**Authentication and Password Reset API Documentation**

This documentation, created by Nithin Settibathula, provides details about the API endpoints and functionality implemented in the provided Node.js code for user authentication and password reset using Express, MongoDB, bcrypt, jwt, and nodemailer.

Table of Contents

1 Introduction

2 Endpoints

=> 1. Register

=> 2. Login

=> 3. Forget Password (Send Reset Link)

=> 4. Reset Password

=> 5. Get User Info

3 Middleware

=> 1. Authenticate Token

4 Running the Server

5 Conclusion

**Introduction**

This Node.js application, authored by Nithin Settibathula, serves as a basic user authentication system with features such as user registration, login, password reset, and retrieval of user information. The application uses a MongoDB database to store user data securely, bcrypt for password hashing, jwt for token-based authentication, and nodemailer for sending password reset emails.

**Endpoints**

1. Register

Endpoint: POST /register

Request Body:

email (String, required): User's email address.

password (String, required): User's password.

Response:

Success (Status Code: 201):

json

Copy code

{ "message": "User registered successfully" }

Failure (Status Code: 500):

json

Copy code

{ "error": "Internal Server Error" }

2. Login

Endpoint: POST /login

Request Body:

email (String, required): User's email address.

password (String, required): User's password.

Response:

Success (Status Code: 200):

json

Copy code

{ "token": "your\_generated\_token" }

Invalid Credentials (Status Code: 401):

json

Copy code

{ "error": "Invalid credentials" }

Failure (Status Code: 500):

json

Copy code

{ "error": "Internal Server Error" }

3. Forget Password (Send Reset Link)

Endpoint: POST /forget-password

Request Body:

email (String, required): User's email address.

Response:

User Not Found (Status Code: 404):

json

Copy code

{ "error": "User not found" }

Success (Status Code: 200):

json

Copy code

{ "message": "Reset link sent successfully" }

Failure (Status Code: 500):

json

Copy code

{ "error": "Internal Server Error" }

4. Reset Password

Endpoint: POST /reset-password

Request Body:

newPassword (String, required): User's new password.

Response:

Success (Status Code: 200):

json

Copy code

{ "message": "Password reset successful" }

Failure (Status Code: 500):

json

Copy code

{ "error": "Internal Server Error" }

5. Get User Info

Endpoint: GET /user-info

Request Headers:

Authorization (String, required): Bearer token obtained during login.

Response:

Success (Status Code: 200):

json

Copy code

{ "email": "user@example.com" }

Unauthorized (Status Code: 401):

json

Copy code

{ "error": "Unauthorized" }

Forbidden (Status Code: 403):

json

Copy code

{ "error": "Forbidden" }

**Middleware**

1. Authenticate Token

This middleware ensures that the request has a valid JWT token in the Authorization header.

Usage: Applied to protected routes (e.g., /user-info, /reset-password).

Behavior:

If the token is missing, returns 401 Unauthorized.

If the token is invalid, returns 403 Forbidden.

If the token is valid, sets req.user to the decoded user information and allows the request to proceed.

Running the Server

To run the server, execute the following command:

bash

Copy code

node your\_filename.js

Replace your\_filename.js with the name of the file containing your Node.js code.

**Conclusion**

This API provides basic user authentication and password reset functionality. Ensure to customize the code, including the MongoDB connection details, JWT secret key, and nodemailer configurations, based on your specific requirements and environment.