

# Users

- [Introduction](#)
- [How Does it Work?](#)
- [Creating a User Entry](#)
- [View Users Details](#)
- [Database Schema](#)
- [API / Web Access](#)
  - [API Routes](#)
  - [Web Application Routes](#)

## Introduction

The User endpoint allows you to manage user accounts within Open-Audit.

## How Does it Work?

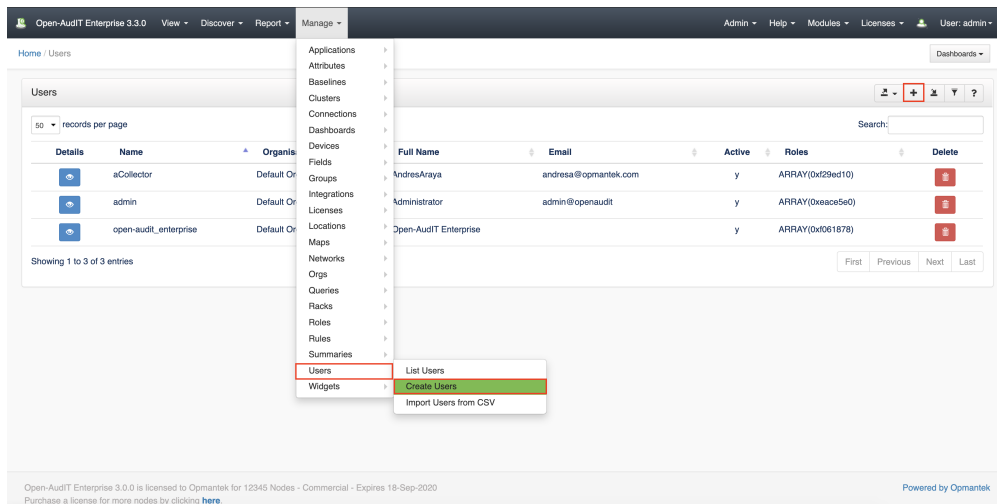
A user has a primary Org as well as a list of Orgs they can access. A user combines this with a list of assigned 'Roles' that define what actions they can take on items assigned to the Orgs they have access to. The combination of a users 'orgs' and 'roles' define what they can and cannot do within Open-Audit.

## Creating a User Entry

Join Paul McClendon, an Opmantek Support Engineer, as he demonstrates how to create a new user in Open-Audit

A user entry can be created using the web interface if the current user logged in has a role that contains the user::create permission. Go to menu: Manage -> Users -> Create Users. Also can be created from the Users View, using the "+" button.

To add a new user to Open-Audit you have to provide the details of that person, assign the organization, select the relevant Roles (multiple roles can be selected), select if the user is active or not, etc. In addition, you must grant permission to that user to access one or more organisations. It is important to notice that selecting a parent organization will automatically provide access to its children.



**Users**

Name:

Org ID:

Password:

Full Name:

Email:

Roles:

Lang:

Active:

Ldap:

Type:

Dashboard ID:

Organisations ID	Name	Parent	Grant Permission
1	Default Organisation	Default Organisation	<input type="checkbox"/>

Note - Selecting a parent will automatically provide access to its children (although it won't be indicated here).

**About**

Users and Roles in Open-Audit are key items. A user has a primary Org as well as a list of Orgs they can access. A user combines this with a list of assigned 'Roles' that define what actions they can take on items assigned to the Orgs they have access to. The combination of a users "orgs" and "roles" define what they can and cannot do within Open-Audit.

For more detailed information, check the Open-Audit Knowledge Base.

**Notes**

If the following conditions are met:

- a Role has an assigned id\_group
- an Org has an assigned id\_group
- an LDAP Server has use\_roles set to y
- a user exists in LDAP (per Active Directory or OpenLDAP) and is in the assigned id\_groups

That user can log on to Open-Audit without an account in Open-Audit needing to be created. Open-Audit will query the LDAP in question and if the user is in the required groups but not in Open-Audit their user attributes (name, full name, email, roles, orgs, etc) within Open-Audit will be automatically populated and they will be logged on.

## View Users Details

Go to menu: Manage-> Users -> List Users.

You will see a list of users. You can view a user by clicking on the blue view icon. You can also edit or delete users.

**Users**

50 records per page

Search:

Details	Name	Organisation	Full Name	Email	Active	Roles	Delete
<input type="button" value="View"/>	admin	Default Organisation	Administrator	admin@openaudit	y	ARRAY(0xec39410)	<input type="button" value="Delete"/>
<input type="button" value="View"/>	open-audit_enterprise	Default Organisation	Open-Audit Enterprise		y	ARRAY(0xd28b1b0)	<input type="button" value="Delete"/>

Showing 1 to 2 of 2 entries

First Previous Next Last

Open-Audit Enterprise 3.3.0 is licensed to Opmantek for 12345 Nodes - Commercial - Expires 18-Sep-2020  
Purchase a license for more nodes by clicking [here](#)

Powered by Opmantek

## Database Schema

The schema for the database is below. It can also be found in the application if the user has database::read permission by going to menu: Admin -> Database -> List Tables, then clicking on the "users" table.

```
CREATE TABLE `roles` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(200) NOT NULL DEFAULT '',
  `description` text NOT NULL,
  `permissions` text NOT NULL,
  `ad_group` varchar(100) NOT NULL DEFAULT '',
  `edited_by` varchar(200) NOT NULL DEFAULT '',
  `edited_date` datetime NOT NULL DEFAULT '2000-01-01 00:00:00',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8;
```

A typical entry looks as below.

```
id: 1
name: admin
org_id: 1
password:
0ab0a153e0bb0d80c50a02d08c07f0c87080eb0502f5057d30e308d204408c8068e9f4075c0eb61056107b050dd30090650b20b02109230b
210be540903ca0b6
full_name: Administrator
email: admin@openaudit
roles: ["admin", "org_admin"]
orgs: [1]
lang: en
active: y
ldap:
edited_by: Administrator
edited_date: 2017-05-31 21:40:59
```

## API / Web Access

You can access the /users collection using the normal Open-Audit JSON based API. Just like any other collection. Please see the API documentation for further details.

Access is provided as part of a roles permissions. Users is a standard resource and can have create, read, update and delete permissions.

The API routes below are usable from both a JSON Restful API and the web interface. The Web application routes are specifically designed to be called from the web interface (a browser).

### API Routes

Request Method	ID	Action	Resulting Function	Permission Required	URL Example	Notes	Example Response
POST	n		create	users::create	/users	Insert a new user entry.	<a href="#">users_create.json</a>
GET	y		read	users::read	/users/{id}	Returns a user details.	<a href="#">users_read.json</a>
PATCH	y		update	users::update	/users/{id}	Update an attribute of a user entry.	<a href="#">users_update.json</a>
DELETE	y		delete	users::delete	/users/{id}	Delete a user entry.	<a href="#">users_delete.json</a>
GET	n		collection	users::read	/users	Returns a list of users.	<a href="#">users_collection.json</a>

### Web Application Routes

Request Method	ID	Action	Resulting Function	Permission Required	URL Example	Notes
GET	n	create	create_form	users::create	/users/create	Displays a standard web form for submission to POST /users.
GET	y	update	update_form	users::update	/users/{id}/update	Show the user details with the option to update attributes using PATCH to /users/{id}