

Deploying a Web Service in the Cloud

Luigi Libero Lucio Starace

luigiliberolucio.starace@unina.it

Università degli Studi di Napoli Federico II, Naples, Italy

November 19, 2021

Goals of this demonstration

- Download and run a simple Node.js web service locally
 - Requirements: Node.js (recommended v14.16.1 or above)
- Deploy the web service in the Cloud using AWS (on an AWS EC2 instance)

Download the web service

Clone the repo: <https://github.com/luistar/cloud-ws-demo.git>

```
>> git clone https://github.com/luistar/cloud-ws-demo.git
Cloning into 'cloud-ws-demo'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 11 (delta 2), reused 7 (delta 1), pack-reused 0
Unpacking objects: 100% (11/11), done.
```

Then move to the newly-created directory

```
>> cd cloud-ws-demo
```

Check out the app.js file (if you want)

Two get endpoints: /greet and /fortune

```
const express = require('express')
const app = express()
const port = 3000

app.get('/greet', (req, res) => {
  res.send('Hello World!')
})

app.get('/fortune', (req, res) => {
  // [...] computes a random fortune message (omitted for brevity)
  res.send(fortune)
})
```

Install dependencies and run the web service

Install dependencies

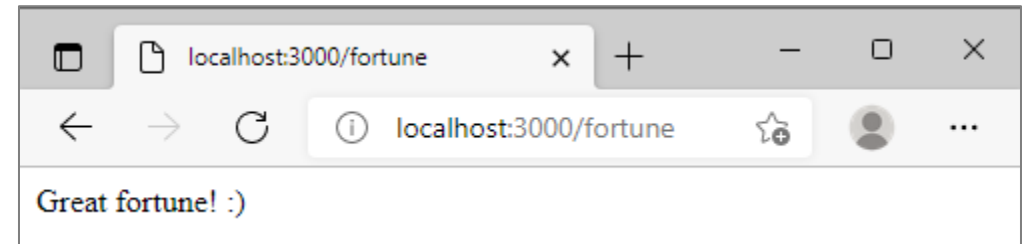
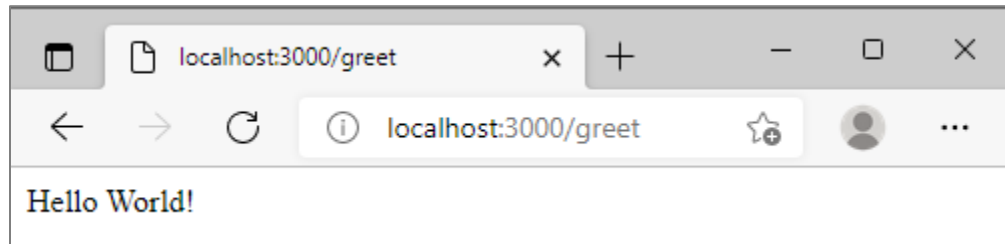
```
>> npm install
```

Run the web service

```
>> node app.js
```

```
Web service listening at http://localhost:3000
```

And try it with your web browser!



Let's move to the Cloud

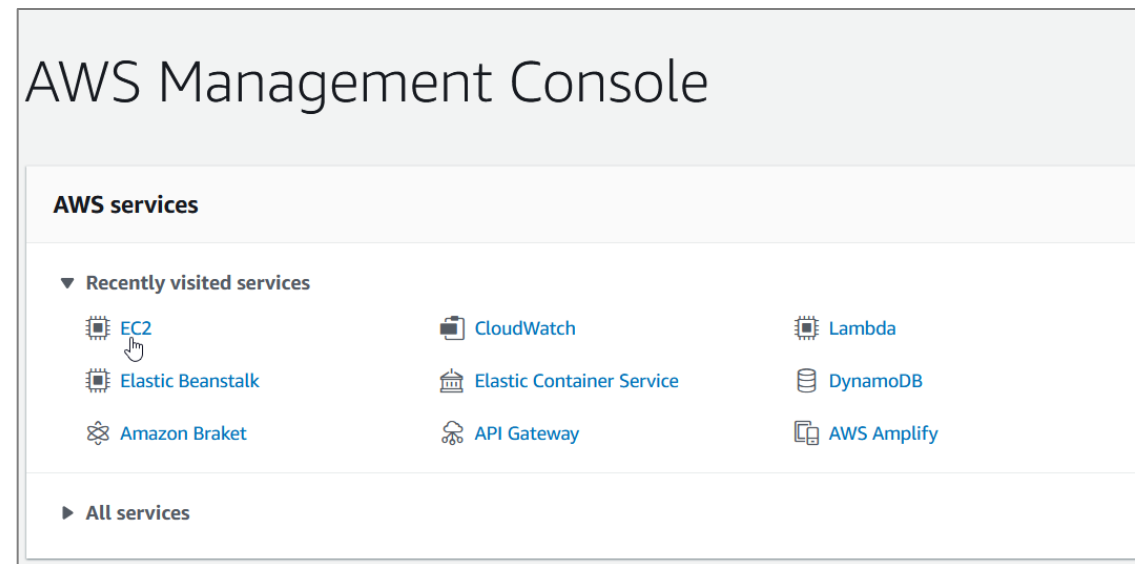
We'll use AWS for this tutorial

Requirements

- To replicate this demo you'll need an AWS account. If you're eligible for the free trial, it'll cost you nothing!
- If you prefer a different provider (such as Azure) the procedure is basically the same (so a little of google-fu will surely suffice!)

EC2 deployment

- We'll start from the most basic deployment: running the web service on a virtual server in the Cloud.
- Let's start by selecting the EC2 service from the AWS web console



Click on the «Launch instance» button

The screenshot displays the AWS Management Console interface for the EC2 service. The top navigation bar includes the AWS logo, a search bar, and user information. The left sidebar shows the 'EC2 Dashboard' with various navigation links. The main content area is divided into several sections:

- Resources:** A table showing the number of EC2 resources in the US East (N. Virginia) Region. The 'Launch instance' button is highlighted with a mouse cursor.
- Launch instance:** A section with a description of EC2 instances and a 'Launch instance' button.
- Service health:** A section showing the status of the EC2 service, indicating it is 'operating normally'.
- Account attributes:** A section showing account details such as 'Supported platforms', 'Default VPC', and 'Settings'.
- Explore AWS:** A section with promotional content for AWS Graviton2.

Resource	Count
Instances (running)	0
Elastic IPs	0
Key pairs	0
Placement groups	0
Snapshots	0
Dedicated Hosts	0
Instances	0
Load balancers	0
Security groups	1
Volumes	0

Select the Ubuntu Server 20.04 LTS Machine Image

The screenshot shows the AWS Management Console interface for selecting an Amazon Machine Image (AMI). The top navigation bar includes the AWS logo, a search bar, and user information. The main content area is titled 'Step 1: Choose an Amazon Machine Image (AMI)' and features a list of available AMIs. The first AMI, 'Ubuntu Server 20.04 LTS (HVM), SSD Volume Type', is highlighted with a 'Free tier eligible' badge. The second AMI, 'Ubuntu Server 18.04 LTS (HVM), SSD Volume Type', is also listed. The page includes a 'Cancel and Exit' link in the top right corner.

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

Volume Type: Amazon EC2 AMI Tools preinstalled; Apache 2.2, MySQL 5.5, PHP 5.3, and Ruby 1.8.7 available.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Free tier eligible

Ubuntu Server 20.04 LTS (HVM), SSD Volume Type -
ami-083654bd07b5da81d (64-bit x86) / ami-04fe9398b2a27a600 (64-bit Arm)

Ubuntu Server 20.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Ubuntu Server 18.04 LTS (HVM), SSD Volume Type -
ami-0279c3b3186e54acd (64-bit x86) / ami-0528007a60177dd84 (64-bit Arm)

Select

Select

☐ 64-bit (Arm)

☒ 64-bit (x86)

☐ 64-bit (Arm)

Select the t2.micro instance type

aws

Services

Search for services, features, blogs, docs, and more

[Alt+S]

vocstartsoft/user1358543=luigiliberolucio.starace@unina.it @ ...

N. Virginia

Support

1. Choose AMI

2. Choose Instance Type

3. Configure Instance

4. Add Storage

5. Add Tags

6. Configure Security Group

7. Review

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 families

Current generation

[Show/Hide Columns](#)

Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
<input type="checkbox"/>	t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	t2	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.large	2	8	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.xlarge	4	16	EBS only	-	Moderate	Yes
<input type="checkbox"/>	t2	t2.2xlarge	8	32	EBS only	-	Moderate	Yes

Cancel

Previous

Review and Launch

Next: Configure Instance Details

Generate a new key pair to connect to the instance

- Make sure to download the .pem file

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. Amazon EC2 supports ED25519 and RSA key pair types.

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 type

☒ RSA ☐ ED25519

Key pair name

EC2 key pair

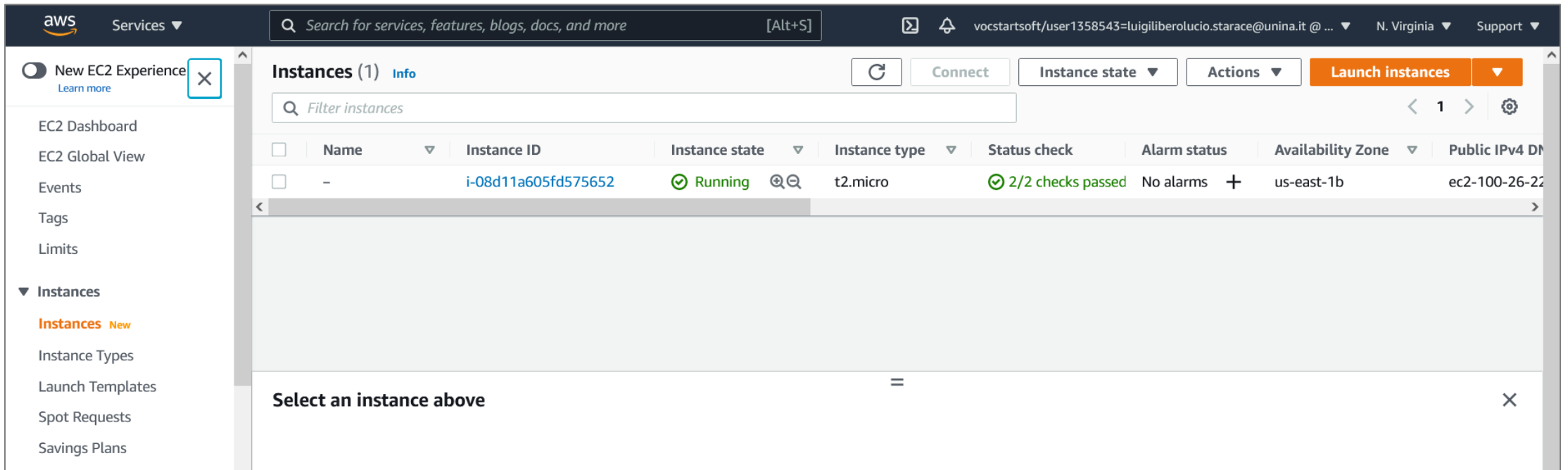
Download Key Pair

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

Wait for the instance to start

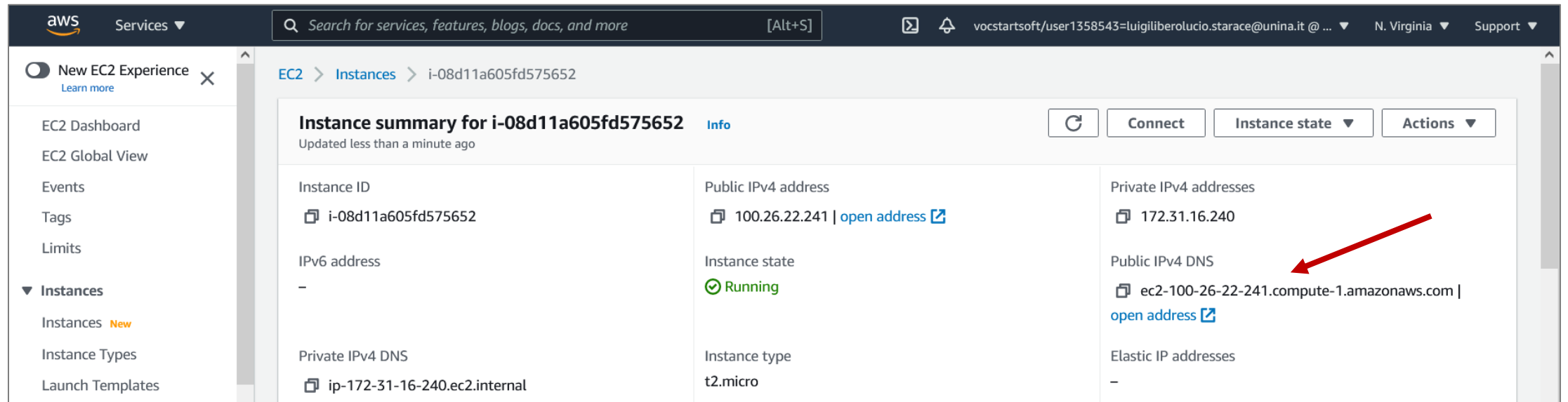


The screenshot displays the AWS Management Console interface. At the top, the navigation bar includes the AWS logo, a search bar, and user information. The left-hand navigation menu is open, showing various services like EC2 Dashboard, Events, and Tags. The 'Instances' section is selected, and the 'Instances (1)' page is displayed. This page features a table with one instance listed. The instance is named 'i-08d11a605fd575652', is in the 'Running' state, and has a 't2.micro' instance type. The status check shows '2/2 checks passed'. Below the table, there is a message that says 'Select an instance above'.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 D
-	i-08d11a605fd575652	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-100-26-22

Click on the instance id to know more

Note the public IPv4 DNS name!



The screenshot shows the AWS Management Console interface for an EC2 instance. The breadcrumb navigation indicates the path: EC2 > Instances > i-08d11a605fd575652. The instance summary section displays various details:

Instance summary for i-08d11a605fd575652 Info		
Updated less than a minute ago		
Instance ID i-08d11a605fd575652	Public IPv4 address 100.26.22.241 open address	Private IPv4 addresses 172.31.16.240
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-100-26-22-241.compute-1.amazonaws.com open address
Private IPv4 DNS ip-172-31-16-240.ec2.internal	Instance type t2.micro	Elastic IP addresses -

A red arrow points to the 'Public IPv4 DNS' field, highlighting the value 'ec2-100-26-22-241.compute-1.amazonaws.com'.

SSH into your new (virtual) server

SSH into your instance

```
>> ssh -i /path/to/key.pem ubuntu@<the-ipv4-dns-name-we-noted>
```

If everything worked, you should be in!

```
ubuntu@ip-172-31-16-240:~$
```

Notice: you might be getting an «**unprotected private key file**» error. In that case, make sure to correctly assign permissions to the pem file, so that only you can see it. Use `chmod` if on a unix derivative, or see [this](#) if you're on Windows.

Install Node.js on your virtual server

Update and then install nodejs and npm

```
ubuntu@ip-172-31-16-240:~$ sudo apt update
```

```
ubuntu@ip-172-31-16-240:~$ sudo apt install nodejs npm
```


Then clone the web service

Clone and install dependencies

```
ubuntu@ip-172-31-16-240:~$ git clone https://github.com/luistar/cloud-ws-demo.git  
ubuntu@ip-172-31-16-240:~$ cd cloud-ws-demo  
ubuntu@ip-172-31-16-240:~/cloud-ws-demo$ npm install
```

And then you're ready to start our Web Service

```
ubuntu@ip-172-31-16-240:~/cloud-ws-demo$ node app.js  
Web service listening at http://localhost:3000
```

We're almost there!

Our web service is up and running in our virtual server, but we still cannot reach it from the internet!



Allow incoming traffic

- Select the security group in the security tab in your instance summary page

November 19, 2021

The screenshot displays the AWS Management Console interface. At the top, the AWS logo and 'Services' dropdown are visible. A search bar contains the text 'Search for services, features, blogs, docs, and more'. The left sidebar shows the navigation menu with categories like 'New EC2 Experience', 'Instances', 'Images', 'Elastic Block Store', and 'Network & Security'. The main content area shows the 'Instance summary for i-08d11a605fd575652' page. The 'Security' tab is selected, and a red arrow points to the 'Security groups' section, which lists 'sg-081860d1acce4b99 (launch-wizard-1)'.

aws Services

Search for services, features, blogs, docs, and more

EC2 > Instances > i-08d11a605fd575652

Instance summary for i-08d11a605fd575652 Info

Updated less than a minute ago

Instance ID
i-08d11a605fd575652

IPv6 address
-

Private IPv4 DNS
ip-172-31-16-240.ec2.internal

VPC ID
vpc-5b07b826

Subnet ID
subnet-820b39cf

Details Security Networking Storage Status checks

▼ Security details

IAM Role
-

Security groups
sg-081860d1acce4b99 (launch-wizard-1)

Add a new inbound traffic rule

Type: Custom TCP

Port range: 3000

Source: Anywhere IPv4

The screenshot shows the AWS Management Console interface for editing inbound rules on a Security Group. The breadcrumb trail indicates the path: EC2 > Security Groups > sg-081860d1acce4b99 - launch-wizard-1 > Edit inbound rules. The page title is 'Edit inbound rules' with an 'Info' link. A subtitle states: 'Inbound rules control the incoming traffic that's allowed to reach the instance.'

The 'Inbound rules' section shows a table with one rule, 'Inbound rule 1'. The rule configuration is as follows:

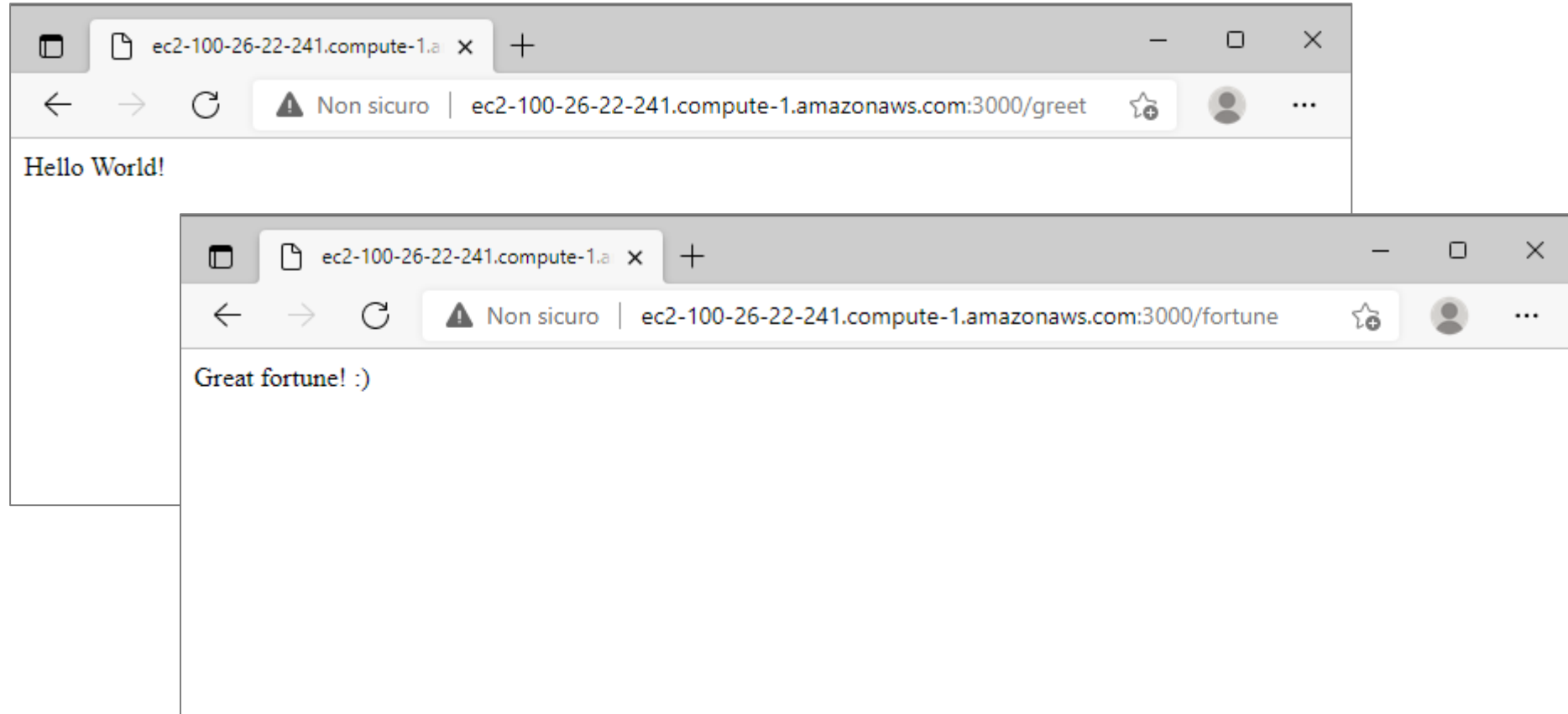
Security group rule ID	Type	Protocol
sgr-0c5783850cdb9e099	Custom TCP	TCP

Additional configuration details for the rule:

- Port range: 3000
- Source type: Anywhere-IPv4
- Source: 0.0.0.0/0
- Description - optional: Allow users to reach our web service

The 'Add rule' button is located at the bottom of the rule configuration area.

Guess what? Now it works!



Creating an image from our instance

Updating the repositories, installing Node.js and npm, downloading and installing our web service was **fun**, sure.

Can we skip all that next time? Yes, we can create an image!

The screenshot displays the AWS Management Console interface. At the top, a green banner indicates a successful operation: "Successfully created ami-0111ce16133f6fa5f from instance i-08d11a605fd575652." Below this, the "Instances (1/1)" section is visible, featuring a search bar and a table of instances. The table contains one instance with the ID "i-08d11a605fd575652", which is in a "Running" state. To the right of the table, the "Actions" dropdown menu is open, showing options such as "Connect", "View details", "Manage instance state", "Instance settings", "Networking", "Security", "Image and templates", and "Monitor and troubleshoot". The "Create image" option is highlighted, and a sub-menu is visible with options "Create image", "Create template from instance", and "Launch more like this".

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
<input checked="" type="checkbox"/>	-	i-08d11a605fd575652	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-100-26-22-241

Terminate our instance

- Remember that we're paying as long as that instance is up! Terminate it when it's not needed anymore!

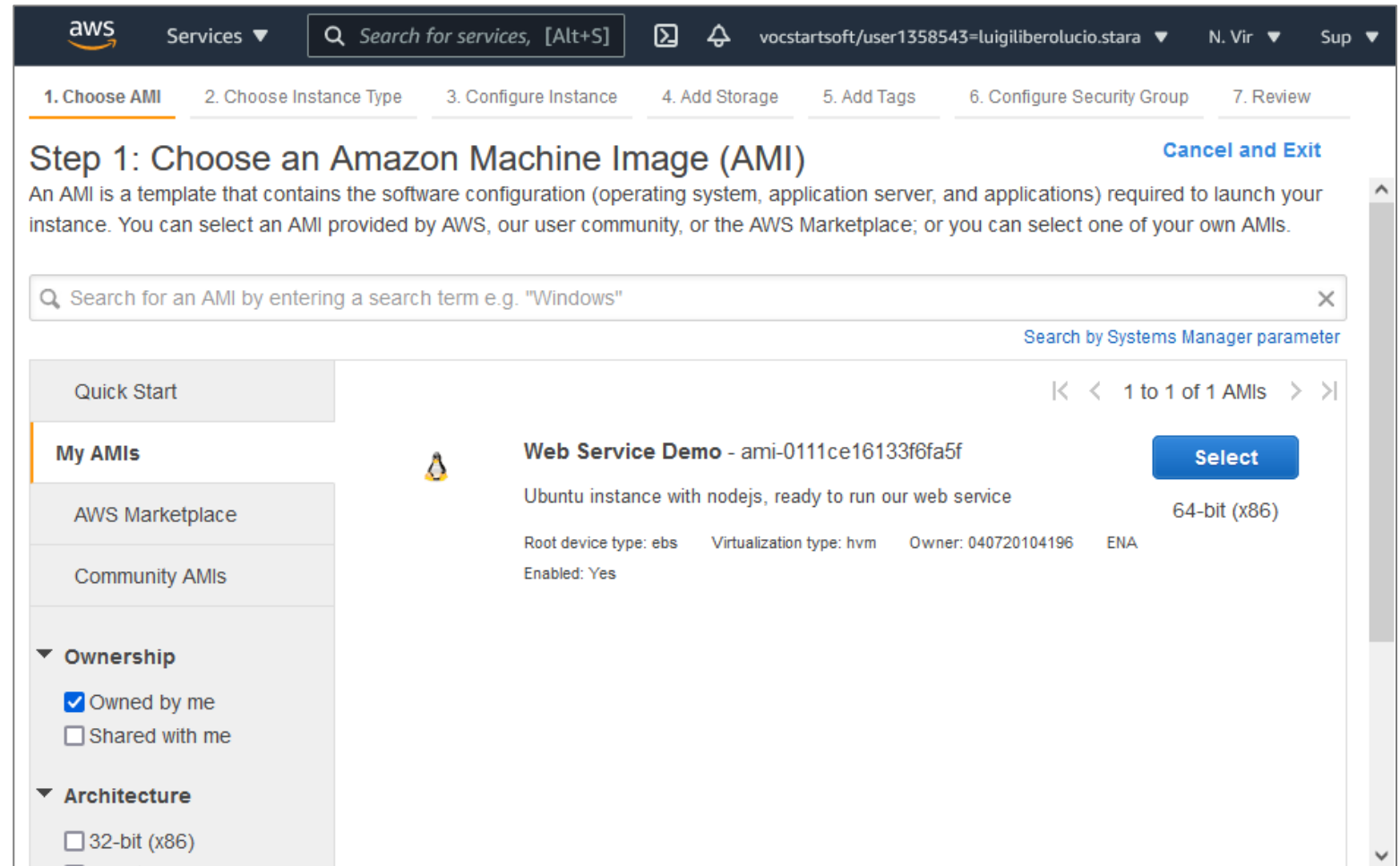
The screenshot shows the AWS Management Console interface. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, a search bar, and user information. A green banner at the top of the console area says 'Successfully created ami-0111ce16133f6fa5f from instance i-08d11a605fd575652.' Below this, the 'Instances (1/1)' section is active. A table lists the instance details:

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input checked="" type="checkbox"/>	-	i-08d11a605fd575652	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b

The 'Instance state' dropdown menu is open, showing the following options: Stop instance, Start instance, Reboot instance, Hibernate instance, and Terminate instance. The 'Launch instances' button is visible in the top right corner of the console area.

Start the instance from the image we saved

This time we can select our image!



Running the image we saved

- This time, when starting the EC2 instance, make sure to use the same security group we already set up! (otherwise, you will need to add the inbound rule to port 3000 again!)

Suggested next steps

- i. Check out **AWS Elastic IP Address** if you need a static IP that does not change when you stop and restart an instance
- ii. Try adding an **auto-scaling group**!
- iii. After doing (i) and (ii), try deploying our web service using **AWS ElasticBeanstalk**!