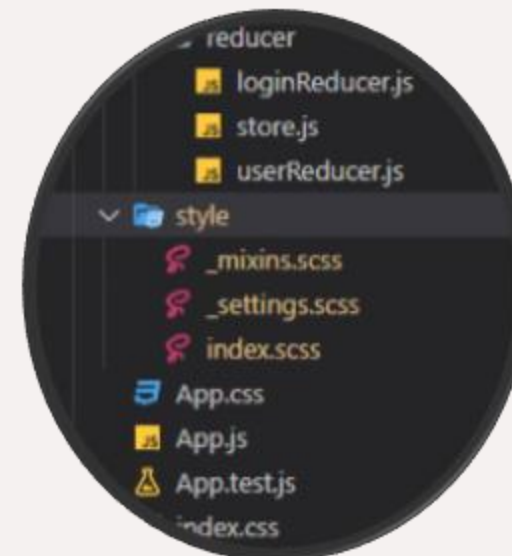
Components



Datas - Pages



Reducers - Styles

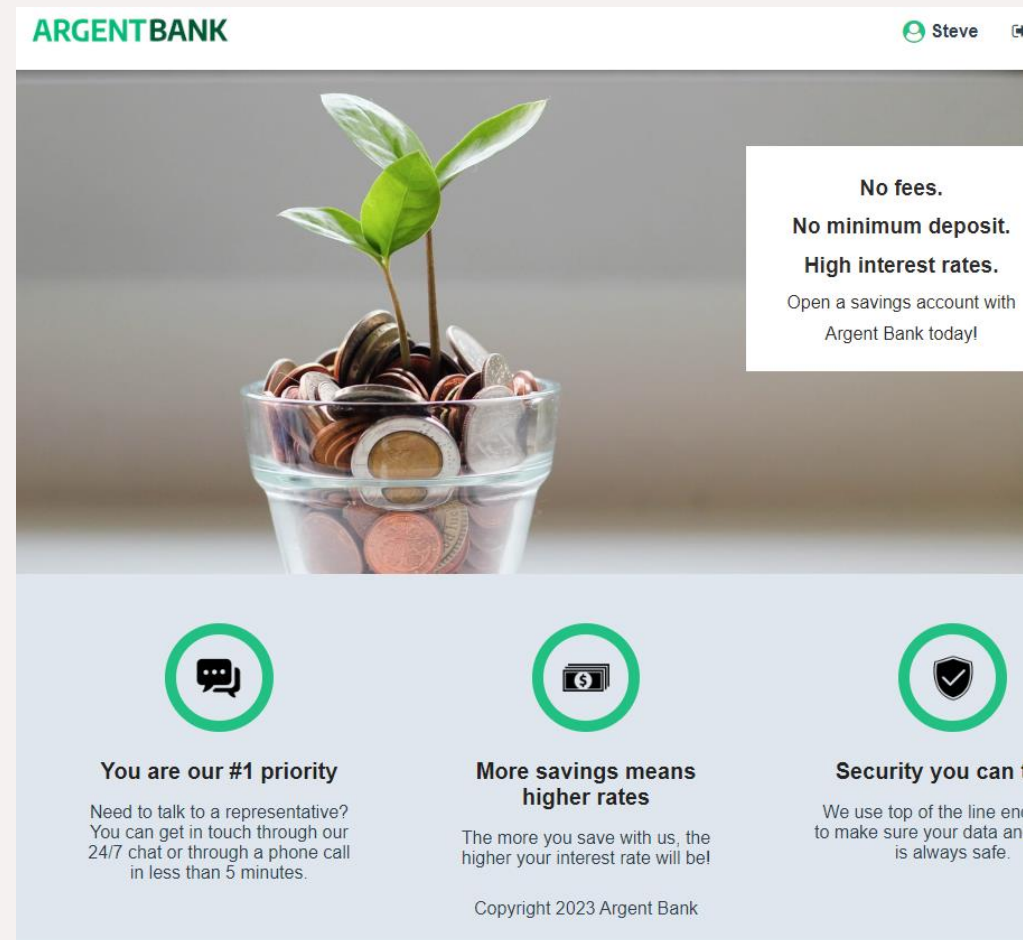


# PAGE HOME

Constituée du component **Banner** qui affiche une image d'une plante – Un texte l'accompagne pour mettre en avant les avantages d'ouvrir un compte épargne avec **ARGENTBANK**



Le component **Feature** qui utilise les datas du fichier dataMockedFeature - il utilise .map pour parcourir le [dataMOckedFeature] en rendu JSX  
Icône paragraphe et titre.



# Component HEADER



ARGENTBANK

 Sign In

Si l'utilisateur n'est pas connecté, le composant affiche un bouton "SignIn" qui redirige l'utilisateur vers la page de connexion.

ARGENTBANK

 Steve  Sign Out

Le composant **Header** représente l'entête de la page. Il affiche le logo **Argent Bank** et des boutons pour se connecter ou se déconnecter. Le composant utilise React Router et Redux (les hooks) pour gérer la navigation et l'état de connexion.

```
const Header = () => {
  const navigate = useNavigate();
  const dispatch = useDispatch();
  const isLoggedIn = useSelector((state) => state.login.connected);
  let firstName = useSelector((state) => state.profile.firstName);

  // Si le prénom n'est pas disponible dans le state, on le récupère
  if (localStorage.getItem("token") && !firstName) {
    firstName = localStorage.getItem("firstName");
  }

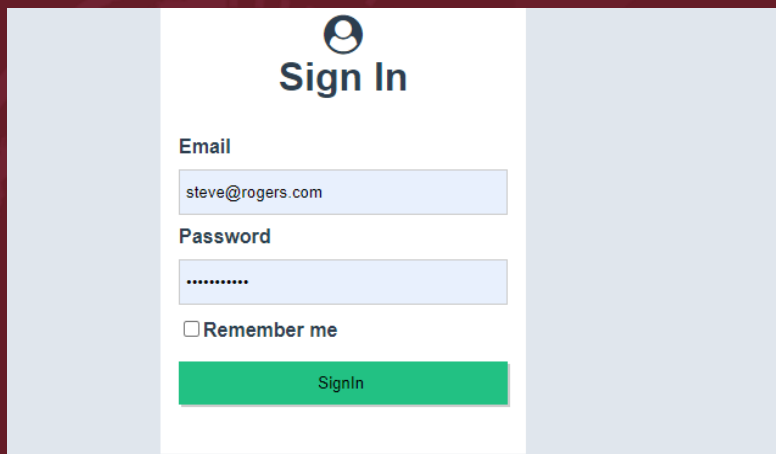
  /**
   * Gère la déconnexion de l'utilisateur.
   */
  const logout = () => {
    dispatch(getLoggedOut());
    navigate("/");
  };
};
```

# PAGE LOGIN

Le composant **Login** permet à l'utilisateur de se connecter à son compte en saisissant son adresse e-mail et son mot de passe.

Il contient un formulaire avec deux champs de saisie .

Lorsque le formulaire est soumis, la fonction **handleLogin** est appelée. Cette fonction appelle l'API login avec l'adresse e-mail et le mot de passe . Si la connexion réussit, le TOKEN est stocké dans le localStorage et l'utilisateur est redirigé vers la page /user. Sinon : gestion des erreurs



The image shows a mockup of a 'Sign In' page. It features a white central card on a light blue background. At the top of the card is a user icon and the text 'Sign In'. Below this are two input fields: 'Email' with the value 'steve@rogers.com' and 'Password' with masked characters '.....'. There is a checkbox labeled 'Remember me' and a green 'SignIn' button at the bottom.

```
const Login = () => {
  const navigate = useNavigate();
  const dispatch = useDispatch();
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [errors, setErrors] = useState({ email: null, password: null });

  useEffect(() => {
    if (localStorage.getItem("token")) {
      navigate("/user");
    }
  }, [navigate]);

  const handleLogin = async () => {
    const data = await login(email, password);
    console.log(data);
    if (data.body) {
      // Stocker le jeton dans localStorage
      localStorage.setItem("token", data.body.token);
      setErrors({ email: null, password: null });
      dispatch(getLoggedIn(data.body.token));
      navigate('/user');
    } else {
      const errorMessage = data.message.replace('Error: ', '');
      if (errorMessage.includes('email')) {
        setErrors({ email: errorMessage, password: null });
      } else if (errorMessage.includes('password')) {
        setErrors({ email: null, password: errorMessage });
      } else {
        setErrors({ email: errorMessage, password: null });
      }
    }
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    handleLogin();
  };
};
```



# Page USER

**ARGENTBANK**

SteveSign Out

Welcome back  
Steve Rogers

Edit Name

Argent Bank Checking (x8349)  
**\$2,082.79**  
Available Balance

View transactions

Argent Bank Savings (x6712)  
**\$10,928.42**  
Available Balance

View transactions

Argent Bank Credit Card (x8349)  
**\$184.30**  
Current Balance

View transactions

Welcome back

Steve

Rogers

Save

Cancel

La page **UserProfile** utilise les composants **CardName** et **AccountCard** pour afficher les informations du profil de l'utilisateur. Le composant **CardName** permet à l'utilisateur de visualiser et de modifier son prénom et son nom. Le composant **AccountCard** affiche les informations d'un compte sous forme de carte.

Voici un résumé simple que vous pourriez utiliser pour votre présentation: La page `UserProfile` affiche le profil de l'utilisateur en utilisant les composants `CardName` et `AccountCard`. Elle récupère les données du profil à partir d'une API lors du montage du composant et met à jour le store Redux avec ces données.

```
const AccountCard = ({dataAccount}) => {
  return(
    <section className="accountCard">
      <div className="accountCard__content">
        <h3 className="accountCard__title">{dataAccount.title}</h3>
        <p className="accountCard__amount">${dataAccount.amount}</p>
        <p className="accountCard__description">{dataAccount.description}</p>
      </div>
      <div className="accountCard__content">
        <NavLink to="/transactions" className="accountCard__view">
          <button className="accountCard__button">View transactions</button>
        </NavLink>
      </div>
    </section>
  );
}

export default AccountCard;
```



## 2 COMPONENTS

```
10 const CardName = () => {
11   const dispatch = useDispatch();
12
13   let firstName = useSelector((state) => state.profile.firstName) || localStorage.getItem("firstName");
14   let lastName = useSelector((state) => state.profile.lastName) || localStorage.getItem("lastName");
15
16   /**
17    * Fonction de soumission du formulaire.
18    * @param {Event} e - L'événement de soumission du formulaire.
19    */
20   const handleSubmit = async (e) => {
21     e.preventDefault();
22     e.target[0].value = '';
23     e.target[1].value = '';
24     console.log(`Submitting new values: ${firstName}, ${lastName}`);
25
26     try {
27       const token = localStorage.getItem("token");
28
29       const updatedProfile = await updateProfile(firstName, lastName, token);
30       console.log('Profile updated:', updatedProfile);
    }
  }
}
```

**AccountCard** affiche les informations d'un compte sous forme de carte. Titre, Montant la Description du compte et permet à l'utilisateur de se diriger vers la vue des transactions

**CardName** permet à l'utilisateur de visualiser et de modifier son firstName et son LastName. Il utilise les Hooks **UseDispatch** et **UseSelector** De Redux pour accéder au store.

Il contient un formulaire de 2 champs, la fonction handleSubmit est appelée qui met à jour le profil en appelant l'API

**UpdateDataProfil** se chargera de l'action de mise à jour dans le store




# Evolution de l'application (Implémentation non sollicitée)

## PAGE TRANSACTIONS


View transactions

Au click sur la page UserProfile view

ARGENTBANK



Steve





Sign Out

Argent Bank Checking (x8349)

\$ 2,082.79

Available Balance

	DATE	DESCRIPTION	AMOUNT	BALANCE
▼	20 juin 2022	Golden Sun Bakery	\$ 5.00	\$ 2082.79
▼	21 juin 2022	Golden Sun Bakery	\$ 10.00	\$ 2087.79
▼	22 juin 2022	Golden Sun Bakery	\$ 30.00	\$ 2117.79
▲	23 juin 2022	Golden Sun Bakery	\$ 40.00	\$ 2147.79
<div>Transaction Type: electronic</div> <div>Category: food </div> <div>Notes: Lorem ipsum dolor sit amet, consectetur adipiscing elit. </div>				
▼	24 juin 2022	Golden Sun Bakery	\$ 50.00	\$ 2187.79

# Synthèse - points clés du projet

Le projet **Argent Bank** utilise plusieurs composants React, ils affichent différentes parties de l'application **Header Banner Feature AccountCard CardName Login**.

Ils utilisent les hooks de React et Redux pour gérer l'état local et global, les effets de bord, les actions Redux et la navigation.

Les données du profil de l'utilisateur sont récupérées à partir d'une API et stockées dans le store Redux. Les composants peuvent accéder aux données du profil à partir du store Redux.

La page `UserProfile` utilise les composants `CardName` et `AccountCard` pour afficher les informations du profil de l'utilisateur.

Les prochaines étapes pour le projet dépendent des objectifs et des priorités de l'équipe. De nouvelles fonctionnalités seraient envisageables comme améliorer l'expérience utilisateur ou de travailler sur l'optimisation des performances.