



Workshop

# App Deployment & Security with DigitalOcean



localhost



DigitalOcean



***Our Mission is to Empower Hackers.***

**65,000+**  
HACKERS

**12,000+**  
PROJECTS CREATED

**400+**  
CITIES

*We hope you learn something awesome today!*  
Find more resources: <http://mlh.io/>

# Table of Contents

-  **0.** Welcome to MLH Localhost
- 1.** What are you building?
- 2.** Deploying to DigitalOcean
- 3.** DevOps 101
- 4.** Review & Next Steps

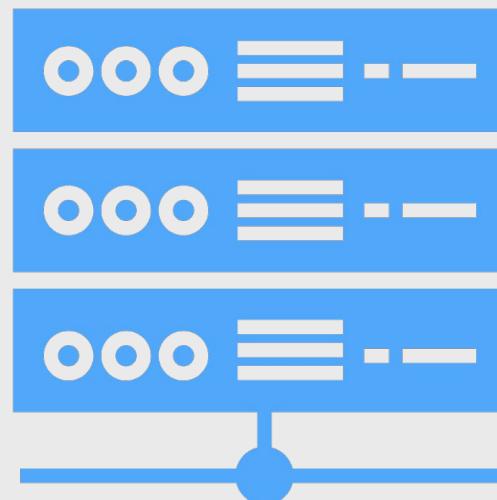
# What will you **learn today?**

- 1 How to deploy an application to a DigitalOcean Droplet
- 2 How to make your application more secure and available
- 3 What DevOps is, and why it's so important to software development

# What's the Cloud?

Every website or application you visit over the internet is stored somewhere: **a server**.

Your university will have a server room; businesses can create their own servers; you can even create local servers on your own devices with tools like **Express** and **Flask**.



# What's the Cloud?



Some companies offer servers for hosting or processing power.

Due to the increase in internet connection speeds, you can access these servers in milliseconds, anywhere in the world.

This is known as **Cloud Computing**.

# What's the Cloud?

DigitalOcean is a company that provides cloud infrastructure.

It allows us to create **Droplets** - virtual servers that you can use to host a web application.



In this workshop, we're going to create a web application, and deploy it on a DigitalOcean Droplet.

# Table of Contents

- 0. Welcome to MLH Localhost
-  1. What are you building?
- 2. Deploying to DigitalOcean
- 3. DevOps 101
- 4. Review & Next Steps

# Operation BeachClean!

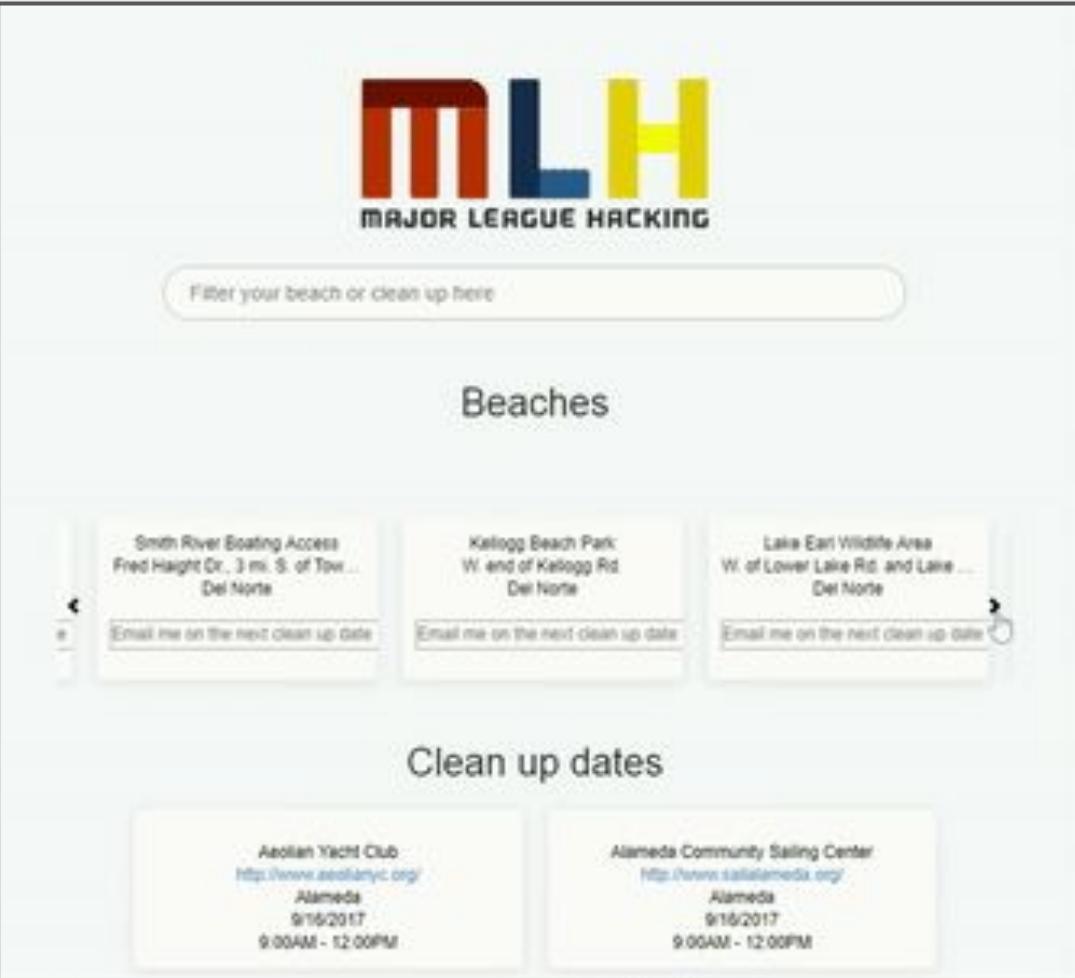
We're going to deploy an application to DigitalOcean. What does it do?

It's a web application that will allow us to:

- Find beach cleanup events
- Search by beach and location
- Allow users to subscribe to upcoming beach cleanups



# What are you going to deploy?



The screenshot shows a web application interface for Major League Hacking (MLH). At the top is the MLH logo and a search bar with the placeholder "Filter your beach or clean up here". Below the search bar is a section titled "Beaches" containing three cards:

- Smith River Boating Access  
Fred Haight Dr., 3 mi. S. of Town Del Norte  
[Email me on the next clean up date](#)
- Kellogg Beach Park  
W. end of Kellogg Rd  
Del Norte  
[Email me on the next clean up date](#)
- Lake Earl Wildlife Area  
W. of Lower Lake Rd. and Lake ...  
Del Norte  
[Email me on the next clean up date](#)

Below the beaches section is a section titled "Clean up dates" containing two cards:

- Aeolian Yacht Club  
<http://www.aeolianyc.org/>  
Alameda  
9/16/2017  
9:00AM - 12:00PM
- Alameda Community Sailing Center  
<http://www.sailalameda.org/>  
Alameda  
9/16/2017  
9:00AM - 12:00PM

**Find yourself a partner, and ask each other  
these questions:**

**If you could build a website, what would  
you create?**

**Let's talk about DevOps.**

# What is DevOps?

When you think about making software, applications, or websites, sometimes you only think about the code itself - a software engineer writing lots of code to build something.

But there's **so much more** to developing!

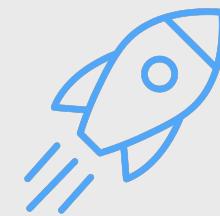
That's where Development and Operations (DevOps) comes in.



# What is DevOps?

What kind of things is a DevOps engineer responsible for?

**Deployment:** Launching code into the real world! This could mean hosting a website on a server, or creating installation files for desktop software.



**Maintenance:** Code needs to be updated, and fixed when it breaks - a DevOps engineer safeguards these processes.

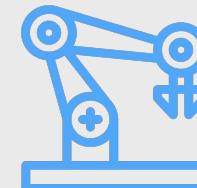
# What is DevOps?

**Monitoring:** A DevOps engineer keeps track of the system health - are the servers running smoothly?



**Security:** Are the devices protected from internal and external attack?

**Automation:** For all of these jobs - can you do them automatically to save human time?

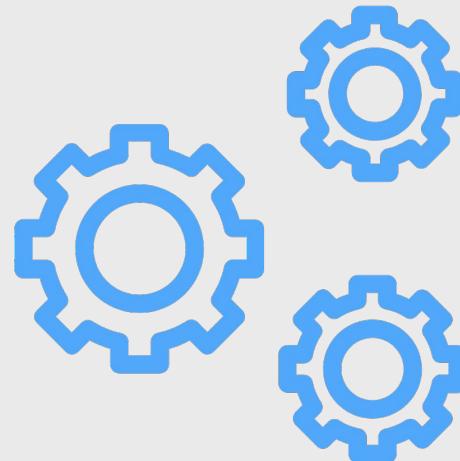


# DevOps and DigitalOcean

We're going to deploy our application to DigitalOcean and learn some vital DevOps skills along the way.

By the end of the workshop, you'll know how to:

- Deploy code to a DigitalOcean Droplet.
- Keep your Droplet secure.
- Monitor the health of your Droplets.



**Let's get started!**

# Table of Contents

- 0. Welcome to MLH Localhost
- 1. What are you building?
- 2. Deploying to DigitalOcean
- 3. DevOps 101
- 4. Review & Next Steps

# Join the DigitalOcean team

Head to your email inbox and you'll have an invitation to join a DigitalOcean team for this workshop.

You have been invited to join mlh-localhost on DigitalOcean  Inbox 

# Join your Workshop's Team

Click the **Team up** button to begin!



**DigitalOcean**

**Join the ranks of mlh-localhost's team!**

Hi there!

You've been invited to join mlh-localhost on DigitalOcean—now you can share resources and work collaboratively with your teammates. Just click the button below to join mlh-localhost.

(And don't worry; you can switch between your personal account and the team at any time.)

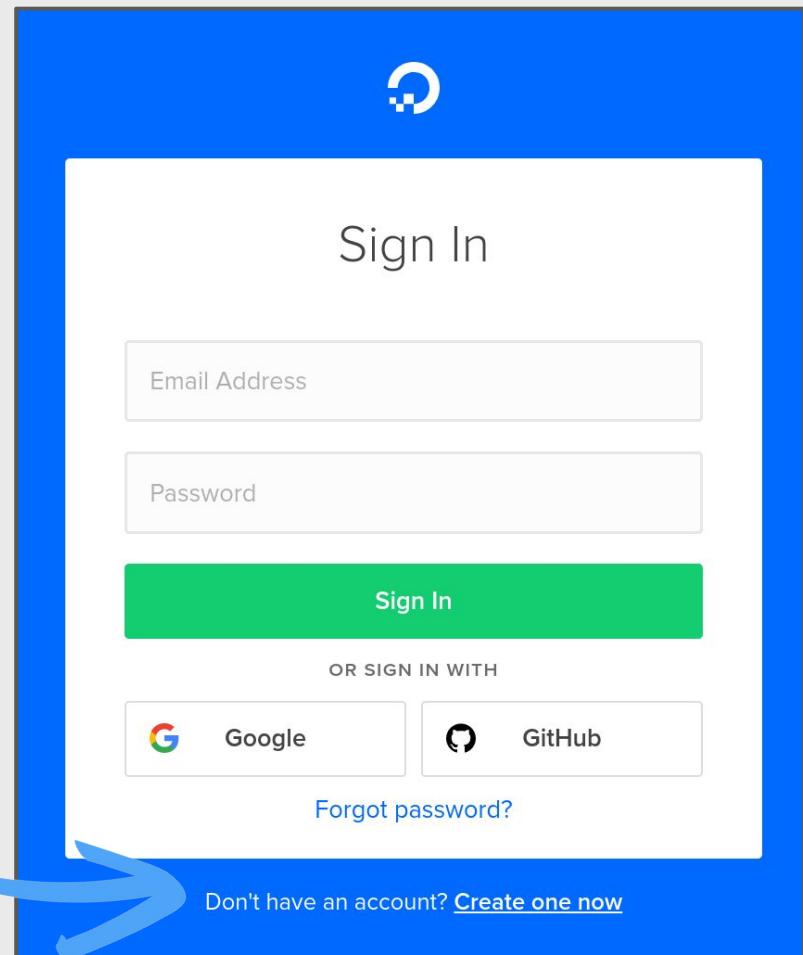
**Team up with mlh-localhost**

© DigitalOcean

# Create a DigitalOcean account

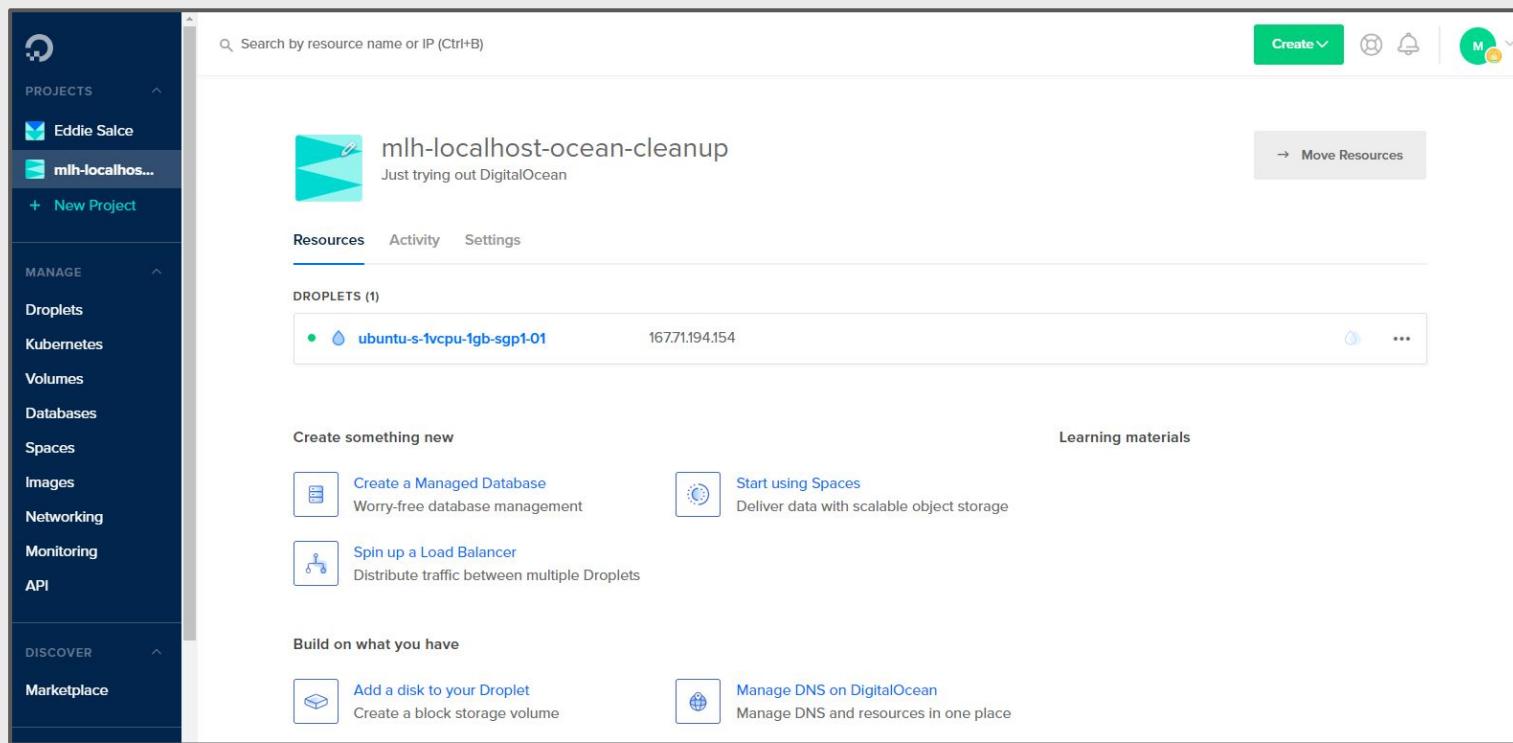
Click on **Create** to make a DigitalOcean account.

You can use your Google account or GitHub, or create one with your email address.

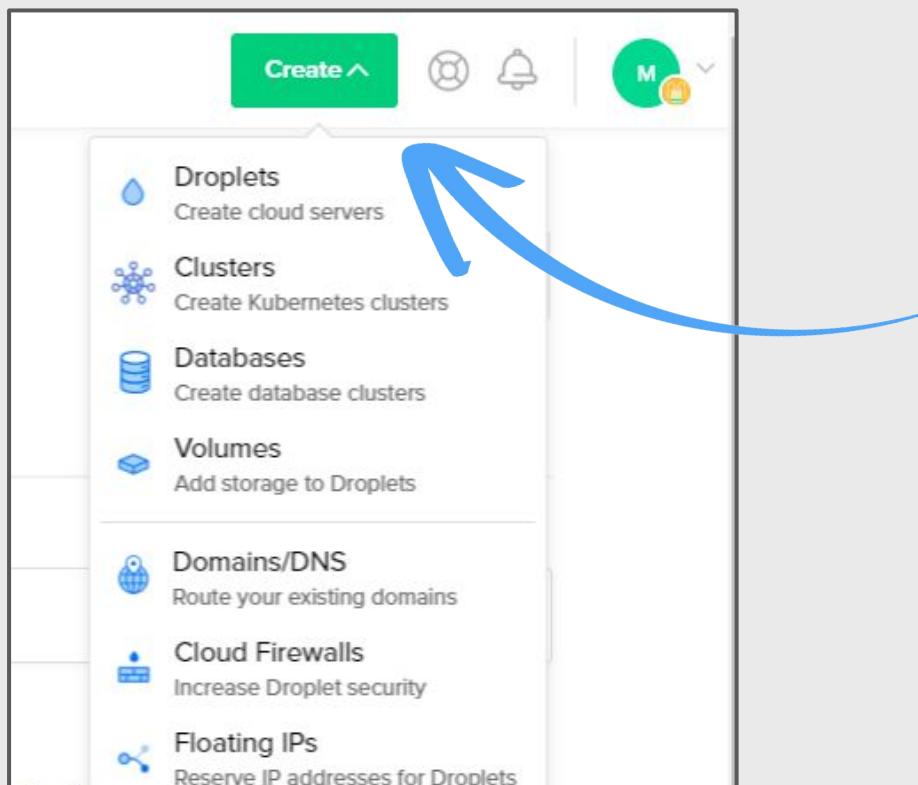


# The DigitalOcean Dashboard

The link will open the DigitalOcean Dashboard. Here's where you can keep track of our projects and Droplets.



# Create a new Droplet



Click the green **Create** button and then select **Droplets**.

# Choose an Operating System

We can select which operating system you want our Droplet to have. Select **Ubuntu 18.04** if it's not already highlighted.

## Create Droplets

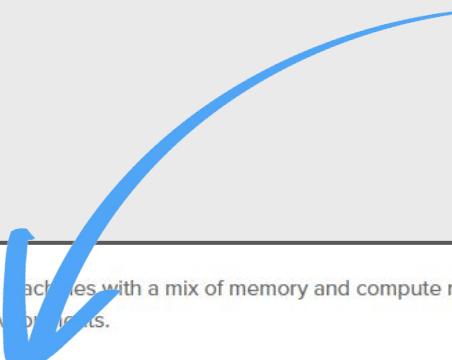
Choose an image [?](#)

[Distributions](#) [Container distributions](#) [Marketplace](#) [Custom images](#)

 Ubuntu <b>18.04 (LTS) x64</b>	 FreeBSD Select version	 Fedora Select version	 Debian Select version	 CentOS Select version
---	--	--	---	---

# Select a Plan

Scroll down and select the smallest plan - **1GB / 1 CPU**. That will be enough for our needs!



Standard virtual machines with a mix of memory and compute resources. Best for small projects that can handle variable levels of CPU performance, like blogs, web apps and dev/test environments.

<b>\$5/mo</b> \$0.007/hour
1 GB / 1 CPU 25 GB SSD disk 1000 GB transfer

<b>\$10/mo</b> \$0.015/hour
2 GB / 1 CPU 50 GB SSD disk 2 TB transfer

<b>\$15/mo</b> \$0.022/hour
3 GB / 1 CPU 60 GB SSD disk 3 TB transfer

<b>\$15/mo</b> \$0.022/hour
2 GB / 2 CPUs 60 GB SSD disk 3 TB transfer

<b>\$15/mo</b> \$0.022/hour
1 GB / 3 CPUs 60 GB SSD disk 3 TB transfer

<b>\$20/mo</b> \$0.030/hour
4 GB / 2 CPUs 80 GB SSD disk 4 TB transfer

Currently selected: 8 GB / 4 CPUs

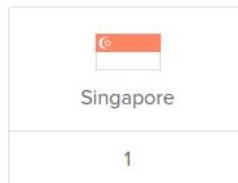


[Show all plans](#)

# Datacenter Region

Select the Datacenter region which is closest to you.

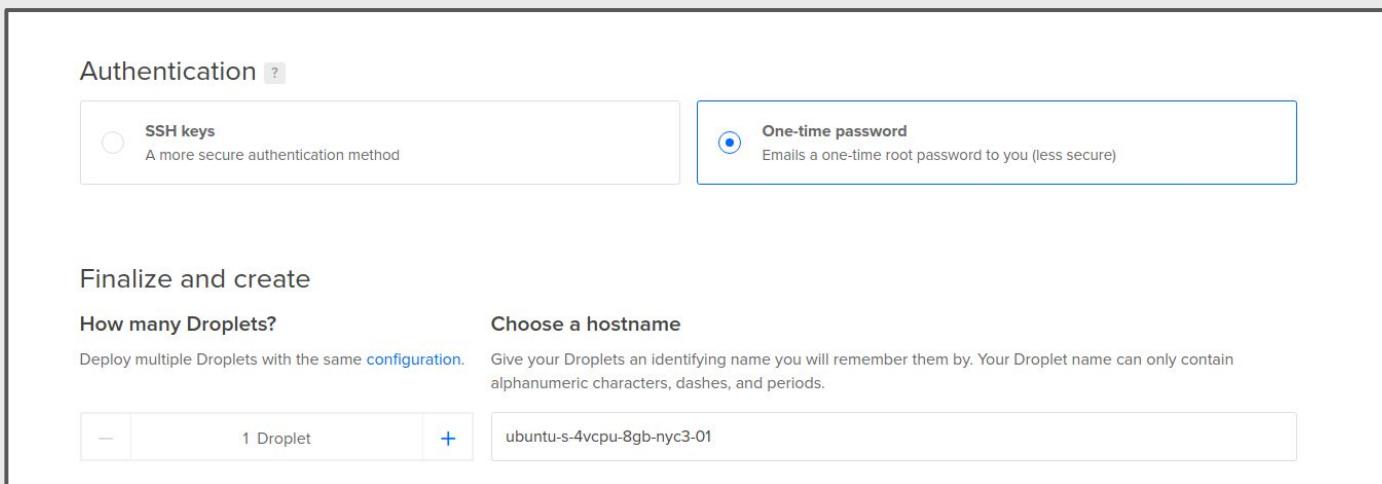
Choose a datacenter region



# Creating a Droplet

Under authentication, select **One-time password**.

Lastly, give your Droplet a name to identify it as yours, such as your first name initial and last name.



The screenshot shows the 'Authentication' section of the DigitalOcean Droplet creation interface. It offers two options: 'SSH keys' (radio button unselected) and 'One-time password' (radio button selected). Below this, there's a 'Finalize and create' section. Under 'How many Droplets?', it says '1 Droplet'. Under 'Choose a hostname', it shows 'ubuntu-s-4vcpu-8gb-nyc3-01'. A note states: 'Deploy multiple Droplets with the same configuration.' and 'Give your Droplets an identifying name you will remember them by. Your Droplet name can only contain alphanumeric characters, dashes, and periods.'

# Assign to a Project

We can leave the rest of the settings on their default values.

Scroll to the bottom, and click **Create Droplet!**

Select Project

Assign Droplets to a project

 mlh-localhost-ocean-cleanup

**Create Droplet**

Back on the Project Dashboard, you can see that your Droplet has been created.

Now you can start using it!

 mlh-localhost-ocean-cleanup  
Just trying out DigitalOcean

[Resources](#)   [Activity](#)   [Settings](#)

---

DROPLETS (2)

•  <a href="#">eddie-droplet-mlh-workshop-ubuntu</a>	165.22.214.27
•  <a href="#">sam-droplet-mlh-workshop-ubuntu</a>	165.22.209.4

# Get your Droplet Details

Check your email and you should receive an email containing the details of your newly created Droplet.

Your New Droplet: [ubuntu-s-4vcpu-8gb-blr1-01](#)

 [Inbox](#) 

 **DigitalOcean** <[support@support.digitalocean.com](mailto:support@support.digitalocean.com)>  
to me ▾

Your new Droplet is all set to go! You can access it using the following credentials:

Droplet Name: [ubuntu-s-4vcpu-8gb-blr1-01](#)  
IP Address: 165.22.209.4  
Username: root  
Password: [d4afd49296bfdb10aaa1bcd44](#)

# Get your Droplet Details

**Droplet Name:** How you identify your Droplet on the DigitalOcean dashboard.

**IP Address:** Your Droplet's address on the internet.

**Username/Password:** How you can log in to your Droplet with an administrator (root) account.

Droplet Name: `ubuntu-s-4vcpu-8gb-blr1-01`

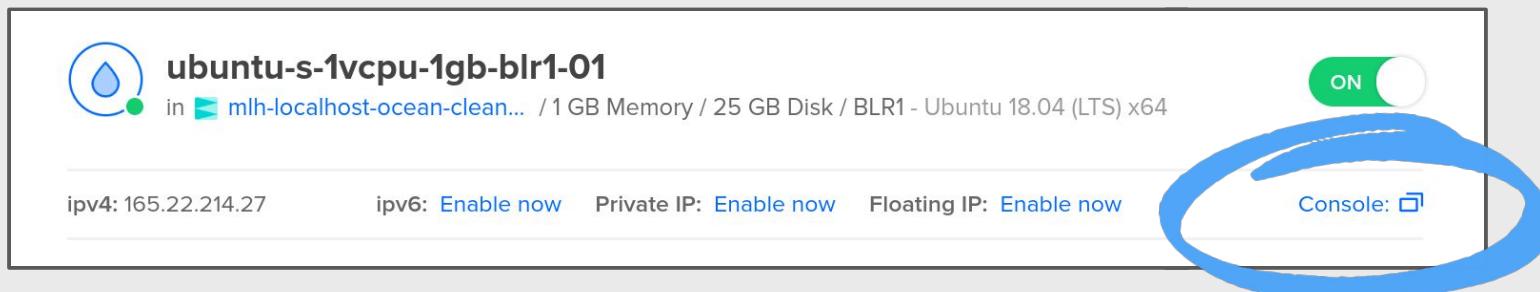
IP Address: `165.22.209.4`

Username: `root`

Password: `d4afd49296bfdb10aaa1bcd44`

# Connecting to your Droplet

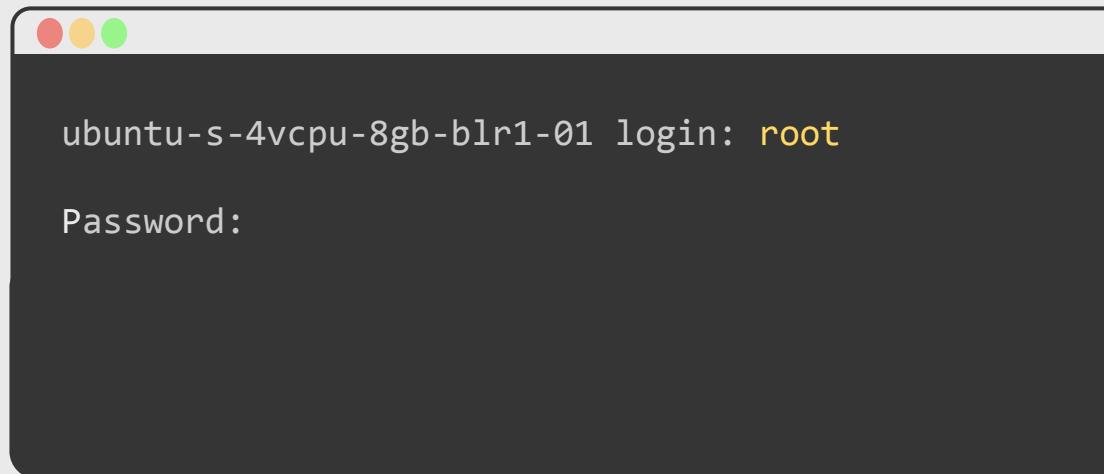
On the DigitalOcean dashboard, select your new Droplet, and click **Console**.



# Connecting to your Droplet

Enter the username **Root**. Enter the password you received in the email.

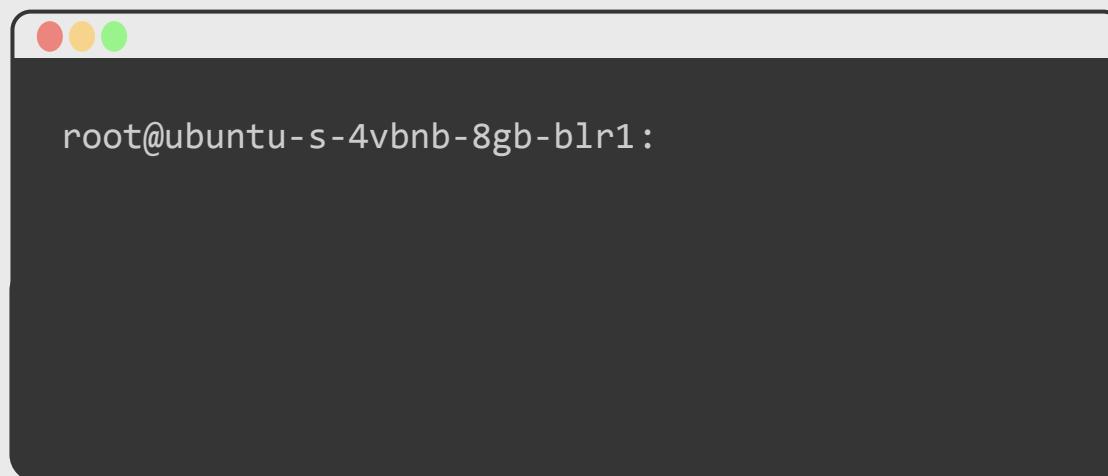
For your first time, you'll be asked to change your password. Make it something unique to you!



# Connecting to your Droplet

Check out what happened to your Console. You're now inside the **Root** (administrator) account of your Droplet!

This means that you are completely controlling the Droplet from your own device. Awesome!



# Adding new Users

An important DevOps responsibility is Security.

One way you can implement this is **Access Control**: only giving users the privileges and power that they need, and no more.



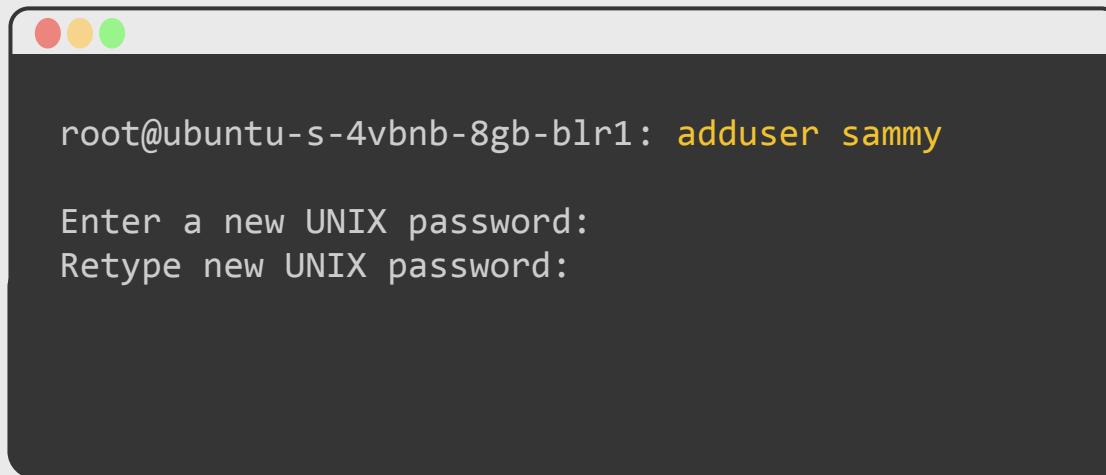
The Root account can do everything to our Droplet - you don't want this power falling into the wrong hands!

Let's make a user account that doesn't have quite as much power as Root.

# Creating a new Account

Let's create a new user account. Type in the **adduser** command with a name and press Enter.

Give your account a password (different to the Root password you just created!) and verify it.

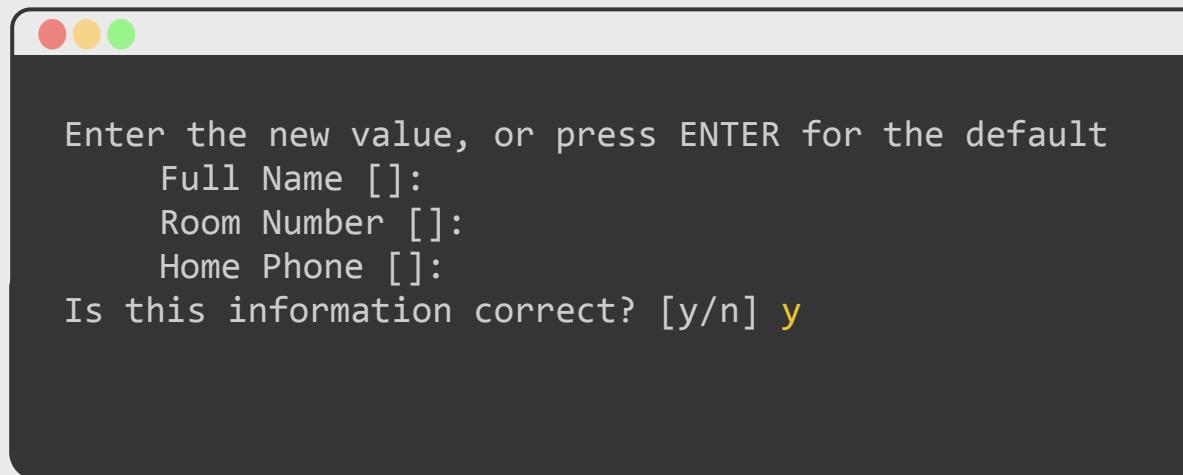


```
root@ubuntu-s-4vbnb-8gb-blr1: adduser sammy
Enter a new UNIX password:
Retype new UNIX password:
```

# Creating a new Account

The Console will ask for some personal details - you can leave these blank and press Enter to continue.

At the end, type **y** to confirm.



# Creating a new Account

Next, you'll give your new account some minimal administrative privileges so that you can install new packages and updates.

Type in the following command, replacing **sammy** with your account name.

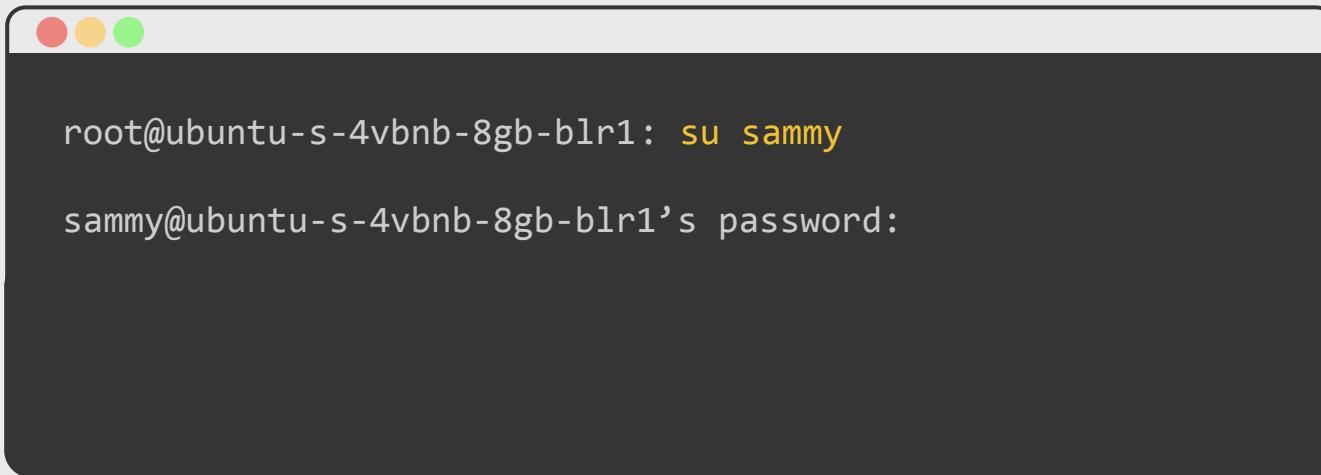


```
root@ubuntu-s-4vbnb-8gb-blr1: usermod -aG sudo sammy
```

# Creating a new Account

Now, log in to your newly created account. Use the **su** command (Switch User) to log in to your new account.

Replace **sammy** with the name you gave your account.



A screenshot of a terminal window with a dark background and light-colored text. The window has three colored window control buttons (red, yellow, green) at the top left. The terminal output shows:

```
root@ubuntu-s-4vbnb-8gb-blr1: su sammy
sammy@ubuntu-s-4vbnb-8gb-blr1's password:
```

# The Advanced Package Tool

Next, we're going to install some software on our Droplet using the Advanced Package Tool.

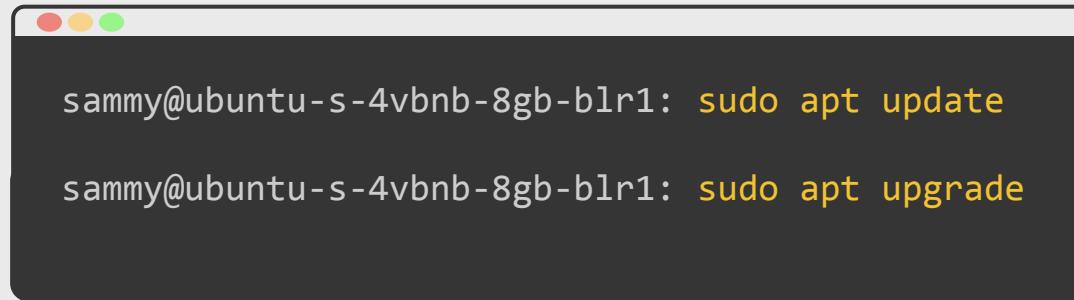
We use the `apt` command to install or update a package (piece of software).

By using APT, you ensure that we're installing the package from a trusted repository, and that any updates you install will be secure.

# Installing Updates

Let's make sure our Droplet is up to date and has all of the packages it needs. Use the **apt** command, and enter your account password.

This might take a minute or two!



```
sammy@ubuntu-s-4vbnb-8gb-blr1: sudo apt update  
sammy@ubuntu-s-4vbnb-8gb-blr1: sudo apt upgrade
```

A screenshot of a macOS-style terminal window. The window title bar shows three colored dots (red, yellow, green). The main area of the terminal contains two lines of text in white font on a black background. The first line is "sammy@ubuntu-s-4vbnb-8gb-blr1: sudo apt update" and the second line is "sammy@ubuntu-s-4vbnb-8gb-blr1: sudo apt upgrade".

Why do we update *and* upgrade?

**Update** returns a list of available packages and new versions, then **upgrade** actually installs them.

# Installing Node.js

We need to install a couple of packages that our application requires to be able to run.

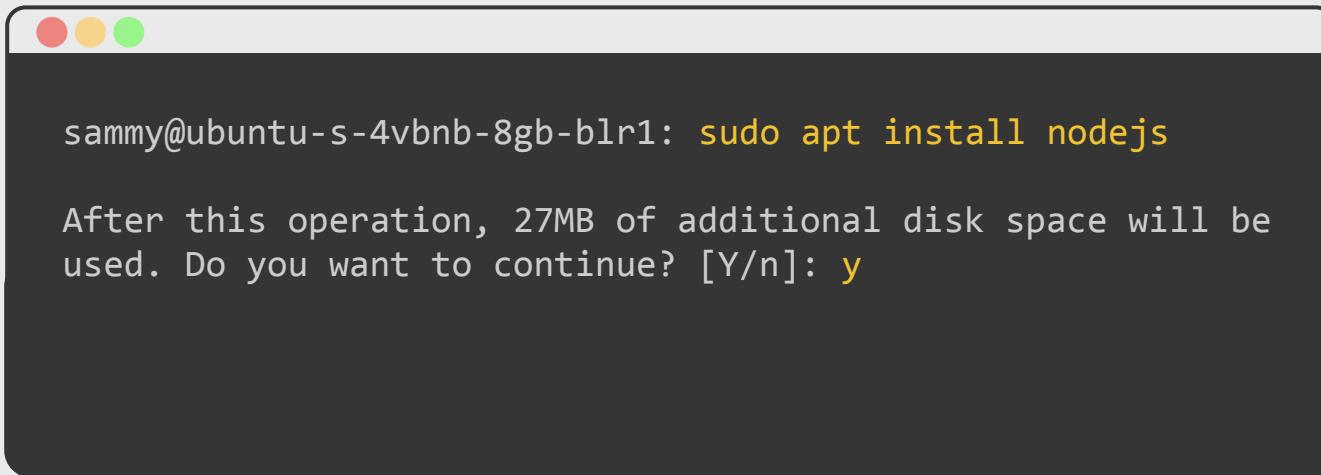
The first is Node.js, a server-side JavaScript environment.



# Installing Node.js

To install Node.js, use the **apt install** command.

If it asks for permission, type **y** for yes.

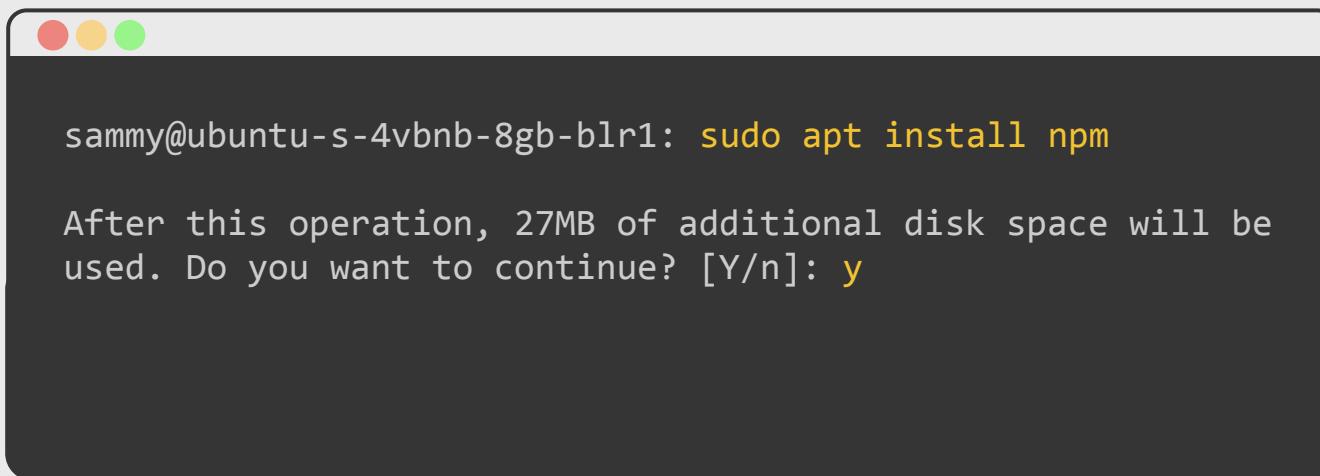


A screenshot of a terminal window on a Linux system. The window has a dark background and light-colored text. At the top, there are three small colored circles (red, yellow, green) which are standard window control buttons. The text in the terminal is as follows:

```
sammy@ubuntu-s-4vbnb-8gb-blr1: sudo apt install nodejs
After this operation, 27MB of additional disk space will be
used. Do you want to continue? [Y/n]: y
```

# Install Node Package Manager

Let's do the same thing again to install Node Package Manager.



```
sammy@ubuntu-s-4vbnb-8gb-blr1: sudo apt install npm

After this operation, 27MB of additional disk space will be
used. Do you want to continue? [Y/n]: y
```

# Get the Application Code

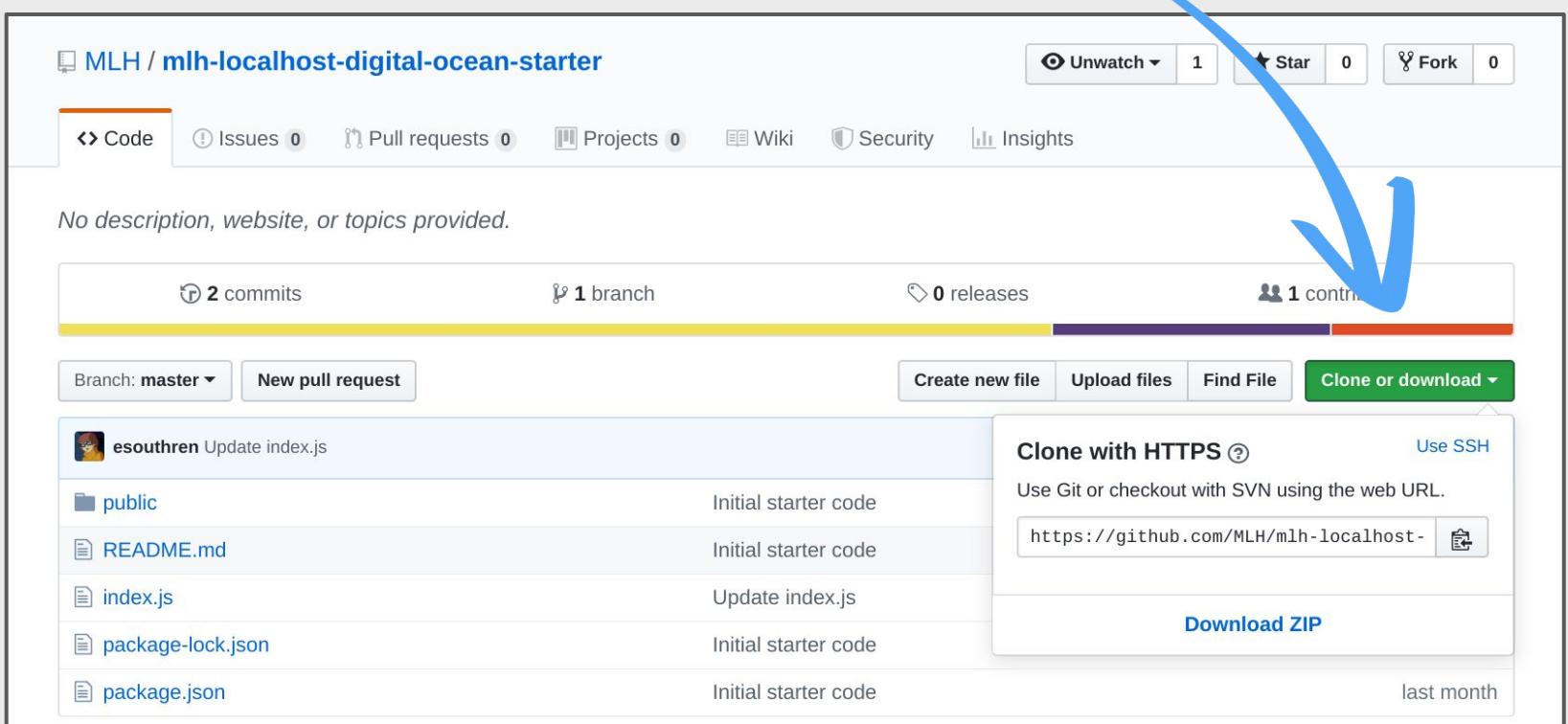
Next you need to add the application code to our Droplet.

Head to this GitHub repository, where the code is stored:

**<http://mlhlocal.host/digitalocean-project>**

# Get the Application Code

Click the green **Clone or Download** button and copy the URL link.



The screenshot shows a GitHub repository page for "MLH / mlh-localhost-digital-ocean-starter". The page includes a navigation bar with "Unwatch", "Star", and "Fork" buttons. Below the navigation bar, there are links for "Issues 0", "Pull requests 0", "Projects 0", "Wiki", "Security", and "Insights". A message states "No description, website, or topics provided." Below this, there are statistics: "2 commits", "1 branch", "0 releases", and "1 contributor". The "Branch: master" dropdown and "New pull request" button are visible. At the bottom, there are buttons for "Create new file", "Upload files", "Find File", and the prominent green "Clone or download" button. To the right of the "Clone or download" button is a "Clone with HTTPS" field containing the URL "https://github.com/MLH/mlh-localhost-". A blue curved arrow points from the text above to the "Clone or download" button. The repository's history shows commits by "esouthern" and files like "index.js", "README.md", "package-lock.json", and "package.json".

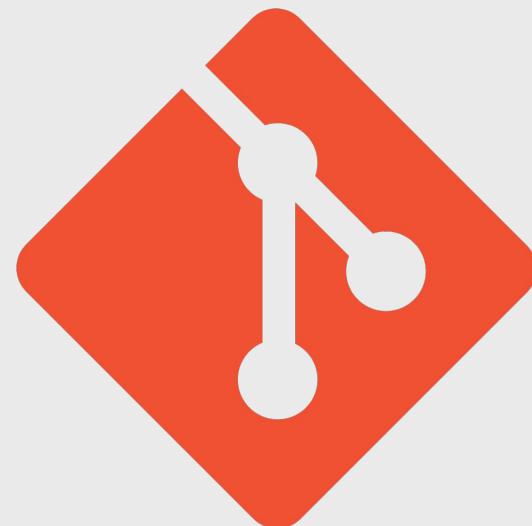
# Using Git for Version Control

GitHub is a tool for storing and sharing code online.

It uses Git as a form of **version control**. That means, keeping track of changes to the code over time.

Why is this important to DevOps?

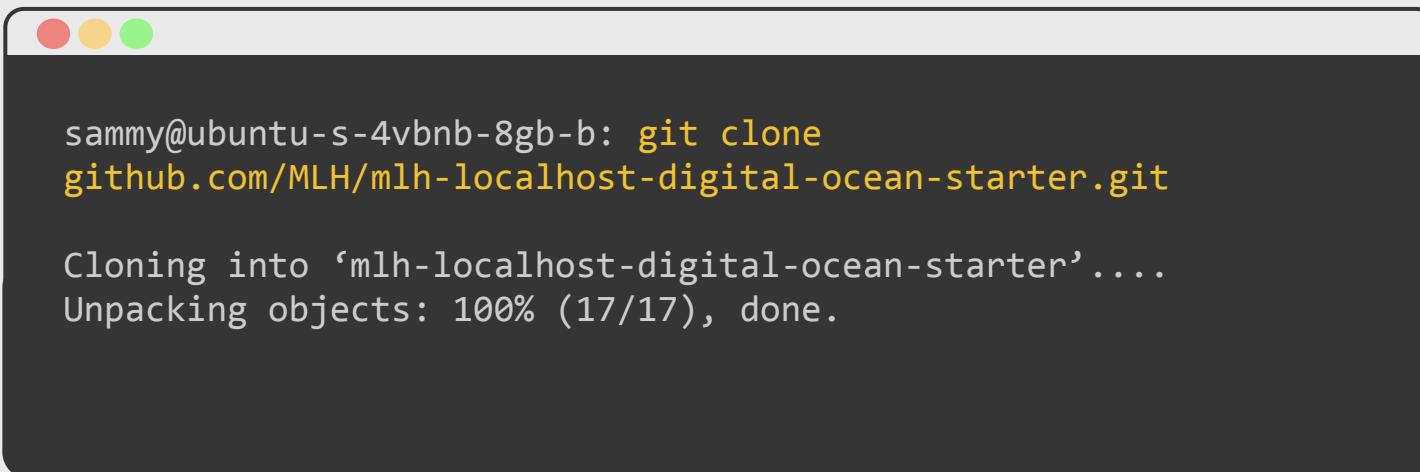
It allows us to have one central source for our code. We can then do things like automatically **deploy** this code when we want to update our application.



# Copy the Project Files

Back in your terminal window, you'll use the **git clone** command to copy this repository into our droplet.

Type **git clone** then paste (Ctrl + V) the GitHub link you copied.



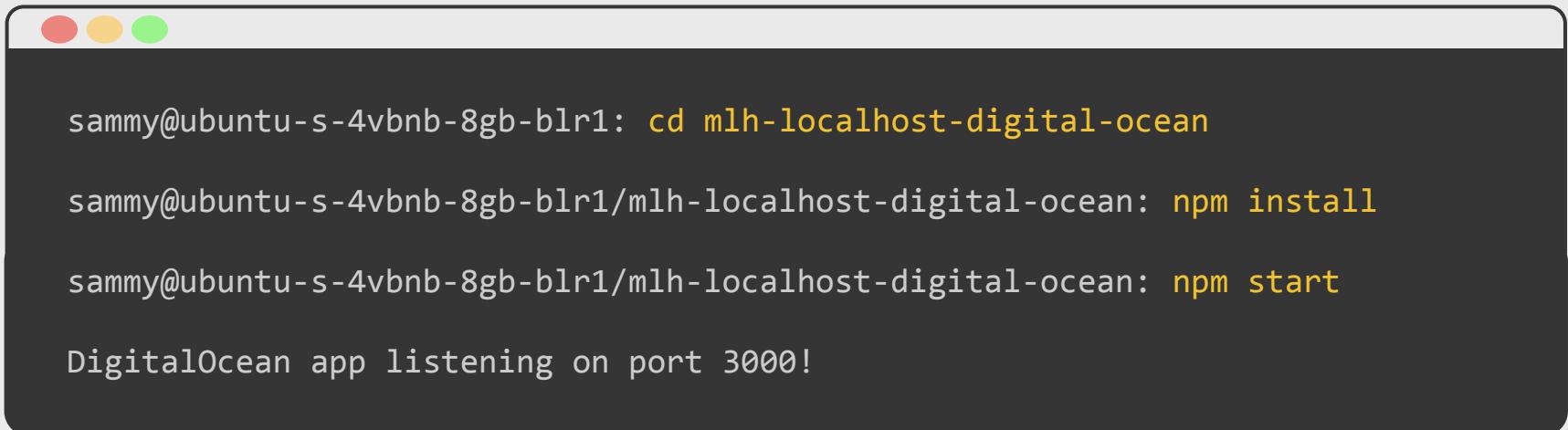
```
sammy@ubuntu-s-4vbnb-8gb-b: git clone
github.com/MLH/mlh-localhost-digital-ocean-starter.git

Cloning into 'mlh-localhost-digital-ocean-starter'....
Unpacking objects: 100% (17/17), done.
```

# Install and Launch the App

You have everything you need!

Let's install the application and launch it.



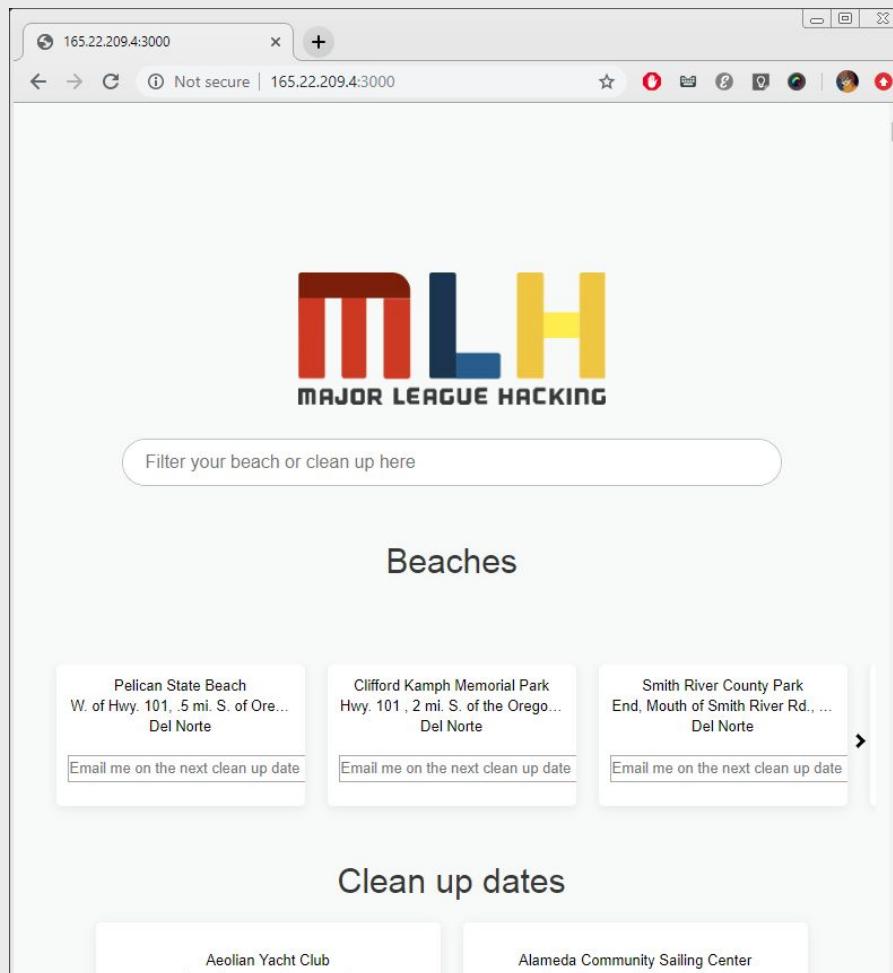
```
sammy@ubuntu-s-4vbnb-8gb-blr1: cd mlh-localhost-digital-ocean
sammy@ubuntu-s-4vbnb-8gb-blr1/mlh-localhost-digital-ocean: npm install
sammy@ubuntu-s-4vbnb-8gb-blr1/mlh-localhost-digital-ocean: npm start
DigitalOcean app listening on port 3000!
```

# Accessing the App

On your device, open a new browser window and head to your Droplet's URL, on port 3000 - substitute your Droplet's IP address.

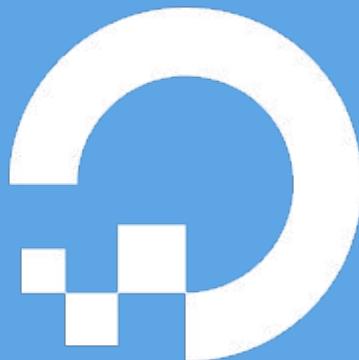
**198.51.100.0:3000**

# Accessing the App



# Success!

You are accessing the application that's  
being hosted in your DigitalOcean Droplet!



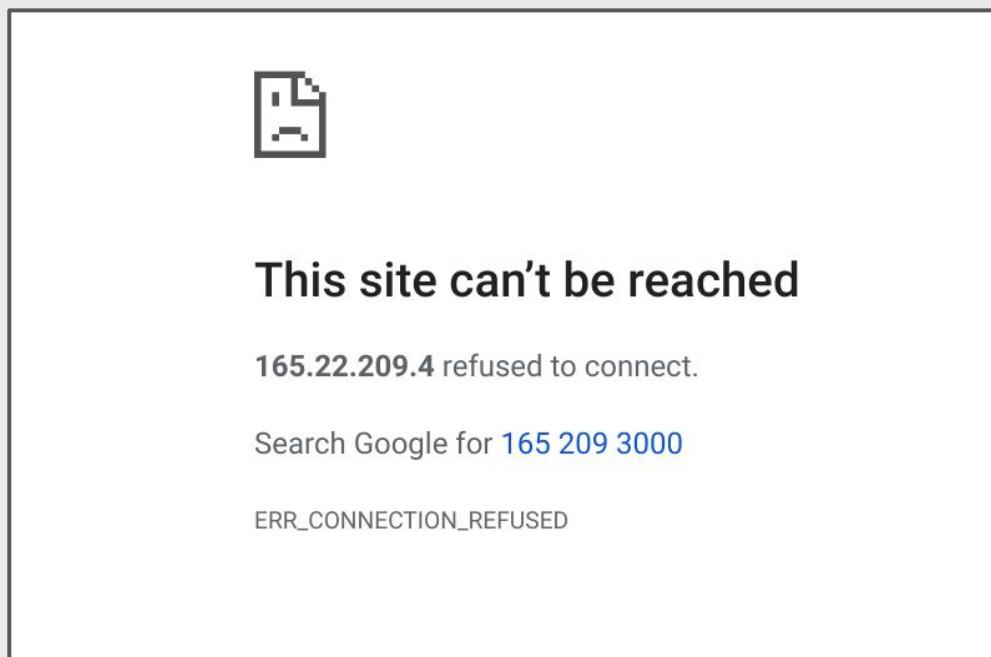
# Table of Contents

0. Welcome to MLH Localhost
1. What are you building?
2. Deploying to DigitalOcean
-  3. DevOps 101
4. Review & Next Steps

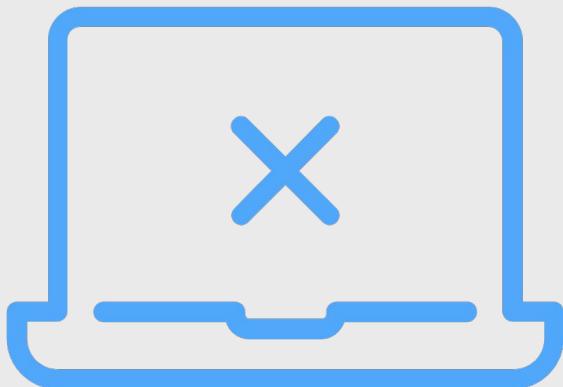
**Let's talk about some common DevOps  
features.**

# Server Availability

Have you ever gone to a website and seen this error message?



# Server Availability



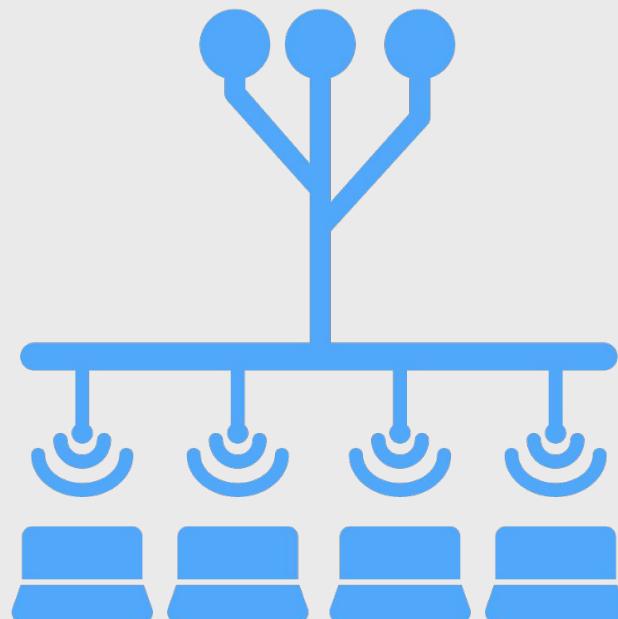
The user can't get to a website: a DevOps nightmare!

What can cause this?

- Servers going down or failing.
- An overload of server requests: too many people trying to connect at once.

# The Solution

Thankfully, DigitalOcean can offer a solution to this: **Load Balancing**.



# Load Balancing

Load balancing distributes network traffic across multiple Droplets. If one server fails, or is overloaded, the user can still make a successful connection.

Load balancers give us three things:

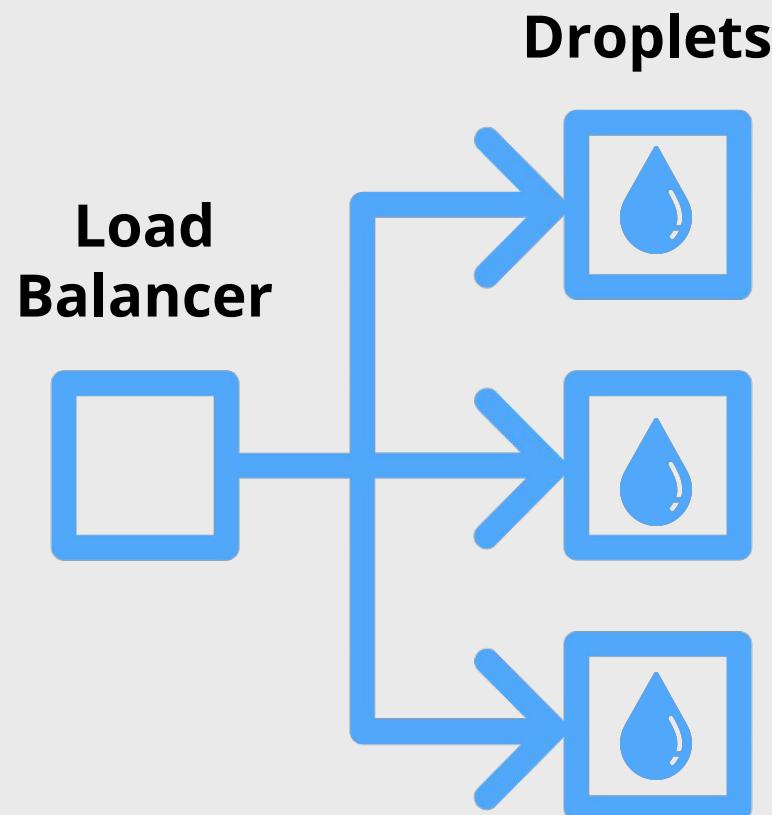
- **Availability:** Load balancers check to see which Droplets are healthy before sending requests to them.
- **Flexibility:** You can make changes to applications running on our servers without worrying that a user won't be able to access the server while you do it.
- **Performance:** You can share the traffic workload across multiple Droplets - so one Droplet doesn't get overwhelmed.

# Load Balancing

How does it work?

Instead of directly accessing the Droplet, you go to the Load Balancer's network address.

The Load Balancer then selects the best Droplet to fulfill the request.



# Load Balancer Health Checks

The Load Balancer will select a **healthy** droplet.

To measure healthiness, the Load Balancer will **ping** our Droplet periodically to check for a successful response.

If it gets a successful response, it will record that the Droplet is currently in a healthy status.

## Key Term

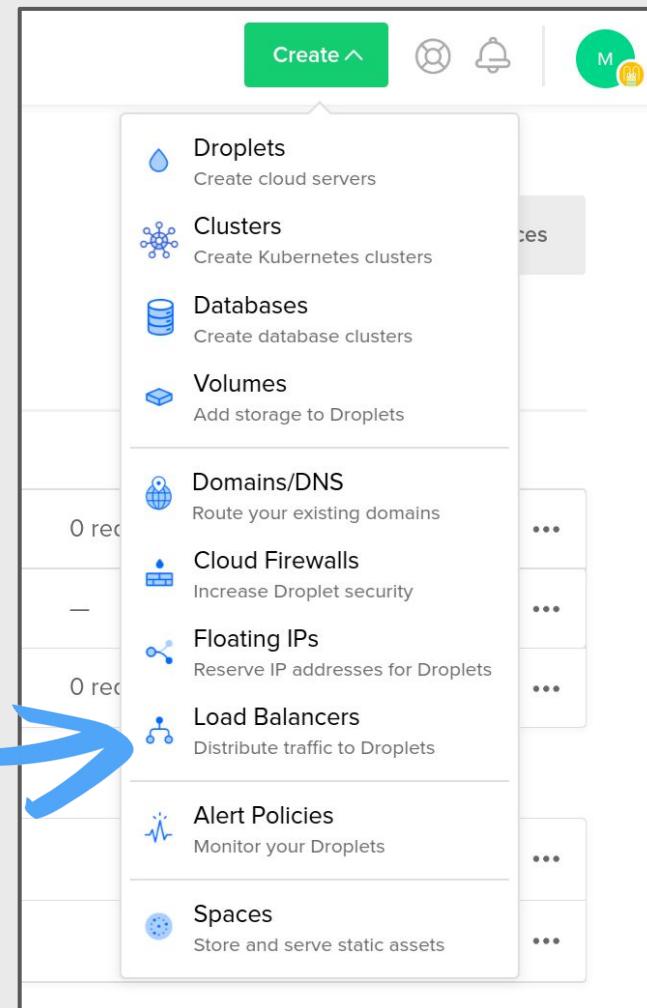
**Ping:** A utility that sends a request to a network device and waits for a response. It measures the time taken and if any data was lost.

# Creating a Load Balancer

Let's make one!

Head back to the DigitalOcean dashboard.

Click **Create**, then **Load Balancers**.



# Creating a Load Balancer

Select the Datacenter region that contains your droplets.

## Choose a datacenter region

Choose the same datacenter as the Droplets you plan to load balance.

		
New York		
1	2	3

	
San Francisco	
1	2

	
Amsterdam	
2	3

 1 Droplet in this region

# Creating a Load Balancer

We want the Load Balancer to redirect to Port 3000, which our application is running on.

Change the Droplet port to **3000**.

### Forwarding rules

Set how traffic will be routed from the Load Balancer to your Droplets. At least one rule is required.

Load Balancer	Droplet
Protocol HTTP	Protocol HTTP
Port 80	Port 3000
→	
New rule	Delete

A blue oval highlights the 'Port 3000' field in the Droplet section.

# Creating a Load Balancer

Scroll down to Advanced Settings and click [Edit](#).

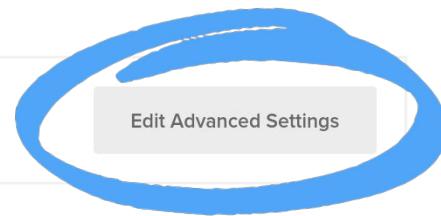
Under Health Checks, change the Port to [3000](#).

### Advanced settings

The most commonly used settings are selected by default. You can change them at any time by clicking “Edit Advanced Settings”.

Algorithm Round Robin	Sticky sessions Off	Health checks <a href="http://0.0.0.0:80/">http://0.0.0.0:80/</a>	SSL No redirect	Proxy Protocol Disabled
--------------------------	------------------------	--	--------------------	----------------------------

[Edit Advanced Settings](#)



### Health checks

Set how often the Load Balancer checks if Droplets are responding. It will automatically stop sending traffic to unresponsive Droplets.

Protocol HTTP	✓ ✓
Port 3000	✓
Path /	✓

# Creating a Load Balancer

Give your Load Balancer a name, and make sure your project matches the workshop you are taking part in.

Lastly, click **Create Load Balancer!**

## Finalize and create

### Choose a name

Load Balancer names must be unique and contain alphanumeric characters, dashes, and periods only.

### Select project

Select an existing project for this Load Balancer to belong to.

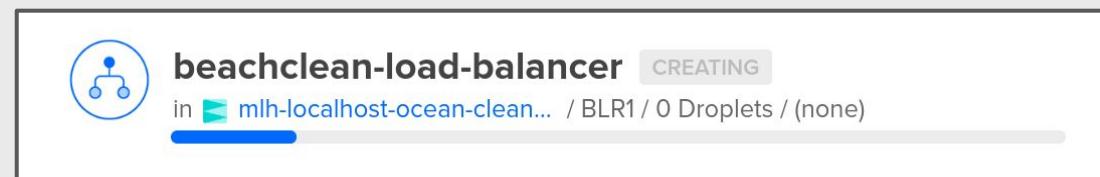
 mlh-localhost-ocean-cleanup 

**Create Load Balancer**

You can add Droplets once the Load Balancer is created.

# Creating a Load Balancer

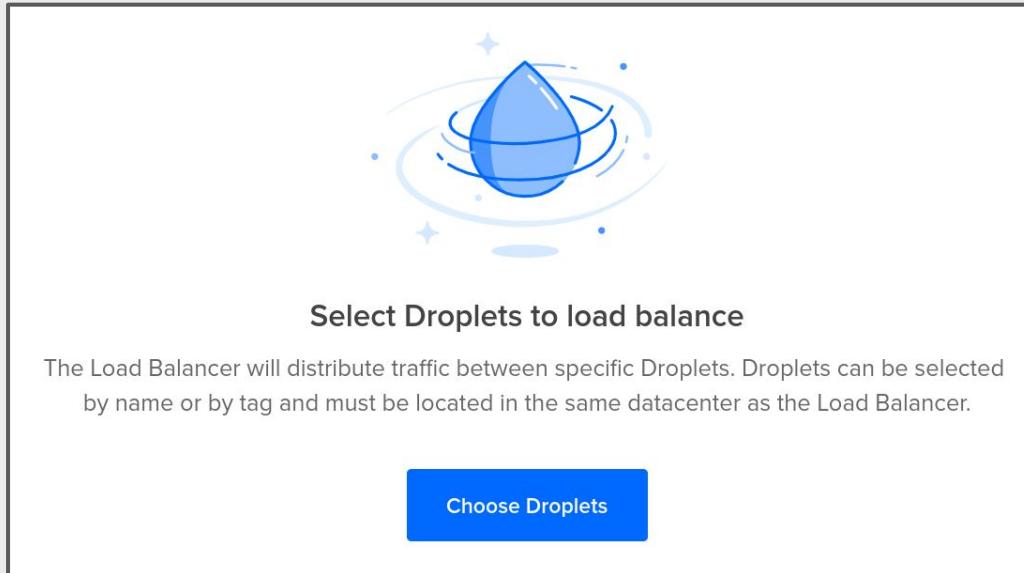
Your load balancer will take a few minutes to be created.



# Adding Droplets

Now that you have a Load Balancer, you need to add some Droplets to it.

Click [Choose Droplets](#).



The interface shows a large blue water droplet icon with a ring of smaller dots around it, symbolizing a load balancer. Below the icon is the text "Select Droplets to load balance". A descriptive paragraph explains that the Load Balancer will distribute traffic between specific Droplets, which can be selected by name or tag and must be located in the same datacenter as the Load Balancer. At the bottom is a prominent blue button labeled "Choose Droplets".

Select Droplets to load balance

The Load Balancer will distribute traffic between specific Droplets. Droplets can be selected by name or by tag and must be located in the same datacenter as the Load Balancer.

Choose Droplets

# Adding Droplets

Add your Droplet to the Load Balancer by searching for its name.

Add two or three more from other workshop participants!

Add Droplets to beachclean-load-balancer

Choose Droplets by adding their name or a tag below. Droplets must be located within BLR1.

Search for a Droplet or a tag

Add Droplets

# Adding Droplets

Wait a few minutes for your Droplets to be configured.

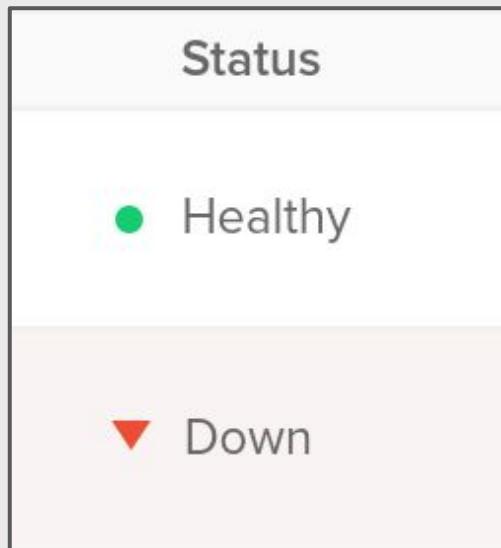
The status will change from **Collecting** to **Healthy** or **Down**.

Name	Status
 <a href="#">ubuntu-s-4vcpu-8gb-blr1-01</a> 8 GB / 160 GB / BLR1	● Collecting...
 <a href="#">ubuntu-s-1vcpu-1gb-blr1-01</a> 1 GB / 25 GB / BLR1	● Collecting...

Name	Status
 <a href="#">ubuntu-s-4vcpu-8gb-blr1-01</a> 8 GB / 160 GB / BLR1	● Healthy
 <a href="#">ubuntu-s-1vcpu-1gb-blr1-01</a> 1 GB / 25 GB / BLR1	▼ Down

# Accessing the Load Balancer



What would cause a Droplet to be **Down**?

- If the Droplet is switched off
- If our application isn't running on Port 3000

We need at least one **Healthy** Droplet to be able to access our Load Balancer.

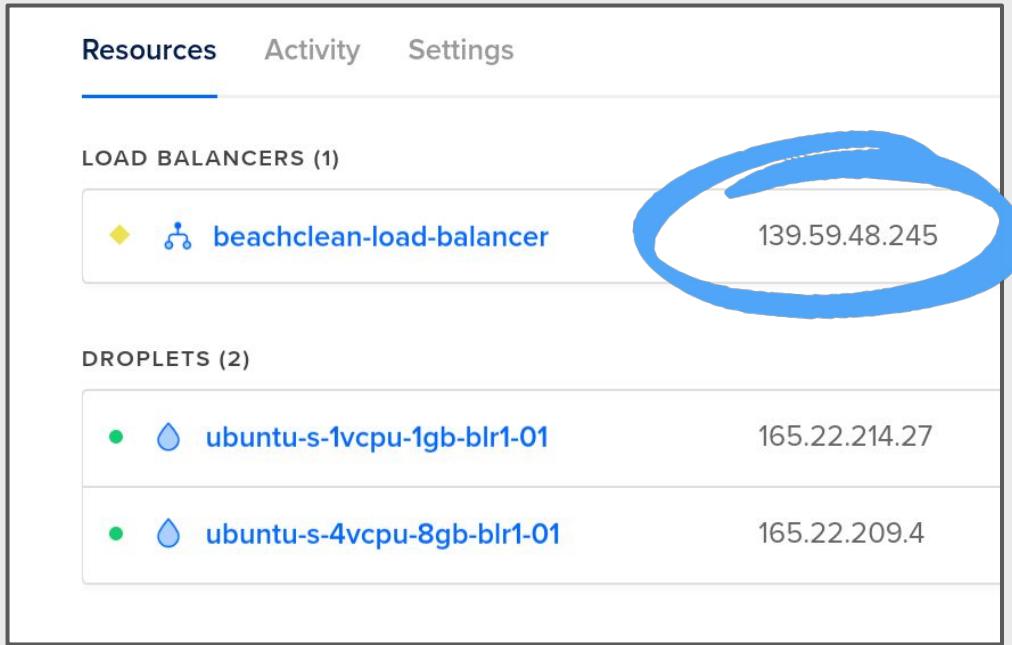
If yours is down, try logging into it and making sure the application is running.

# Accessing the Load Balancer

Let's access our Load Balancer!

Back on the Dashboard, you can find the IP Address of our Load Balancer.

Type it into your address bar and check out the Application!



The screenshot shows the DigitalOcean dashboard under the 'Resources' tab. In the 'LOAD BALANCERS (1)' section, there is one entry: 'beachclean-load-balancer' with the IP address '139.59.48.245'. This IP address is circled in blue. In the 'DROPLETS (2)' section, there are two entries: 'ubuntu-s-1vcpu-1gb-blr1-01' with the IP '165.22.214.27' and 'ubuntu-s-4vcpu-8gb-blr1-01' with the IP '165.22.209.4'.

Resource Type	Name	IP Address
LOAD BALANCERS	beachclean-load-balancer	139.59.48.245
DROPLETS	ubuntu-s-1vcpu-1gb-blr1-01	165.22.214.27
DROPLETS	ubuntu-s-4vcpu-8gb-blr1-01	165.22.209.4

# DevOps Monitoring

A vital part of DevOps is monitoring the servers that host our software.

We can track things such as:

- Server memory usage
- Incoming/Outgoing network traffic
- HTTP Requests

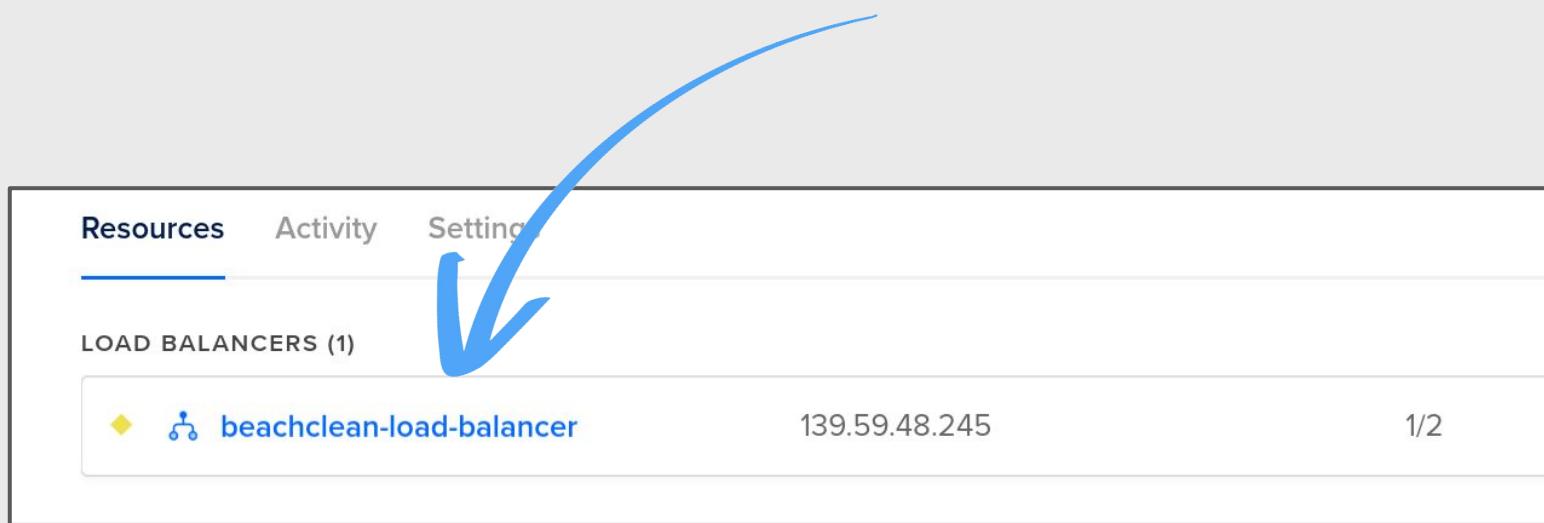


# DevOps Monitoring

Let's take a look at some of the DigitalOcean monitoring tools.

Then you'll add some custom tools of our own!

On your project dashboard, click on your Load Balancer.

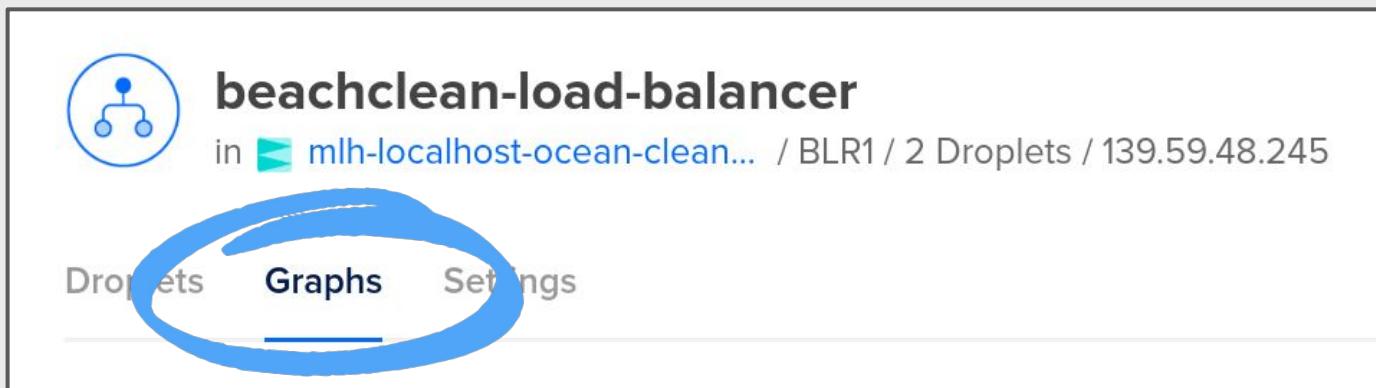


A screenshot of a DigitalOcean project dashboard. At the top, there are three tabs: 'Resources' (underlined in blue), 'Activity', and 'Settings'. Below the tabs, it says 'LOAD BALANCERS (1)'. A blue curved arrow points from the text 'click on your Load Balancer.' in the previous slide to the 'beachclean-load-balancer' entry. The entry itself consists of a small yellow diamond icon, a blue user icon, the text 'beachclean-load-balancer', the IP address '139.59.48.245', and the number '1/2'.

beachclean-load-balancer	139.59.48.245	1/2
◆ ⚙️ beachclean-load-balancer		

# DevOps Monitoring

Select **Graphs** from the tab menu.

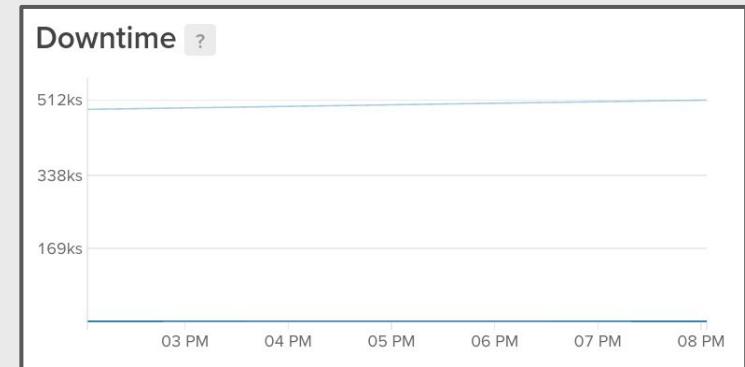
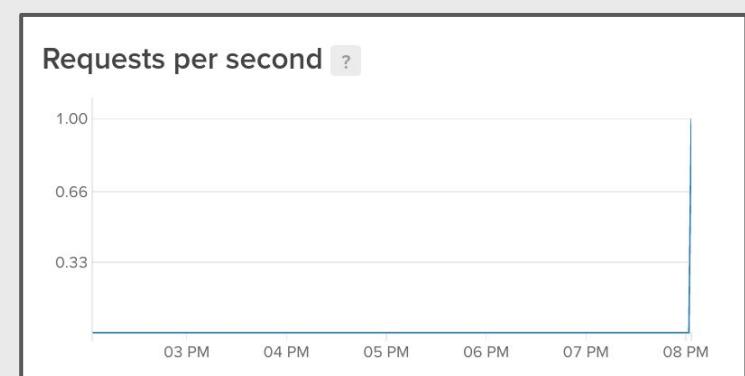


# DevOps Monitoring

Let's see what you can monitor for our Load Balancer.

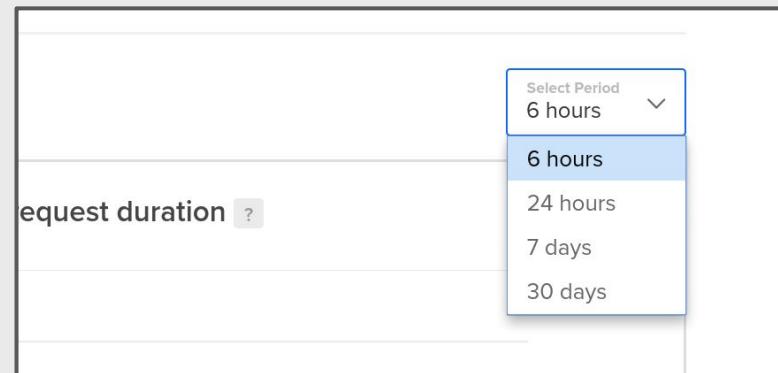
**Requests per second:** How many people are trying to access our Load Balancer every second? Here, you can see spikes that represent connections.

**Downtime:** How often is our server down, and for how long?



# DevOps Monitoring

We can click **Select Period** and monitor activity from the last 6 hours to the last 30 days.



# Alert Policies

An important aspect of monitoring is being alerted to potential issues before they become big problems.

With DigitalOcean you can create **Alert Policies**.

With an Alert Policy you define a threshold for a metric such as CPU or Memory usage.

When the threshold is breached, it sends us an alert via email.



## Key Term

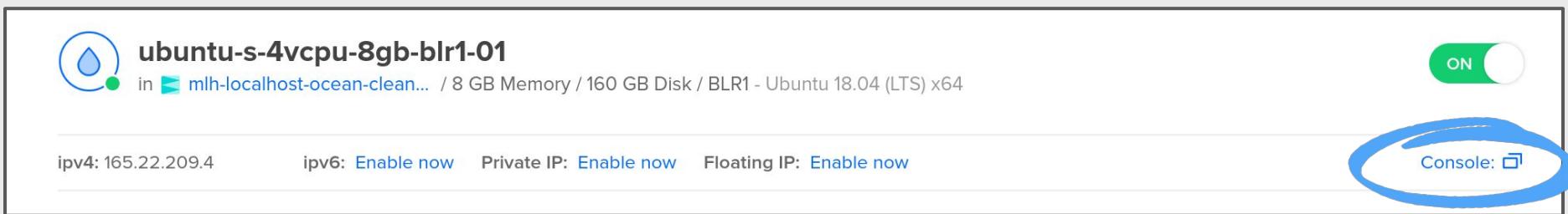
**Metric:** A measurable unit, such as percentage of memory being used, or number of requests received per second.

# Installing the Metrics Agent

To monitor a Droplet, you first need to install a Metrics Agent on it.

This provides some more in-depth monitoring tools.

To get started, click on your Droplet and open the [Console](#).



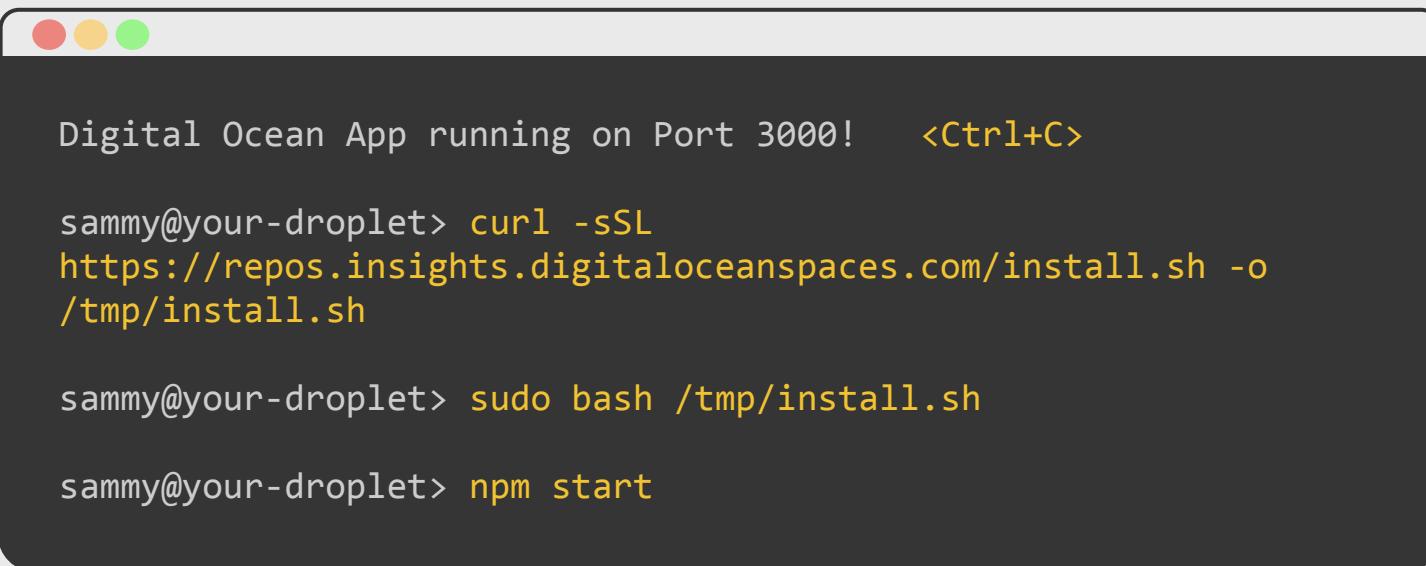
The screenshot shows a Droplet named "ubuntu-s-4vcpu-8gb-blr1-01" in the "mlh-localhost-ocean-clean..." region. The Droplet is currently active, indicated by the "ON" toggle switch. Below the Droplet name, its specifications are listed: 8 GB Memory / 160 GB Disk / BLR1 - Ubuntu 18.04 (LTS) x64. At the bottom of the card, there are four status buttons: "ipv4: 165.22.209.4", "ipv6: [Enable now](#)", "Private IP: [Enable now](#)", and "Floating IP: [Enable now](#)". On the far right, there is a "Console" button with a blue outline, which is highlighted with a large blue oval.

# Installing the Metrics Agent

If your app is currently running, you can stop it by pressing **Ctrl+C**.

Use the **Curl** command to download the Metrics Agent installation script and use the **Bash** command to run it.

Remember to start your application again with **npm start!**



```
Digital Ocean App running on Port 3000!    <Ctrl+C>

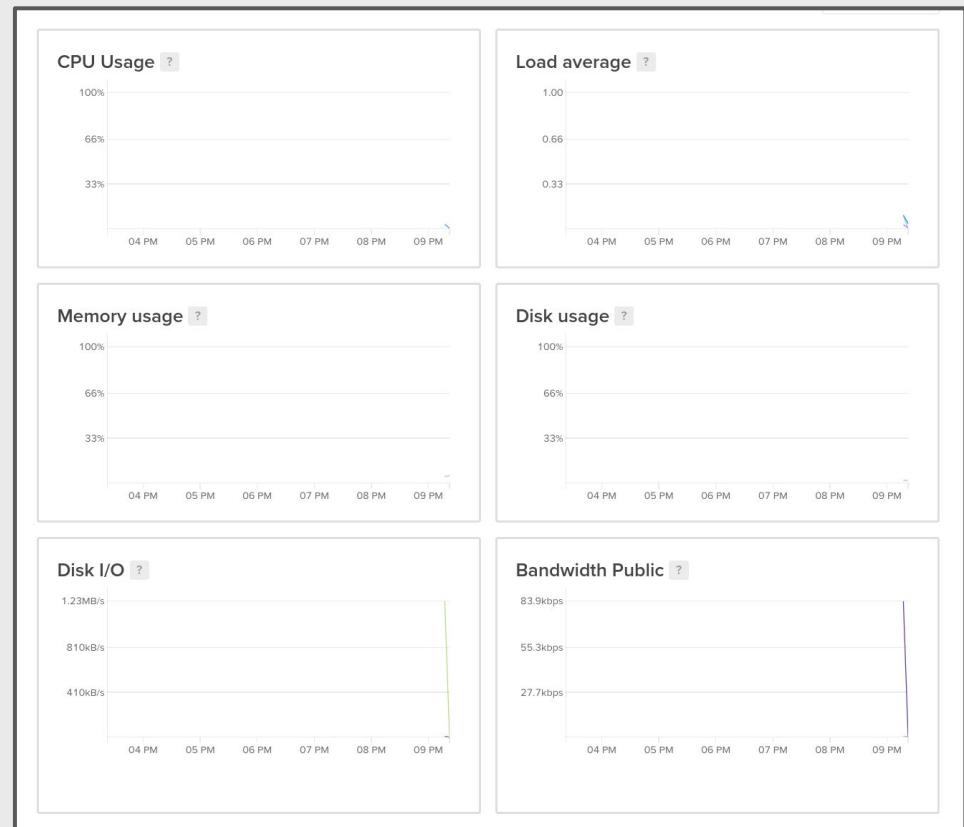
sammy@your-droplet> curl -sSL
https://repos.insights.digitaloceanspaces.com/install.sh -o
/tmp/install.sh

sammy@your-droplet> sudo bash /tmp/install.sh

sammy@your-droplet> npm start
```

# Metrics Agent Graphs

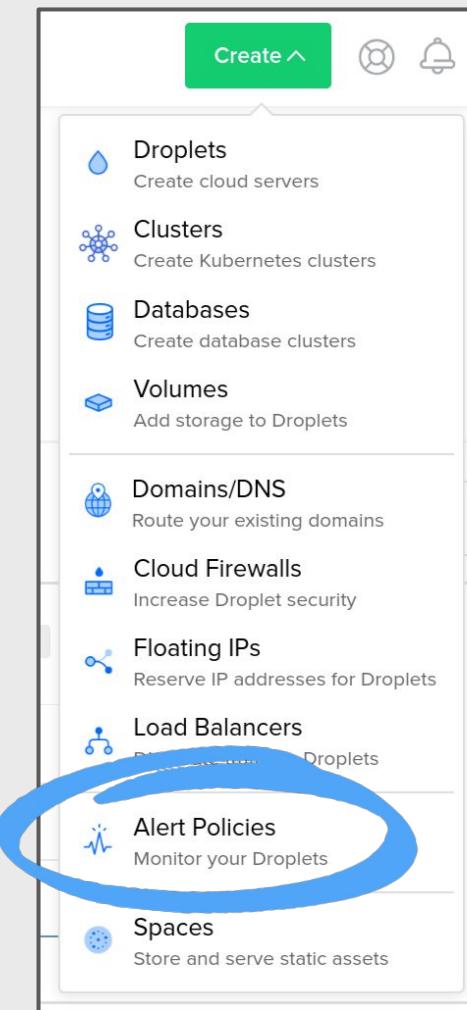
If you click on your Droplet dashboard, you'll see that you now have some more Monitoring graphs.



# Create an Alert Policy

Now you can create an Alert Policy for our Droplet.

To create an Alert Policy, click **Create**, then **Alert Policies**.



# Create an Alert Policy

Let's define the metric and threshold for our Alert Policy.

We want to be notified when the **CPU** usage of our Droplet **is above 70%** for more than **five minutes**.

Select metric & set threshold

CPU	is above	70	%	5 min	>
-----	----------	----	---	-------	---

## Key Term

**CPU (Central Processing Unit)**: The circuit in a computer that performs logical and mathematical operations.

# Create an Alert Policy

Search for your Droplet name and add it to the Alert Policy.

## Select Droplets or Tags

Alerting requires installation of the DigitalOcean metrics agent on your Droplets. When selecting all Droplets or tags, make sure the Droplets have the agent installed. [Get instructions](#) on how to install the metrics agent. The agent is pre-installed on Kubernetes worker nodes. To maintain alert policies with worker node recycling, tags are recommended over node names.

 ubuntu-s-4vcpu-8gb-blr1-01 ×

Please type a Droplet or Tag Name

# Create an Alert Policy

Send alerts via

Email eddie@majorleaguehacking.com [Add more recipients](#)



Click Add more recipients and add your email address.

Lastly, click **Create Alert Policy**.

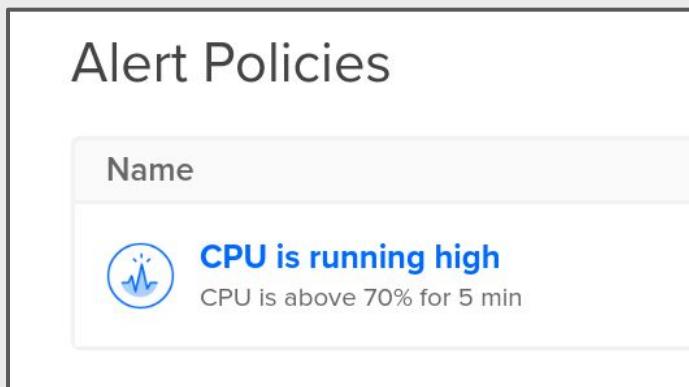
Name and create alert policy

CPU is running high



Create alert policy

# Create an Alert Policy



The screenshot shows a list of alert policies. One policy is highlighted, titled "CPU is running high". The description below it states "CPU is above 70% for 5 min". There is also a small icon of a blue circle with a white waveform inside.

Our Alert Policy is live!

Now, if our Droplet is using more than 70% of its CPU, you'll receive a notification via email.

This means you can take preemptive action before it causes any problems.

# DevOps Principles: Security

Let's focus on another one of the core DevOps principles: **Security**.

Our application is online and could be subject to remote attack over a network.

How can we combat this?

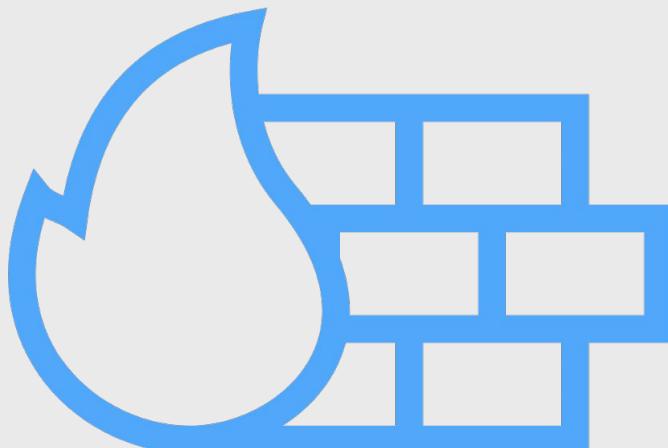
One way is through creating a **Firewall**.



# Adding a Firewall

**Firewalls** control what traffic is allowed to access a server.

If the traffic isn't allowed, it's dropped by the Firewall.



## Key Term

**Firewall:** A security service that monitors incoming and outgoing network traffic and uses rules to block unwanted traffic.

# Ports



Your app has an IP address - its network home.

Each IP address has  $2^{16}$  (more than 65,000!) possible **ports** that can connect to this IP address.

A port is a gateway to a network address and helps us identify what kind of information is being sent.

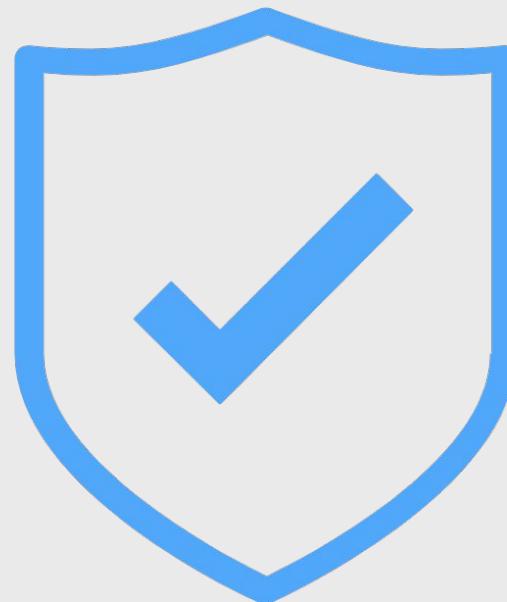
Some ports are reserved - for example, port 80 represents HTTP, or the world wide web.

We can write rules for Firewalls which can block or allow access to specific ports.

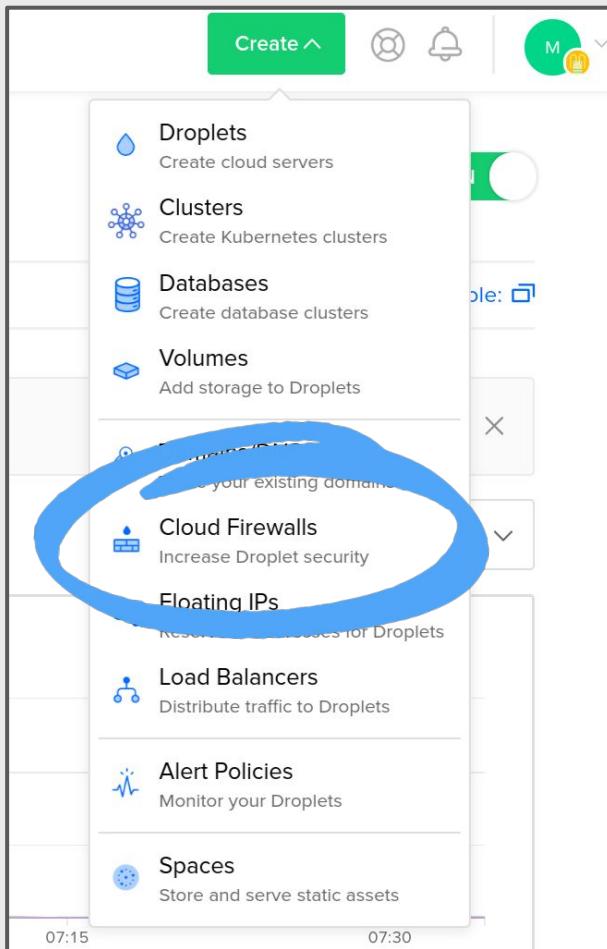
# Adding a Firewall

First, we'll demonstrate what a Firewall can do by creating one with **no rules**: so all traffic will be blocked!

Then, we'll create a rule so that users can only access our app's port, 3000.



# Adding a Firewall



Start by heading to the DigitalOcean dashboard and click **Create -> Cloud Firewall**.

# Adding a Firewall

First of all, give your firewall a name.

Name

Name beach-clean-firewall	
------------------------------	---

# Configuring the Firewall

In the **Inbound Rules** section, remove IPv4 and IPv6 from **Sources**.

This means that our Firewall isn't going to let any traffic though!

## Inbound Rules

Set the Firewall rules for incoming traffic. Only the specified ports will accept inbound connections. All other traffic will be dropped.

Type	Protocol	Port Range	Sources
SSH	TCP	22	<a href="#">Add a source</a>

### Key Term

**IPv4/IPv6:** Different network protocols which represent IP addresses. IPv4 uses 32 bits to store an address, giving 4.3 billion different possible addresses - and we're running out! IPv6 was introduced, which uses 128 bits per address.

# Configuring the Firewall

Lastly, select the Droplets you'd like to apply the Firewall to.

Search for your Droplet, and click **Create Firewall**.

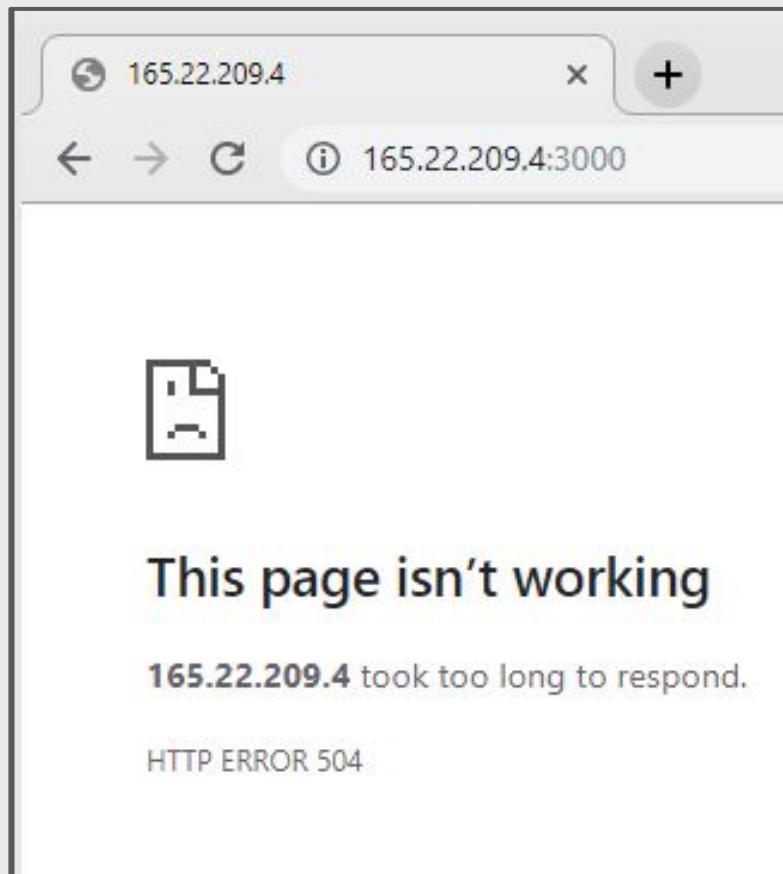
### Apply to Droplets

Select Droplets to apply your Firewall rules to.

ubuntu-s-4vcpu-8gb-blr1-01  Search for a Droplet or a tag

**Return to your application's IP address in  
your browser and try to load it again. What  
happens?**

# Testing the Firewall

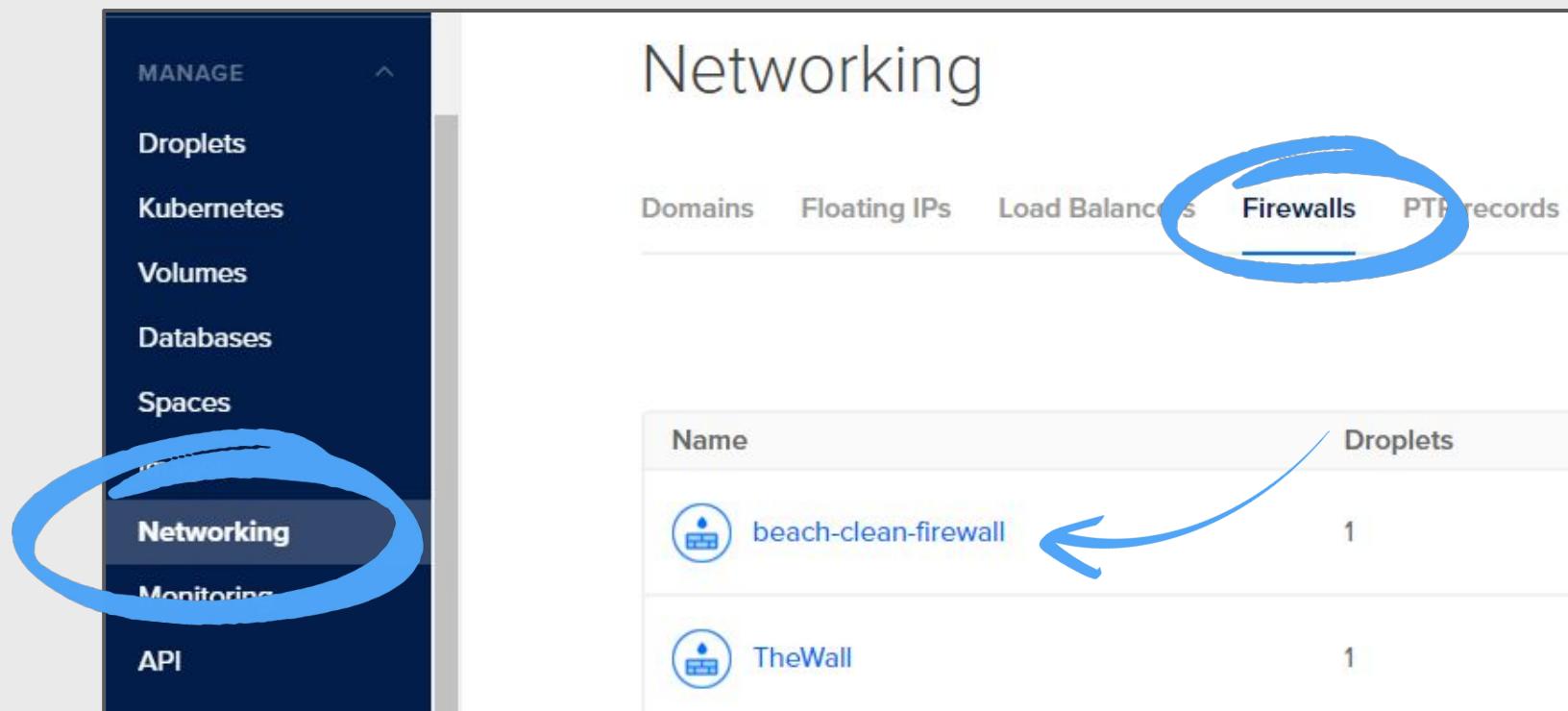


Our Firewall is successfully blocking all traffic, meaning that any requests made to our application are blocked!

We've demonstrated that our Firewall works, but let's add some rules so we can let some traffic through and still access our app.

# Re-Configuring the Firewall

On your Dashboard, click the Networking option, then select the Firewalls tab. Click on the Firewall you just made.



# Re-Configuring the Firewall

In the **Inbound Rules** section, select **More** and then **Edit Rule**.

## Inbound Rules

Set the Firewall rules for incoming traffic. Only the specified ports will accept inbound connections. All other traffic will be blocked.

Type	Protocol	Port Range	Sources	
SSH	TCP	22	(None)	<a href="#">More ^</a>

New rule ▾

[Edit Rule](#)  
[Delete Rule](#)

# Re-Configuring the Firewall

Change the Type to **Custom** and the Ports to **3000**.

This means that our Firewall will only allow traffic to access our server if it travels through Port 3000.

In Sources, add **All IPv4** and **All IPv6**.

## Inbound Rules

Set the Firewall rules for incoming traffic. Only the specified ports will accept inbound connections. All other traffic will be dropped.

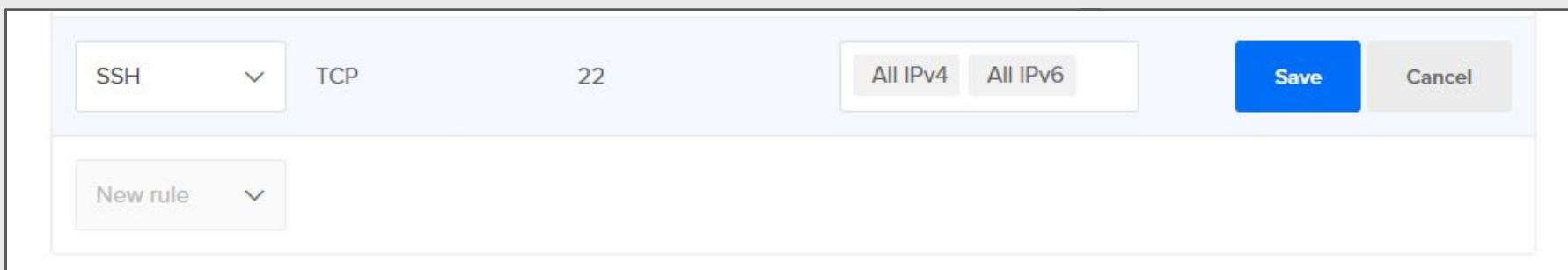
Type	Protocol	Port Range	Sources
Custom	TCP	Ports 3000	All IPv4 All IPv6

# Re-Configuring the Firewall

Add another rule, and select **SSH**.

This will allow us to connect to our application remotely via secure shell,

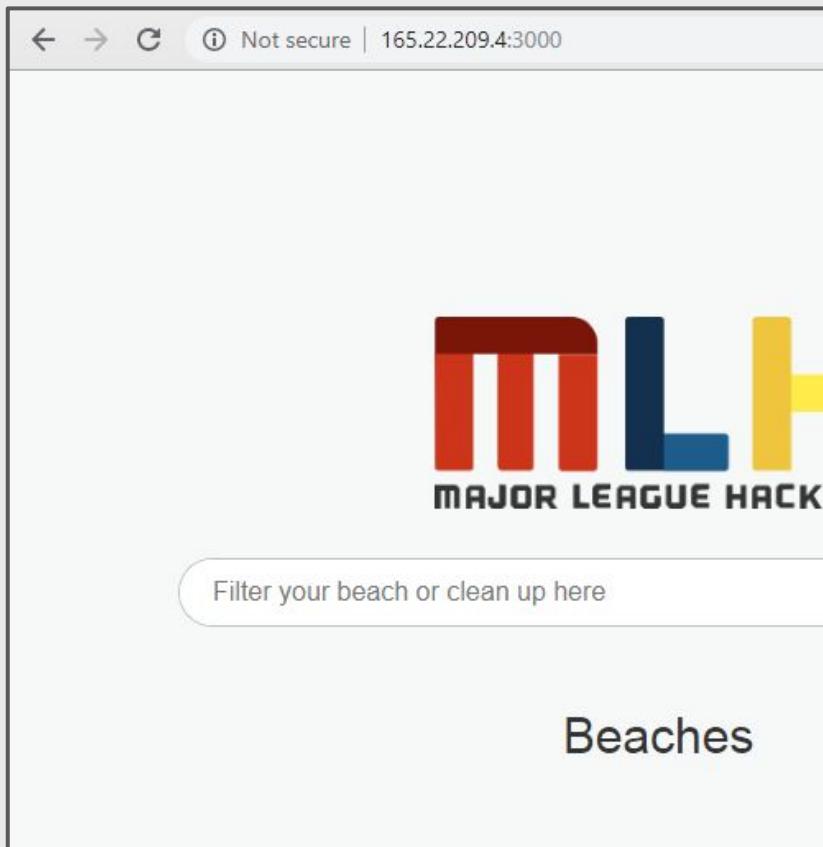
Accept the default values and click **Save**.



# Testing the Firewall

Now try refreshing your app.

Your Firewall rule has updated and allowed traffic on Port 3000, so your app is accessible again!



# Table of Contents

0. Welcome to MLH Localhost
1. What are you building?
2. Deploying to DigitalOcean
3. DevOps 101
-  4. Review & Next Steps

# Let's Recap

*What did you learn?*

- 1 DigitalOcean provides Cloud Computing services and allows us to host applications in Droplets.
- 2 DevOps is the practise of deploying, securing and updating software.
- 3 We can add features such as Firewalls, Load Balancers and Alert Policies to our Droplets.

# What did you learn today?

We created a fun quiz to test your knowledge and see what you learned from this workshop.

**<http://mlhlocal.host/quiz>**

# Learning shouldn't stop when the workshop ends...

**Check your email for access to:**



- These workshop slides
- Practice problems to keep learning
- Deeper dives into key topics
- Instructions to join the community
- More opportunities from MLH!

# Next Steps

- 1 Investigate other Alert Policies you could add to your Droplet.
- 2 Research **Automatic Deployment** - where the app would reload every time you make a change to a GitHub repository.
- 3 Try changing your Alert Policy notifications from email to Slack!



Workshop

# App Deployment & Security with DigitalOcean



localhost



DigitalOcean