TP3 - Firebase

Étape 1 : Installation de Node.js

Node.js est une plateforme permettant d'exécuter du code JavaScript côté serveur. Pour l'installer :

- Téléchargement de Node.js : Allez sur le site officiel de Node.js https://nodejs.org/ et téléchargez la version recommandée pour votre système d'exploitation.
- Installation :
  - Windows : Exécutez le fichier .msi téléchargé et suivez les instructions de l'assistant d'installation.
  - Mac : Exécutez le fichier .pkg téléchargé et suivez les instructions de l'assistant d'installation.
  - Linux : Utilisez le gestionnaire de paquets de votre distribution (ex. apt pour Ubuntu) : `bash sudo apt update sudo apt install nodejs npm`
- Vérification de l'installation : Après l'installation, ouvrez un terminal (ou l'invite de commandes) et tapez les commandes suivantes pour vérifier que Node.js et npm sont correctement installés :

  ```
  node -v
  npm -v
  ```

  Cela devrait afficher les versions installées de Node.js et npm.

Étape 2 : Créer son projet et installer les dépendances

- Créer un nouveau projet Node.js : Dans le terminal, créez un nouveau répertoire pour votre projet et accédez-y :

  ```
  mkdir mon-projet-firebase
  cd mon-projet-firebase
  ```

- Initialisez un projet Node.js avec npm. Cela créera un fichier package.json pour votre projet :

  ```
  npm init -y
  ```

- Installer le SDK Firebase Installez le SDK Firebase via npm en utilisant la commande suivante :

  ```
  npm install firebase
  ```

Étape 3 : Configuration de Firebase

- Créer un projet Firebase : Allez sur le site de la console Firebase : https://console.firebase.google.com/ Cliquez sur "Ajouter un projet" et

suivez les instructions pour créer un nouveau projet.

- Ajouter une application à votre projet Firebase : Dans la console Firebase, sélectionnez votre projet. Cliquez sur l'icône web "</>" pour ajouter une application web. Suivez les instructions et copiez les configurations Firebase fournies (API key, authDomain, projectId, etc.).
- Configurer Firebase dans votre projet : Créez un fichier firebase-config.js dans votre projet et ajoutez-y le code suivant, en remplaçant les valeurs par celles de votre configuration Firebase :

```
// firebase-config.js
import { initializeApp } from 'firebase/app';

const firebaseConfig = {
  apiKey: "YOUR_API_KEY",
  authDomain: "YOUR_AUTH_DOMAIN",
  projectId: "YOUR_PROJECT_ID",
  storageBucket: "YOUR_STORAGE_BUCKET",
  messagingSenderId: "YOUR_MESSAGING_SENDER_ID",
  appId: "YOUR_APP_ID"
};

const app = initializeApp(firebaseConfig);

export default app;
```

Pour savoir quoi renseigner, il faut créer une application sur votre projet firebase, puis dans "Vue d'ensemble", "ajouter une application". Ensuite mettez lui un nom et cliquer sur "</>" pour que firebase vous donne directement votre firebaseConfig.

Étape 4 : Lancer le serveur de développement

- Créez un fichier auth.js dans votre projet et ajoutez-y le code suivant pour vérifier que Firebase est correctement configuré :

```
// auth.js
import app from './firebase-config.js';

console.log('Firebase app initialized:', app);
```

- Pour que l'import fonctionne correctement, ajouter cette ligne à votre package.json:

```
"type": "module"
```

- Dans le terminal, exécutez la commande suivante pour lancer votre application :

```
node auth.js
```

Étape 5 : Inscrivez un utilisateur

Il faut vérifier que votre `firebase-config.js` est bien renseigné et que vous pouvez inscrire un utiliateur.

- Suivez https://firebase.google.com/docs/auth/web/start?hl=fr pour écrire et tester une fonction permettant de s'inscrire.

Étape 6 : Configurer un serveur express

Lorsque vous exécutez votre fichier javascript avec `node auth.js`, le code contenu dans le fichier `auth.js` sera lancé. Donc pour l'instant, nous n'avons aucune page HTML, simplement un programme exécutable.

- Installez la bibliothèque https://expressjs.com/fr/ avec `npm install express`.
- Créez un fichier `index.js` dans lequel vous lancer une application express "Hello World" comme ceci :

```
// index.js
import express from "express";

const app = express()
const port = 3000

app.get('/', (req, res) => {
  res.send('Hello World!')
})

app.listen(port, () => {
console.log('Server started at http://localhost:' + port);
})
```

- Lancez le serveur avec la commande `node index.js`.
- Vérifiez le fonctionnement en ouvrant un navigateur à l'adresse http://localhost:3000
- Créez un dossier `public/` et créez-y ce fichier `index.html`:

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Projet Firebase</title>
    <link rel="stylesheet">
```

```
</head>
<body>
    <h1>Projet Firebase</h1>
</body>
</html>
```

- Modifiez `index.js` pour renvoyer votre fichier HTML à la place de votre "Hello World" :

```
// Nouveaux imports
import path from "path";
import { dirname } from 'path';
import { fileURLToPath } from 'url';

// Recherche du chemin absolu vers le dossier actuel
const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);

// Pour remplacer le Hellow World
app.get('/', function(req, res) {
res.sendFile(path.join(__dirname, '/public/index.html'));
});
```

Étape 7 : Créez la page d'inscription

- Dans `public/` créez un fichier HTML `signup.html`, celui-ci doit contenir un formulaire d'inscription.
- Modifiez `index.js` en ajoutant une route `/signup` permettant d'accéder à cette page depuis http://localhost:3000/signup
- Utilisez votre fonction d'inscription de `auth.js` pour pouvoir inscrire un utilisateur lors de la soumission du formulaire.

Étape 8 : Créez la page de connexion

- Aidez-vous de votre inscription pour pouvoir maintenant faire une page de connexion fonctionnelle.

Étape 9 : Déconnexion

- Ajoutez un message sur votre page HTML quand l'utilisateur est connecté (exemple : "vous êtes connecté").
- Ajoutez un bouton permettant à l'utilisateur de se déconnecter.