



Bérénger AKODO & Hadjar Adam ABAKAR

Projet TP React Native

Introduction

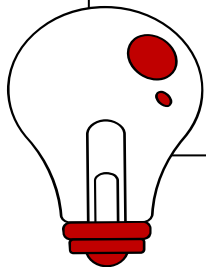
Votre tâche est de développer l'interface utilisateur d'une application bancaire en utilisant React Native. Cette application permettra aux utilisateurs de s'enregistrer, de se connecter, d'effectuer des virements, de commander des chèquiers et de consulter des informations de test.

Fonctionnalités à implémenter

1. Enregistrement d'un utilisateur
2. Connexion d'un utilisateur
3. Effectuer un virement
4. Commander un chéquier
5. Afficher une route de test

Le projet

NativeBank est une application mobile développée en React Native, offrant des fonctionnalités bancaires de base telles que l'ouverture de compte, la connexion, et la commande de chèquiers. Ce projet utilise un contexte API pour gérer les appels réseau et assurer la communication avec un backend.



Fonctionnalités

Utilisateur



RegisterScreen


NativeBank

CRÉER UN COMPTE

Nom

Prénom

test@test.com

Mot de passe

Solde

CONNEXION

Ouverture de compte



login

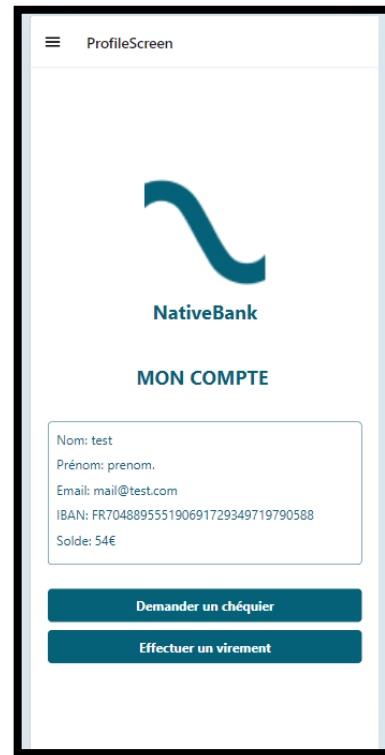

NativeBank

CONNEXION


Adresse email

Mot de passe

Ecran de connexion



ProfileScreen


NativeBank

MON COMPTE

Nom: test
Prénom: prenom.
Email: mail@test.com
IBAN: FR704889555190691729349719790588
Solde: 54€

Ecran mon compte

Fonctionnalités

Mouvement bancaire

≡ AddTransfer


NativeBank

EFFECTUER UN VIREMENT

-- Sélectionner un destinataire --

Montant

EFFECTUER LE VIREMENT

Virement

≡ CommandCheck


NativeBank

COMMANDER UN CHÉQUIER

Nom: test
Prénom: prenom.
Email: mail@test.com
IBAN: FR704889555190691729349719790588
Solde: 54€

Demander un chéquier pour moi-même

Commander un chéquier pour un autre utilisateur

Sélectionner un utilisateur

Saisir l'IBAN de l'utilisateur

COMMANDER POUR UN AUTRE UTILISATEUR

Commande de chéquier

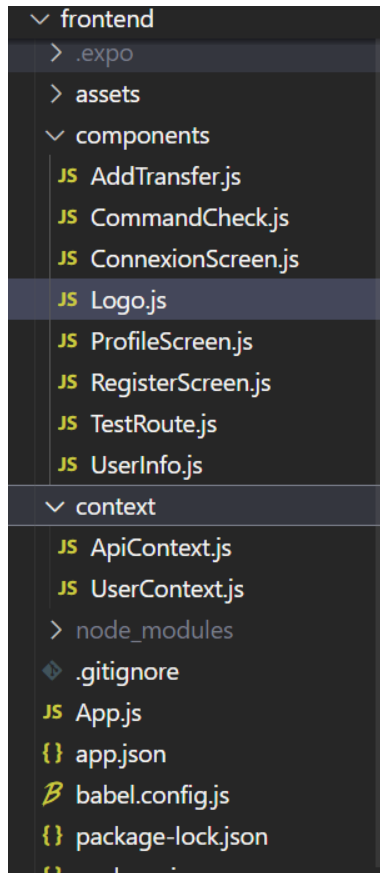
Fonctionnalités

Test



Test de l'API

Structure du projet



```
<ApiProvider>
  <UserProvider>
    <NavigationContainer>
      <Drawer.Navigator>
        <Drawer.Screen name="login" component={ConnexionScreen} options={{ drawerLabel: "Connexion" }}/>
        <Drawer.Screen name="RegisterScreen" component={RegisterScreen} options={{ drawerLabel: "Création de compte" }}/>
        <Drawer.Screen name="ProfileScreen" component={ProfileScreen} options={{ drawerLabel: "Profil" }}/>
        <Drawer.Screen name="AddTransfer" component={AddTransfer} options={{ drawerLabel: "Effectuer un virement" }}/>
        <Drawer.Screen name="CommandCheck" component={CommandCheck} options={{ drawerLabel: "Commander un chéquier" }}/>
        <Drawer.Screen name="TestRoute" component={TestRoute} options={{ drawerLabel: "Tester les routes de l'API" }}/>
      </Drawer.Navigator>
    </NavigationContainer>
  </UserProvider>
</ApiProvider>
```

```
export default function ProfileScreen({ route, navigation }) {
  const { user } = useUser();
  const [successMessage, setSuccessMessage] = useState("");
  const [showWelcomeMessage, setShowWelcomeMessage] = useState(false);

  useEffect(() => {
    if (route.params?.successMessage) {
      setSuccessMessage(route.params.successMessage);
      setShowWelcomeMessage(true);
    }
  }, [route.params?.successMessage]);

  useEffect(() => {
    if (showWelcomeMessage) {
      const timer = setTimeout(() => {
        setShowWelcomeMessage(false);
        setSuccessMessage("");
        navigation.setParams({ successMessage: null });
      }, 2000);
      return () => clearTimeout(timer);
    }
  }, [showWelcomeMessage]);
```

Merci

Projet disponible ici: [Dépôt Git](#)