**1. Setup**

First, make sure you have the necessary packages installed:

npm init -y

npm install express pg bcrypt jsonwebtoken dotenv

**2. Create a .env File**

Create a .env file to store environment variables:

DATABASE\_URL=postgresql://username:password@localhost/db\_name

JWT\_SECRET=your\_secret\_key

**3. Database Connection**

Set up the PostgreSQL connection using the pg library:

javascript

// db.js

const { Pool } = require('pg');

require('dotenv').config();

const pool = new Pool({

connectionString: process.env.DATABASE\_URL,

});

module.exports = pool;

**4. User Model and Authentication Logic**

javascript

// authController.js

const bcrypt = require('bcrypt');

const jwt = require('jsonwebtoken');

const pool = require('./db');

require('dotenv').config();

const registerUser = async (req, res) => {

const { username, password } = req.body;

try {

// Check if user exists

const userExists = await pool.query('SELECT \* FROM users WHERE username = $1', [username]);

if (userExists.rows.length > 0) {

return res.status(400).json({ message: 'User already exists' });

}

// Hash the password

const hashedPassword = await bcrypt.hash(password, 10);

// Insert new user

const newUser = await pool.query(

'INSERT INTO users (username, password) VALUES ($1, $2) RETURNING \*',

[username, hashedPassword]

);

res.status(201).json({ message: 'User registered successfully', user: newUser.rows[0] });

} catch (error) {

console.error(error.message);

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

}

};

const loginUser = async (req, res) => {

const { username, password } = req.body;

try {

// Find user by username

const user = await pool.query('SELECT \* FROM users WHERE username = $1', [username]);

if (user.rows.length === 0) {

return res.status(400).json({ message: 'Invalid credentials' });

}

// Check password

const validPassword = await bcrypt.compare(password, user.rows[0].password);

if (!validPassword) {

return res.status(400).json({ message: 'Invalid credentials' });

}

// Generate JWT

const token = jwt.sign({ id: user.rows[0].id }, process.env.JWT\_SECRET, { expiresIn: '24h' });

res.json({ token });

} catch (error) {

console.error(error.message);

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

}

};

module.exports = { registerUser, loginUser };

**5. Express Setup and Routes**

javascript

// server.js

const express = require('express');

const { registerUser, loginUser } = require('./authController');

require('dotenv').config();

const app = express();

app.use(express.json());

// Routes

app.post('/api/register', registerUser);

app.post('/api/login', loginUser);

// Start the server

const PORT = process.env.PORT || 5000;

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

**6. Create the Users Table in PostgreSQL**

Run the following SQL command to create the users table:

sql

CREATE TABLE users (

id SERIAL PRIMARY KEY,

username VARCHAR(255) UNIQUE NOT NULL,

password VARCHAR(255) NOT NULL

);

**7. Testing the API**

You can test the API using tools like Postman or cURL.

* **Register a User**:

http

POST /api/register

Content-Type: application/json

{

"username": "user1",

"password": "password123"

}

* **Login a User**:

http

Copy code

POST /api/login

Content-Type: application/json

{

"username": "user1",

"password": "password123"

}