Secure Backend Development Assignment Report

Name: Aditya Chavhan

Course: 6th Semester Honours

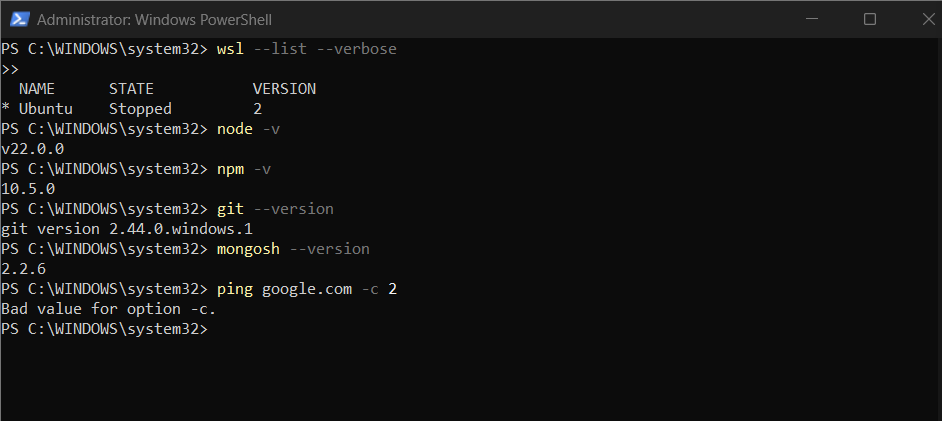
Assignment: Secure Backend Development using WSL, Express.js, and MongoDB Atlas

Submission Date: May 15, 2025

------------------------------------------------------------

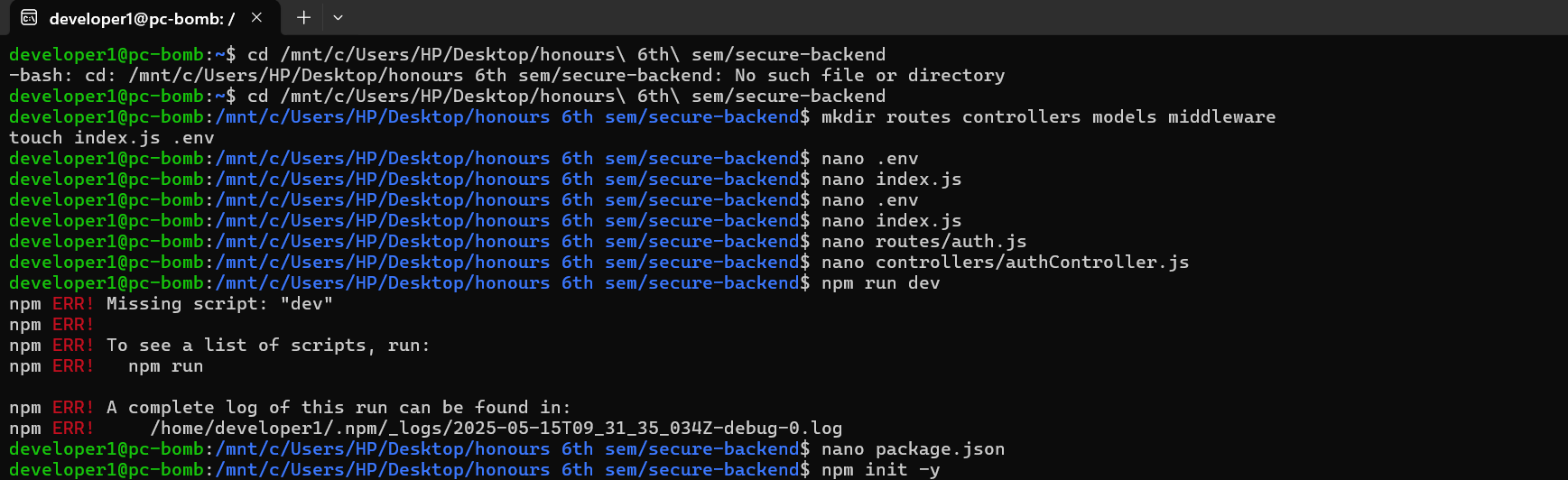
# Step 1: Environment Setup (WSL + Ubuntu)

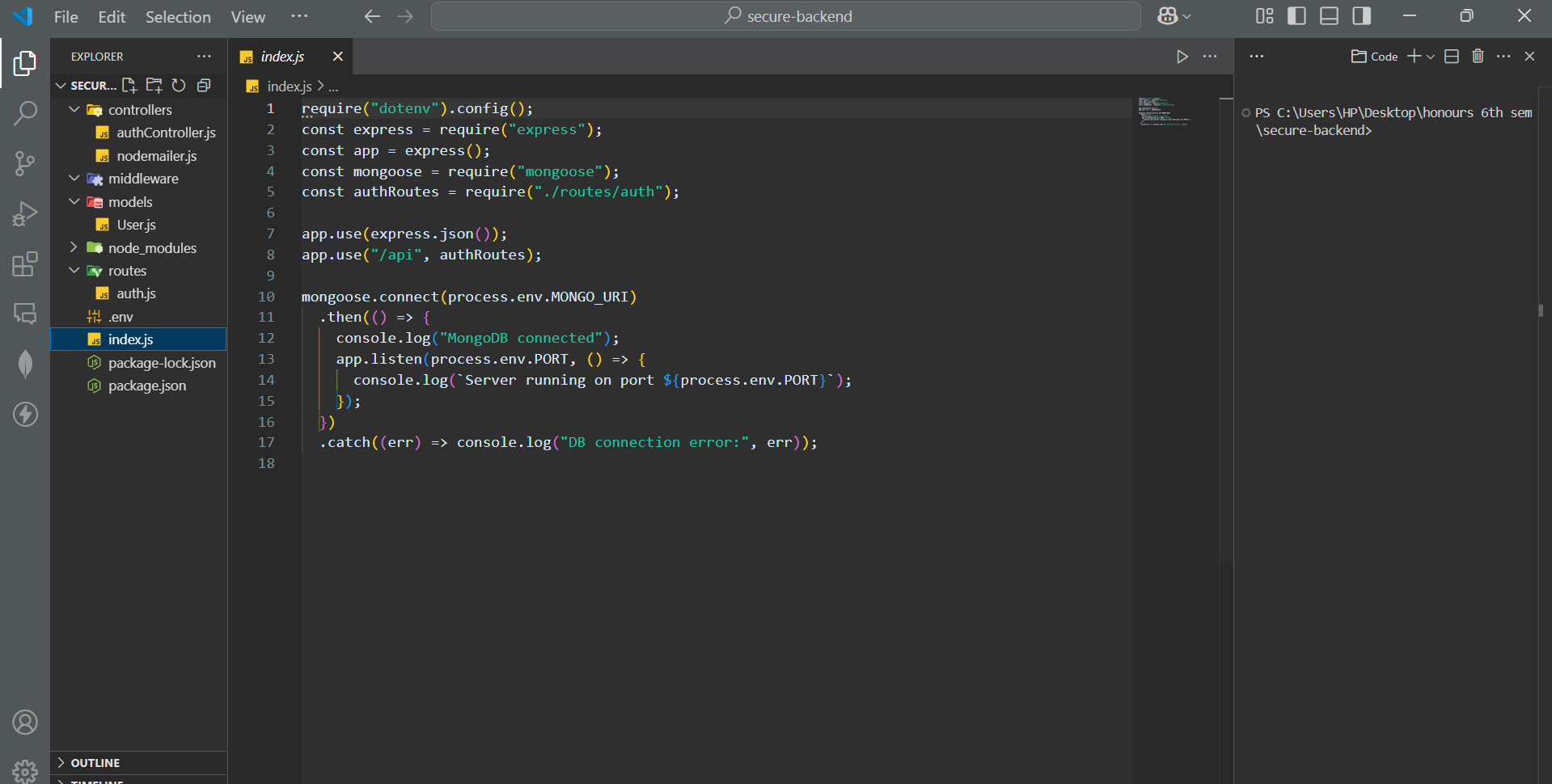
- Installed and configured WSL with Ubuntu terminal.  
- Installed Node.js, npm, and Git in the Ubuntu environment.  
- Verified internet access using ping.  
- Ensured correct versions using node -v, npm -v, git --version.



# Step 2: Project Initialization and Structure

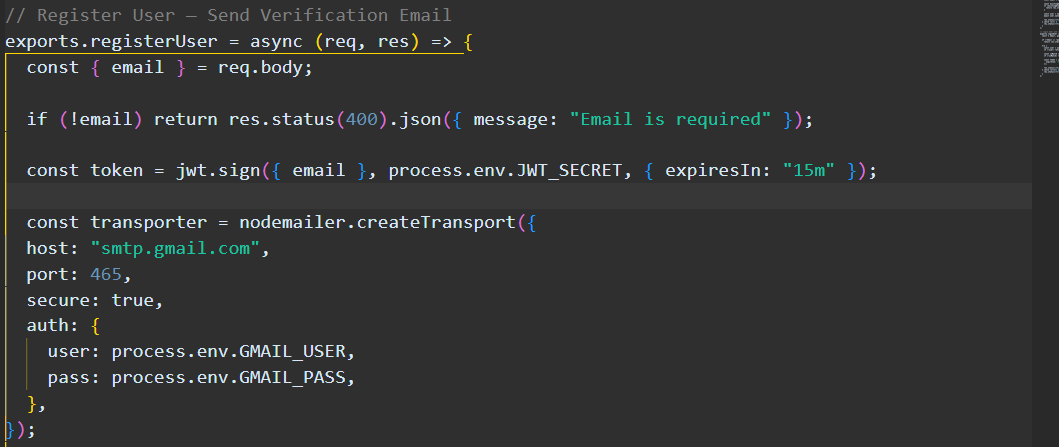
- Created project folder on Windows Desktop using WSL path.  
- Initialized Node.js project with `npm init -y`.  
- Installed dependencies: express, mongoose, dotenv, nodemailer, bcryptjs, jsonwebtoken.  
- Created folder structure: routes, controllers, models, middleware.  
- Added nodemon for development and configured dev script in package.json.

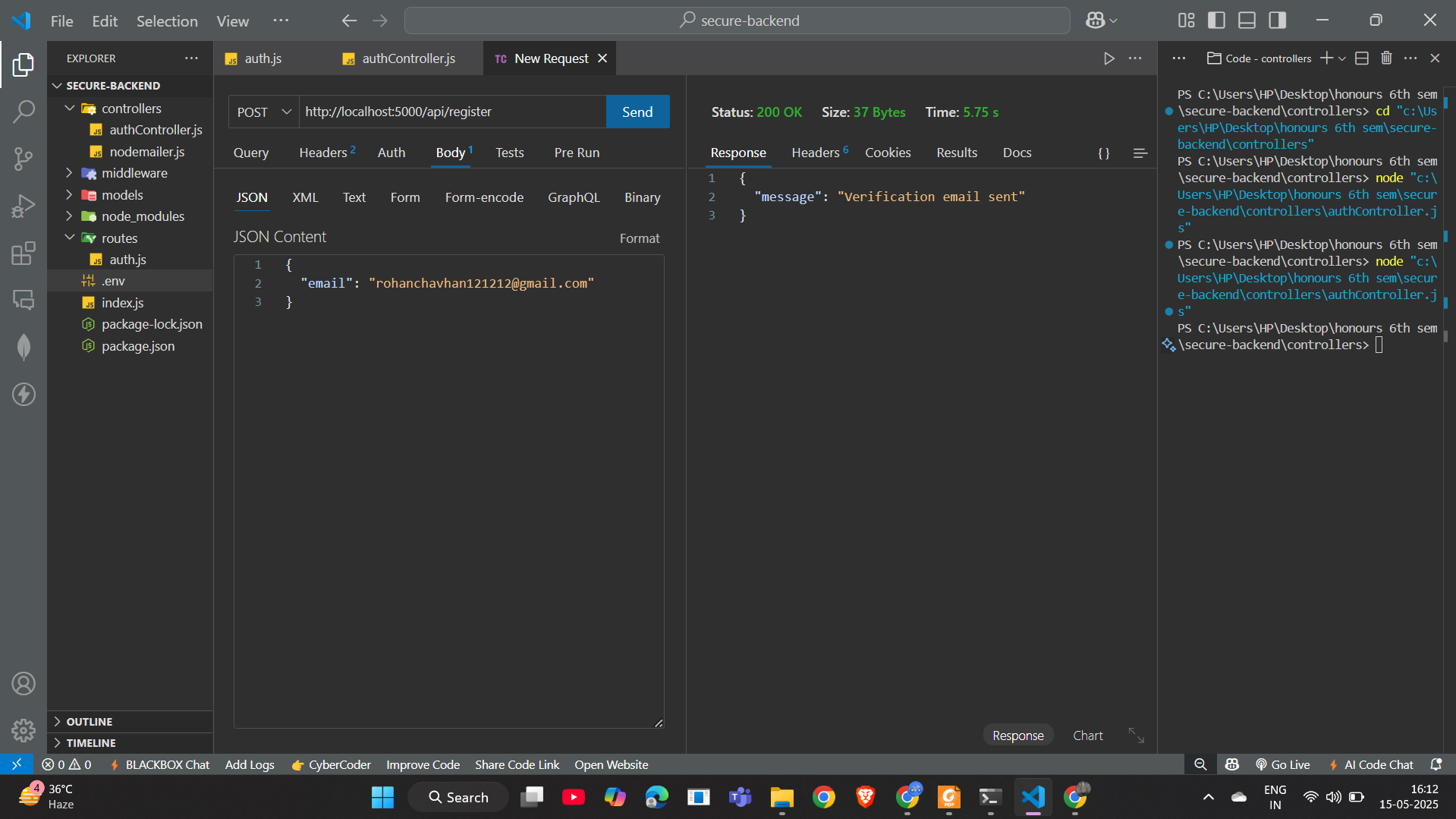




# Step 3: Email Registration Endpoint (/register)

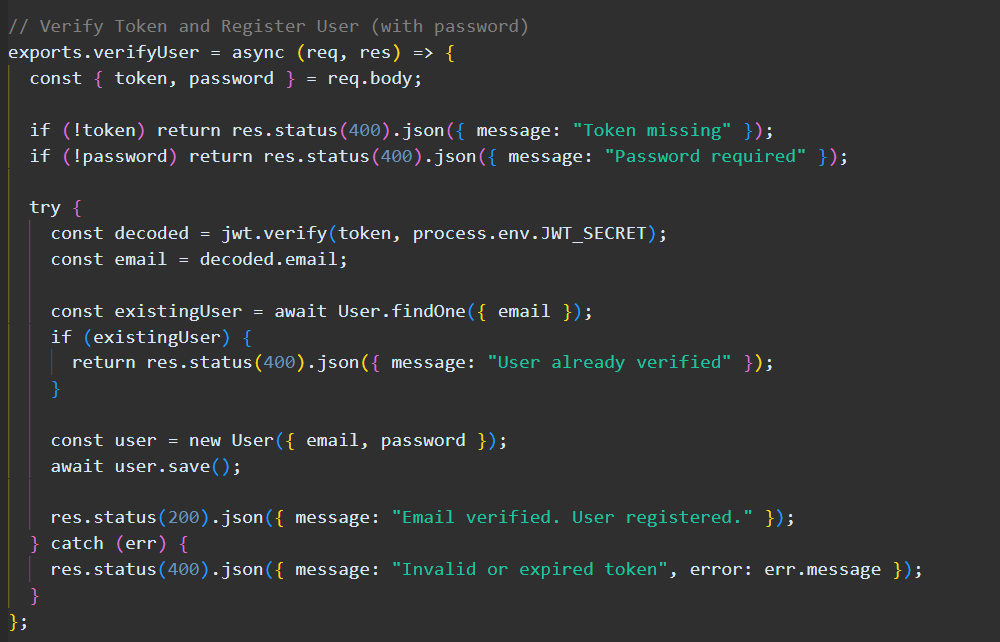
- Setup .env file with Gmail credentials and MongoDB Atlas URI.  
- Implemented /register endpoint to send verification link to user's email using Nodemailer and Gmail SMTP.  
- Verified email credentials with SMTP test script.  
- Used full SMTP config to avoid 534 errors from Gmail.

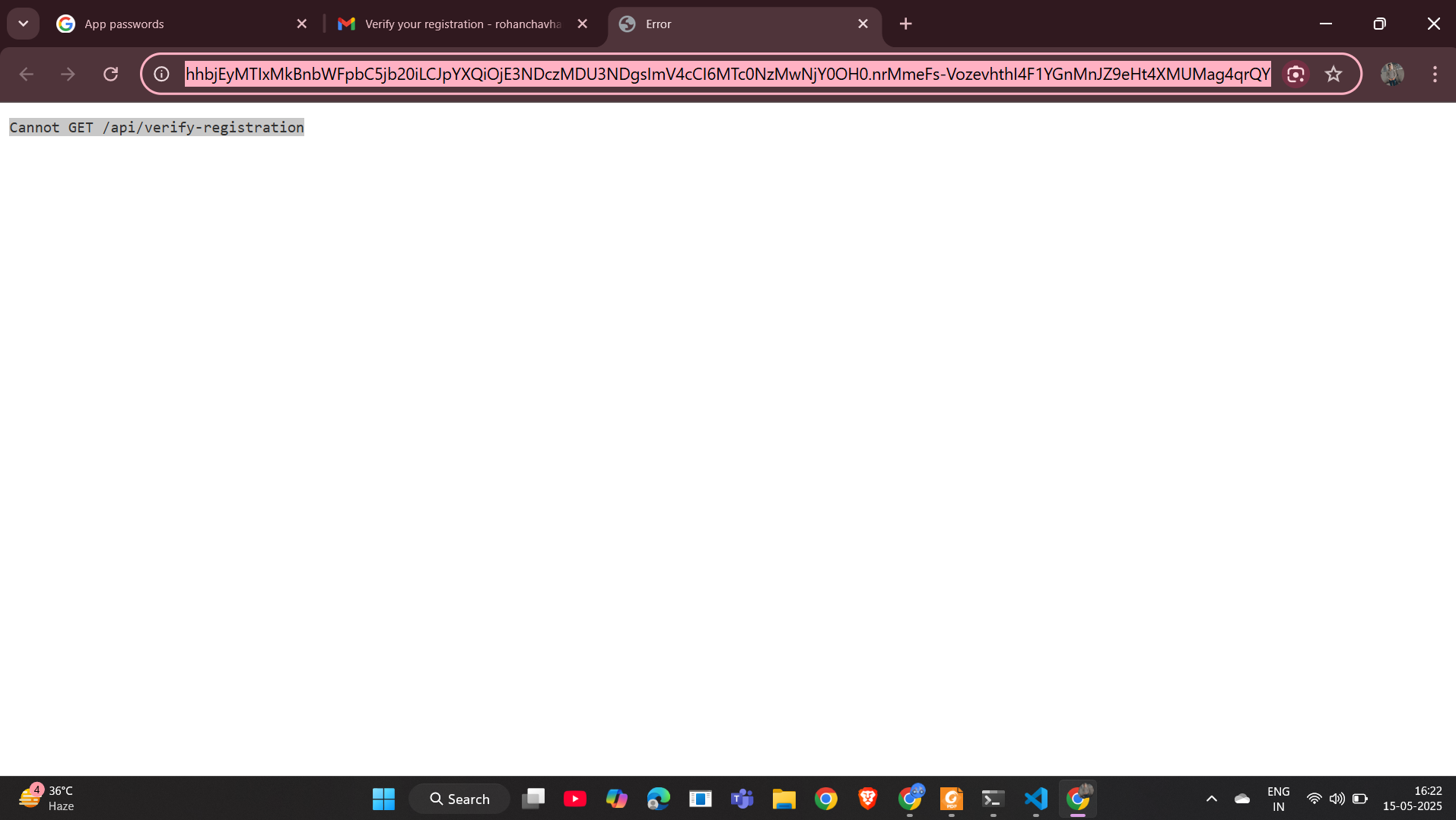
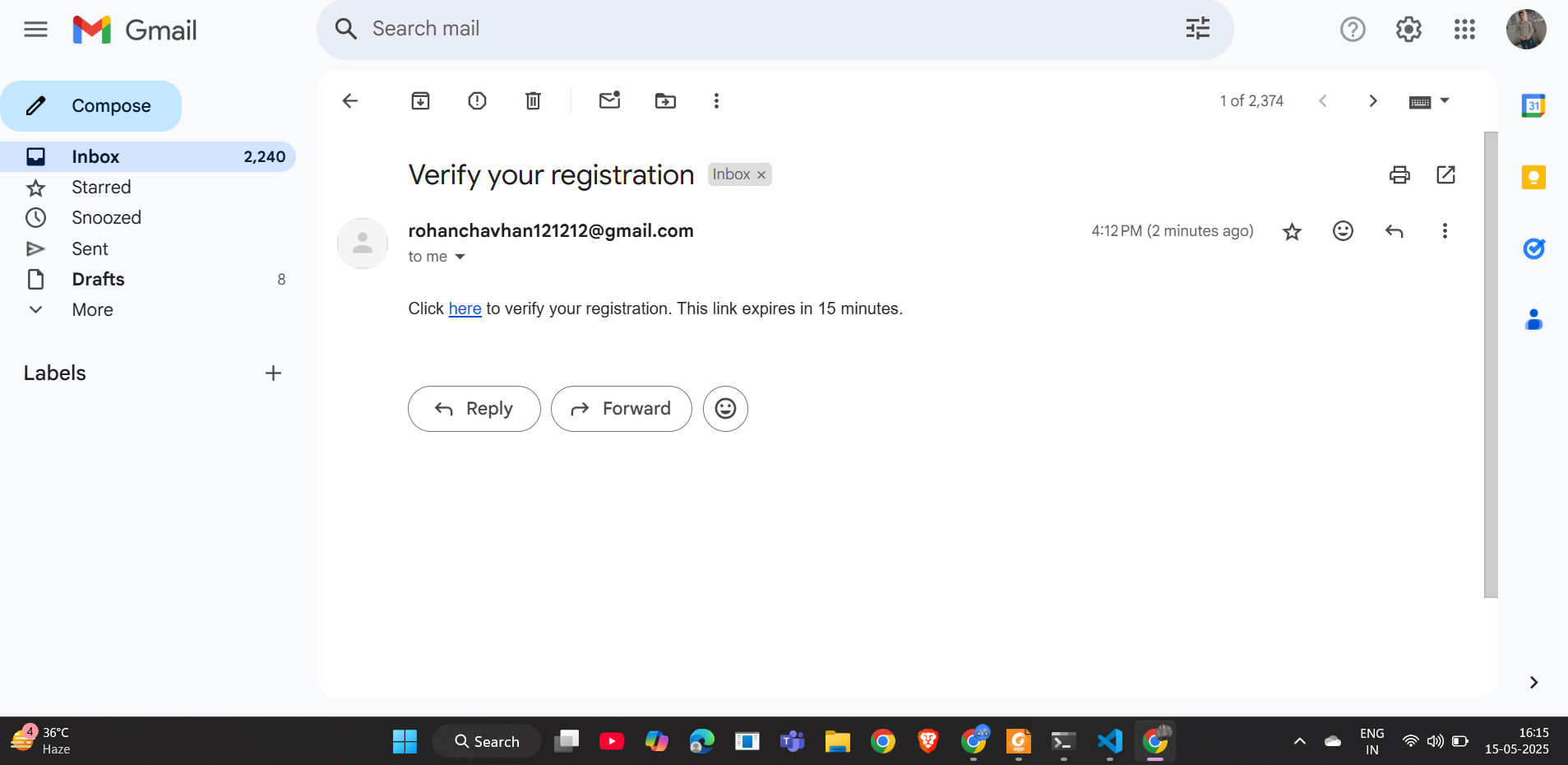




# Step 4: Email Verification (/verify-registration)

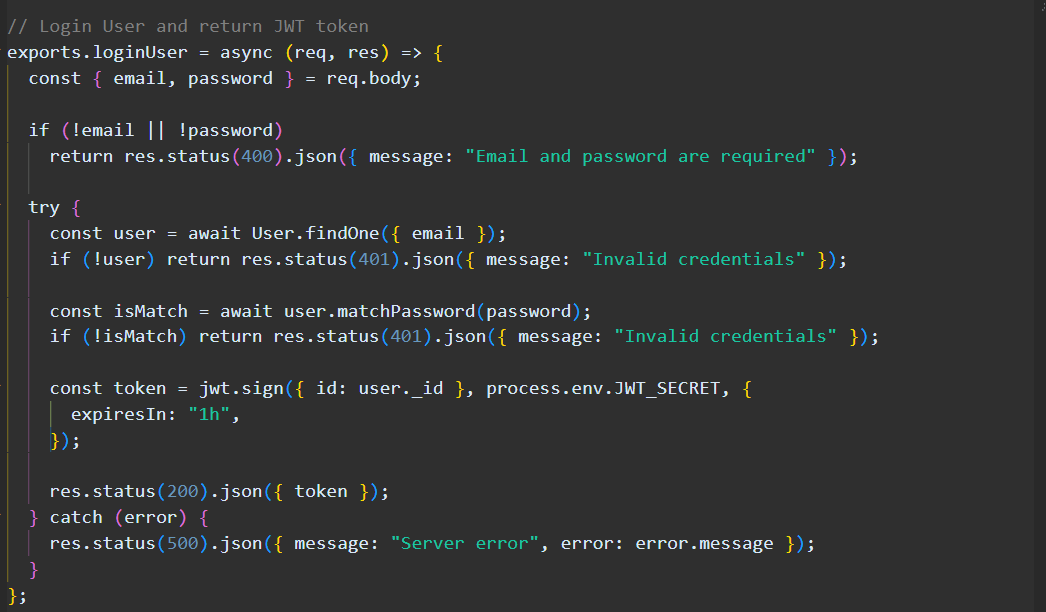
- Created verifyUser controller to handle token verification.  
- Modified route to use POST instead of GET.  
- Updated logic to decode JWT token and register the user with hashed password using bcrypt.  
- Ensured token and password are passed in POST body.

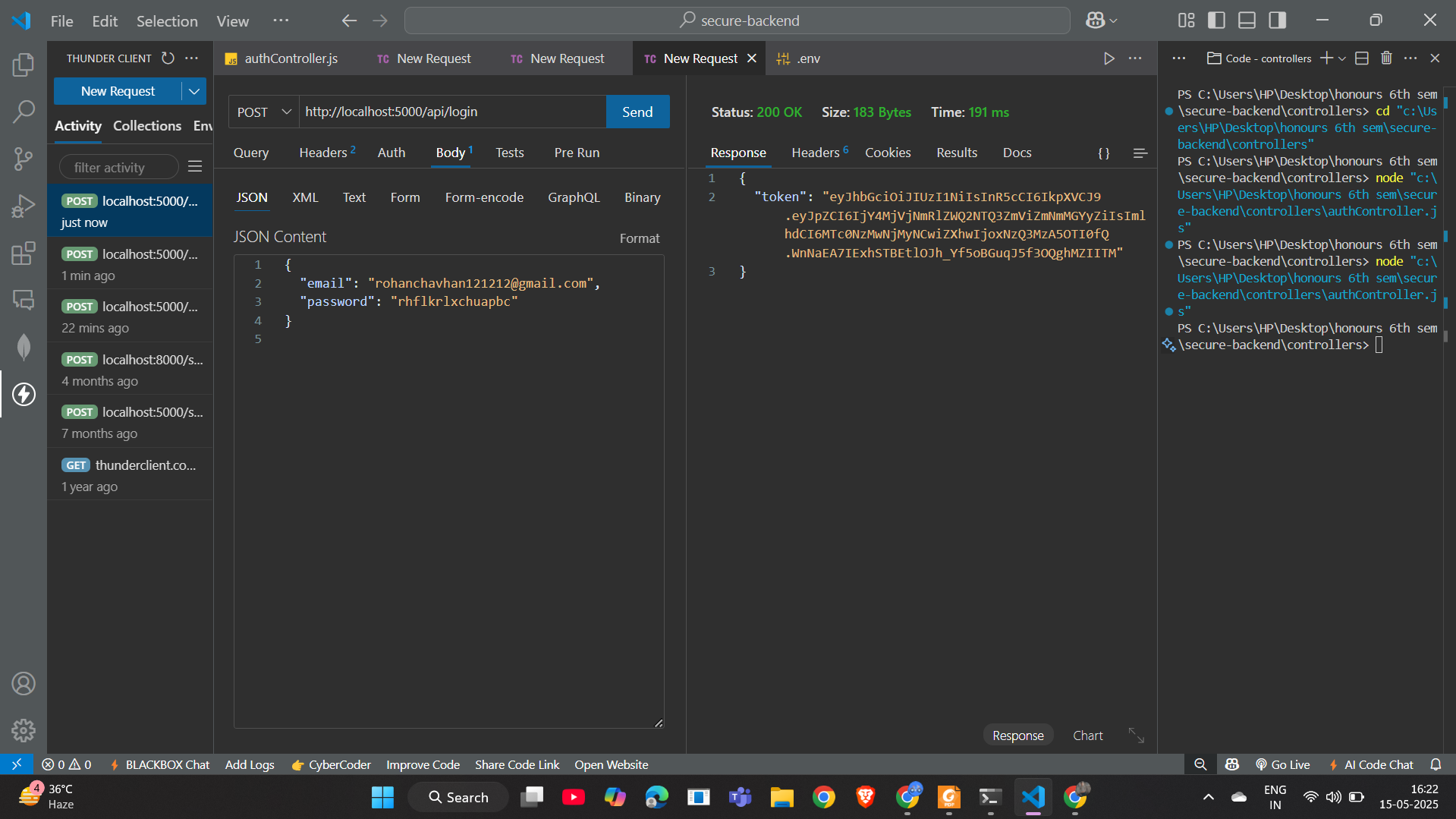




# Step 5: Login Functionality (/login)

- Added password field to User model and hashing logic using mongoose middleware.  
- Created /login endpoint to authenticate users using email and password.  
- Compared password using bcrypt and issued JWT token on success.  
- Returned JWT token in login response.





# Step 6: Postman Testing

- Tested all routes (register, verify-registration, login) using Postman.  
- Used POST method for all endpoints.  
- Verified that registration sends email, verification creates user, and login returns JWT token.

