

The logo for Remede Agency, with 'Remede' in orange and 'Agency' in red, both in a bold, sans-serif font.

Embauchée au sein de :

Agence spécialisée dans le développement d'applications web

Nouvelle banque qui démarre et essaie de percer dans le secteur

The logo for ARGENTBANK, with 'ARGENT' in green and 'BANK' in dark green, both in a bold, sans-serif font.

Utiliser une API pour un compte utilisateur bancaire avec React

Authentification des utilisateurs

Création d'une application web permettant aux clients de se connecter et de gérer leurs comptes et leur profil.

- **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.**

Éléments clés à spécifier à chaque endPoint de l'api

- La méthode HTTP à utiliser (POST, GET, PUT, DELETE)
 - La route à utiliser (/store/inventory)
- La description de ce à quoi correspond l'endPoint (récupère...)
- Les parametres pour tenir compte des differents scenarios
- Les differentes reponses avec les codes de status HTTP (ex : "200 OK - La requête a réussi")

En phase de conception

- ✓ de visualiser toutes leurs transactions pour le mois en cours
- ✓ de visualiser les détails d'une transaction dans une autre vue
- ✓ d'ajouter, de modifier ou de supprimer des informations sur une transaction

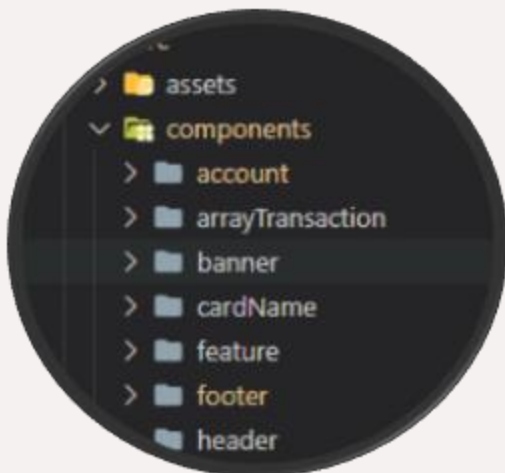


Technologies - Outils utilisés pour développer l'application Argent Bank

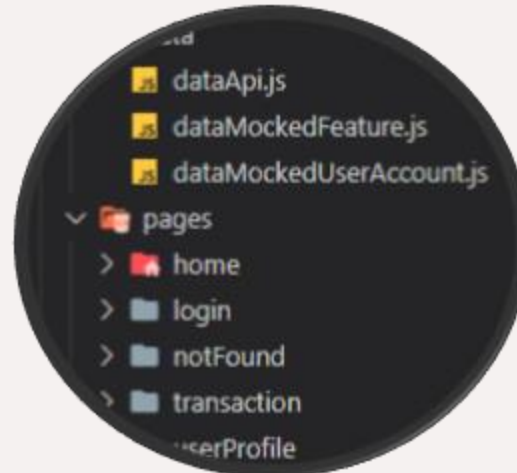
React version **18.2.0**,
Redux/toolkit version **1.9.5**,
sass version **1.62.1**,
react-router-dom version **18.2.0**, et
react-redux version **8.0.5**.

Architectures . Sources = src

Components



Datas - Pages



Reducers - Styles



Component HEADER

ARGENTBANK

Sign In



Utilisateur n'est pas connecté

ARGENTBANK



Steve

Sign Out



Utilisateur veut se deconnecter

Composant **Header** représente l'en-tête de la page, affiche le logo **Argent Bank** et des boutons pour se connecter ou se déconnecter .

Il utilise 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");
  }

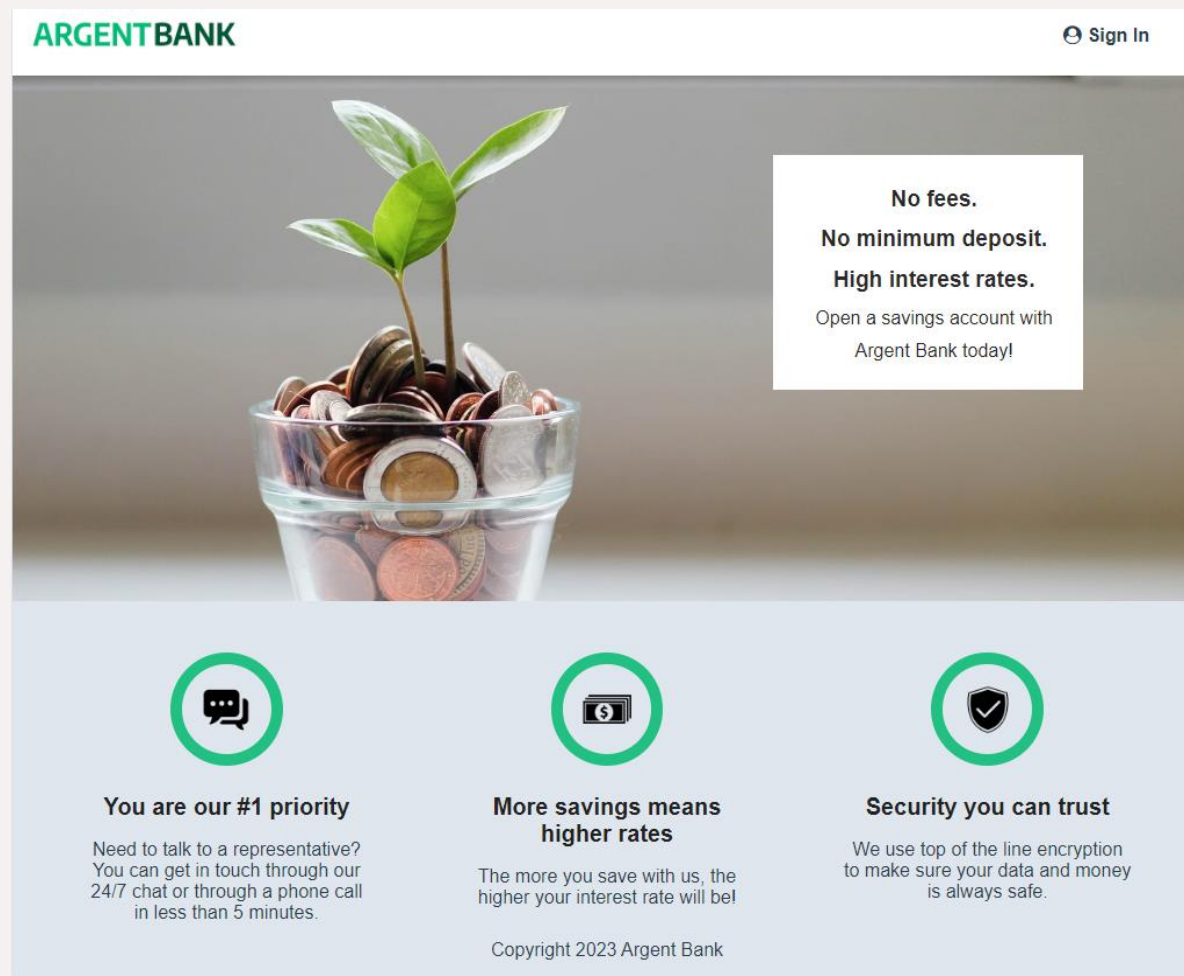
  /**
   * Gère la déconnexion de l'utilisateur.
   */
  const logout = () => {
    dispatch(getLoggedOut());
    navigate("/");
  };
};
```

PAGE HOME

Component **Banner** : affiche une image d'une plante - Un texte l'accompagne pour mettre en avant les avantages d'ouvrir un compte épargne avec **ARGENTBANK**



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



PAGE LOGIN

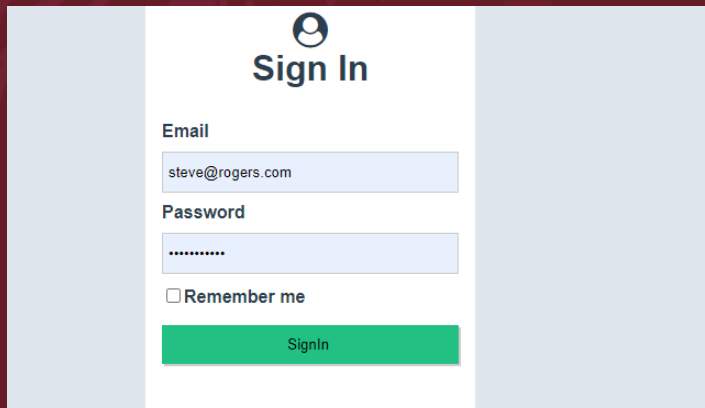
Comprend :

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

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



```
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();
  };
};
```


Fichier DataApi.js

Fn getProfile.js

```
const getProfile = async (token) => {
  // Envoie une requête POST à l'API pour obtenir le profil
  const response = await fetch("http://localhost:3001/api/v1/user/profile", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
      Authorization: `Bearer ${token}`,
    },
  });
  console.log(response);
  // Récupère les données de la réponse
  const data = await response.json();
  // Si la requête a réussi, renvoie les données du profil
  if (data.status === 200) {
    return data.body;
  } else {
    return "error";
  }
};
```

Fn updateProfile.js

```
const updateProfile = async (firstName, lastName, token) => {
  // Envoie une requête PUT à l'API pour mettre à jour le profil
  const response = await fetch("http://localhost:3001/api/v1/user/profile", {
    method: "PUT",
    headers: {
      "Content-Type": "application/json",
      Authorization: `Bearer ${token}`,
    },
    body: JSON.stringify({ firstName: firstName, lastName: lastName }),
  });
  // Récupère les données de la réponse
  const data = await response.json();
  console.log(data);
  // Si la mise à jour a réussi, renvoie les données du profil mis à jour
  if (data.status === 200) {
    return data.body;
  } else {
    throw new Error("Échec de la mise à jour du profil");
  }
};
```

3 Fonctions qui fournissent les fonctionnalités nécessaires pour se connecter, obtenir et mettre à jour le profil en utilisant une api

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

Page USER

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.

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;
```

AccountCard affiche les informations d'un compte sous forme de carte.

Il permet à l'utilisateur de se diriger vers la vue des transactions



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);
    }
  }
}
```

CardName permet à l'utilisateur de visualiser et de modifier son firstName et son LastName.

Il utilise les Hooks **UseDispatch** et **UseSelector** pour accéder au store.

Il contient un formulaire de 2 champs

La fonction **handleSubmit** est appelée et 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

SWAGGER

Documentation response 200

Curl

```
curl -X POST "http://localhost:3001/api/v1/user/login" -H "accept: application/json" -H "Content-Type: application/json" -d '{"email": "tony@stark.com", "password": "password123"}'
```

Request URL

http://localhost:3001/api/v1/user/login

Server response

Code	Details
200	<p>Response body</p> <pre>{ "status": 200, "message": "User successfully logged in", "body": { "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZiI6IjY0MzRkMTI3NjdlYzI3MDk3YzAzMTASNyIsIm1ldCI6MTY4NjQyNTU0NiwiZXN1IjoxNjg2NTExOTQ2fQ.QkdlU0F3L-Kzu-Ss12o8BUp6khLktXTSDxPHLprUE" }}</pre> <p>Response headers</p> <pre>access-control-allow-origin: * connection: keep-alive content-length: 245 content-type: application/json; charset=utf-8 date: Sat 10 Jun 2023 19:32:26 GMT etag: W/"f5-8Ut0TCzq5nFBU4f8gGeKrw0DLgo" keep-alive: timeout=5 x-powered-by: Express</pre>

Responses

Documentation Swagger

User Module



POST /user/login Login

POST /user/signup Signup

POST /user/profile User Profile API



PUT /user/profile User Profile API



200

Response body

```
{
  "status": 200,
  "message": "User successfully logged in",
  "body": {
    "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.e"
  }
}
```

Response headers

```
access-control-allow-origin: *
connection: keep-alive
content-length: 245
content-type: application/json; charset=utf-8
date: Sat10 Jun 2023 19:42:11 GMT
etag: W/"f5-Q1VY8rvqFpY3W+/i2vWRUzckgEs"
keep-alive: timeout=5
x-powered-by: Express
```

B. Phase 2 : Transactions

- Encore en phase de conception , nous mettons au point une fonctionnalité pour les transactions qui doit pouvoir permettre aux utilisateurs :
 - de voir leurs transactions pour le mois en cours
 - de visualiser les détails d'une transaction dans une autre vue
 - de modifier une transaction
 - de supprimer une transaction
 - de créer une nouvelle transaction

Fichier .yaml

Fournir un jeton JWT dans l'en-tête `Authorization` et l'ID du compte et le mois dans le corps de la requête.
Avec fichier YAML On peut effectuer pour la phase 2 Transactions :

Transaction

GET /user/profile/monthTransactions Get transactions of the provided month

GET /user/profile/transaction retrieve the provided transaction

POST /user/profile/addTransactionNote Create a new transaction note

PUT /user/profile/updateTransactionNote Update transaction note

DELETE /user/profile/DeleteTransactionNote delete transaction note

Transaction Type: electronic

Category: food

Notes: Lorem ipsum dolor sit amet, consectetur adipiscing elit.

RECAPITULATIF du PROJET

Action	EndPoint	Méthode
Login	/user/login	POST
Signup	/user/signup	POST
Profile	/user/profile	POST
Profile	/user/profile	PUT
UserAccounts	/user/profile/userAccounts	POST
Account	/user/profile/account	POST
Account	/user/profile/addAccount	POST
Account	/user/profile/updateAccount	PUT
MonthTransactions	/user/profile/monthTransactions	GET
Transaction	/user/profile/transaction	GET
Transaction	/user/profile/addTransactionNote	POST
Transaction	/user/profile/updateTransactionNote	PUT
Transaction	/user/profile/DeleteTransactionNote	DELETE

MongoDB Compass

argentBankDB.users

Documents

Aggregations

Schema

Explain Plan

Filter



Type a query: { field: 'value' }

+ ADD DATA

EXPORT DATA

```
{
  "_id": ObjectId('647dd12767ec27097c031097'),
  "email": "tony@stark.com",
  "password": "$2b$12$/z0J.5laWmgNawtY/awFweViupUv",
  "firstName": "tony",
  "lastName": "Stark",
  "createdAt": "2023-06-05T12:12:23.825+00:00",
  "updatedAt": "2023-06-22T06:54:35.258+00:00",
  "__v": 0
}
```

```
{
  "_id": ObjectId('647dd12767ec27097c031098'),
  "email": "steve@rogers.com",
  "password": "$2b$12$A3IU7AQs2vKlEUdHpdk04.SjUnyH",
  "firstName": "Steve",
  "lastName": "Rogers",
  "createdAt": "2023-06-05T12:12:23.830+00:00",
  "updatedAt": "2023-06-22T06:51:24.969+00:00",
  "__v": 0
}
```

Avant

argentBankDB.users

Documents

Aggregations

Schema

Explain Plan

Filter



Type a query: { field: 'value' }

+ ADD DATA

EXPORT DATA

```
{
  "_id": ObjectId('647dd12767ec27097c031097'),
  "email": "tony@stark.com",
  "password": "$2b$12$/z0J.5laWmgNawtY/awFweViupUv",
  "firstName": "banby",
  "lastName": "christophe",
  "createdAt": "2023-06-05T12:12:23.825+00:00",
  "updatedAt": "2023-06-22T08:47:33.101+00:00",
  "__v": 0
}
```

```
{
  "_id": ObjectId('647dd12767ec27097c031098'),
  "email": "steve@rogers.com",
  "password": "$2b$12$A3IU7AQs2vKlEUdHpdk04.SjUnyH",
  "firstName": "Steve",
  "lastName": "Rogers",
  "createdAt": "2023-06-05T12:12:23.830+00:00",
  "updatedAt": "2023-06-22T06:51:24.969+00:00",
  "__v": 0
}
```

Après

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 . Les composants peuvent accéder aux données du profil à partir du store .

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.