

User Dashboard with Goal Tracking

This project is a React-based User Dashboard for tracking goals. It uses Spring Boot for the backend and MySQL for database storage. The frontend is built with React and Bootstrap for a responsive UI.

Tech Stack

- Backend: Spring Boot, MySQL, Postman (for API testing)
- Frontend: React, Bootstrap
- Tools: VS Code (Frontend), IntelliJ IDEA 2024.3.2 (Backend)

Features

- User registration, login, and authentication
- Create, update, and delete goals
- View and track active goals
- Responsive UI

Setup Backend

1. Clone the repo and navigate to the backend folder.
2. Set up MySQL (goal_tracking database).
3. Run the Spring Boot application in IntelliJ IDEA.

Frontend

1. Navigate to the frontend folder.
2. Run npm install and npm start.

API Testing (Postman)

- POST /api/users/register: Register a new user
- POST /api/users/login: Log in and get JWT token
- GET /api/goals: Get user goals
- POST /api/goals: Create a new goal

This project was developed as a React-based user dashboard with goal tracking functionality. I tried my best to integrate both the frontend and backend to provide a seamless experience for users to track their goals. I hope this project helps in learning full-stack development using modern technologies like React, Spring Boot, and MySQL.

Build a React User Dashboard with Goal Tracking

Create a React.js dashboard for managing users and tracking their goals with the following features:

- **Add User:** Include a form to add a new user (name, email).

Navbar

HomeLoginRegister User

Register User

Name

Shubham Kumar Yadav

Email

sy392307@gmail.com

Password

shubham

Submit

- **User List:** Display a list of users with name, email, and the number of goals they have.

React App

localhost:3000

Navbar

HomeLoginRegister User

Search by Name

ID	User Name	User Email	Action
1	Shubham Kumar Yadav	sy392307@gmail.com	<div>ViewDelete</div>
2	Shubham	sh612@gmail.com	<div>ViewDelete</div>
3	Demo	demo@gmail.com	<div>ViewDelete</div>
4	ram	ramsingh23076@gmail.com	<div>ViewDelete</div>
5	ram sing	ram@gmail.com	<div>ViewDelete</div>

- **Search and Sort:** Add a search bar to filter users by name or email and allow sorting by name.

Navbar

HomeLoginRegister User

Shubham

ID	User Name	User Email	Action
1	Shubham Kumar Yadav	sy392307@gmail.com	<div>ViewDelete</div>
2	Shubham	sh612@gmail.com	<div>ViewDelete</div>

- **Login User: Using Email and Password.**

Navbar

HomeLoginRegister User

Login

Email

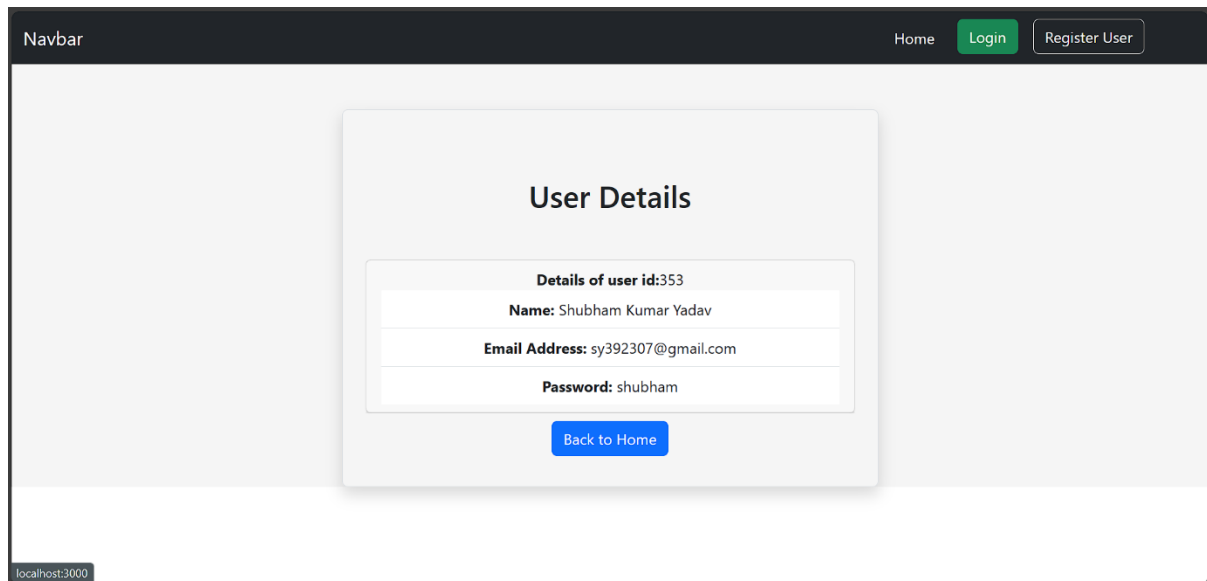
sy392307@gmail.com

Password

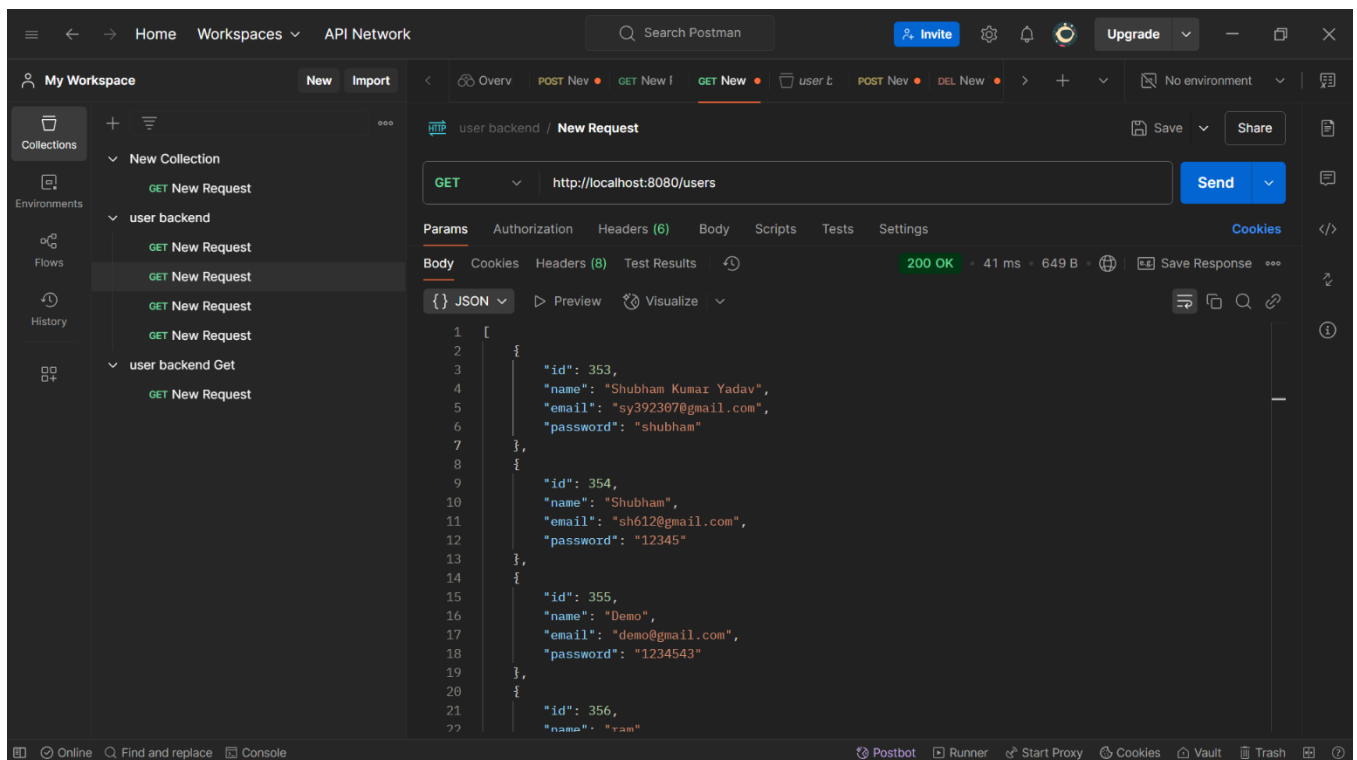
shubham

Login

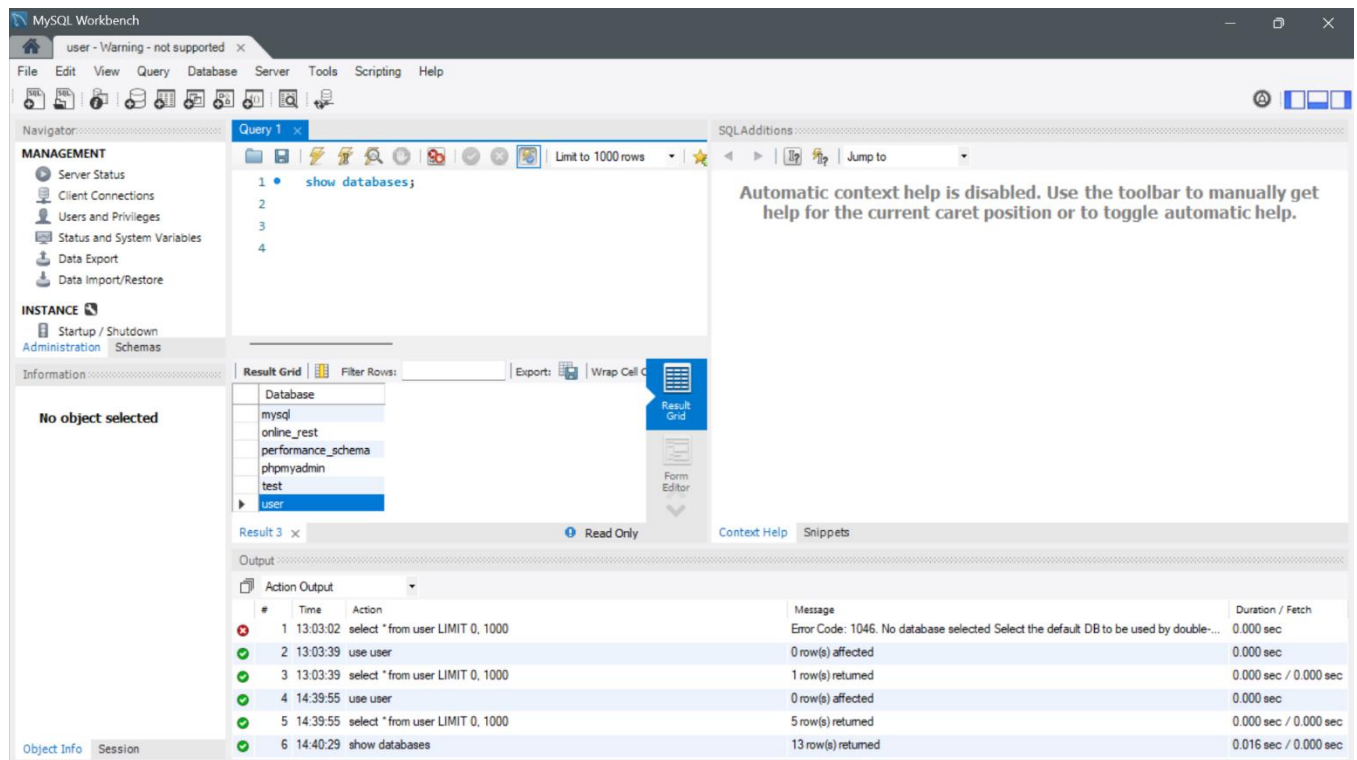
- **User Details:** Clicking on a user shows their goals in a sidebar or modal, with each goal showing its title, deadline, and status (e.g., "In Progress," "Completed").



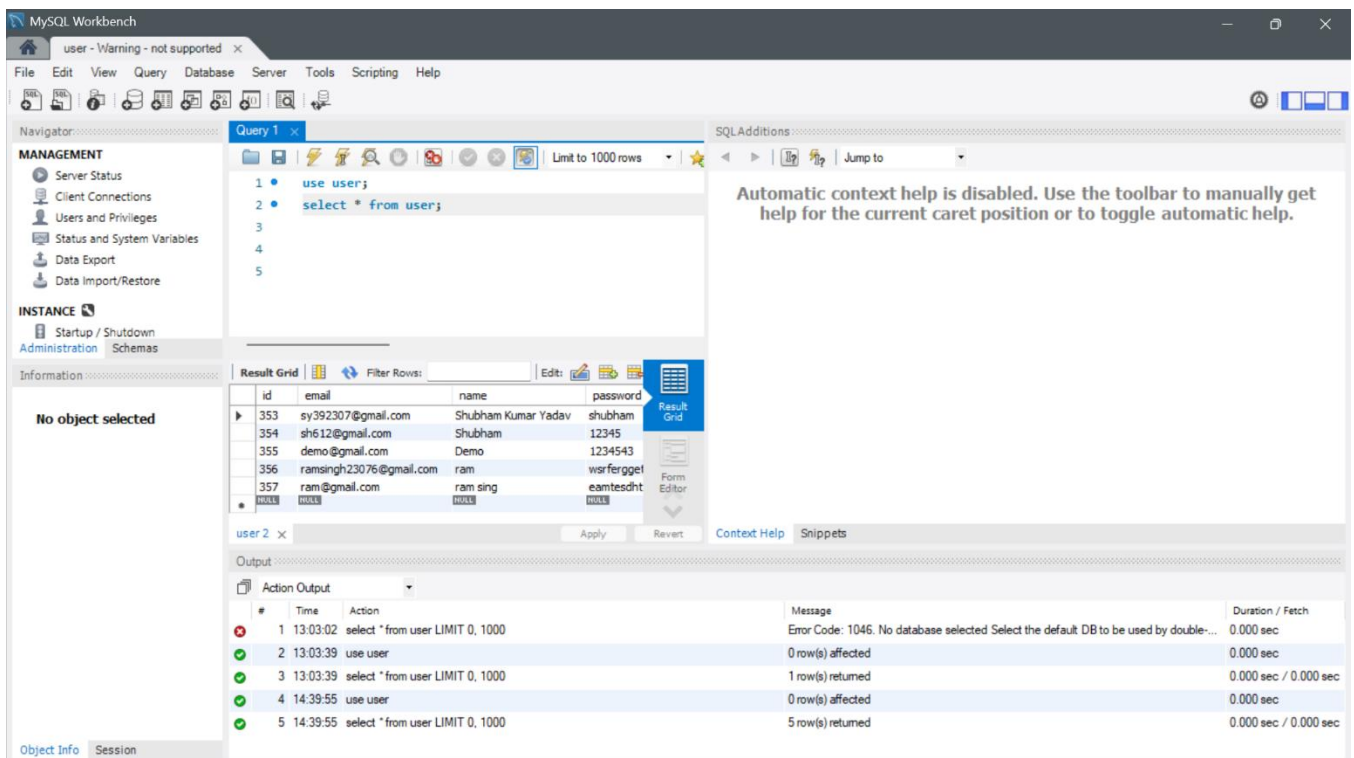
- **POSTMAN:** Data will GET, POST, PUT, DELETE in JSON file



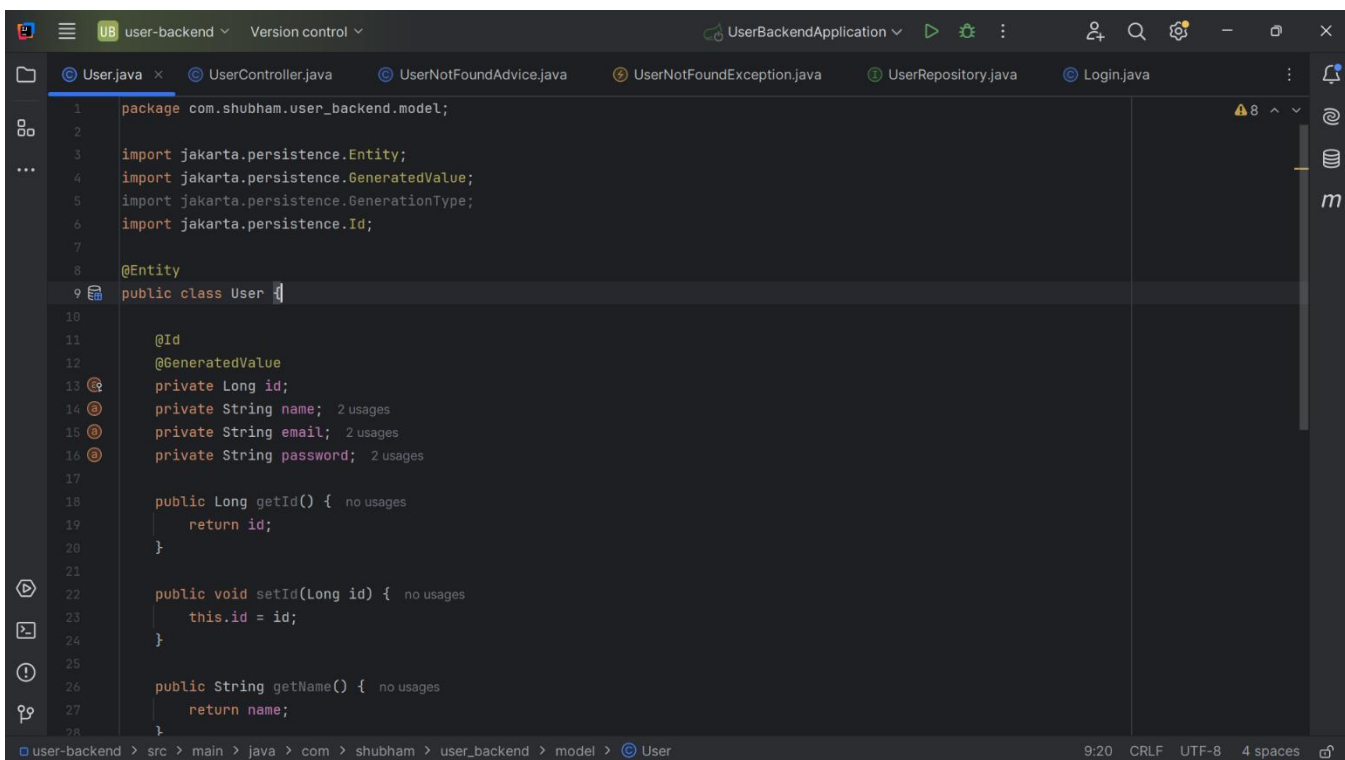
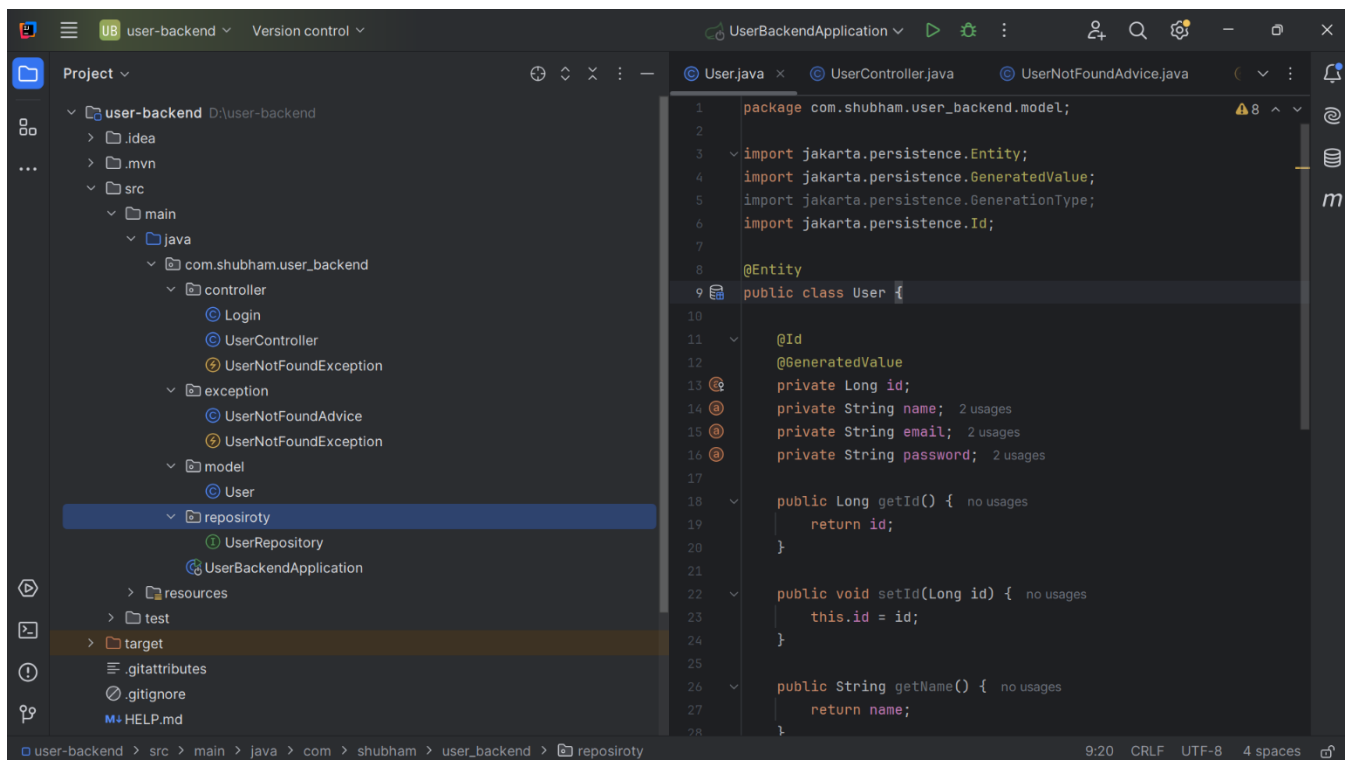
- **MYSQL: Database Name is User**



- **Tables and Data of User which was Added Use of spring Boot, POSTMA, React.**



- IntelliJ IDEA 2024.3.2 (Backend):

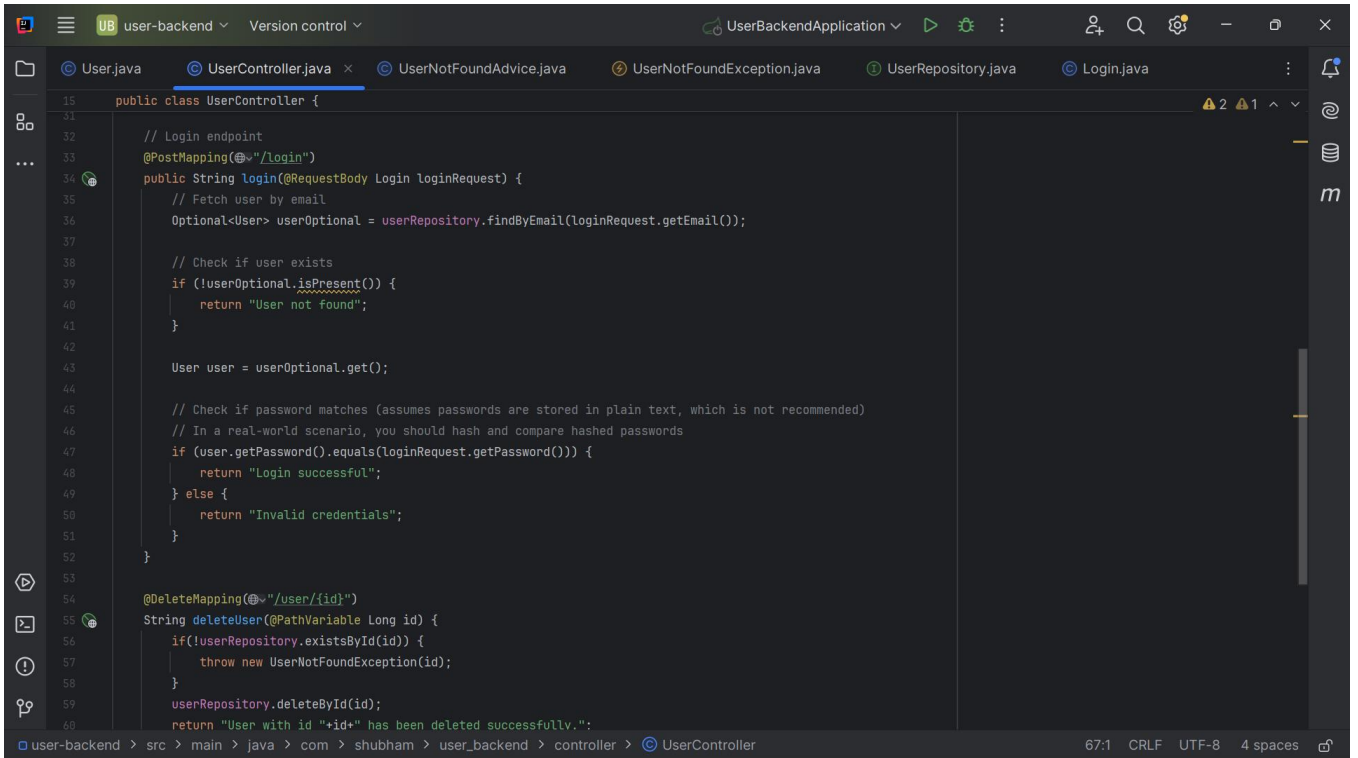


```
9 public class User {
23     this.id = id;
24 }
25
26 public String getName() { no usages
27     return name;
28 }
29
30 public void setName(String name) { no usages
31     this.name = name;
32 }
33
34 public String getEmail() { no usages
35     return email;
36 }
37
38 public void setEmail(String email) { no usages
39     this.email = email;
40 }
41
42 public String getPassword() { 1 usage
43     return password;
44 }
45
46 public void setPassword(String password) { no usages
47     this.password = password;
48 }
49 }
```

user-backend > src > main > java > com > shubham > user_backend > model > User 9:20 CRLF UTF-8 4 spaces

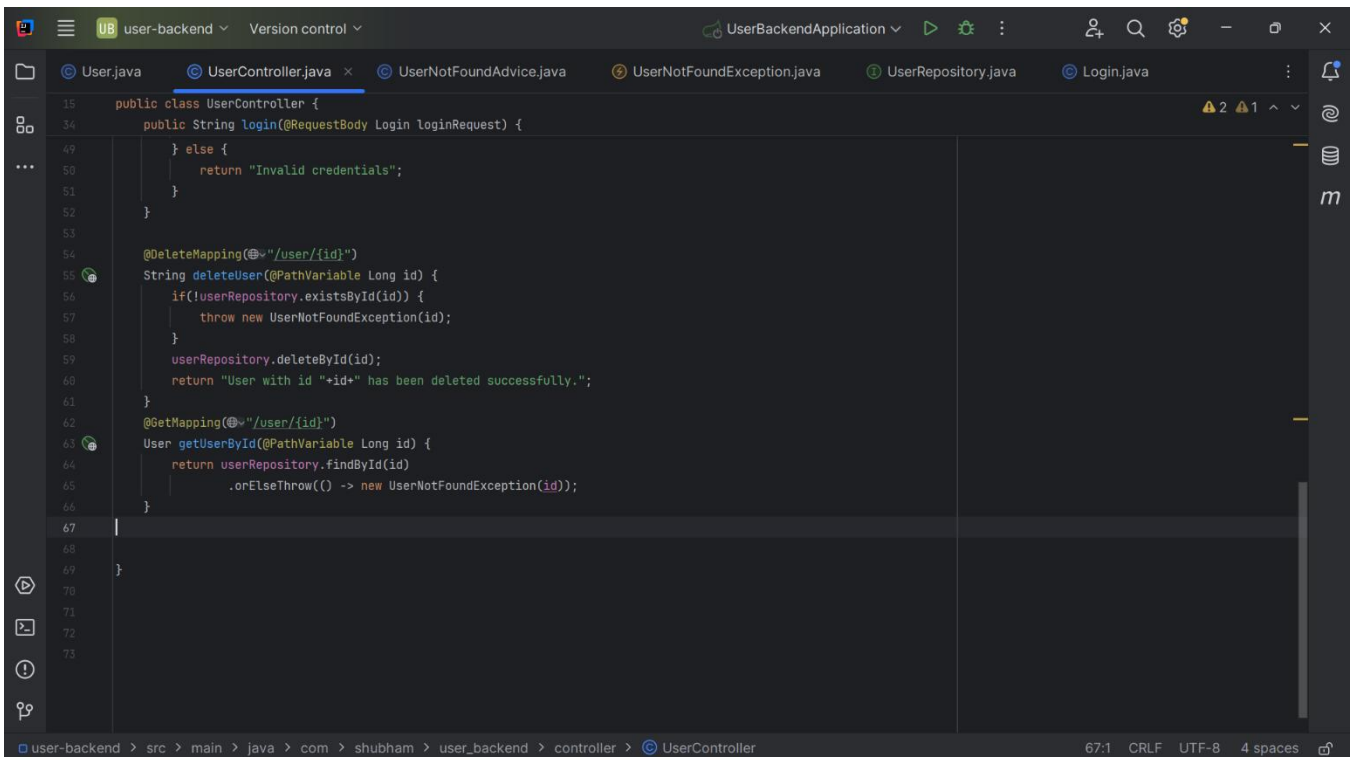
```
1 package com.shubham.user_backend.controller;
2
3 import com.shubham.user_backend.model.User;
4 import com.shubham.user_backend.controller.Login;
5 import com.shubham.user_backend.reposiroty.UserRepository;
6 import org.springframework.beans.factory.annotation.Autowired;
7 import org.springframework.web.bind.annotation.*;
8
9 import com.shubham.user_backend.exception.UserNotFoundException;
10 import java.util.List;
11 import java.util.Optional;
12
13 @RestController
14 @CrossOrigin("http://localhost:3000")
15 public class UserController {
16
17     @Autowired
18     private UserRepository userRepository;
19
20     // Endpoint to create a new user
21     @PostMapping("/user")
22     public User newUser(@RequestBody User newUser) {
23         return userRepository.save(newUser);
24     }
25
26     // Endpoint to get all users (for debugging purposes)
27     @GetMapping("/users")
28     public List<User> getAllUsers() {
29         return userRepository.findAll();
30     }
31 }
```

user-backend > src > main > java > com > shubham > user_backend > controller > UserController 67:1 CRLF UTF-8 4 spaces



```
15 public class UserController {
31
32     // Login endpoint
33     @PostMapping("/login")
34     public String login(@RequestBody Login loginRequest) {
35         // Fetch user by email
36         Optional<User> userOptional = userRepository.findByEmail(loginRequest.getEmail());
37
38         // Check if user exists
39         if (!userOptional.isPresent()) {
40             return "User not found";
41         }
42
43         User user = userOptional.get();
44
45         // Check if password matches (assumes passwords are stored in plain text, which is not recommended)
46         // In a real-world scenario, you should hash and compare hashed passwords
47         if (user.getPassword().equals(loginRequest.getPassword())) {
48             return "Login successful";
49         } else {
50             return "Invalid credentials";
51         }
52     }
53
54     @DeleteMapping("/user/{id}")
55     String deleteUser(@PathVariable Long id) {
56         if (!userRepository.existsById(id)) {
57             throw new UserNotFoundException(id);
58         }
59         userRepository.deleteById(id);
60         return "User with id "+id+" has been deleted successfully.";
61     }
62 }
```

user-backend > src > main > java > com > shubham > user_backend > controller > UserController 67:1 CRLF UTF-8 4 spaces



```
15 public class UserController {
34     public String login(@RequestBody Login loginRequest) {
49         } else {
50             return "Invalid credentials";
51         }
52     }
53
54     @DeleteMapping("/user/{id}")
55     String deleteUser(@PathVariable Long id) {
56         if (!userRepository.existsById(id)) {
57             throw new UserNotFoundException(id);
58         }
59         userRepository.deleteById(id);
60         return "User with id "+id+" has been deleted successfully.";
61     }
62
63     @GetMapping("/user/{id}")
64     User getUserById(@PathVariable Long id) {
65         return userRepository.findById(id)
66             .orElseThrow(() -> new UserNotFoundException(id));
67     }
68
69 }
70
71
72
73 }
```

user-backend > src > main > java > com > shubham > user_backend > controller > UserController 67:1 CRLF UTF-8 4 spaces

LINKS:

Vercel Link: <https://user-dashboard-with-goal-tracking-recat-frontend.vercel.app/>

GitHub Backend Link: [User-Dashboard-with-Goal-Tracking-Recat-Backend/user-backend at main · coshubham/User-Dashboard-with-Goal-Tracking-Recat-Backend](https://github.com/coshubham/User-Dashboard-with-Goal-Tracking-Recat-Backend/tree/main)