Login & Registration System – Overview

This project implements a secure, fullstack user authentication system using NestJS for both the backend API and server-side rendered frontend. The system allows users to register, log in, and access protected content using JWT (JSON Web Token) authentication. It integrates with a PostgreSQL database using TypeORM for data management. All user credentials are securely stored with bcrypt hashing, and API routes are protected with authentication guards. This setup provides a scalable foundation for any application requiring user management and secure access control.

Features:

- User registration & login with JWT authentication

- Secure endpoints using AuthGuard

- PostgreSQL database integration

- TypeORM-based User entity & repository

Backend Setup and run instructions

1. Requirements

- Node.js (v18+)

- PostgreSQL installed & running

- VS Code

- Postman

- pgAdmin

2. Create project:

$ nest new backend

3. Install packages:

$ npm install @nestjs/typeorm typeorm pg bcrypt @nestjs/jwt passport-jwt passport @nestjs/passport

4.Database Configuration:

- Host: localhost

- Port: 5432

- Username: postgres

- Password: root

- Database: Nazmul

5.Run the Backend:

$ cd backend

$ npm install

$ npm run start:dev

API will be available at: http://localhost:3000

6.API Usage – Test via Postman

Register a New User

- Method: POST

- URL: http://localhost:3000/auth/register

- Body (JSON):

```
{
  "email": "rahat@gmail.com",
  "password": "12456"
}
```

Login

- Method: POST

- URL: http://localhost:3000/auth/login

- Body (JSON):

```
{
```

"email": "rahat@gmail.com.",

"password": "123456"

}

- On success, receive an `access_token` (JWT).

Frontend Setup – Using NestJS (SSR)

-Server-Side Rendered views (login, register, dashboard)

- Simple authentication logic with JWT stored in local storage

- Axios/fetch integration to communicate with backend

- Styled components and prebuilt UI elements provided by DaisyUI

Run the Frontend:

$ cd frontend

$ npm install

$ npm run start:dev

Frontend will run at: http://localhost:3001

User Flow:

1. Open browser at http://localhost:3001/register

2. Enter email and password, submit

3. On success → redirected to login

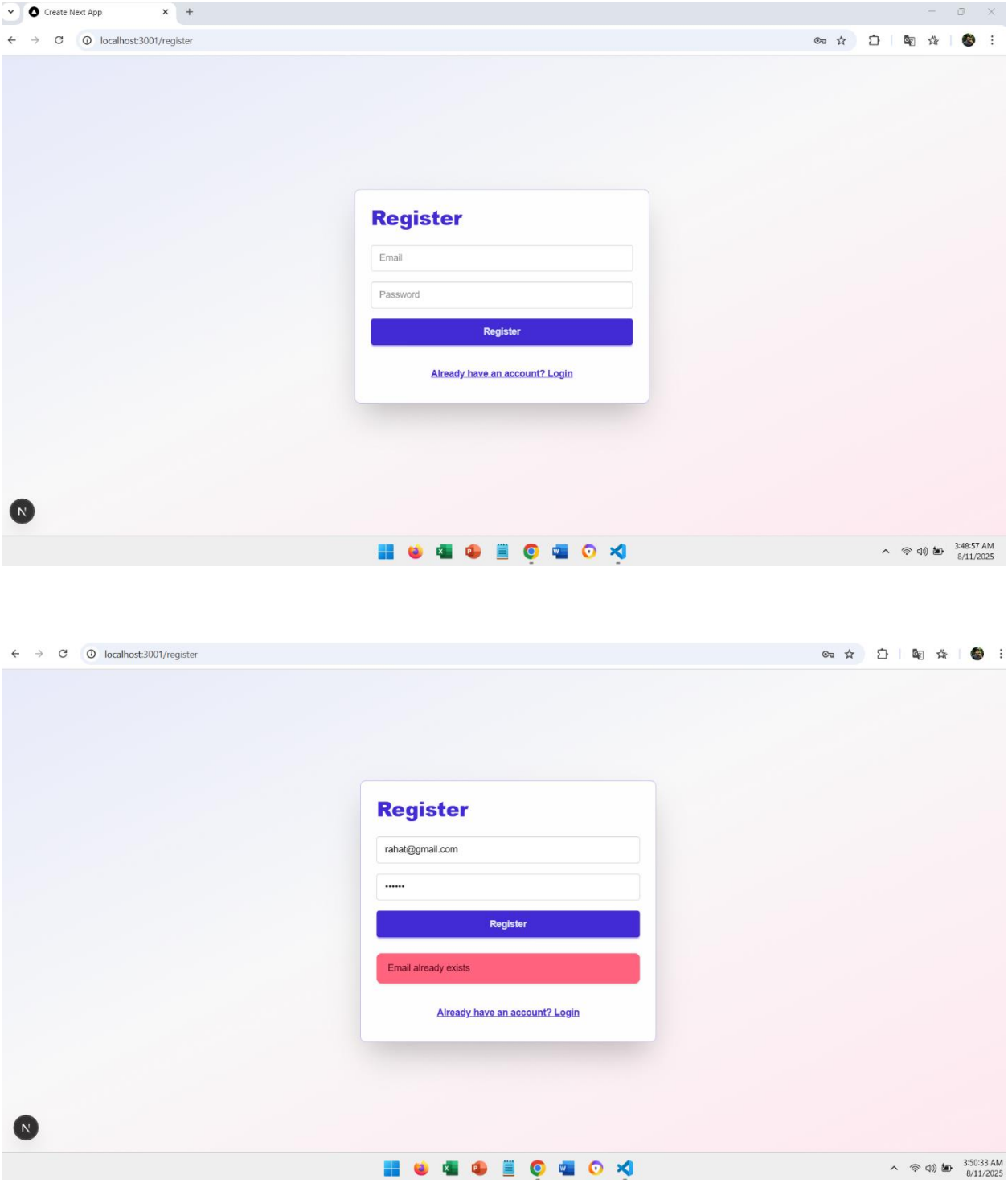
4. After login, dashboard will show:

   - Welcome message

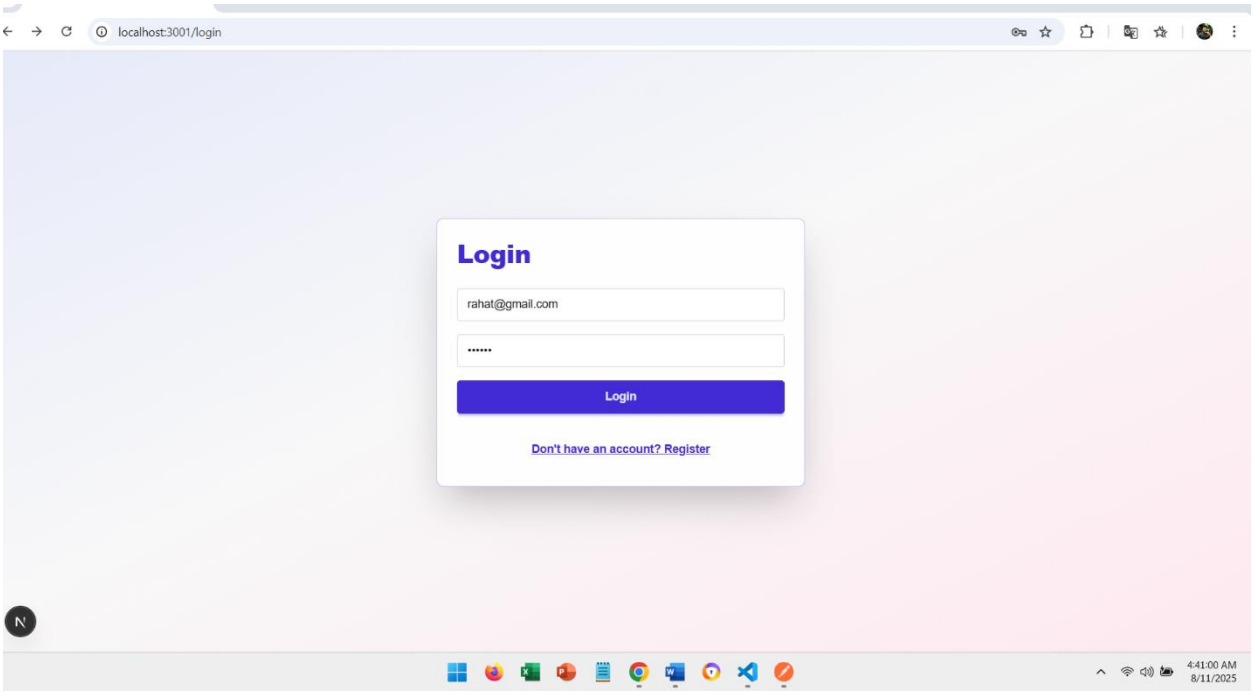
   - User info from backend

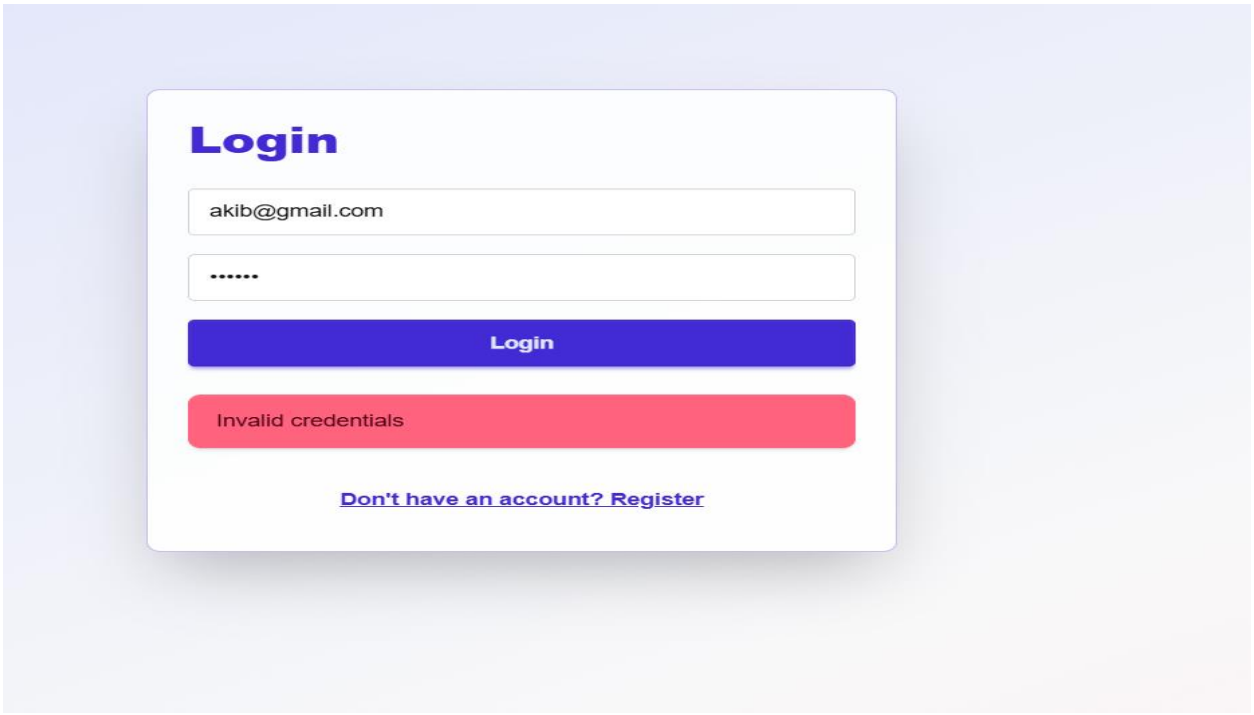
   - Logout button

Screenshots:

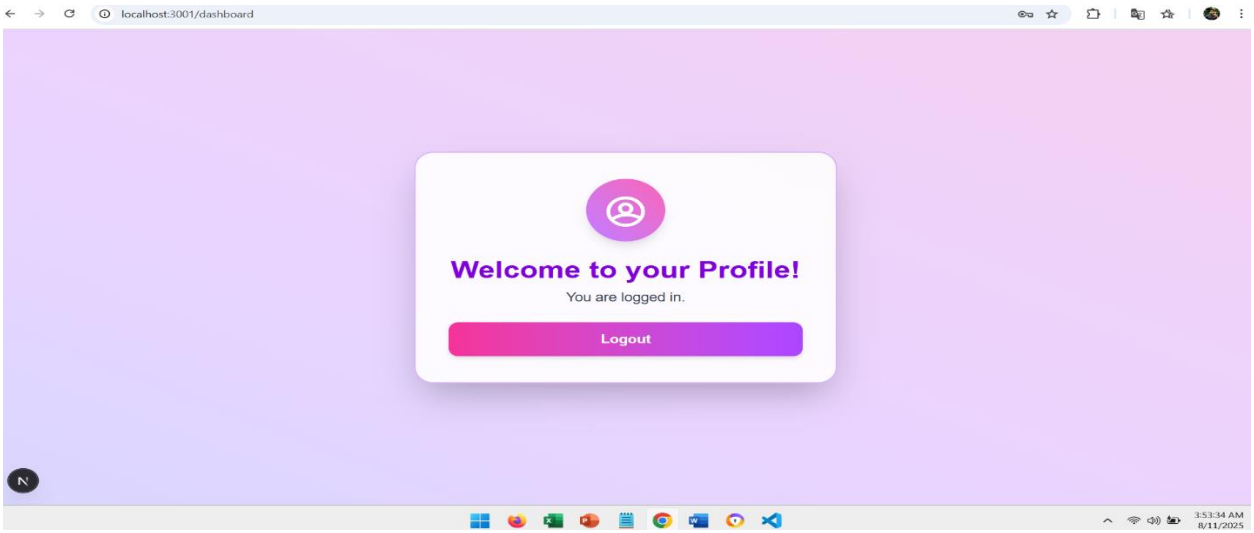


Registration Page


localhost:3001/login
Login
rahat@gmail.com
Login
Don't have an account? Register
4:41:00 AM
8/11/2025

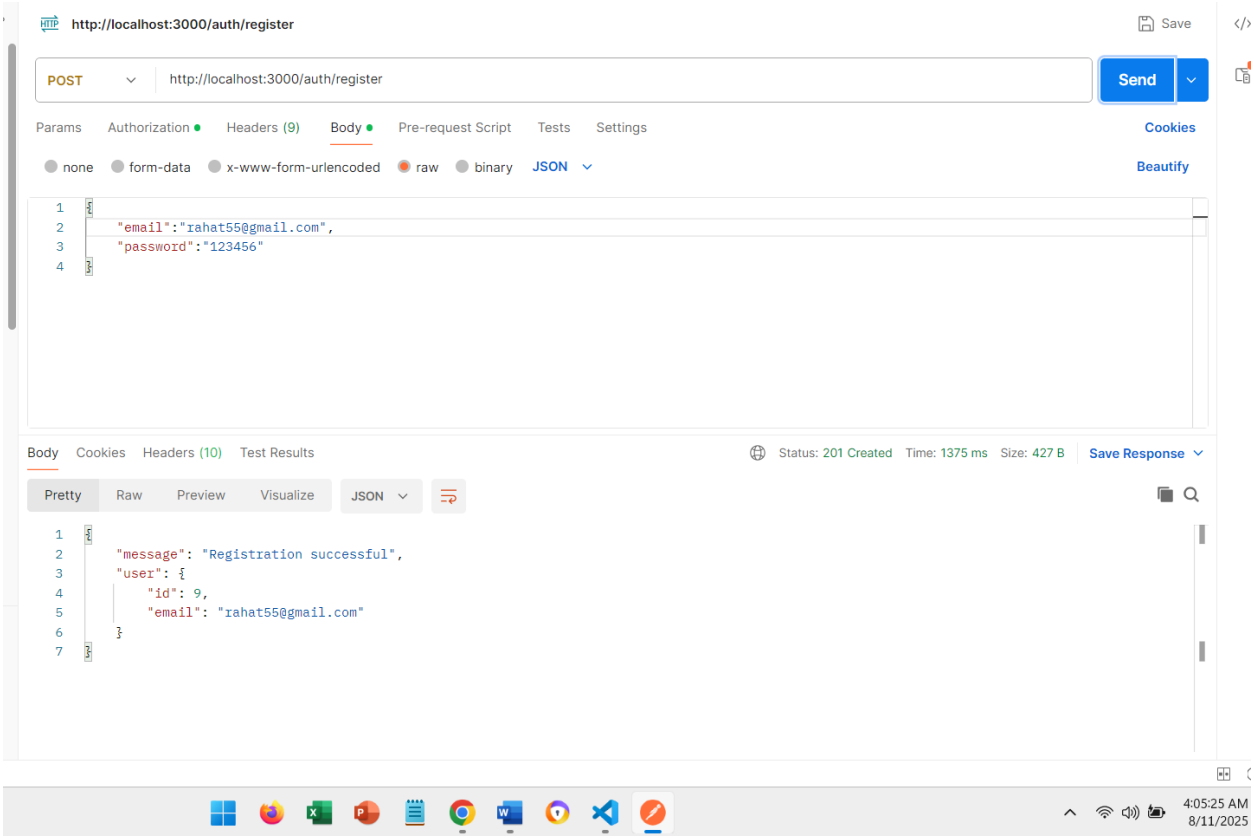

Login
akib@gmail.com
Login
Invalid credentials
Don't have an account? Register

Login Page

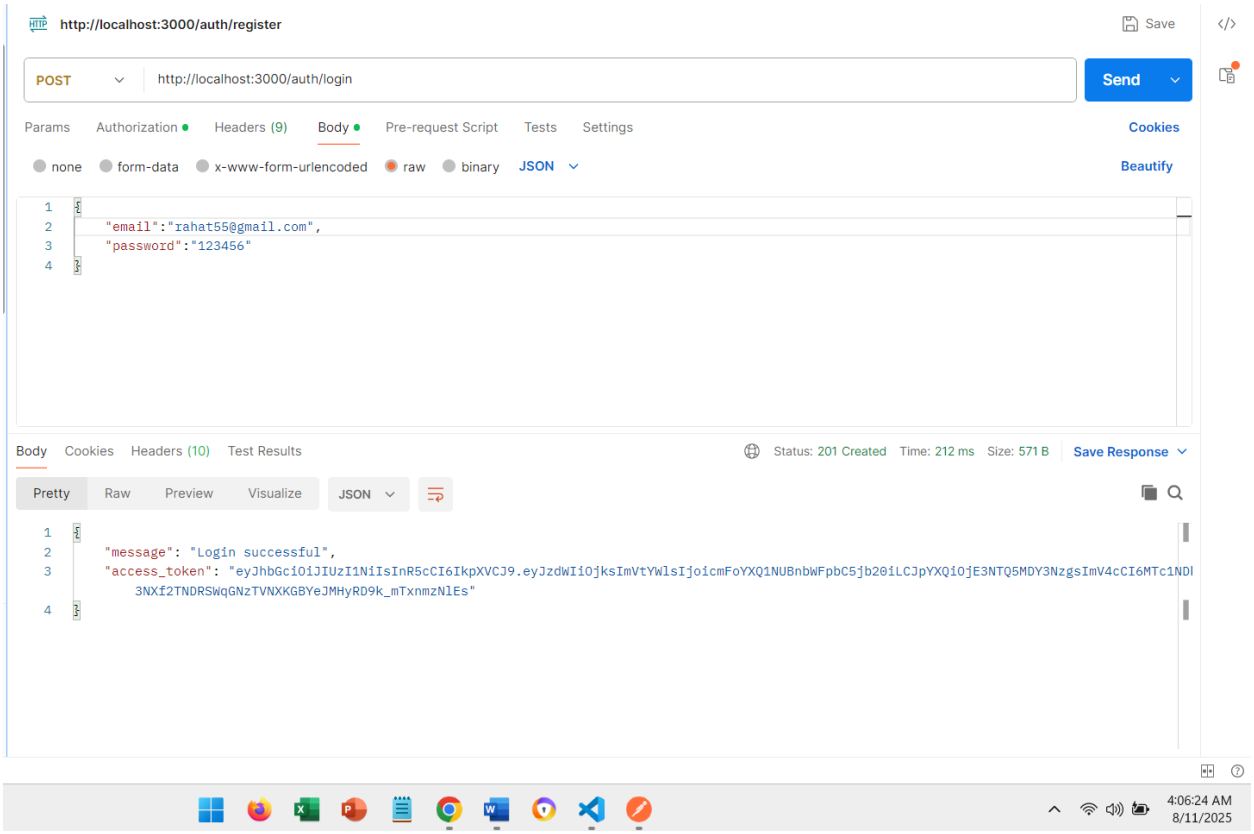


Dashboard

Backend API..



Registration


http://localhost:3000/auth/register
Save
POST
http://localhost:3000/auth/login
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
JSON
Beautify
1
2 "email":"rahat55@gmail.com",
3 "password":"123456"
4
Body
Cookies
Headers (10)
Test Results
Status: 201 Created
Time: 212 ms
Size: 571 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
1
2 "message": "Login successful",
3 "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOjksImVtYWlsIjoicmFoYXQ1NUBnbWFpbC5jb20iLCJpYXQiOjE3NTQ5MDY3NzgsImV4cCI6MTc1NDI
3NXf2TNDRSWqGNzTVNXKGBYeJMHyRD9k_mTxnmzNlEs"
4
4:06:24 AM
8/11/2025

Login