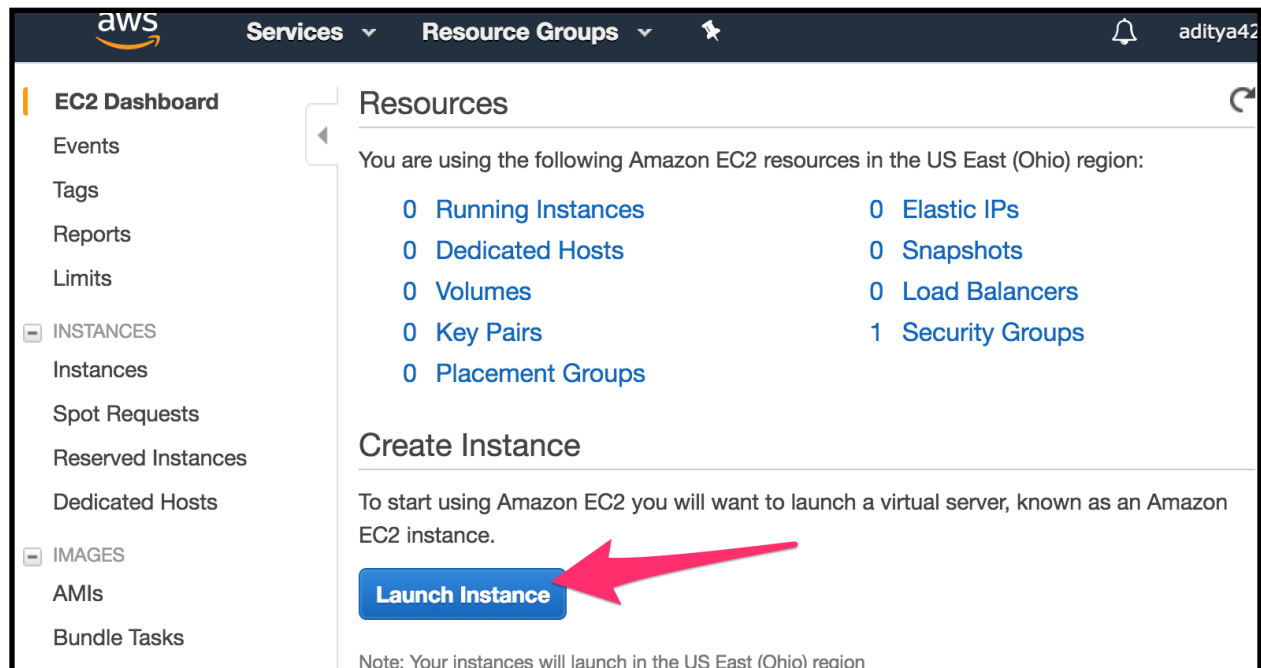


Deploying my Rails web app to AWS

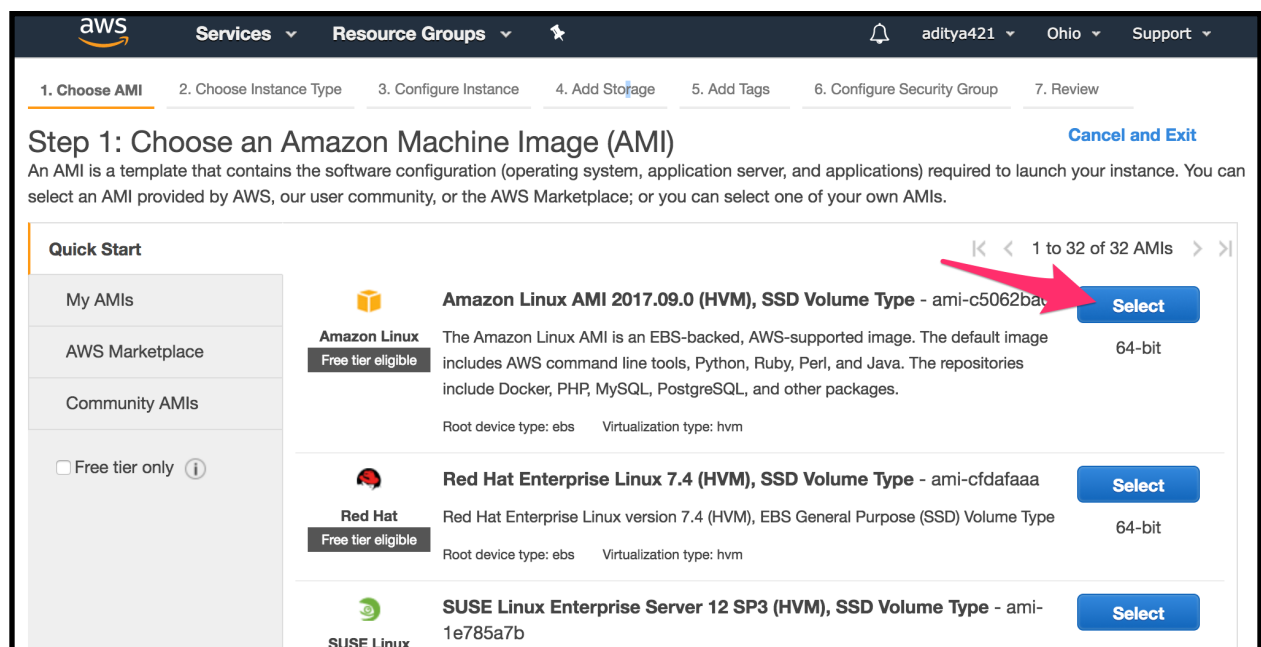
To begin with Login to your AWS account

Setting up EC2 Instance:

1. Launch Instance



2. Choose an AMI (Preferred Amazon Linux AMI)



Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance types Current generation [Show/Hide Columns](#)

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	General purpose	t2.micro <small>Free tier eligible</small>	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Instance Details](#)

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances: [Launch into Auto Scaling Group](#)

Purchasing option: ☐ Request Spot instances

Network: [Create new VPC](#)

Subnet: [Create new subnet](#)

Auto-assign Public IP:

IAM role: [Create new IAM role](#)

Shutdown behavior:

Enable termination protection: ☐ Protect against accidental termination

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Storage](#)

3. Choose instance type:
4. Configure Instance Details:(Leave as it is and hit next)

5. Add Storage: (Leave it as it is and hit next)

aws Services Resource Groups

aditya421 Ohio Support

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encrypted
Root	/dev/xvda	snap-05d46fafb13fd7ac7	8	General Purpose SSD	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypted

[Add New Volume](#)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Tags](#)

6. Add Tags (Leave as it is and hit next)

aws Services Resource Groups

aditya421 Ohio Support

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. A copy of a tag can be applied to volumes, instances or both. Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

Key (127 characters maximum)	Value (255 characters maximum)	Instances	Volumes
This resource currently has no tags			
Choose the Add tag button or click to add a Name tag . Make sure your IAM policy includes permissions to create tags.			

[Add Tag](#) (Up to 50 tags maximum)

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Security Group](#)

aws

Services

Resource Groups

aditya421

Ohio

Support

1. Choose AMI

2. Choose Instance Type

3. Configure Instance

4. Add Storage

5. Add Tags

6. Configure Security Group

7. Review

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group ☐ Select an existing security group

Security group name: mywebsite-sg

Description: launch-wizard-1 created 2017-10-15T23:35:15.763-05:00

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Anywhere 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

Add Rule

Warning

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Cancel

Previous

Review and Launch

7. Configure security group

8. Review and Launch Instance

aws

Services

Resource Groups

aditya421

Ohio

Support

1. Choose AMI

2. Choose Instance Type

3. Configure Instance

4. Add Storage

5. Add Tags

6. Configure Security Group

7. Review

Step 7: Review Instance Launch

Root Device Type: ebs Virtualization type: hvm

▼ Instance Type [Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

▼ Security Groups [Edit security groups](#)

Security group name mywebsite-sg

Description launch-wizard-1 created 2017-10-15T23:35:15.763-05:00

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	0.0.0.0/0	
SSH	TCP	22	::/0	

Cancel

Previous

Launch

Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

Create a new key pair

Key pair name
mywebsite-keypair

Download Key Pair

You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

Cancel

Launch Instances

9. Create a new key-pair and save it locally(ex: Desktop) and launch the instance (This key will be used to access your EC2 instance through ssh) **Keep it safe!**
10. Browse to AWS dashboard > EC2 > You should be able to see 1 running instance:

aws

Services

Resource Groups

EC2 Dashboard

Events

Tags

Reports

Limits

INSTANCES

Instances

Resources

You are using the following Amazon EC2 resources in the US East (Ohio) region:

1 Running Instances

0 Elastic IPs

0 Dedicated Hosts

0 Snapshots

1 Volumes

0 Load Balancers

1 Key Pairs

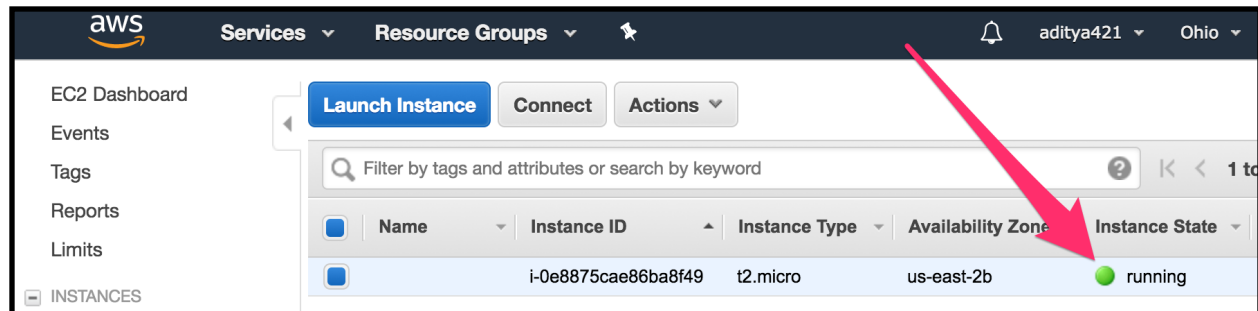
2 Security Groups

0 Placement Groups

EC2 setup is now complete!

Accessing your EC2 Instance:

1. Browse to AWS dashboard > EC2 > Running Instances:

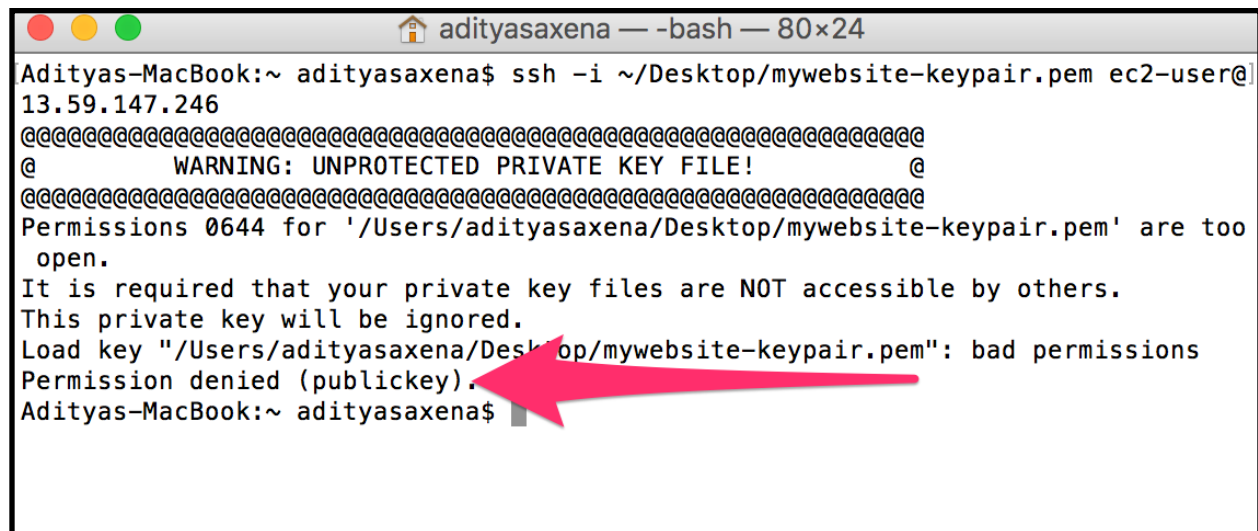


- ## 2. SSH into your instance:

Open your terminal and use below command (replace the IP with the IP of your instance)

```
ssh -i ~/Desktop/mywebsite-keypair.pem ec2-user@yourEC2IPAddress
```

As you can see it failed!



You need to update the permissions:

```
chmod 400 <path to your .pem file>
```



and run ssh command again: Access successful!

Deploying my web app from repository on GitHub:

My rails app is on GitHub repository and in order to get it on our EC2 instance we need to install git on our EC2 server:

1. As we can see nothing is installed on our EC2 server right now:

```
adityasaxena — ec2-user@ip-172-31-18-14:~ — ssh -i ~/Desktop/mywebsi
[ec2-user@ip-172-31-18-14 ~]$ ls -aF
./ ../ .bash_logout .bash_profile .bashrc .ssh/
[ec2-user@ip-172-31-18-14 ~]$
```

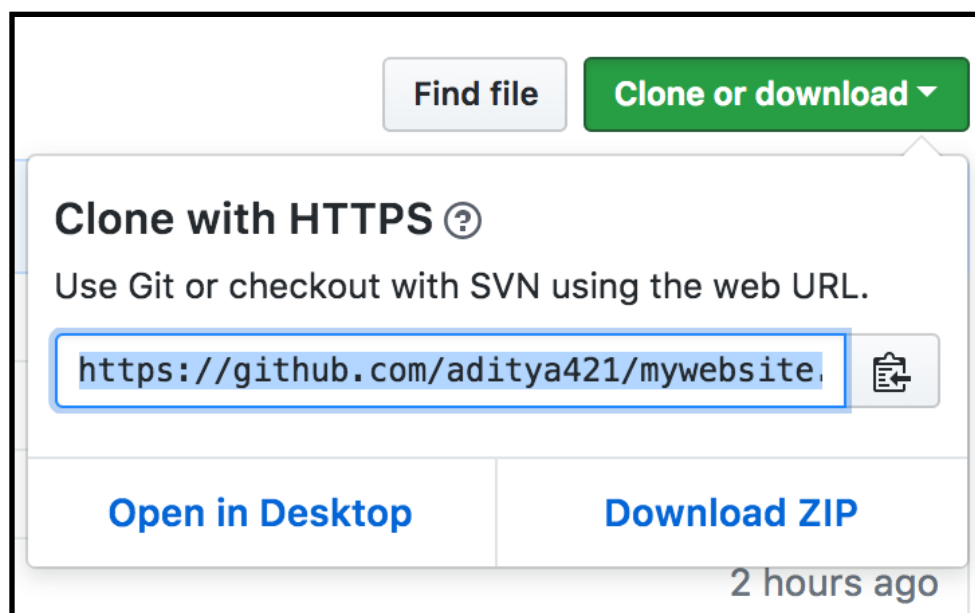
2. We will now install git on our EC2 instance
(Amazon linux uses yum as the default packet manager)

```
$ sudo yum install git-all
```

Once installed you can check that we have installed git:

```
Complete!
[ec2-user@ip-172-31-18-14 ~]$ which git
/usr/bin/git
[ec2-user@ip-172-31-18-14 ~]$
```

3. Go to your repository on GIT and get the link and clone it as follows:




```
adityasaxena — ec2-user@ip-172-31-18-14:~ — ssh -i ~/Desktop/mywebsite-...
[ec2-user@ip-172-31-18-14 ~]$ git clone https://github.com/aditya421/mywebsite.git
Cloning into 'mywebsite'...
remote: Counting objects: 108, done.
remote: Compressing objects: 100% (92/92), done.
remote: Total 108 (delta 2), reused 108 (delta 2), pack-reused 0
Receiving objects: 100% (108/108), 190.29 KiB | 5.14 MiB/s, done.
Resolving deltas: 100% (2/2), done.
[ec2-user@ip-172-31-18-14 ~]$ ls -al
total 32
drwx----- 5 ec2-user ec2-user 4096 Oct 16 05:46 .
drwxr-xr-x 3 root      root    4096 Oct 16 04:47 ..
-rw-r--r-- 1 ec2-user ec2-user   18 Aug 30 19:00 .bash_logout
-rw-r--r-- 1 ec2-user ec2-user  193 Aug 30 19:00 .bash_profile
-rw-r--r-- 1 ec2-user ec2-user  124 Aug 30 19:00 .bashrc
drwxrwxr-x 13 ec2-user ec2-user 4096 Oct 16 05:46 mywebsite
drwxrw---- 3 ec2-user ec2-user 4096 Oct 16 05:46 .pki
drwx----- 2 ec2-user ec2-user 4096 Oct 16 04:47 .ssh
[ec2-user@ip-172-31-18-14 ~]$
```

As you can see my repository (mywebsite) appears on our ec2 instance.

Now that we have our rails app available on our EC2 server, we need rails installed in order to run my app.

For that follow instructions online and set up rails in order to run the app.

After setup test your app as follows:

<https://rvm.io/>

gem install bundler

bundle install

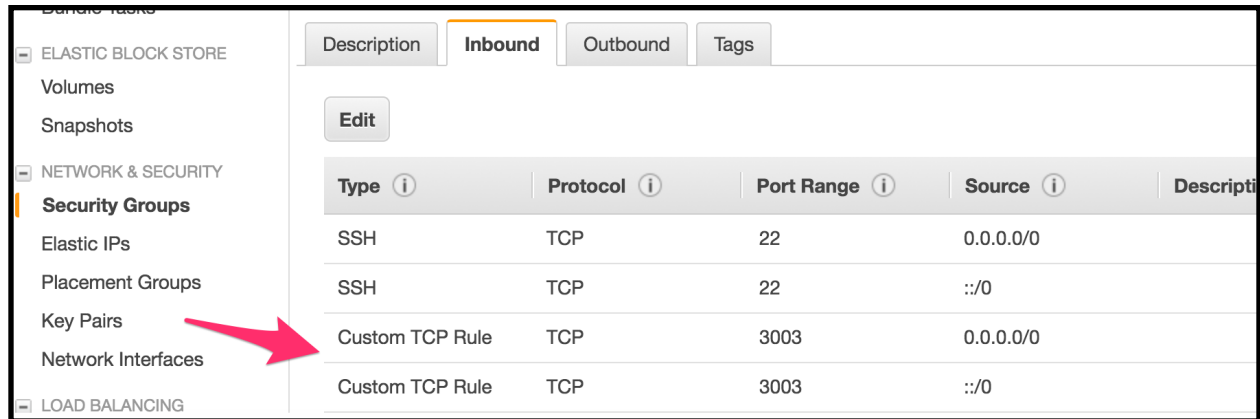
ruby -v && rails -v

install nodejs to support your rails app

```
sudo ln -sf /usr/bin/nodejs /usr/local/bin/node
```


In order to view your rails app through your EC2 IP Address:

Add the following entry to allow traffic on port 3003 in EC2>Network Security>Security Groups



The screenshot shows the AWS Management Console interface for a Security Group. The left sidebar contains a navigation menu with categories: ELASTIC BLOCK STORE, NETWORK & SECURITY, and LOAD BALANCING. Under NETWORK & SECURITY, the following items are listed: Volumes, Snapshots, Security Groups (highlighted with an orange bar), Elastic IPs, Placement Groups, Key Pairs, and Network Interfaces. A red arrow points to the 'Network Interfaces' link. The main content area shows the 'Inbound' tab selected, with an 'Edit' button and a table of inbound rules.

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	0.0.0.0/0	
SSH	TCP	22	::/0	
Custom TCP Rule	TCP	3003	0.0.0.0/0	
Custom TCP Rule	TCP	3003	::/0	

Browse to your project directory and run the following command to start your app:

```
$ rails s -p 3003 -b 0.0.0.0
```

Access your web app through the IP of your instance: (Remember to add port :3003 in front of the IP address)