Project Setup Documentation for Main Service and Public API

Overview

This repository consists of two services:

- Main Service: This service handles user authentication and authorization using JWT (JSON Web Token). It provides secure endpoints for user management, including creating and retrieving. JWT tokens are generated upon successful login, and they are used to secure user-specific routes.

- Public API: This service offers publicly accessible endpoints that require an API key for authorization. It allows third-party applications or clients to interact with certain functionalities (like retrieving user profiles and candidates) without needing to authenticate using a username or password.

**** _ JWT Authentication _****

JWT is used in the Main Service to securely authenticate users. When a user logs in with their credentials, the service generates a JWT token. This token is included in the response and can be used for subsequent requests to access secured endpoints. JWTs help in maintaining stateless authentication, meaning the server doesn't need to store session data.

**** _ API Key Generation _****

In the main service API keys are generated for authorized users during registration. The keys are created using the crypto module in Node.js, providing secure, random, and unique API keys. These keys are stored in the database and are used by the Public API to authorize incoming requests. The API key should be sent in the request header for accessing public endpoints.

**** _ Interaction Between Main Service and Public API _****

The Public API generates an API key upon user registration and uses it for authorization.

The Main Service securely manages user data, including generating JWT tokens for authenticated users.

Both services communicate with each other, with the Public API using the Main Service's data (like user profiles and candidates) based on valid API keys.

Dependencies

Main Service Dependencies:

Here's a breakdown of the dependencies and devDependencies Main Service:

- @prisma/client: Prisma Client for interacting with MongoDB.

- bcrypt: A library used for password hashing and validation.

- body-parser: Middleware to parse incoming request bodies.

- express: Web framework for building the API.

- jsonwebtoken: Used for generating and verifying JWT tokens for authentication.

- mongodb: MongoDB client for connecting to MongoDB (used with Prisma).

DevDependencies:

- prisma: Prisma ORM for working with the database, used during development to manage the database schema.

Service Setup: Main Service

1. **# Clone the Repository**

Start by cloning the project repository:

```bash
git clone <repository_url>
cd Recruit_Assign/main-service
```

2. Install Dependencies

To install all the necessary dependencies for the Main Service, use the following command:

```bash
npm install
```

This will install the dependencies listed in the dependencies and devDependencies sections of the package.json file, including express, bcrypt, jsonwebtoken, prisma, and others.

3. Set Up Environment Variables

Create a .env file in the root of the Main Service folder:

```bash
touch .env
```

Add the following variables to the .env file:

DATABASE_URL="mongodb://<your_mongodb_connection_string>"

JWT_SECRET="your_jwt_secret"

DATABASE_URL: The connection string to your MongoDB instance (local or cloud).

JWT_SECRET: A secret key used to sign and verify JWT tokens.

4. Initialize Prisma Client

Once the dependencies are installed, run the following command to generate the Prisma Client:

```bash
npx prisma generate
```

This will ensure that Prisma's generated client is ready for use in your application.

5. Run the Main Service

Start the Main Service by running the following:

```bash
npm run index.js
```

It will run on http://localhost:8000.

6. Testing the Main Service

You can now test the endpoints of the Main Service using Postman or cURL:

POST /register – Register a new user

POST /login – Log in and get a JWT token

POST /candidate – Add a new candidate

GET /candidate – Retrieve candidates for the logged-in user

****_ Service Setup: Public API _****

1. Clone the Repository

Clone the Public API repository:

```bash
cd ..
git clone <repository_url>
cd Recruit_Assign/public-api
```

2. Install Dependencies

To install the necessary dependencies for the Public API, run the following command:

```
npm install
```

3. Set Up Environment Variables

Create a .env file in the root of the Public API folder:

```
touch .env
```

Add the following variables to the .env file:

DATABASE_URL="mongodb://<your_mongodb_connection_string>"

Ensure that the DATABASE_URL is the same as the one used in the Main Service since both services share the same MongoDB database.

4. Initialize Prisma Client

Generate the Prisma Client for the Public API by running:

npx prisma generate

5. Run the Public API

Start the Public API on a different port , here it will run on 8001 port

npm run index.js

6. Testing the Public API

Test the following Public API endpoints:

GET /public/profile – Retrieve the user profile based on the API key.

GET /public/candidate – Retrieve candidates associated with the user (based on the API key).

_ Commands to Initialize the Project _

1. Install Dependencies (Main Service and Public API)

To install dependencies for both services, run:

cd main-service

npm install

cd ../public-api

npm install

2. Generate Prisma Client (Main Service and Public API)

Generate the Prisma client for both services:

cd main-service

npx prisma generate

cd ../public-api

npx prisma generate

3. Running Both Services

After setting up the environment variables and generating Prisma clients, run both services:

Main Service (on port 8000):

cd main-service

npm run index.js

Public API (on port 8001):

cd public-api

npm run index.js