URL Shortener

Description:

This project is a fully functional URL shortener service built using Node.js, Express, MongoDB, and JWT authentication. It allows users to create short links, manage them, and track analytics securely. Users must sign up and log in to access most features, ensuring each user's data is private and secure.

Features:

- User signup and login with JWT authentication
- Create short URLs (random or custom slug)
- Redirect to original URLs using short links
- Track visit history (IP, device, timestamp)
- disable and enable short URLs
- Update the URLs
- Search URLs in the database
- Secure backend using middleware authentication

Tech Stack:

- Node.js
- Express.js
- Mongoose
- JWT for authentication
- express-useragent for device tracking

Project Setup

Step 1 : Download the Repository:

Execute this command in your terminal or download the zip of this repository

```
git clone https://github.com/husainhakim/URL-Shortener.git
```

Step 2 : Navigate to the Folder:

Execute this command to move to the backend folder

```
cd backend
```

Step 3 : Download the Dependencies:

Execute this command to download all the dependencies

```
npm install
```

Step 4 : Make .env file:

Run this command

```
touch .env
```

Step 5 : Put the content into the .env file:

Here are all the variables which need to be put in the .env file

```
PORT=
DB_URL=
JWT_SECRET=
JWT_Expiry_Time=
```

Step 6 : Execute the Code:

```
npm start
```

Endpoints

POST /auth/signup -

Register a new user with email and password.
Example Request Body:

{

“Name” : “Husain Hakim”,
"email": "husain@gmail.com",
"password": "husain@1234"
}
Response : JWT token for authentication

POST /auth/login -

Login an existing user using email and password.
Example Request Body:
{
"email": "husain@gmail.com",
"password": "husain@1234"
}
Response : JWT token for authentication.

POST /url -

Create a new short URL (random or custom slug).
Requires Authorization header with Bearer token.

Example Request Body:

{ "url": "https://www.google.com",

"customSlug": "google"
}
Response : The shortId and redirect URL.

PUT /url/:shortId -

Update an existing short URL or change its custom slug.
Requires Authorization header with Bearer token.
Example Request Body:
{
"url": "https://www.example.com",

"customSlug": "newslug" // optional

}

POST /url/:shortId/disable -

Disable a short URL.
Requires Authorization header with Bearer token.
Response : Message confirming the URL is disabled.

POST /url/:shortId/enable -

Enable a previously disabled short URL.
Requires Authorization header with Bearer token.
Response : Message confirming the URL is enabled

GET /url/:shortId/analytics -

Get analytics for a specific short URL.
Requires Authorization header with Bearer token.
Response : Total clicks, visit history (IP, device, timestamp).

GET /url -

Get all short URLs for the logged-in user.
Requires Authorization header with Bearer token.
Response : List of URLs.

GET /search?q=<query>&analytics=<true|false> -

Search URLs by original URL, shortId, or custom slug.
Requires Authorization header with Bearer token.
Response : Matching URLs; optionally includes analytics

GET /:shortId -

Redirect to the original URL using the short link.
Public endpoint, does not require JWT.
Records visit in visit history (IP, device, timestamp)

Thank You