User Management API Documentation (English Version)

Overview This API is used for **user authentication, user management, token handling, and role management**. **JWT** is used for access and refresh tokens, and **MySQL** is used to store user information. Only users with the **admin** role can access certain endpoints, such as viewing all users, deleting a user, or changing roles.

Base URL: `http://<server_ip>:8080` **Authentication:** Bearer JWT (for protected endpoints)

Endpoints

1. Sign Up

Register a new user. - **URL:** `/signup` - **Method:** `POST` - **Headers:** `Content-Type: application/json` - **Request Body:**

```
{
  "username": "john",
  "password": "mypassword"
}
```

- **Responses:** - **201 Created** ✅ `{ "message": "user registered successfully" }` - **401 Unauthorized** ⚠️: `{ "message": "user registered was not successfully" }` - **400 Bad Request** ⚠️: `{ "error": "username or password are not valid" }` - **Example curl:**

```
curl -X POST http://localhost:8080/signup
     -H "Content-Type: application/json"
     -d '{"username": "john", "password": "mypassword"}'
```

2. Login

Authenticate a user and receive access & refresh tokens. - **URL:** `/login` - **Method:** `POST` - **Headers:** `Content-Type: application/json` - **Request Body:**

```
{
  "username": "john",
  "password": "mypassword"
}
```

- **Responses:** - **200 OK** ✅

```
{
  "success": true,
  "role": "user",
  "message": "Login successful",
  "access_token": "<ACCESS_TOKEN>",
  "refresh_token": "<REFRESH_TOKEN>"
}
```

- **401 Unauthorized** ⛔– Invalid username or password - **400 Bad Request** ⚠️– Empty username or password - **Example curl:**

```
curl -X POST http://localhost:8080/login
     -H "Content-Type: application/json"
     -d '{"username": "john", "password": "mypassword"}'
```

3. Refresh Token

Obtain a new access token using a valid refresh token. - **URL:** `/refresh` - **Method:** `POST` - **Headers:** `Content-Type: application/json` - **Request Body:**

```
{
  "refresh_token": "<REFRESH_TOKEN>"
}
```

- **Responses:** - **200 OK** ✅ `{ "access_token": "<NEW_ACCESS_TOKEN>", "message": "new access token generated", "success": true }` - **401 Unauthorized** ⛔– Refresh token is expired or invalid - **400 Bad Request** ⚠️– Refresh token is missing - **Example curl:**

```
curl -X POST http://localhost:8080/refresh
     -H "Content-Type: application/json"
     -d '{"refresh_token": "<REFRESH_TOKEN>"}'
```

4. Get All Users (Admins Only)

Retrieve a list of all registered users. - **URL:** `/users` - **Method:** `GET` - **Headers:** `Authorization: Bearer <ACCESS_TOKEN>` - **Responses:** - **200 OK** ✅– Returns an array of users

```
[
  { "username": "john", "role": "user" },
  { "username": "admin", "role": "admin" }
]
```

- **403 Forbidden** ⛔– Non-admin users cannot access - **401 Unauthorized** ⛔– Token missing or invalid - **Example curl:**

```
curl -X GET http://localhost:8080/users
     -H "Authorization: Bearer <ACCESS_TOKEN>"
```

5. Delete User (Admins Only)

Delete a user by username. - **URL:** `/users/{username}` - **Method:** `DELETE` - **Headers:** `Authorization: Bearer <ACCESS_TOKEN>` - **Responses:** - **200 OK** ✅– User deleted successfully - **404 Not Found** ⚠️– User does not exist - **403 Forbidden** ⛔– Non-admin or invalid token - **400 Bad Request** ⚠️– Invalid username - **Example curl:**

```
curl -X DELETE http://localhost:8080/users/john
     -H "Authorization: Bearer <ACCESS_TOKEN>"
```

6. Change User Role (Admins Only)

Change a user's role. - **URL:** `/users/{username}` - **Method:** `PUT` - **Headers:** `Authorization: Bearer <ACCESS_TOKEN>` - **Request Body:**

```
{
  "new_role": "admin"
}
```

- **Responses:** - **200 OK** ✅– Role changed successfully - **404 Not Found** ⚠️– User does not exist - **403 Forbidden** ⛔– Non-admin or invalid token - **400 Bad Request** ⚠️– Invalid username or role - **Example curl:**

```
curl -X PUT http://localhost:8080/users/john
     -H "Authorization: Bearer <ACCESS_TOKEN>"
     -H "Content-Type: application/json"
     -d '{"new_role": "admin"}'
```

Authentication

- **Access Token:** Short-lived JWT (1 hour), required for protected endpoints (`/users`, `/users/{username}` DELETE & PUT).
- **Refresh Token:** Long-lived JWT (10 days), used to obtain new access token via `/refresh`.
- Tokens must be sent in the `Authorization` header as `Bearer <token>`.

Status Codes

Status Code	Description
200	✅Success
201	✅Resource created
400	⚠️Bad request (missing or invalid parameters)
401	⛔Unauthorized (invalid credentials or token)
403	⛔Forbidden (insufficient permissions)
404	⚠️Not found
500	Server error (DB or internal error)