**SkillSwap User Management (Auth) API Documentation  
  
User Data Format:**{ “email”: “String” (required),

“role”: “String” (required, “admin” || “Basic”, default: ”Basic”),

“username”: “String” (required),

“password”: “String” (required),

“description”: “String”,

“isAvailable”: “Boolean” }

**Base URL**

The base URL for all API requests is **https://ws24-skillswap.onrender.com**

**Endpoints**

All endpoints are prefixed with **/api/auth**

**1. Register**

Allows new users to register.

* **URL:** **/api/auth/register**
* **Method:** **POST**
* **Body Parameters:**
  + **username**: String (required)
  + **email**: String (required)
  + **password**: String (required)
* **Success Response:**
  + **Code:** 201 (Created)
  + **Content:** **{ message: "User successfully registered", user: { id, username, email } }**
  + **Token Delivery:** The JWT token is sent as an HTTP-only cookie named **jwt**. This cookie is not accessible via JavaScript and is automatically sent with subsequent requests to the server.
* **Error Response:**
  + **Code:** 400 (Bad Request)
  + **Content:** **{ message: "Error message" }**

**2. Login**

Authenticates a user and returns a session token.

* **URL:** **/api/auth/login**
* **Method:** **POST**
* **Body Parameters:**
  + **email**: String (required)
  + **password**: String (required)
* **Success Response:**
  + **Code:** 200 (OK)
  + **Content:** **{ message: "Login successful", user: user.\_id }**
  + **Token Delivery:** The JWT token is sent as an HTTP-only cookie named **jwt**. This cookie is not accessible via JavaScript and is automatically sent with subsequent requests to the server.
* **Error Response:**
  + **Code:** 400 (Bad Request) / 401 (Unauthorized)
  + **Content:** **{ message: "Login not successful", error: "User not found" or "Invalid password" }**

**3. Update User Data**

Allows logged-in users to update their own data.

* **URL:** **/api/auth/update/{userId}**
* **Method:** **PUT**
* **Headers:**
  + **Authorization**: **Bearer <JWT Token>** (Token retrieved from login cookie)
* **URL Parameters:**
  + **userId**: ID of the user to update
* **Body Parameters:** Any user field(s) to update.
* **Success Response:**
  + **Code:** 200 (OK)
  + **Content:** **{ message: "User updated successfully", user: { updated fields } }**
* **Error Response:**
  + **Code:** 401 (Unauthorized) / 403 (Forbidden)
  + **Content:** **{ message: "Error message" }**

**4. Delete User Data**

Allows admins to delete a user's data.

* **URL:** **/api/auth/delete/{userId}**
* **Method:** **DELETE**
* **Headers:**
  + **Authorization**: **Bearer <JWT Token>** (Token retrieved from login cookie)
* **URL Parameters:**
  + **userId**: ID of the user to delete
* **Success Response:**
  + **Code:** 200 (OK)
  + **Content:** **{ message: "User deleted successfully" }**
* **Error Response:**
  + **Code:** 401 (Unauthorized) / 403 (Forbidden)
  + **Content:** **{ message: "Error message" }**

**5. Fetch All Users (Admin Only)**

* **Endpoint:** **/api/auth/all\_users**
* **Method:** GET
* **Access:** Admins only
* **Description:** Fetches data of all users. Accessible only by users with an admin role.
* **Authorization:** Required. JWT token must be included in the HTTP-only cookie.

**6. Fetch User Data by ID**

* **Endpoint:** **/api/auth/{userId}**
* **Method:** GET
* **Access:** Users (for their own data) and Admins
* **Description:** Fetches data for a specific user by their ID. Users can access their own data, and admins can access any user's data.
* **Authorization:** Required. JWT token must be included in the HTTP-only cookie.
* **URL Parameters:** **id** - The ID of the user whose data is being requested.