

Step 1: Getting Started

CapRover can be installed on various cloud providers, including **AWS (Amazon Web Services)**, **Google Cloud**, **Azure**, and **DigitalOcean**. The recommended approach is to select a server that allows SSH access, making it easy to manage your installation.

Video Guide

Step1 Getting Started



Creating a Server

Begin by creating a server instance on your chosen cloud provider. Ensure that the server has the following specifications for optimal performance:

- **At least 2GB of RAM**
- **1 CPU**

This configuration will provide sufficient resources for running applications effectively. Most cloud providers allow you to upgrade your server resources later if needed.

i Through out this setup, we've used DigitalOcean as our cloud provider, however you can use any cloud provider as stated earlier. You can do same using this link: [Get](#)

Selecting a Server Location

When selecting a location for your server, choose a region that minimizes latency by being geographically closer to you. This will enhance performance and responsiveness.

Setting Up SSH Key

For secure and convenient access to your server, it's highly recommended to use an SSH key. This simplifies authentication and enhances security.

i If your cloud provider gives you Remote Access through browser then you can avoid below given steps.

Here's how to set it up:

1. Generate an SSH Key (if you don't have one):

- Open your terminal.
- Run the command:

```
ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
```

- Accept the default file location by pressing **Enter**.
- Optionally, set a passphrase for extra security.

2. Copy Your SSH Key:

- Use one of the following methods to copy your public key:
 - **Command Line:**

```
cat ~/.ssh/id_rsa.pub
```

Copy the output manually.

- **File Location:** Navigate to your file location & open `~/.ssh/id_rsa.pub` using a text editor and copy the contents.

3. Add SSH Key to Your Provider:

- Log in to your cloud provider's dashboard.
- Locate the section for SSH keys (often found under **Security** or **Settings**).
- Paste your copied SSH key into the designated field and give it a recognizable name.

Once your SSH key is set up, you're ready to proceed with installing CapRover and configuring your server. Remember to **note your server's IP address** for future use.

Last updated on October 11, 2024

We expect that you have created Server on your cloud provider. Now it's time to access the server and install the Caprover.

Video Guide

Step2 General Configuration



Step 2: CapRover Installation

First, open the terminal or console of your **Ubuntu server**. You can do this by using SSH to connect to your server. or use browser based SSH to connect. For example:

```
ssh root@[YOUR_SERVER_IP]
```

Docker Installation:

Run the following command to uninstall all conflicting packages:

```
for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg;
```

Installing using `apt` Repository

Step 1: setup docker's `apt` Repository

Copy and paste the given commands to terminal.

```
# Add Docker's official GPG key:
```

```
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \
$(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

Step 2: Install the Docker packages.

Copy and paste below command to your server/SSH terminal.

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

Step 3: Verify Docker Installation.

to verify the installation run following command.

```
sudo docker run hello-world
```

This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

Step 4: Configuring Firewall

Copy and paste the below given command to SSH terminal, to modify firewall settings.

Note: Some providers like AWS restricts the firewall settings, However they provide dedicated page to manage Inbound & Outbound Ports. You'll have to look for the your cloud provider's guide to setup the firewall.

For DigitalOcean it's straight forward, just paste the below given command, and that's it.

```
ufw allow 80,443,3000,996,7946,4789,2377,3306,22/tcp; ufw allow 7946,4789,2377/udp;
```



You have now successfully installed and started Docker Engine.

Installing Caprover using Docker:

Once Docker is installed, run the following command to install CapRover:

```
docker run -p 80:80 -p 443:443 -p 3000:3000 -e ACCEPTED_TERMS=true -v /var/run/docker.sock:/var/run/docker.sock -v /captain:
```

⚠️ NOTE: Do not change the port mappings. CapRover only works on the specified ports.

You will see a bunch of outputs on your screen. Once CapRover is initialized, visit `http://[IP_OF_YOUR_SERVER]:3000` in your browser and log in to CapRover using the **default password** `captain42` . You can change your password later. However, do not make any changes in the dashboard. We'll use the command line tool to set up the server (recommended).

STEP 3 : Custom Domain Configuration

To set up your own domain to point to your server, follow these steps:

1. Find Your Server's External IP Address:

- You can find your server's IP address in your hosting provider's control panel. This is referred to as the **Primary IP / Public IP** or **External IP**.

2. Login to Your Domain Registrar:

- Go to the website where you purchased your domain (e.g. Hostinger, GoDaddy, Namecheap).
- Locate the **DNS management panel** for your domain. This is where you'll configure your domain records.

3. Add an A Record:

- You'll need to add an **A Record** to point your domain to your server's external IP address.
- Create an A Record for a **wildcard subdomain** (e.g., `*.yourdomain.com`) to point to your server's IP.

For example, if your server's Public IP is `45.210.99.200` and you're using a wildcard subdomain like `*.apps` , your A Record will look like this:

Type	Host	Value	TTL
A Record	*.apps	45.210.99.200	Automatic
A Record	backend	45.210.99.200	Automatic
A Record	@	45.210.99.200	Automatic

4. Finish DNS Setup:

- Once the A Records are added, DNS is configured, and you won't need to change anything further in your domain settings.

That's it! You're ready to move on to the next step in setting up your server.

Set Up a Swap File (Optional)

If your server doesn't have enough RAM, you might encounter issues, especially when building Docker images. Instead of purchasing more RAM, you can create a swap file, which acts as virtual memory. Here's how you can add 1GB of swap to your server (you can adjust the size by replacing `1G` with `2G` for 2GB, etc.):

Video Guide

Step3 1 Swap file



In your server SSH Terminal, paste the following commands to create SWAP.

Steps to Add 1GB Swap File:

1. Create the Swap File:

```
sudo fallocate -l 1G /swapfile
```

If you get an error with `fallocate`, use this alternative command:

```
sudo dd if=/dev/zero of=/swapfile bs=1024 count=1048576
```

2. Set the Correct Permissions (only root should access it):

```
sudo chmod 600 /swapfile
```

3. Set Up the Swap Area:

```
sudo mkswap /swapfile
```

4. Enable the Swap:

```
sudo swapon /swapfile
```

To make the change permanent open the `/etc/fstab` file and append the following line:

```
/swapfile swap swap defaults 0 0
```

5. Verify Swap Activation:

```
sudo swapon --show
```

Example output:

NAME	TYPE	SIZE	USED	PRI0
/swapfile	file	1024M	507M	-1

You can also use:

```
sudo free -h
```

Example output:

total	used	free	shared	buff/cache	available	
Mem:	488M	158M	83M	2.3M	246M	217
Swap:	1.0G	506M	517M			



This step is optional, but it helps avoid memory-related issues, especially on servers with limited RAM.

Step 4: Setting Up CapRover CLI

To manage your CapRover installation, you need to install Node.js on your local machine. Node.js allows you to access NPM (Node Package Manager), which is essential for installing the CapRover CLI. You can download Node.js from the [official website](#).

Video Guide

Step4 Caprover serversetup & CLI



Install CapRover CLI

After installing Node.js, open your terminal and run the following command to install the CapRover CLI:

```
npm install -g caprover
```

Configure Your Server

Once the CapRover CLI is installed, you'll need to set up your server. Use the following command to start the setup process:

```
caprover serversetup
```

During this process, you'll be prompted to enter your root domain. This root domain should match the domain or subdomain you configured to point to your server's IP address.

For example, if you set up ***.apps.yourdomain.com** as a wildcard domain in your DNS settings, your root domain would be **apps.yourdomain.com**. This allows CapRover to manage subdomains like **captain.apps.yourdomain.com** for the admin dashboard and other apps.

This command will guide you through several steps to configure your server. Remember to choose a secure password during this process.

Logging into CapRover

Now that everything is configured, you can log in to your CapRover server. Make sure your domain points to your server's IP address. To access the dashboard, go to:

```
https://captain.apps.yourdomain.com
```

Replace **yourdomain.com** with your actual domain name. Use the password you set during the CLI setup to log in.

Last updated on October 11, 2024

Step 5: Deploy MySQL Database

Video Guide

Step5 Setup Database



5.1 Create a New App

- Login to your Caprover dashboard by visiting `https://captain.apps.yourdomain.com`, replace `yourdomain.com` with your actual domain.
- In the Caprover dashboard, click on **Apps** in the left menu.
- Navigate to **One-Click Apps/Databases**.

Create A New App

my-amazing-app

Create New App

☐ Has Persistent Data 

Or Select From

One-Click Apps/Databases

- Search for or select **MySQL** from the list of available apps.

One Click Apps

Choose an app, a database or a bundle (app+database) from the list below. The rest is magic, well... Wizard!

One click apps are retrieved from the official [CapRover One Click Apps Repository](#) by default. You can add other public/private repositories if you want to.

mysql



Directus + MySQL + Redis

Directus is an open-source headless CMS and an Open Data Platform built to democratize the database. It provides a dynamic API and intuitive Admin Panel App.



Mysql-Backup

mysql-backup is a simple way to do MySQL database backups and restores when the database is running in a container.



MySQL

MySQL is a relational database management system based on SQL



Redmine (MySQL)

Redmine is a flexible project management web application written using Ruby on Rails framework. This app is packaged with MySQL.



- Fill in the required details:
 - **App Name:** (e.g., `mysql-db`)
 - **MySQL Root Password:** Choose a strong password.
 - **MySQL Version:** Prefer the LTS version for stability.
- Click **Deploy** to initiate the process.

App Name

This is your app name. Pick a name such as my-first-1-click-app

MySQL Version

Check out their Docker page for the valid tags <https://hub.docker.com/r/library/mysql/tags/>

MySQL Root password

Deploy

5.2 Access Database Credentials

- After the deployment is complete, return to the **Apps** section.
- Click on your newly created `mysql-db` app.
- Navigate to **App Configs** to set up the port mapping. Add the necessary port, then click **Save** and **Restart** to apply the changes.

Port Mapping ⓘ

Server Port	Container Port
3306	3306

Add Port Mapping

Persistent Directories ⓘ

Path in App	Label	
/var/lib/mysql	mysql-db-db-data	Set specific host path

Add Persistent DirectoryDelete AppSave & Restart

- Next, Go to **HTTP Settings**:
 - Remove the option to not expose as a web app.
 - Enable **HTTPS** for secure connections.

Your app is internally available as `srv-captain--mysql-db-db` to other Captain apps. In case of web-app, it is accessible via `http://srv-captain--mysql-db-db` from other apps.

☐ Do not expose as web-app ⓘ

Your app is publicly available at:

<https://mysql-db-db.apps.minimalpos.com>

ⓘ

- Save & Restart to Apply Changes

5.3 Connect to Your MySQL Database

- Use a MySQL client like MySQL Workbench or any other MySQL Client to connect from your application using the following credentials:
 - **Hostname:** Displayed in **HTTP Settings** beside **Enable HTTPS**.

 `mysql-db-db` 

Your app is internally available as `srv-captain--mysql-db-db` to other Captain apps.

☐ Do not expose as web-app ⓘ

Your app is publicly available at:

<https://mysql-db-db.apps.minimalpos.com>

- **Port:** `3306` (the default MySQL port).
- Create a new connection using the credentials you set earlier.

Setup New Connection

Connection Name: Type a name for the connection

Connection Method: Method to use to connect to the RDBMS

Parameters SSL Advanced

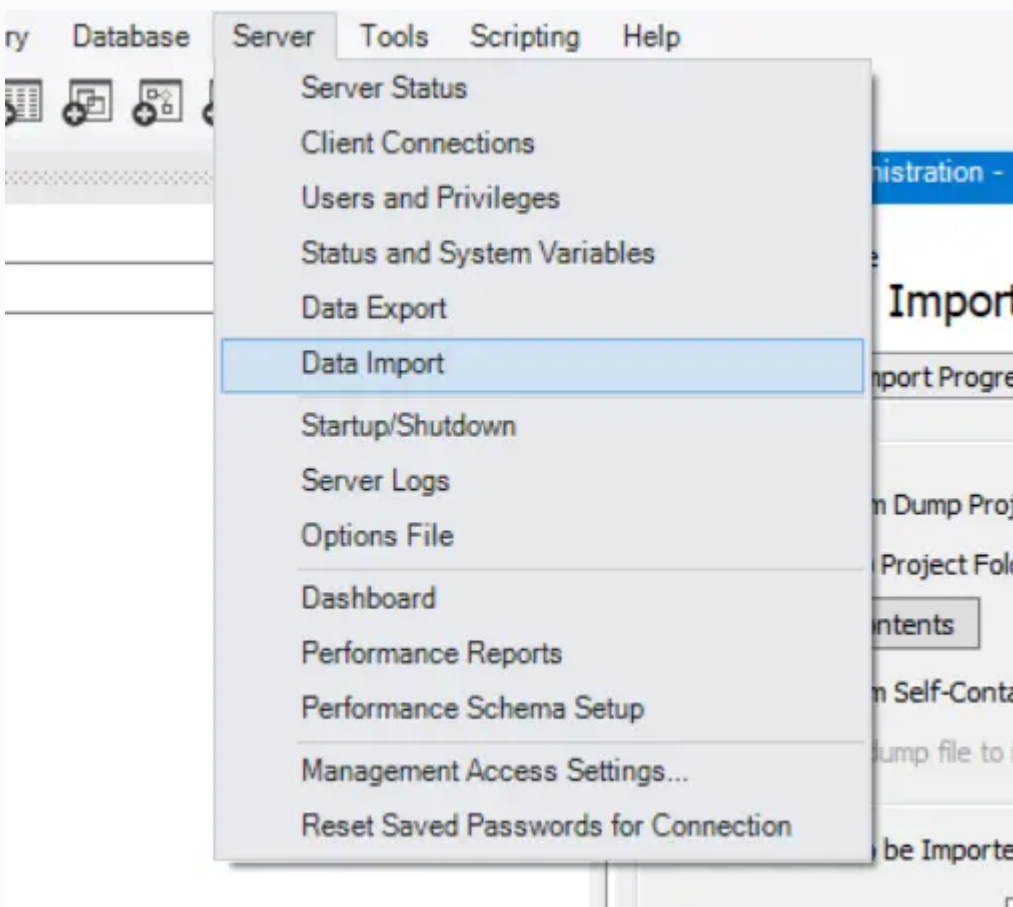
Hostname: Port: Name or IP address of the server host - and TCP/IP port.

Username: Name of the user to connect with.

Password: The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

- After successfully setting up the connection, import your database file (e.g., `restropro_saas.sql`). Which is provided in the zip file that you downloaded.





example
Data Import

Import from Disk

Import Progress

Import Options

☐ Import from Dump Project Folder

D:\Documents\dumps

...

Select the Dump Project Folder to import. You can do a selective restore.

Load Folder Contents

☒ Import from Self-Contained File

E:\UIFlow\restropro-saas\restropro_saas.sql

...

Select the SQL/dump file to import. Please note that the whole file will be imported.

Default Schema to be Imported To

Default Target Schema:

restropro_saas

▼

New...

The default schema to import the dump into.
NOTE: this is only used if the dump file doesn't contain its schema,
otherwise it is ignored.

Select Database Objects to Import (only available for Project Folders)

Imp... Schema

Imp... Schema Objects

Dump Structure and Dat ▼

Select Views

Select Tables

Unselect All

Press [Start Import] to start...

Start Import

Congratulations! Your MySQL database has been successfully deployed and is ready for use.
Enjoy managing your data with ease!

5.4: Add SuperAdmin User

To create superadmin user, first you will need to create encrypted password. to do so visit [bcrypt-generator](#) and set the number of **rounds to 10** . Use this generated password in the insert query.

\$2a\$10\$BB0NRi.4IM8W XK.vG8.2y.bBikAD/qjPup8ay8j8WoN/AnYzuuMve

Encrypt

Rounds

— 10 +

In the below given SQL Query, replace the following details:

YOUR_EMAIL

YOUR_ENCRYPTED_PASSWORD

YOUR_NAME

```
INSERT INTO `superadmins`  
VALUES  
(  
  'YOUR_EMAIL',  
  'YOUR_ENCRYPTED_PASSWORD',  
  'YOUR_NAME'  
);
```

Now open your MySQL Client, and paste & run this query.

  You've successfully created superadmin user. to access the SuperAdmin Dashboard, your superadmin URL will look like this
<https://yourdomain.com/superadmin>

However, we still have Backend and Frontend setup pending so the URL will not be accessible yet.

Last updated on October 11, 2024

Step 6: Deploy Your Backend Application

Video Guide

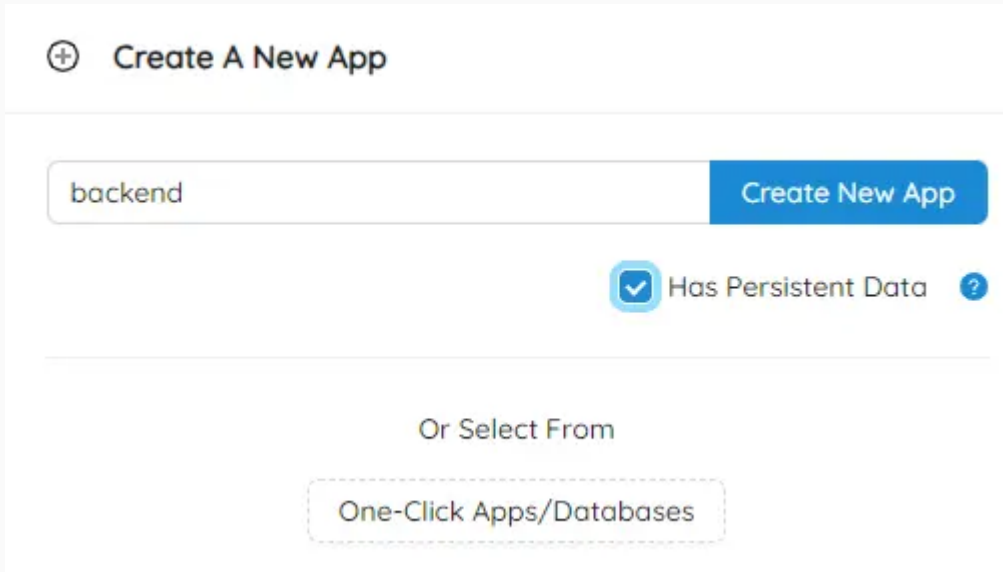
Step6-Setup Backend



6.1 Create a New App

- In the dashboard, click on **Apps** in the left menu.
- Click on the **Create New App** button.
- Enter a name for your app (e.g., `backend`) and click **Create New App**.

- Ensure you select the option for persistent data.



⊕ Create A New App

backend Create New App

☒ Has Persistent Data ?

Or Select From

One-Click Apps/Databases

6.2 Setup .env file in App Config

- Configure Your Environment Variables: Each environment variable defines specific settings for your app. Here's a general breakdown of the important variables and how they might change based on your app's domain:

6.2.1 Database URL

- This URL connects your app to the database. It typically includes the database type (mysql in our case), username, password, host, port, and database name.

```
DATABASE_URL='mysql://[username]:[password]@[host]:[port]/[database_name]'
```

Replace the host, password as per configured in new app created. We get the host in HTTP Settings of the created DB (mysql-db) app as shown.



Eg :

```
DATABASE_URL='mysql://root:YOURPASSWORD@srv-captain--mysql-db-db:3306/restropro_saas'
```

6.2.2 Jwt Secret

- The JWT_SECRET is used to sign JSON Web Tokens (JWT) for authentication. Set this value as a secure, unique string to ensure the safety of your app

6.2.3 Frontend Domain & Cookie

- These values define your frontend's URL and its cookie domain. Update these according to the domain name where your frontend is hosted. For example: If frontend is hosted on `https://yourdomain.com`, the cookie value will be `yourdomain.com`.

```
FRONTEND_DOMAIN="https://yourdomain.com"  
FRONTEND_DOMAIN_COOKIE="yourdomain.com"
```

6.2.4 Stripe Credentials

- To configure subscriptions for the app, follow the mentioned steps & then get your stripe credentials - STRIPE_SECRET & STRIPE_WEBHOOK_SECRET as required.

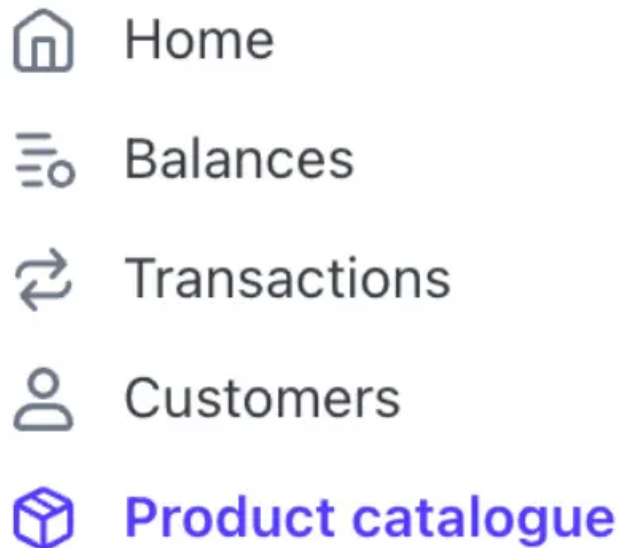
6.2.4.1 Stripe Subscription Setup

- Goto [Stripe](#) and set up your account according to your business registration type. Once you activate your account, follow the given procedure.
- Goto Stripe Dashboard and Select "Product catalogue", then Add new product.

2. Stripe Subscription Setup

Goto stripe.com and set up your account according to your business registration type. Once you activate your account, follow the given procedure.

Goto Stripe Dashboard and Select "Product catalogue", then Add new product.



- Fill the details as per your business requirements.

Add a product

Close preview ×

Name (required)

Name of the product or service, visible to customers.

My Subscription

Description

Appears at checkout, on the customer portal, and in quotes.

Monthly Subscription

Image

Appears at checkout. JPEG, PNG or WEBP under 2MB.

Upload

More options ▾

Recurring

One-off

Amount (required)

US\$ 10.00

USD

Billing period

Monthly

More pricing options

Cancel

Add product

Preview

Estimate totals based on pricing model, unit quantity, and tax.

Unit quantity

1

1 × US\$10.00 = **US\$10.00**

Subtotal **US\$10.00**

Tax [Start collecting tax](#)

Total per month **US\$10.00**

Billed at the start of the period

- Once you create the Subscription Product, you will see it like this.

Product catalogue

+ Add product

All products

Features

Coupons

Shipping rates

Tax rates

Pricing tables

Meters

All
2

Active
2

Archived
0

⊕ Created

⊕ Status

📄 Export prices

📄 Export products

⚙️ Edit columns

Name	Pricing	Created	Updated	
 RestroPRO Month	US\$5.00 🔄 Per month	7 Jun	7 Jun	...
 RestroPRO Subscription	US\$5.00 🔄 Per month	5 Jun	5 Jun	...

2 results

- Now go back to Stripe Dashboard and open the developers page, you will get API keys from there. Look for a secret key, and copy it. This will be your `STRIPE_SECRET`.

UIFLOW

Search

Developers

Overview **API keys** Webhooks Events Logs Apps

API keys [Learn more about API authentication →](#)

Standard keys
Create a key that unlocks full API access, enabling extensive interaction with your account. [Learn more](#)

NAME	TOKEN	LAST USED	CREATED
Publishable key		10 Jun	12 May
<u>Secret key</u>	Reveal test key	11 Jun	13 May

Restricted keys
Create a key with specific access limits and permissions for greater security. [Learn more](#) [+ Create restricted key](#)

NAME	TOKEN	LAST USED	CREATED
No restricted keys			

- Now we will set up the webhook. For that open Stripe Dashboard and goto developers page again. Goto Webhooks Tab, then click on Add endpoint.

Developers

Overview API keys **Webhooks** Events Logs Apps

[Try Workbench](#) View, create, inspect, and edit your webhooks from anywhere in the dashboard. [Learn more](#) [×](#)

Hosted endpoints [+ Add endpoint](#)

- Then provide the webhook URL, this will be our backend route webhook handler, so we're yet to setup the backend, so you just need to provide the url. For example if i'm setting Backend on domain "api.example.com" then the webhook URL will look like

this.Webhook URL: <https://api.example.com/api/v1/auth/stripe-webhook>. Replace the domain, to your backend domain. After placing endpoint URL, click on the 'select events to listen to'

× | Listen to Stripe events

Listen to Stripe events

[Add an endpoint](#) [Test in a local environment](#)

Set up your webhook endpoint to receive live events from Stripe or [learn more about webhooks](#).

Endpoint URL

Description

Listen to

☒ Events on your account ☐ Events on Connected accounts ⓘ

Select events to listen to

[+ Select events](#) ←

[Add endpoint](#) [Cancel](#)

- Look for the following events and select them.

customer.subscription.created

customer.subscription.deleted

customer.subscription.paused

customer.subscription.pending_update_applied

customer.subscription.pending_update_expired

customer.subscription.resumed

customer.subscription.trial_will_end

customer.subscription.updated

- Once you add it open the created webhook endpoint in Stripe Dashboard.
- Click on the 'Reveal' to copy the Webhook Signing Secret.

Status

Enabled

Listening for

8 events

API version

2024-04-10 ⓘ

Signing secret

Reveal

Configuration

View logs

- Paste the copied secret in STRIPE_WEBHOOK_SECRET

```
STRIPE_SECRET=YOUR_STRIPE_SECRET
```

```
STRIPE_WEBHOOK_SECRET=YOUR_STRIPE_WEBHOOK_SECRET
```

6.2.5 SMTP Credentials

- These values configure email sending. Adjust them based on your SMTP provider. Update the host, port, username, and password to match your email service.

Sample .env

```
DATABASE_URL=mysql://root:thisismypassword@srv-captain--mysql-db-db:3306/restropro_saas
JWT_SECRET=SECRET
JWT_EXPIRY=15m
JWT_EXPIRY_REFRESH=30d
COOKIE_EXPIRY=300000
COOKIE_EXPIRY_REFRESH=2592000000
PASSWORD_SALT=10
FRONTEND_DOMAIN=https://yourdomain.com
FRONTEND_DOMAIN_COOKIE=yourdomain.com
STRIPE_SECRET=sk_live_51PFbvQSCWiCS3BjbcjbcjoWGPCZkgLEmtRksf0oTpbRwPQ0H840rD0r000QNu0Zkm
STRIPE_WEBHOOK_SECRET=whsec_nqjVQsMQP7PHshrWbkTbdjsbjvhZWl33W85sWaMz
SMTP_HOST=sandbox.smtp.mailtrap.io
SMTP_PORT=2525
SMTP_EMAIL=4d049cab28855f
SMTP_PASSWORD=d388ajdsb2ddfae7
ENCRYPTION_KEY=uiflow
```

- Navigate to the **App Configs** section of your newly created app.
- Paste the contents of your `.env` file into the environmental variables section using **Bulk Edit** option
- ensure all environment variables are set correctly.

backend

HTTP Settings

App Configs

Deployment

Environmental Variables

Bulk Edit ☒

DATABASE_URL=mysql://root:database@srv-captain--mysql-db-db:3306/restropro_saas

JWT_SECRET=restro_jwt_secret

JWT_EXPIRY=15m

JWT_EXPIRY_REFRESH=30d

COOKIE_EXPIRY=300000 # 15 minute

COOKIE_EXPIRY_REFRESH=259200000 # 30 day

PASSWORD_SALT=10

FRONTEND_DOMAIN=https://minimalpro.com

- Also add the persistent directory as shown

Persistent Directories

Path in App

/usr/src/app/public

Label

backend

Set specific host path

Add Persistent Directory

6.3 Connect New Domain in HTTP Settings

- Navigate to the **HTTP Settings** section.
- Enter your custom domain (e.g., `backend.yourdomain.com`) and click on **Connect New Domain**.
- Ensure to enable **HTTPS** for secure communication.
- Select the option for **Force Https & WebSocket Support** to allow real-time communication.

Your app is internally available as `srv-captain--backend` to other Captain apps. In case of web-app, it is accessible via `http://srv-captain--backend` from other apps.

☐ Do not expose as web-app ⓘ

Your app is publicly available at:

[Enable HTTPS](#) <https://backend.apps.minimalpos.com>

[Enable HTTPS](#) [Remove](#) backend.minimalpos.com

Redirect all domains to: No redirects ▾

[Connect New Domain](#) ⓘ

[Edit Default Nginx Configurations](#)

[Container HTTP Port](#)

☒ Force HTTPS by redirecting all HTTP traffic to HTTPS ⓘ

☒ Websocket Support ⓘ

[Edit HTTP Basic Auth](#) Current State: **inactive**

6.4 CapRover Deploy

Here are two convenient methods for deploying your application to CapRover: Drag & Drop, which allows for quick uploads, and Git, which enables seamless version control integration. Choose the method that best fits your workflow. However caprover supports more methods of deployments, you can explore more at [here](#).

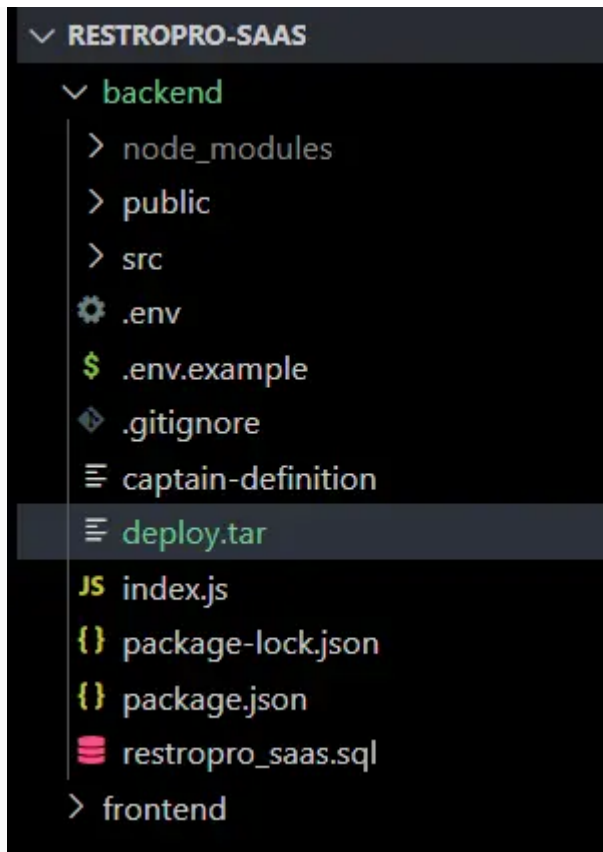
6.4.1 Method 1: Drag & Drop

Package Your Application

- Open the terminal preferably Git Bash. Navigate to where your backend folder is located.
- Create a tarball (.tar) containing the source code and necessary files for deployment (e.g., Dockerfile, app files, etc.) using the below command.

```
tar -cvf ./deploy.tar --exclude='.map' --exclude='node_modules' --exclude='.git' ./captain-definition ./
```

- A new file deploy.tar will be created in the same directory. This is the file to be uploaded for deployment.



Drag & Drop to deploy

- Go the deployments & Scroll to Tarball section.

🔗 Method 2: Tarball

You can simply create a tarball (`.tar`) of your project and upload it here via upload button.



Click or drag TAR file to this area to upload

Must contain `captain-definition` file.

Upload & Deploy

- Simply drag your generated file `deploy.tar` into the “Upload & Deploy” section of the app page.

🔗 Method 2: Tarball

You can simply create a tarball (`.tar`) of your project and upload it here via upload button.



Click or drag TAR file to this area to upload

Must contain `captain-definition` file.

 `deploy.tar`

Upload & Deploy

- CapRover will handle the deployment process based on your uploaded package.

6.4.2 Method 2: Using Git

This method uses git to deploy your backend application to CapRover with the latest changes from your selected branch.

- Open your terminal preferably Git Bash and navigate to the backend folder where your Git repository is configured.
- Make sure to select the appropriate branch (e.g., `main`) that you want to push changes from.
- Run the following command to deploy your application:

```
caprover deploy
```

- On Success, you will get this output.

```
Deployed successfully backend  
App is available at https://backend.apps.minimalpos.com
```



Congratulations! Your backend application has been successfully deployed on CapRover and is ready to handle requests.
Enjoy building and scaling your application with ease!

Last updated on October 12, 2024

Step 7: Deploy Your Frontend Application

Video Guide

Step7 Frontend



Optional: Customize Your App's Logo and Name

Before proceeding with the deployment, you might want to customize your app's branding by changing the logo and app name. This step is completely your wish but can help personalize your application for your business.

Changing the Logo

- Go to vite.config.js file & change the icons.

i In the `vite.config.js` file, two different logo sizes are specified for use in various contexts, such as web app icons or progressive web app (PWA) installations:

- **192x192 logo:** This smaller version (`logo_192.png`) is typically used for mobile and app icons. Its size makes it optimal for displaying on smaller screens or as a favicon in browsers.
- **512x512 logo:** The larger version (`logo.png`) is used for high-resolution displays or larger contexts, such as app splash screens or more detailed icons.

```
icons: [  
  {  
    src: "/logo_192.png",  
    sizes: "192x192",  
    type: "image/png",  
  },  
  {  
    src: "/logo.png",  
    sizes: "512x512",  
    type: "image/png",  
  },  
],
```

- Next, change the favicon from `index.html` file. To change the favicon, simply replace the `favicon.png` file with your desired image.

 The favicon is the small icon displayed in browser tabs and bookmark

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/png" href="/favicon.png" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Restro PRO - App for Restaurant, Cafe, Hotel, Bar.</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.jsx"></script>
  </body>
</html>
```

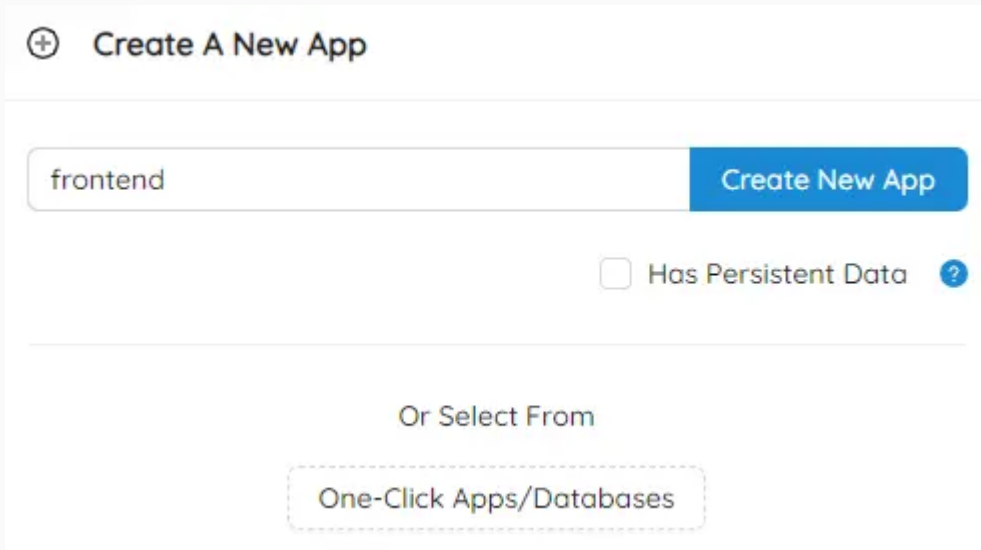
Updating the App Name

- To customize the app's name, simply open the `index.html` file and modify the `<title>` tag. This will update the name displayed in the browser tab when users visit your app. Adjust it to reflect your app's branding or purpose.

7.1 Create a New App

- In the dashboard, click on **Apps** in the left menu.

- Click on the **Create New App** button.
- Enter a name for your app (e.g., `frontend`) and click **Create New App**.



⊕ Create A New App

frontend **Create New App**

☐ Has Persistent Data ?

Or Select From

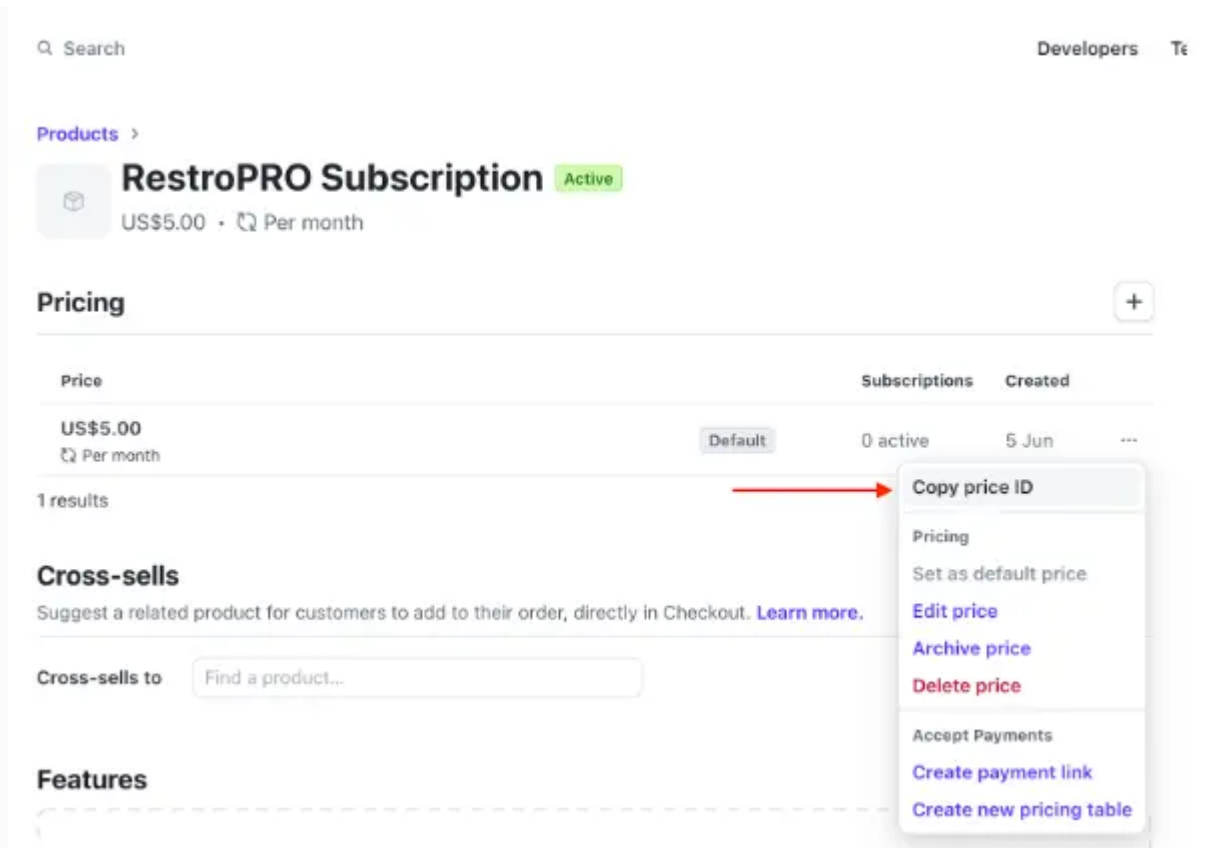
One-Click Apps/Databases

7.2 Setup .env file in App Config

- Configure Your Environment Variables: Each environment variable defines specific settings for your app. Here's a general breakdown of the important variables and how they might change based on your app's domain

7.2.1 Stripe Credentials

- Go to stripe & click on your created Subscription Product as made earlier. Look for Pricing, and click on options three dots.
- Set subscriptionPrice to the price that you want for the subscription. Copy the price id. This will be your `VITE_STRIPE_PRODUCT_SUBSCRIPTION_KEY`



Sample .env

```
VITE_BACKEND=https://backend.YOURDOMAIN.com/api/v1
VITE_BACKEND_SOCKET_IO=https://backend.YOURDOMAIN.com
VITE_BACKEND_IMAGES_BASE_URL=https://backend.YOURDOMAIN.com
VITE_FRONTEND_DOMAIN=https://YOURDOMAIN.com
VITE_STRIPE_PRODUCT_SUBSCRIPTION_KEY=price_1P0sjYSCWiCS3Basdhuiwheuw
```

- Navigate to the **App Configs** section of your newly created app.

- Paste the contents of your `.env` file into the environmental variables section using **Bulk Edit**.
- ensure all environment variables are set correctly.

[HTTP Settings](#) [App Configs](#) [Deployment](#)

Environmental Variables ⓘ

Bulk Edit ☒

```
VITE_BACKEND=https://backend.minimalpos.com/api/v1
VITE_BACKEND_SOCKET_IO=https://backend.minimalpos.com
VITE_BACKEND_IMAGES_BASE_URL=https://backend.minimalpos.com
VITE_FRONTEND_DOMAIN=https://minimalpos.com
VITE_STRIPE_PRODUCT_SUBSCRIPTION_KEY="price_1POsjYSCWiCS3BoQN2lnBMTz"
```

7.3 Connect New Domain in HTTP Settings

- Navigate to the **HTTP Settings** section.
- Enter your domain (e.g., `yourdomain.com`) and click on **Connect New Domain**.
- Ensure to enable **HTTPS** for secure communication.
- Select the option for **Force Https & WebSocket Support** to allow real-time communication.

Your app is publicly available at:

Enable HTTPS

<https://frontend.apps.minimalpos.com>

Enable HTTPS

Remove

minimalpos.com

Redirect all domains to:

No redirects



www.the-best-app-in-the-world.com

Connect New Domain



Edit Default Nginx Configurations

Container HTTP Port

☒ Force HTTPS by redirecting all HTTP traffic to HTTPS

☒ Websocket Support

Edit HTTP Basic Auth

Current State: **inactive**

7.4 CapRover Deploy

Here are two convenient methods for deploying your application to CapRover: Drag & Drop, which allows for quick uploads, and Git, which enables seamless version control integration. Choose the method that best fits your workflow. However caprover supports more methods of deployments, you can explore more at [here](#).

7.4.1 Method 1: Drag & Drop

Package Your Application

- Open the terminal preferably Git Bash. Navigate to where your frontend folder is located.
- Create a tarball (.tar) containing the source code and necessary files for deployment (e.g., Dockerfile, app files, etc.) using the below command.

```
tar -cvf ./deploy.tar --exclude='.map' --exclude='node_modules' --exclude='.git' ./captain-definition ./
```

- A new file deploy.tar will be created in the same directory. This is the file to be uploaded for deployment.

Drag & Drop to deploy

- Go the deployments & Scroll to Tarball section.

🔗 Method 2: Tarball

You can simply create a tarball (`.tar`) of your project and upload it here via upload button.



Click or drag TAR file to this area to upload

Must contain `captain-definition` file.

Upload & Deploy

- Simply drag your `.tar` into the “Upload & Deploy” section of the app page.

🔗 Method 2: Tarball

You can simply create a tarball (`.tar`) of your project and upload it here via upload button.



Click or drag TAR file to this area to upload

Must contain `captain-definition` file.

 `deploy.tar`

Upload & Deploy

- CapRover will handle the deployment process based on your uploaded package.

7.4.2 Method 2: Using Git

- Open your terminal and navigate to the frontend folder where your Git repository is configured.
- Make sure to select the appropriate branch (e.g., `main`) that you want to push changes from.
- Run the following command to deploy your application:

```
caprover deploy
```

```
Deployed successfully frontend  
App is available at https://frontend.apps.minimalpos.com
```



Congratulations! Your Frontend application has been successfully deployed on CapRover and is ready to handle requests.
Enjoy building and scaling your application with ease!

Last updated on October 12, 2024