Compte Rendu TP1 Développement Avancé

Louis Prigneaux 304

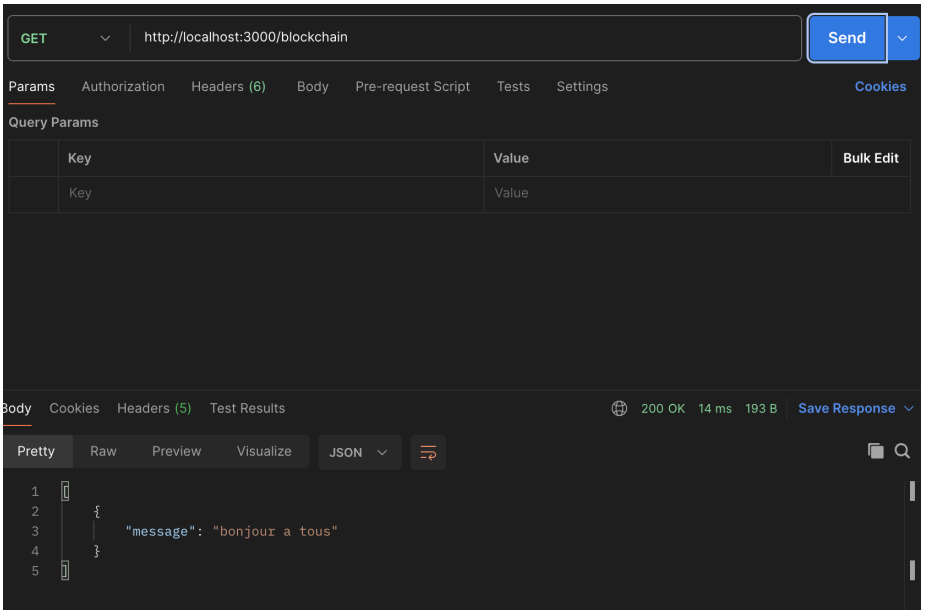
Etape 1 :

J'ai ajouté à chaque accès de endpoint sous forme de console.log :

console.log("GET request received on /blockchain");
console.log("POST request received on /blockchain");

Etape 2 :

- Création du fichier blockchain.json dans le dossier Data
- Changement du const Path : *const path = join(process.cwd(), "data", "blockchain.json");*



Etape 3 :

- Gestion de l'id à la création du bloc grâce à: id: uuidv4(),
- Modification de getDate :

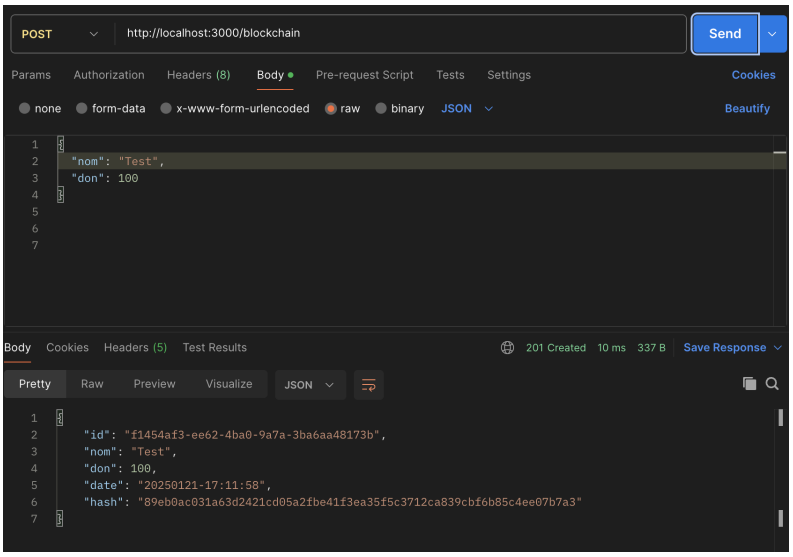
```
export function getDate() {
    const now = new Date();
    const year = now.getFullYear();
    const month = String(now.getMonth() + 1).padStart(2, '0'); // Mois de 0 à 11
    const day = String(now.getDate()).padStart(2, '0');
    const hours = String(now.getHours()).padStart(2, '0');
    const minutes = String(now.getMinutes()).padStart(2, '0');
    const seconds = String(now.getSeconds()).padStart(2, '0');

    return `${year}${month}${day}-${hours}:${minutes}:${seconds}`;
}
```

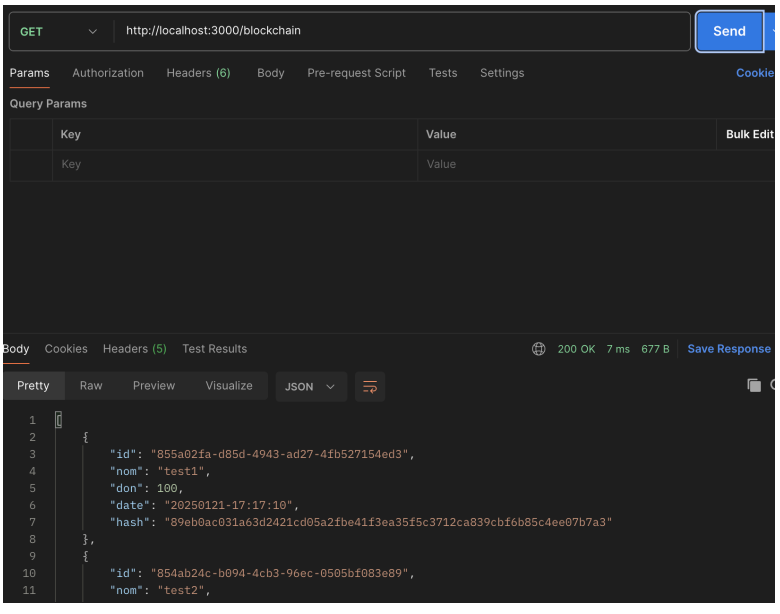
Etape 4 :

- Modification de findLastBlock :

```
export async function findLastBlock() {
  try {
    const blockchain = await findBlocks();
    return blockchain.length > 0 ? blockchain[blockchain.length - 1] : null;
  } catch (error) {
    console.error("Erreur lors de la recherche du dernier bloc :", error);
    throw error;
  }
}
```

- Hash généré de cette façon:
  const previousHash = generateHash(JSON.stringify(lastBlock));
  newBlock.hash = previousHash;





Etape 5 :

- Validation de la chaine lors des ajout :

```
/**
 * Vérifie l'intégrité de la blockchain.
 * @return {Promise<boolean>} Retourne true si la chaîne est valide, sinon false.
 */
export async function verifBlocks() {
    try {
        const blockchain = await findBlocks();

        if (blockchain.length <= 1) return true;

        for (let i = 1; i < blockchain.length; i++) {
            const currentBlock = blockchain[i];
            const previousBlock = blockchain[i - 1];

            const expectedHash = generateHash(JSON.stringify(previousBlock));

            if (currentBlock.hash !== expectedHash) {
                console.error(`Bloc invalide détecté à l'index ${i}.`);
                return false;
            }
        }

        return true;
    } catch (error) {
        console.error("Erreur lors de la vérification de l'intégrité de la blockchain :", error);
        throw error;
    }
}
```

```
Restarting 'src/server.js'
Server is listening on http://localhost:3000
GET request received on /blockchain
POST request received on /blockchain
Bloc valide !
```