# User Manual: Server

## 1. Introduction

This server is designed for JWT-based authentication and role-based access authorization. It supports various user roles, including user, supervisor, and admin, with specific functionalities for each role.

## 2. How to Turn On the Server

**Prerequisites:**

1. Node.js installed on your machine.
2. MongoDB running either locally or remotely.
3. Create a `.env` file in the root directory with the following content:

SECRET=your\_jwt\_secret\_key  
MONGO\_URI=your\_mongodb\_connection\_string

**Steps:**

1. Install dependencies:  
   npm install
2. Start the server:

npm run dev

This will start the server using `nodemon` for automatic restarts on code changes.  
The server will be listening on `http://localhost:3001`.

## 3. How to Use

**Available Endpoints:**

**Sign-up:**

**URL:** */api/signUp* **Method:** POST **Description:** Registers a new user. **Body Parameters:**

{

'email': 'user@example.com',   
'password': 'Password123'

}

**/\* ADD HTTPS explanation \*/**

**Sign-in:**

**URL:** */api/signIn* **Method:** POST **Description:** Authenticates a user and provides a JWT token. **Body Parameters:**

{

'email': 'user@example.com',  
 'password': 'Password123'

}

**Get Homepage:**

**URL:** */api/homeBoard* **Method:** GET **Description:** Returns home page content.

**Get User Homepage:**

**URL:** */api/userBoard* **Method:** GET **Description:** Returns user board content. **Headers:**

{

'x-access-token': '<JWT\_TOKEN>'

}

**Get Supervisor Homepage:**

**URL:** */api/supervisorBoard* **Method:** GET **Description:** Returns supervisor board content. **Headers:**

{

'x-access-token': '<JWT\_TOKEN>'

}

**Get Admin Homepage:**

**URL:** */api/adminBoard* **Method:** GET **Description:** Returns admin board content. **Headers:**

{

'x-access-token': '<JWT\_TOKEN>'

}

**Add Supervisor:**

**URL:** */api/addSupervisor* **Method:** PUT **Description:** Adds supervisor role to a user. **Headers:**

{

'x-access-token': '<JWT\_TOKEN>'

}

**Body Parameters:**

{

'email': 'user@example.com'

}

**Remove Supervisor:**

**URL:** */api/removeSupervisor* **Method:** PUT **Description:** Removes supervisor role from a user. **Headers:**

{

'x-access-token': '<JWT\_TOKEN>'

}

**Body Parameters:**

{

'email': 'user@example.com'

}

## 4. Functionalities Overview

1. User Registration and Authentication: Allows users to sign up and sign in using their email and password.
2. Passwords are hashed using bcrypt for security.
3. Role-Based Access: Provides different access levels for users, supervisors, and admins.
4. JWT Authentication: Uses JSON Web Tokens for session management and secure API access.
5. Middleware for Validation: Includes middleware functions for validating passwords, checking if an email is already used, and verifying JWT tokens.
6. Admin Capabilities: Admins can promote users to supervisors and demote supervisors to regular users.
7. Rate Limiting: Limits the number of requests from a single IP to prevent abuse.
8. Logging: Uses `morgan` to log HTTP requests to a file for monitoring and debugging.
9. HTTPS Support: Can be configured to run over HTTPS with self-signed certificates for secure communication.