Activité – Mise en œuvre d une API de hachage de mots de passe avec Bcrypt

Objectifs :

- Créer une API REST sécurisée pour la gestion des mots de passe utilisant le package bcrypt.

Étapes de réalisation

1. Initialisation du projet :

a) Créer un nouveau dossier pour le projet

```
mkdir password-hashing-api
cd password-hashing-api
```

b) Initialiser un projet Node.js

```
npm init -y
```

c) Installer les dépendances (express, nodemon et bcrypt)

```
npm install express
npm install nodemon
npm install bcrypt
```

d) Configurer package.json pour utiliser les modules ES

```
"main": "index.js",
"type": "module",
"scripts": {
    "start": "nodemon index.js"
  },
```

2. Création de la structure de l'API index.js :

a) Configurer Express & Bcrypt

```
import express from 'express';
import bcrypt from 'bcrypt';
const app = express();
```

b) Mettre en place le middleware pour parser le JSON

```
app.use(express.json());
```

c) Définir le port d'écoute

```
const PORT = 5000;
app.listen(PORT, () => {
    console.log(`Server running on port ${PORT}`);
      });
```

3. Implémentation du hachage simple (20 min)

1. Créer la route POST /hash, puis implémenter la logique de hachage avec bcrypt

```
// Exemple d'itinéraire pour démontrer le hachage direct
app.post('/hash', async (req, res) => {
    try {
        const { password } = req.body;
        const hashedPassword = await bcrypt.hash(password, 10);
        res.json({ hashedPassword });
    } catch (error) {
        res.status(500).json({ error: error.message });
    }
});
```

2. Tester la route avec Postman ou Powershell

```
Invoke-RestMethod -Uri "http://localhost:5000/hash" -Method Post -Body
'{"password":"P@ssw0rd"}' -ContentType "application/json"
```

4. Gestion des utilisateurs (25 min)

a) Créer la structure de données pour stocker les utilisateurs

```
// Stockage en mémoire (à remplacer par une base de données)
const users = [];
```

b) Implémenter la route POST /register

- Générer le salt
- Hacher le mot de passe
- Stocker l'utilisateur

```
// Enregistrez un nouvel utilisateur
app.post('/register', async (req, res) => {
    try {
        const { username, password } = req.body;

        // Générer du salt
        const salt = await bcrypt.genSalt(10);

        // Hacher le mot de passe avec le salt généré
        const hashedPassword = await bcrypt.hash(password, salt);

        // Enregistrer l'utilisateur
        users.push({
            username,
            password: hashedPassword
        });

        res.status(201).json({ message: 'Utilisateur sauvegardé' });
    } catch (error) {
```

```
            res.status(500).json({ error: error.message });
        }
    });
```

c) Tester l'enregistrement d'un utilisateur

Invoke-RestMethod -Uri "http://localhost:5000/register" -Method Post -Body '{"username":"test","password":"P@ssw0rd"}' -ContentType "application/json"

5. Authentification (20 min)

a) Implémenter la route POST /login et ajouter la logique de comparaison des mots de passe

```
// Itinéraire de connexion
app.post('/login', async (req, res) => {
    try {
        const { username, password } = req.body;

        // Chercher un utilisateur
        const user = users.find(u => u.username === username);
        if (!user) {
            return res
                   .status(404)
                   .json({ message: 'Utilisateur introuvable' });
        }

        // Comparer le mot de passe
        const isMatch = await bcrypt.compare(password, user.password);

        if (isMatch) {
            res.json({ message: ' Connexion réussie ' });
        } else {
            res.status(401).json({ message: ' Mot de passe invalide' });
        }
    } catch (error) {
        res.status(500).json({ error: error.message });
    }
});
```

b) Tester la connexion avec différents scénarios

Invoke-RestMethod -Uri "http://localhost:5000/login" -Method Post -Body '{"username":"test","password":"P@ssw0rd"}' -ContentType "application/json"

6. Réaliser les tâches suivantes

a) Hacher un mot de passe simple

b) Enregistrer un nouvel utilisateur

c) Tenter une connexion avec les bons identifiants

d) Tenter une connexion avec un mauvais mot de passe

e) Tenter une connexion avec un utilisateur inexistant