**Diapositive 1 : Introduction au Projet**

* **Titre** : Projet Argent Bank - Application Web de Gestion de Profil Utilisateur
* **Texte** : Présentation du projet, objectifs (authentification, gestion de profil) et technologies utilisées (React, Redux, API RESTful).

**Diapositive 2 : Choix de l'Outil de Build**

* **Titre** : Pourquoi utiliser Vite ?
* **Texte** :
  + **Vite** est une alternative moderne et performante à Create-React-App.
  + **Avantages** : Compilation rapide, support des modules ES, optimisation automatique.
* **Exemple de commande** :

bash

npm create vite@latest my-react-app -- --template react

**Diapositive 3 : Utilisation de Redux**

* **Titre** : Gestion de l'État avec Redux
* **Texte** :
  + **Redux** est utilisé pour gérer l'état global de l'application.
  + **Avantages** : Prévisibilité et facilité de débogage.
* **Exemple de code** (userSlice.js) :

javascript

**import** { createSlice } **from** '@reduxjs/toolkit';

**const** userSlice = createSlice({

name: 'user',

initialState: {

user: **null**,

token: **null**,

error: **null**,

},

reducers: {

setToken(state, action) {

state.token = action.payload;

},

logoutUser(state) {

state.user = **null**;

state.token = **null**;

state.error = **null**;

},

},

extraReducers: (builder) => {

builder

.addCase(getUserProfile.fulfilled, (state, action) => {

state.user = action.payload;

state.error = **null**;

})

.addCase(getUserProfile.rejected, (state, action) => {

state.error = action.payload;

});

},

});

**export** **const** { setToken, logoutUser } = userSlice.actions;

**export** **default** userSlice.reducer;

**Diapositive 4 : Utilisation de createSlice**

* **Titre** : Pourquoi utiliser createSlice ?
* **Texte** :
  + **createSlice** simplifie la création de reducers et actions dans Redux.
  + **Avantages** : Réduit la quantité de code boilerplate.
* **Exemple de code** (userSlice.js) :

javascript

**import** { createSlice } **from** '@reduxjs/toolkit';

**const** userSlice = createSlice({

*// ...*

});

**Diapositive 5 : Gestion des Actions Asynchrones**

* **Titre** : Utilisation de createAsyncThunk
* **Texte** :
  + **createAsyncThunk** permet de créer des actions asynchrones pour les requêtes API.
  + **Avantages** : Simplifie la gestion des requêtes et des erreurs.
* **Exemple de code** (userSlice.js) :

javascript

**import** { createAsyncThunk } **from** '@reduxjs/toolkit';

**import** axios **from** 'axios';

**export** **const** getUserProfile = createAsyncThunk(

'user/getUserProfile',

**async** (\_, thunkAPI) => {

**try** {

**const** token = thunkAPI.getState().user.token;

**const** response = **await** axios.get('http://localhost:3001/api/v1/user/profile', {

headers: {

Authorization: `Bearer ${token}`,

},

});

**return** response.data.body;

} **catch** (error) {

**return** thunkAPI.rejectWithValue(

error.response?.data?.message || 'Erreur lors de la récupération des informations utilisateur'

);

}

}

);

**Diapositive 6 : Utilisation de React Router**

* **Titre** : Navigation avec React Router
* **Texte** :
  + **React Router** est utilisé pour gérer la navigation entre les pages.
  + **Avantages** : Simplifie la gestion des routes et des paramètres.
* **Exemple de code** (App.js) :

javascript

**import** { BrowserRouter, Routes, Route } **from** 'react-router-dom';

**import** Home **from** './pages/Home';

**import** Login **from** './pages/Login';

**import** User **from** './pages/User';

**function** App() {

**return** (

<BrowserRouter>

<Routes>

<Route path="/" element={<Home />} />

<Route path="/login" element={<Login />} />

<Route path="/user" element={<User />} />

</Routes>

</BrowserRouter>

);

}

**Diapositive 7 : Conclusion**

* **Titre** : Résumé du Projet
* **Texte** :
  + Présentation des technologies utilisées et des fonctionnalités implémentées.
  + **Avantages** : Scalabilité, maintenabilité et facilité d'utilisation.
* **Image** : Capture d'écran de l'application.

Cette structure devrait vous aider à présenter clairement votre projet et les choix techniques effectués. N'oubliez pas d'inclure des exemples de code pour illustrer chaque point !