SCIENCE EVENTS CAMEROON's Tutorial GitAws: GitHub+AWS

December 20 - Polytechnic School of Yaounde

Amazon Web Service (AWS)

AWS is a cloud platform among the most comprehensive and broadly used one. AWS offers many data center services and allows application providers to securely host their applications.

Creating an Account

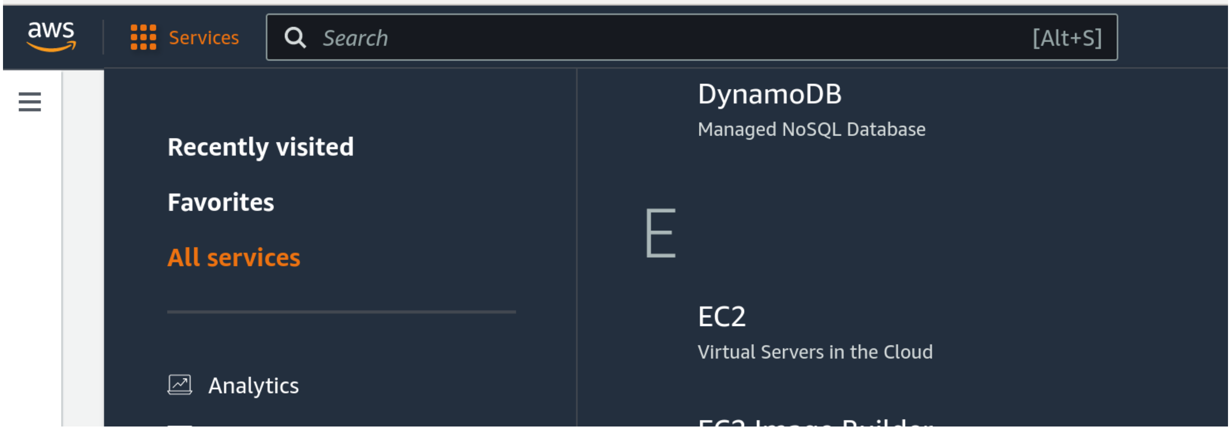
You can create a free account on the AWS website: fill in your user email and name and simply follow the instructions. Make sure to fill in an available and valid email address as a verification code will be sent to the latter.

For the purpose of this tutorial, and to allow everyone to efficiently follow the tutorial, we give you a temporary acces to Stella's account (because you need a payment method to create an account). The user name is `ScientificDays22` and the password `Tutorials22`

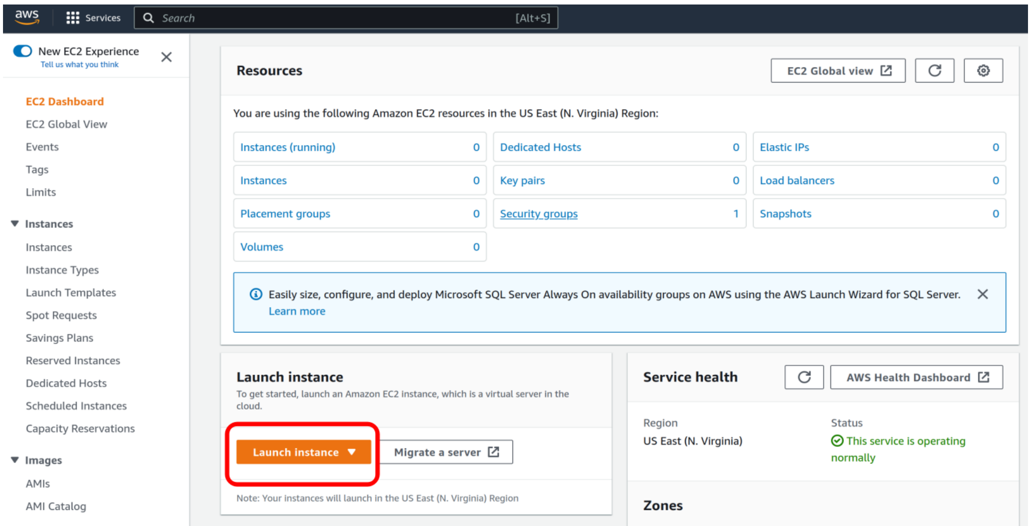
Launching an Instance

You can create a virtual machine (VM) by launching an instance

1. On the home page, click on "Services" at the top of the page, then "All services" and "EC2", like in the picture below:



2. This leads you to the EC2 dashboard where you can have an overview of all your resources. To create a new VM, click on the "Launch instance" button



Resources

You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:

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

Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. Learn more

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Note: Your instances will launch in the US East (N. Virginia) Region

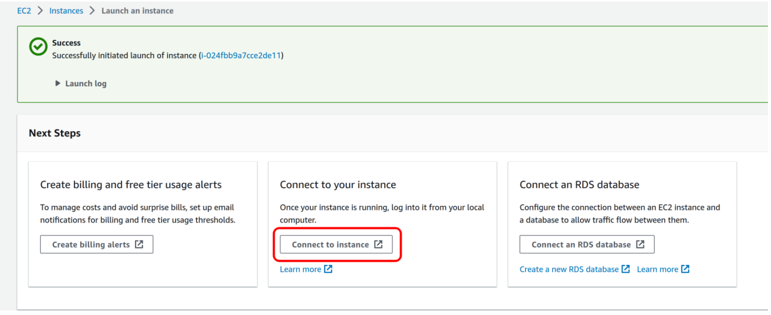
Service health

Region: US East (N. Virginia)

Status: This service is operating normally

3. Enter a name for your VM, chose the operating system (OS) -`Linux` *for this tutorial*-, the instance type -*chose a free tier instance: by default,* `t2.micro` *with 1GB RAM and 1vCPU*-, the key pair -*you must have a key pair to enable ssh connection to your instance*-. Chose the `ScientificDays22.pem` key pair created for this tutorial downloadable here. Everything else can be left at the default config.

4. On the creation success page, you can directly connect to your newly created instance:



Success
Successfully initiated launch of instance (i-024fbb9a7cce2de11)

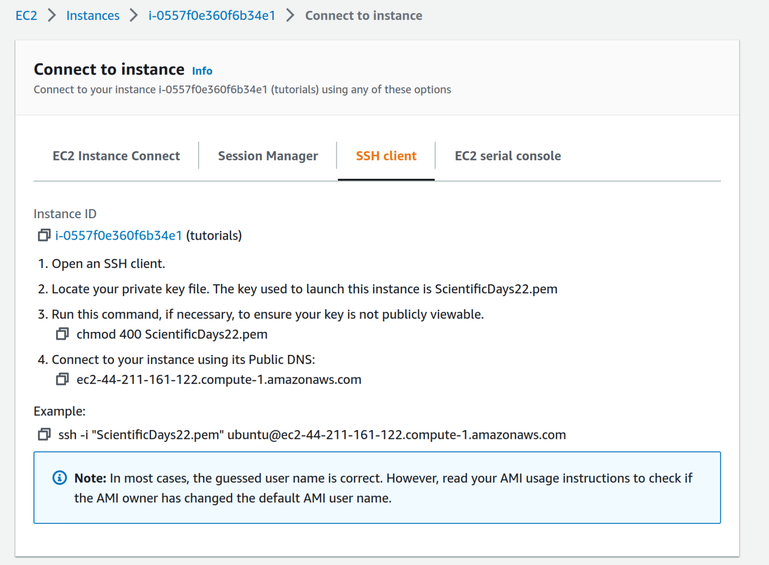
Next Steps

Create billing and free tier usage alerts
To manage costs and avoid surprise bills, set up email notifications for billing and free tier usage thresholds.

Connect to your instance
Once your instance is running, log into it from your local computer.

Connect an RDS database
Configure the connection between an EC2 instance and a database to allow traffic flow between them.

Next, follow the instructions on the "SSH client" tab to access your VM from a terminal:



Connect to instance

Connect to your instance i-0557f0e360f6b34e1 (tutorials) using any of these options

Instance ID
i-0557f0e360f6b34e1 (tutorials)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is ScientificDays22.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   ```
   chmod 400 ScientificDays22.pem
   ```
4. Connect to your instance using its Public DNS:
   ```
   ec2-44-211-161-122.compute-1.amazonaws.com
   ```

Example:

```
ssh -i "ScientificDays22.pem" ubuntu@ec2-44-211-161-122.compute-1.amazonaws.com
```

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

GitHub

GitHub is a cloud-based Git (a version control system) repository that helps you store, manage, track and control changes to your projects.

Installation

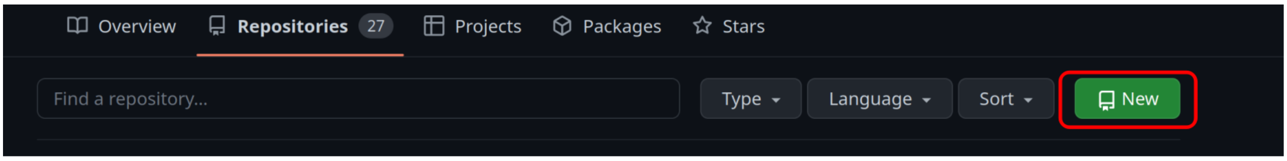
1. On Linux:

```
sudo apt update
sudo apt install git-all
```

2. On Windows:
    - Download the windows installer that we made available here
    - At the 'PATH environment' installation phase, chose *Git from the command line and also from 3rd party software* ==> this will provide a git bash terminal which understands all git commands.
3. Creating an account: creating a GitHub account is as simple as easy. Just sign up and follow the steps (enter a valide an accessible email address, a user name, and a password).

Gitaws

1. Create a new repository in your GitHub account (by clicking on new like in the following picture):



Fill in the name and the description and click on `Create repository` at the bottom of the page: you will be provided with a list of commands to link your local *code/project* to the newly created repo.

2. In your laptop, chose a project that you would like to save in the cloud, move to the corresponding directory and type:

```
echo "# xxxx" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:xxxx (the link of your repo)
git push -u origin main
```

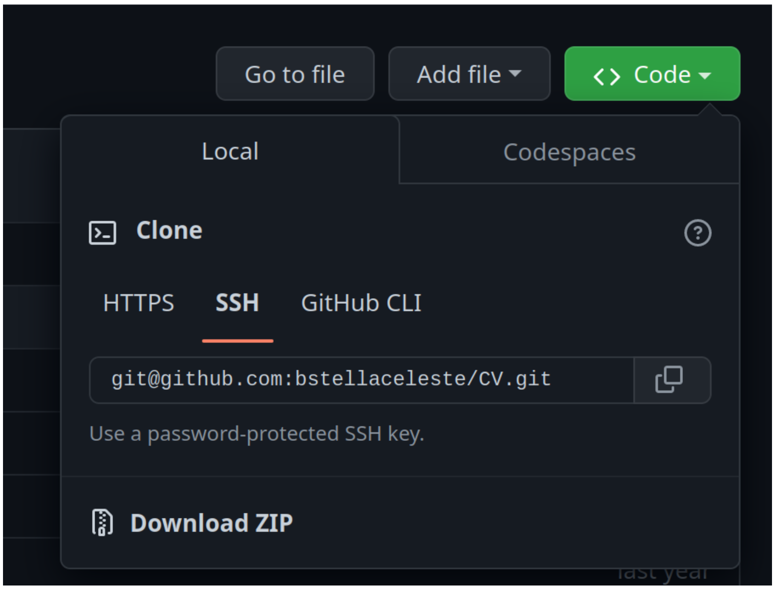
At the end, reload your git page and your project should be available.

3. Now, move back to your AWS' VM to clone your repo. Before cloning, you must establish the ssh connection with your laptop and your VM. Clik on the profile's button at the top right and go to `settings` -> `ssh and GPG keys` -> `new SSH key`. Give a name to your key and enter the public key of your device. Open a terminal (on Linux, or the Git bash terminal on Windows) and first generate an ssh key (if you don't yet have one): `ssh-keygen`.

   > You can leave all the answer's fields blank and at the end, the tool will print the path to your public key (generally at `$HOME/.ssh/id_rsa.pub`)

   Once your keys are generated, print the public one: `cat path/to/your/id_rsa.pub` and copy it to your GitHub account settings.

   You can now clone your repo both in your VM and your laptop. Copy the link as in the picture below, and in a terminal, type:

   ```
   git@github.com:bstellaceleste/CV.git
   ```

   Use a password-protected SSH key.

4. In your VM, make some changes in your project, and upload them:

   ```
   git status (to know which files have been modified and need to be
   saved)
   git add xxx (xxx is for the files you want to save; you can replace
   xxx by '.' to include everything)
   git commit -m "a message that informs about the backup"
   git push (to upload the files to save)
   ```

   If you refresh the GitHub page, you can see your changes.

5. Now, go to your laptop and browse your files: obviously you still have the previous version of your project, i.e., the changes made from your VM are not visible in your laptop. To have the same version as your VM (and the remote one), you need to `pull` the changes: `git pull`.

6. Let us simulate a failure, i.e., you *accidentaly* delete your directory in one or both machines (your VM or your laptop). If you do so, recovery is as simple as just re-cloning the project like in **step 3**.

7. Let us now imagine that you `push` something you didn't want to. Don't panic! You can revert to the previous commit like this: `git reset --soft HEAD~`. This will move your project (remote) to the version before the unfortunate commit.

   > A '**--hard**' reset will also move your local files to the previous version.