

How to install OnTrack software on Linux

Ontrack is a simple yet powerful self-hosted budgeting software for Linux. It works by creating a beautiful interface where you can list your budget, review your finances, and even track your spending history.

Ontrack is a simple yet powerful self-hosted budgeting software for Linux. It works by creating a beautiful interface where you can list your budget, review your finances, and even track your spending history. This article shows you how to install Ontrack on Ubuntu Linux 22.04 using Docker Compose and Caddy.

Get the dependencies for Ontrack

The first step in installing Ontrack budgeting software on Linux is to download Docker and Caddy. Docker will run the entire web application in an isolated container, while Caddy will let you broadcast it to the Internet.

To get started, find the signing key for the Docker repository from the developer's website:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keys/docker.gpg  
sudo chmod a+r /etc/apt/keys/docker.gpg
```

Create new apt repository file for Docker:

```
sudo nano /etc/apt/sources.list.d/docker.list
```

Write the following line of code in your repository file:

```
deb [arch=amd64 signed-by=/etc/apt/keys/docker.gpg] https://download.docker.com/linux/ubuntu focal-stable
```

Download the signing key for the Caddy project repository:

```
curl -fsSL 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg --dearmor -o /etc/apt/keys/caddy.gpg
```

Fetch the Caddy project's repository file by running the following command:

```
curl -fsSL 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | sudo tee /etc/apt/sources.list.d/caddy.list
```

Apply your new repository and make sure your system is completely updated:

```
sudo apt update && sudo apt upgrade
```

Install Docker, Docker Compose and Caddy using apt:

```
sudo apt install docker-ce docker-ce-cli containerd.io docker-compose-plugin doc
```

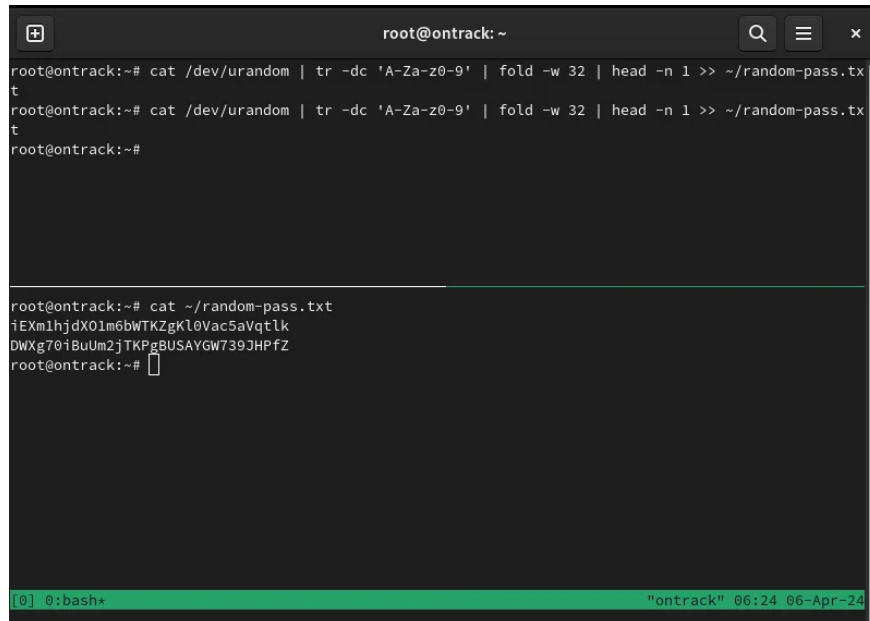
Download Docker containers Ontrack

Fetch the current Ontrack repository for Linux from the developer's Github page:

```
git clone https://github.com/inoda/ontrack.git && cd ./ontrack
```

Run the following command 2 times to generate two random long text strings:

```
cat /dev/urandom | tr -dc 'A-Za-z0-9' | fold -w 32 | head -n 1 >> ~/random-pass.txt
```



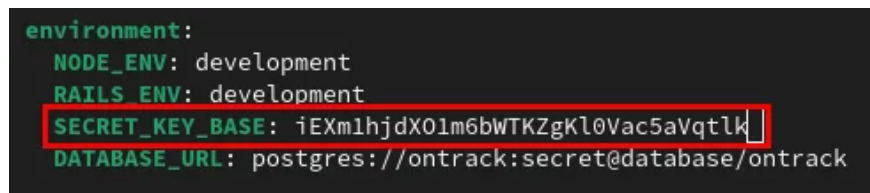
```
root@ontrack:~# cat /dev/urandom | tr -dc 'A-Za-z0-9' | fold -w 32 | head -n 1 >> ~/random-pass.txt
t
root@ontrack:~# cat /dev/urandom | tr -dc 'A-Za-z0-9' | fold -w 32 | head -n 1 >> ~/random-pass.txt
t
root@ontrack:~#

root@ontrack:~# cat ~/random-pass.txt
iEXm1hjdX01m6bWTKZgKl0Vac5aVqtlk
DWXg70iBuUm2jTKPgBUSAYGW739JHPfZ
root@ontrack:~#
```

Open Ontrack's 'docker-compose.yml' file with your favorite text editor:

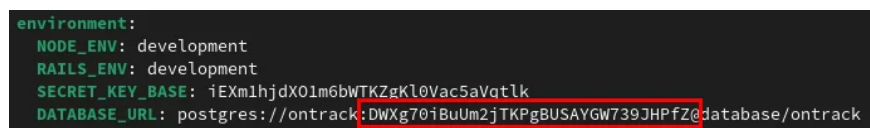
```
nano ./docker-compose.yml
```

Scroll to the variable 'SECRET_KEY_BASE' and change the value from 'super-secret' to your first random string.



```
environment:
  NODE_ENV: development
  RAILS_ENV: development
  SECRET_KEY_BASE: iEXm1hjdX01m6bWTKZgKl0Vac5aVqtlk
  DATABASE_URL: postgres://ontrack:secret@database/ontrack
```

Replace the string 'secret' on the variable 'DATABASE_URL' with your second random string.



```
environment:
  NODE_ENV: development
  RAILS_ENV: development
  SECRET_KEY_BASE: iEXm1hjdX01m6bWTKZgKl0Vac5aVqtlk
  DATABASE_URL: postgres://ontrack:DWXg70iBuUm2jTKPgBUSAYGW739JHPfZ@database/ontrack
```

Scroll down to the 'POSTGRES_PASSWORD' variable and replace the 'secret' value with your second random string.

```
environment:
  NODE_ENV: development
  RAILS_ENV: development
  SECRET_KEY_BASE: iEXm1hjdX01m6bWTKZgKl0Vac5aVqtlk
  DATABASE_URL: postgres://ontrack:DWXg70iBuUm2jTKPgBUSAYGW739JHPfZ@database/ontrack

database:
  image: postgres
  container_name: ontrack-db
  volumes:
    - database:/var/lib/postgresql/data
  environment:
    POSTGRES_PASSWORD: DWXg70iBuUm2jTKPgBUSAYGW739JHPfZ
    POSTGRES_USER: ontrack
    POSTGRES_DB: ontrack
```

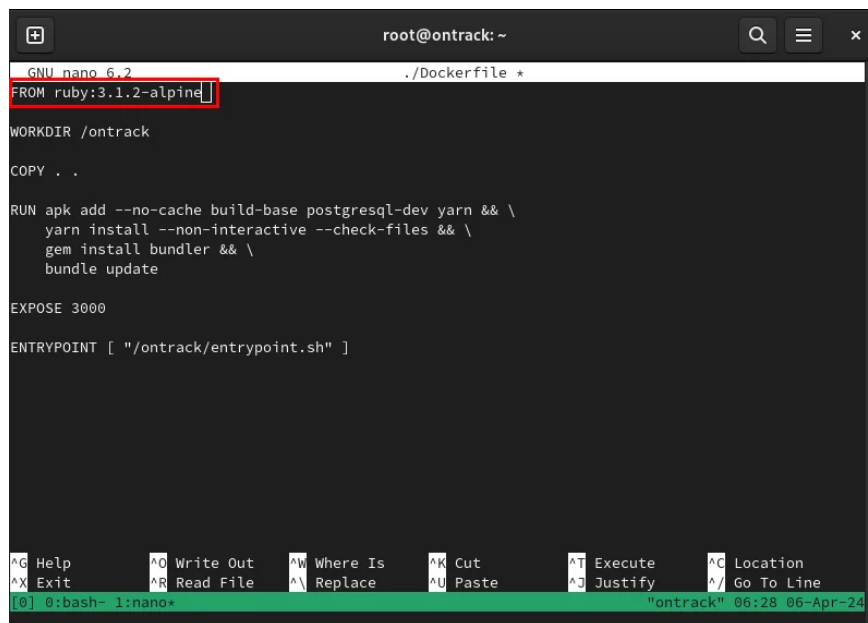
Updating and building Ontrack containers

Open the Dockerfile for Ontrack using your favorite text editor:

```
nano ./Dockerfile
```

Replace the value of the FROM variable with the following value:

```
FROM ruby:3.1.2-alpine
```



```
root@ontrack: ~
GNU nano 6.2 ./Dockerfile *
FROM ruby:3.1.2-alpine
WORKDIR /ontrack
COPY . .
RUN apk add --no-cache build-base postgresql-dev yarn && \
  yarn install --non-interactive --check-files && \
  gem install bundler && \
  bundle update
EXPOSE 3000
ENTRYPOINT [ "/ontrack/entrypoint.sh" ]

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^G Location
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^D Justify    ^_/ Go To Line
[0] 0:bash- 1:nano+ "ontrack" 06:28 06-Apr-24
```

Save your modified Dockerfile, then open the 'package.json' file:

```
nano ./package.json
```

Find the line that starts with **@babel/preset-env**, then insert the following code below that line:

```
"babel-plugin-macros": "^3.0.1",
```

```
root@ontrack: ~
GNU nano 6.2 ./package.json *
{
  "name": "ontrack",
  "private": true,
  "dependencies": {
    "@babel/plugin-transform-runtime": "^7.15.0",
    "@babel/preset-env": "^7.15.4",
    "@babel/plugin-macros": "^3.0.1",
    "@babel/preset-react": "^7.0.0",
    "@rails/actioncable": "^6.0.3-2",
    "@rails/activestorage": "^6.0.3-2",
    "@rails/uj": "^6.0.3-2",
    "@rails/webpacker": "^6.0.0-beta.5",
    "axios": "^0.21.3",
    "babel-plugin-transform-react-remove-prop-types": "^0.4.24",
    "chart.js": "^2.8.0",
    "moment": "^2.29.4",
    "prop-types": "^15.7.2",
    "react": "^16.10.1",
    "react-datepicker": "^2.9.6",
    "react-dom": "^16.10.1",
    "react-text-mask": "^5.4.3",
    "sweetalert2": "^8.18.0",
    "text-mask-addons": "^3.8.0",
    "webpack": "^5.11.0",
  }
}
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
[0] 0:~$ nano+ [0,0] "ontrack" 06:28 06-Apr-24
```

Scroll down to the 'devDependency' category , then add the following to the @babel/eslint-parser line :

```
"@babel/plugin-proposal-object-rest-spread": "^7.15.4" ,
```

```
root@ontrack: ~
GNU nano 6.2 ./package.json *
  "react-dom": "^16.10.1",
  "react-text-mask": "^5.4.3",
  "sweetalert2": "^8.18.0",
  "text-mask-addons": "^3.8.0",
  "webpack": "^5.11.0",
  "webpack-cli": "^4.2.0"
},
"version": "0.1.0",
"babel": {
  "presets": [
    "./node_modules/@rails/webpacker/package/babel/preset.js"
  ]
},
"browserslist": [
  "defaults"
],
"devDependencies": {
  "@babel/core": "^7.15.4",
  "@babel/eslint-parser": "^7.15.4",
  "@babel/plugin-proposal-object-rest-spread": "^7.15.4",
  "eslint": "^7.32.0",
  "eslint-plugin-react": "^7.25.1"
}
}
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
[0] 0:~$ nano+ [0,0] "ontrack" 06:29 06-Apr-24
```

Open your instance configuration file with your favorite text editor:

```
nano ~/ontrack/config/environments/development.rb
```

Add the following line of code right below Rails.application.configure:

```
config.hosts = [ "SUBDOMAIN.YOUR-ROOT.DOMAIN" ]
```

```
root@ontrack: ~
GNU nano 6.2 ./config/environments/development.rb *
Rails.application.configure do
  config.hosts = [
    "ontrack.myvpsserver.top"
  ]
  # Settings specified here will take precedence over those in config/application.rb.

  # In the development environment your application's code is reloaded on
  # every request. This slows down response time but is perfect for development
  # since you don't have to restart the web server when you make code changes.
  config.cache_classes = false

  # Do not eager load code on boot.
  config.eager_load = false

  # Show full error reports.
  config.consider_all_requests_local = true

  # Enable/disable caching. By default caching is disabled.
  # Run rails dev:cache to toggle caching.
  if Rails.root.join('tmp', 'caching-dev.txt').exist?
    config.action_controller.perform_caching = true
    config.action_controller.enable_fragment_cache_logging = true

  end

  # Don't care if the current url is hosted by a non-trusted proxy.
  config.action_controller.allow_forgery_protection = true

  # Tell Rails to render requests from untrusted proxies
  config.action_controller.action_on_untrusted_proxy = :deny

  # Enable/disable Sprockets asset management. By default, it is enabled.
  # config.assets.enabled = true

  # It is recommended to run this plugin at the end of this configuration file
  # so you can enable it locally for a particular environment, e.g.,
  # config.assets.enabled = true
end

^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^_ Go To Line
[0] 0:bash- 1:nano+ "ontrack" 06:31 06-Apr-24
```

Save your instance configuration file, then run the following command to build both its Ontrack and Postgres databases:

```
sudo docker compose up --detach
```

Note : Ontrack Docker container building can take 5 to 10 minutes depending on your server resources. If your machine has less than 2GB RAM and no swap file, Docker will crash without reporting any errors on Terminal.

Confirm that Ontrack containers are running properly by listing all active Docker processes:

```
sudo docker ps
```

```
root@ontrack: ~/ontrack# sudo docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
b2a04a5058a6  ontrack-app   "/ontrack/entrypoint..." 12 seconds ago Up 10 seconds 127.0.0.1:3
000->3000/tcp
1089e2e755ce  postgres      "docker-entrypoint.s..." 12 seconds ago Up 11 seconds 5432/tcp
ontrack-db
root@ontrack: ~/ontrack#
```

Create SSL Reverse Proxy with Caddy

At this point, you have Ontrack budgeting software running on port 3000 of your machine. To access this software securely, you need to create an SSL Reverse Proxy that encrypts the connection between you and your server.

Go to your domain's DNS manager, then add a new 'A' record for your Ontrack instance.

DNS records

Type	Hostname	Value	TTL (seconds)	
A	ontrack.myvpserver.top	directs to 178.128.84.109	30	More ▾

Backup the original Caddyfile, then create a new Caddyfile in '/etc/caddy':

```
sudo mv /etc/caddy/Caddyfile ~/Caddyfile.backup sudo nano /etc/caddy/Caddyfile
```

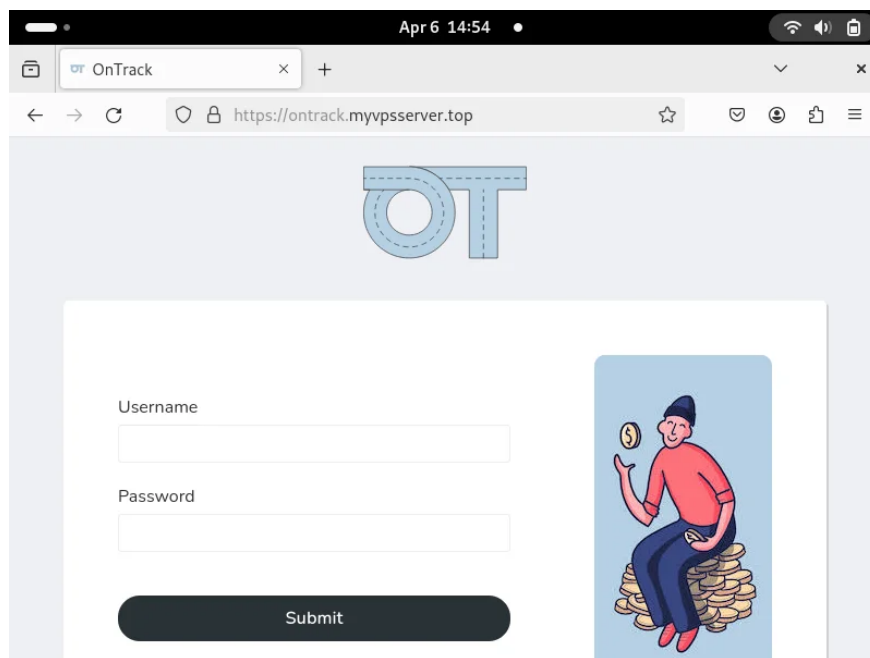
Paste the following block of code into your new Caddyfile:

```
SUBDOMAIN.YOUR-ROOT.DOMAIN { reverse_proxy :3000 }
```

Save your new Caddyfile, then start the Caddy daemon to run your new reverse proxy:

```
sudo systemctl enable --now caddy.service
```

Test if your SSL Reverse Proxy is working by navigating to your URL.



Use Ontrack and create a user account

Once Ontrack is up and running, you can now create your user account. To do this, return to the server's terminal window, then open the shell for the Ontrack container:

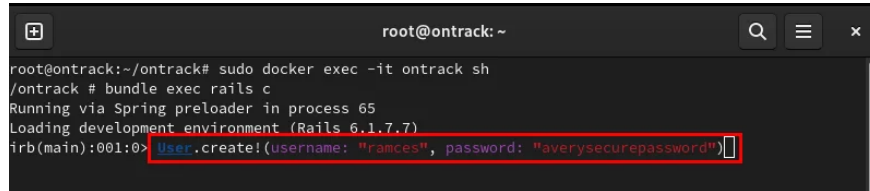
```
sudo docker exec -it ontrack sh
```

Open the database handler inside your Ontrack container:

```
bundle exec rails c
```

Create your new user account by running the following command:

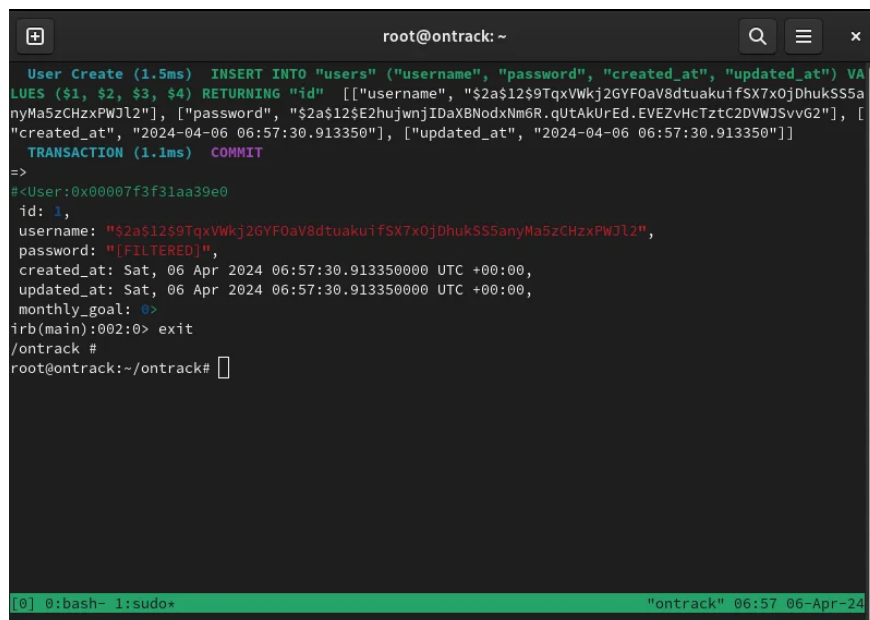
```
User.create!(username: "YOUR-USERNAME", password: "YOUR-SECURE-PASSWORD")
```



```
root@ontrack: ~  
root@ontrack:~/ontrack# sudo docker exec -it ontrack sh  
/ontrack # bundle exec rails c  
Running via Spring preloader in process 65  
Loading development environment (Rails 6.1.7.7)  
irb(main):001:0> User.create!(username: "ramces", password: "averysecurepassword")
```

Type **'exit'** , then press **Enter** to exit the database handler.

Press **Ctrl + D** to leave the Docker container's root shell.



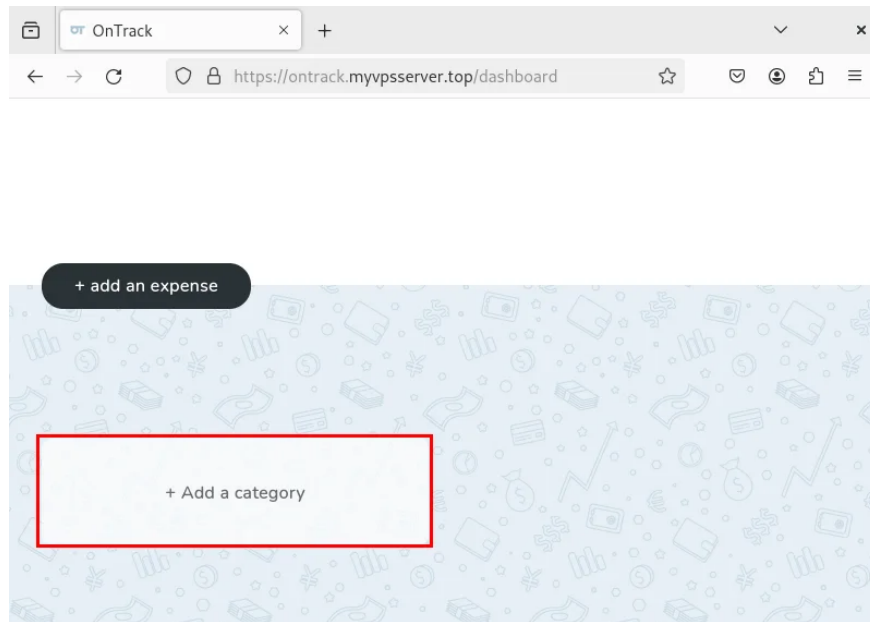
```
root@ontrack: ~  
User Create (1.5ms) INSERT INTO "users" ("username", "password", "created_at", "updated_at") VA  
LUES ($1, $2, $3, $4) RETURNING "id" [["username", "$2a$12$9TqxVWkj2GYFOaV8dtuakuiFSX7x0jDhukSS5a  
nyMa5zCHzxPWJl2"], ["password", "$2a$12$E2hujwnjIDaXBNodxNm6R.qUtAkUrEd.EVEZvHcTztC2DvWJSyv62"], [  
"created_at", "2024-04-06 06:57:30.913350"], ["updated_at", "2024-04-06 06:57:30.913350"]]  
TRANSACTION (1.1ms) COMMIT  
=>  
#<User:0x00007f3f31aa39e0  
id: 1,  
username: "$2a$12$9TqxVWkj2GYFOaV8dtuakuiFSX7x0jDhukSS5anyMa5zCHzxPWJl2",  
password: "[FILTERED]",  
created_at: Sat, 06 Apr 2024 06:57:30.913350000 UTC +00:00,  
updated_at: Sat, 06 Apr 2024 06:57:30.913350000 UTC +00:00,  
monthly_goal: 0>  
irb(main):002:0> exit  
/ontrack #  
root@ontrack:~/ontrack#
```

Test your new account by opening Ontrack in your web browser and logging in to your account.

Create your first trade on Ontrack

To use Ontrack to record transactions, you need to create an expense category. This allows the web app to collate your expenses by group, making it easy to deduce insights into your spending habits.

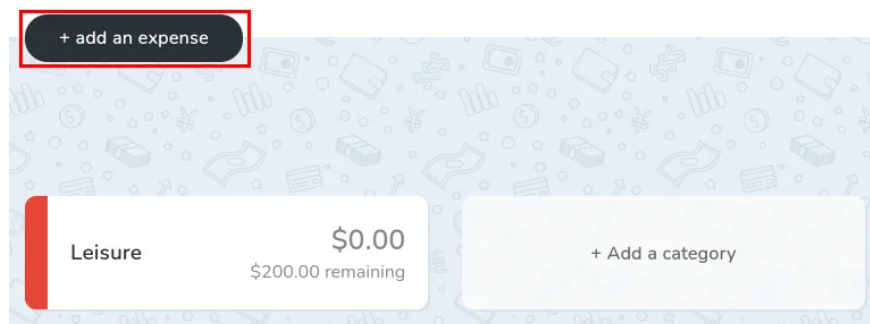
Scroll down to the Ontrack dashboard, then click **Add a category** .



Provide the category name, tag color, and whether the category has spending limits.

A screenshot of a 'Create Category' modal window. It features a close button (X) in the top right corner. The form includes three main sections: 1) 'Name *' with a text input field containing 'Leisure'. 2) 'Color *' with a 3x3 grid of color swatches; the red swatch in the middle row is selected with a white border. Below the grid is a button labeled 'Enter hex instead'. 3) 'Monthly goal' with a text input field containing '\$200'. At the bottom right of the modal is a dark 'Save' button.

Click **Add an expense** on the Ontrack Dashboard.



Fill in your expense details, then click **Save** to transfer it to your Ontrack instance.

✕

Create Expense

Amount *

Category * Date *

Description *

[Import a CSV instead](#)

Finally, confirm that Ontrack successfully saved the transaction by checking your expense history. To do that, scroll up the page, then click **the History** link in the upper right corner of the page.

OT dashboard import history insights logout

All categories Last 90 days Search for description

Date	Category	Amount	Description
04/06/2024	Leisure	\$20.00	Hello, Ma

Showing 1-1 of 1

Installing and implementing your own expense tracking software is just the first step to taking back control of your online and digital life. Learn how you can ensure your privacy when sending emails by installing an alias server like SimpleLogin.

You finished reading the article "**How to install OnTrack software on Linux**" edited by the [TipsMake](#) team. We hope this article has provided you with many useful tech tips and tricks. You can search for similar articles on tips and guides. Thank you for reading and for following us regularly.