

Homework #5 Amazon Web Services (AWS) with Python

This semester we are allowing all students to explore cloud computing as offered by Amazon's Web Services. Using the instructions below one can establish a service at AWS by signing up to AWS Educate. Once established, you will be able to move your Python back-end program developed for Assignment #6 to your AWS instance and have it executed there.

1. Pre-requisites

All new **AWS Educate** student members enrolling at member institutions like USC will now receive **\$100 in AWS usage credit** pre-loaded into an **AWS Educate Starter account** at signup.

Students **no longer** need to enter an **AWS Account ID** or select an account type after registering. They may now go directly into their student portal and enjoy the benefits of AWS Educate.

AWS Educate Starter Accounts also come with a **capped amount of usage** and **do not require a credit card** to sign up, eliminating the risk of members overspending if a service is not shut off.

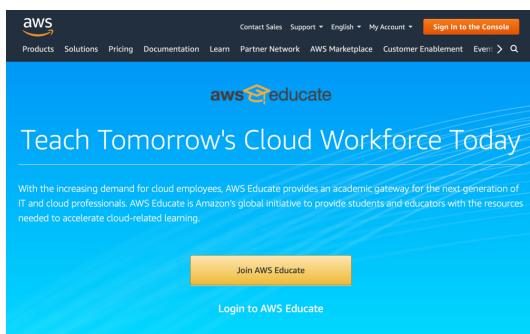
AWS Educate Starter Accounts do have limited AWS service capability. To view the list of services supported by AWS Educate Started Accounts, you can go here:

https://s3.amazonaws.com/awseducate-starter-account-services/AWS_Educate_Starter_Accounts_and_AWS_Services.pdf

2. Sign up for AWS Educate

To sign up for AWS Educate go to:

<http://aws.amazon.com/education/awseducate>



Click on the **Join AWS Educate** button.

On Step 1, click on the arrow in *Student* button.

The screenshot shows the first step of the AWS Educate sign-up process. At the top, there's a logo for 'aws educate' and a link to 'Apply to join AWS Educate'. Below that is a blue header bar with the text 'Step 1/3: Choose your role'. Underneath are four categories: 'Student' (highlighted with a red border), 'Educator', 'US Veteran', 'Institution', and 'Company/Recruiter'. Each category has a small icon and a dropdown arrow. A note at the bottom states: 'Please note that any personal information you provide will be treated in accordance with the [AWS Educate Terms and Conditions](#) and [AWS Privacy Notice](#)'. A blue banner at the bottom says: 'AWS Educate is Amazon's program to help students learn real-world cloud technology skills before graduating. It provides students and educators with the resources needed to accelerate cloud-related learning.'

On Step 2, fill out the form appropriately. **University of Southern California** will auto complete. Leave the **Promo Code** field empty, click the reCAPTCHA "I'm not a robot", and click **NEXT**.

The screenshot shows the second step of the AWS Educate sign-up process. The header is 'Step 2/3: Tell us about yourself'. It includes fields for 'Preferred Language' (set to English), 'School' (set to 'University of Southern California'), 'Major' (set to 'Mouse'), 'Email' (set to 'mickey@usc.edu'), 'Year' (set to '05'), 'Year' (set to '2020'), and 'Promo Code (optional)'. Below these are 'Frequently Asked Questions' and a note about reCAPTCHA. At the bottom is a 'NEXT' button.

On Step 3, you **must scroll** to view the complete **Terms and Conditions**.

The screenshot shows the third step of the AWS Educate sign-up process. The header is 'Terms & Conditions'. It includes a 'Preferred Language' dropdown (English). The main content area contains the '10.0 CONTRACTING ENTITY' section, which states: 'Notwithstanding anything to the contrary in these Terms, 10.1 India Customers: If you are located in India, your contracting party will be Amazon Internet Services Private Limited ("AISPL"), and this Agreement is an agreement between you and AISPL, located at Ground Floor, EROS Plaza, Eros Corporate Centre, Nehru place, New Delhi, India - 110019. If you are located in India, all references to "AWS," "we," or "us" in this Agreement shall be deemed to refer to AISPL. Additionally, if you are located in India, this Agreement shall be deemed to offer the terms and conditions provided in this section.' There is also a note about the Amazon Privacy Notice. At the bottom is a note: 'You must scroll through the entire Terms and Conditions before accepting or declining.' followed by 'I Agree' and 'I Decline' buttons, and a 'SUBMIT' button.

On Step 4, select the checkbox **I Agree**. Click **SUBMIT** and finish the sign-up process.

After your application is reviewed and approved, you will receive a **welcome e-mail** from AWS Educate Support, which includes details to **set your password** and log in to the **AWS Educate Student Portal**, as shown below:

The screenshot shows an email inbox with one message titled "AWS Educate Application Approved". The message is from "AWS Educate Support <support@awseducate.com>" and was sent "Thu, Feb 13, 2:40 PM (2 days ago)". The email content is as follows:

Dear [Recipient],
Congratulations!
Your AWS Educate application has been approved. As a member of the AWS Educate program, you will gain access to the benefits listed below.
AWS Educate Student Portal
The AWS Educate Student Portal is the hub for AWS Educate students around the world to find AWS content to help with classwork, connect to self-paced labs and training resources.
Click Here to set your password and log in to the AWS Educate Student Portal. After logging in, click "AWS Account" at the top of the page to access your AWS Educate Starter Account. Use this feature to gain access to the AWS Console and resources, and start building in the cloud.
Bookmark the AWS Educate Student Portal for easy access, or [click here](#) to sign in directly.
You can access a video walk-through of the AWS Educate Student portal [here](#).
Free AWS Essentials Training
To access our foundational AWS Cloud Practitioner Essentials online learning class for free and find other self-paced labs, you must have either an AWS account or an Amazon ID.

- If you have an AWS account, sign in and [click here](#) to receive these benefits.
- If you do not have an AWS account, [click here](#) and follow the instructions to create an Amazon ID to access these benefits.

Once you access the Training and Certification portal, click "Learning Library" and search for "AWS Cloud Practitioner Essentials" to easily locate and enroll in AWS Cloud Practitioner Essentials on-line training. You can access AWS training any time after setting up your account by clicking [here](#).
Thank you again for participating in AWS Educate and we hope you enjoy the program!
Good luck with your continued studies,
The AWS Educate Team

2.2 Issues Sign up for AWS Educate

If you are having issues signing up for AWS Educate, and your initial application is rejected, you can create a Support Case at <https://console.aws.amazon.com/support>, where you describe the problem and attached a copy of the front and back of your USC ID. Alternatively you could also contact AWS Educate Support directly at:

<http://aws.amazon.com/education/awseducate/contact-us>

3. Login to AWS Educate

Go to: <http://aws.amazon.com/education/awseducate>



Click on **Login to AWS Educate**. Enter your username and password as requested. You should be redirected to the **aws educate** home page.

The screenshot shows the AWS Educate homepage. At the top, there's a navigation bar with links for Portfolio, Career Pathways, Badges, Jobs, AWS Account (which is highlighted with a red box), and Logout. Below the navigation bar, there's a user profile section with a profile picture, Consecutive Days: 2, Pathways Completed: 0, and Badges Earned: 0. A dropdown menu for Preferred Language is set to English. On the left, there's a sidebar with text about cloud technology and opportunities, followed by a video thumbnail titled "What is AWS? Amazon". On the right, there's a "Suggested Jobs" section with two entries: "Analyst Experience - Technology" and "Software Engineering Analyst".

On the top right navigation bar, select **AWS Account**. You will be taken to a page where you can view your **applied credits** and **validity** of your Starter Account.

This screenshot shows the "AWS Educate Starter Account" page on vocareum. The page features a character holding a laptop and the title "AWS Educate Starter Account". It says, "Your cloud journey has only just begun. Use your AWS Educate Starter Account to access the AWS Console and resources, and start building in the cloud!" Below this is an orange button labeled "AWS Educate Starter Account". A note below the button states, "Your account has an estimated 99 credits remaining and access will end on Feb 12, 2021." At the bottom, there's a note about clicking the button leading to a third-party site managed by Vocareum, Inc.

Click on the orange button labelled "**AWS Educate Starter Account**", which will take you to **vocareum**. Scroll down to the end of the **Terms and Conditions** page, and click **I Agree**. You will be directed to the **vocareum Dashboard** as shown below.

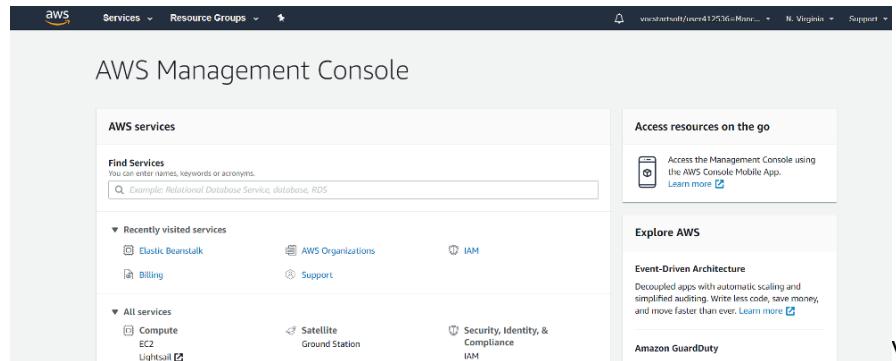
This screenshot shows the vocareum dashboard. On the left, there's a "Welcome to your AWS Educate Account" section with a FAQ and a list of supported regions. On the right, there's a "Your AWS Account Status" section with three icons: Active (yellow), \$99.86 (green), and 2:60 (blue). Below these is a "Account Details" button, which is highlighted with a yellow circle. A note at the bottom says, "Please use AWS Educate Account responsibly. Remember to shut down your instances when not in use to make the best use of your credits. And, don't forget to logout once you are done with your work!"

Clicking on the Account Details button will display a modal window with your Credentials. Click on **Show** to display your **AWS CLI** keys and token.

This screenshot shows a modal window titled "Credentials" from the vocareum dashboard. It contains sections for "AWS Access", "AWS Starter account", and "AWS CLI". The "AWS CLI" section includes a text area with the following content:

```
[default]
aws_access_key_id=
aws_secret_access_key=
aws_session_token=
```

Click the **AWS Console** button and you will be taken to your **AWS Management Console** page. Please note that you may have to **Allow pop-up windows** in your browser settings for this Website, if pop-up windows are **Blocked**.



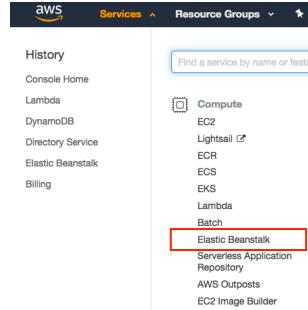
On the top right corner, you can find:

- 1) Your account name – for example vocstartoft/user623430=papa@usc.edu
- 2) AWS Deployment Region – US East (N.Virginia)

Important Note: Notice item 2) above. **AWS Educate Starter Accounts** are limited to use only the **us-east-1** region.

4. Set up the Default Elastic Beanstalk Application

- Click the top left menu named **Services**
- From the list of Amazon Web Services, select **Elastic Beanstalk**, under **Compute**.



- Select **Create New Application** in the top right, right underneath your account name, and follow the Wizard.
- In the **Application Name** field, enter a name for your application.

Create New Application

Application Name	PythonApp
Description My first cloud Python App	
Tags	
Key (127 characters maximum) Value (255 characters maximum)	
50 remaining	
<input type="button" value="Cancel"/> <input type="button" value="Create"/>	

- Click **Create**.
- In the **Environment** section click on the **Create One now** hyperlink

All Applications > PythonApp

Actions ▾

Environments

No environments currently exist for this application. [Create one now.](#)

Application versions

Saved configurations

- In the **Choose an environment tier** dialog select **Web server environment** and click on **Select** button.

Select environment tier

AWS Elastic Beanstalk has two types of environment tiers to support different types of web applications. Web servers are standard applications that listen for and then process HTTP requests, typically over port 80. Workers are specialized applications that have a background processing task that listens for messages on an Amazon SQS queue. Worker applications post those messages to your application by using HTTP.

<input checked="" type="radio"/> Web server environment Run a website, web application, or web API that serves HTTP requests. Learn more	<input type="radio"/> Worker environment Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. Learn more
---	--

- In the **Environment Information** section, select a **Domain** (use the default or check availability of your own subdomain of elasticbeanstalk.com). Click on “**Check availability**” button. Your URL should be green. Otherwise you should change the environment URL.

Create a web server environment

Launch an environment with a sample application or your own code. By creating an environment, you allow AWS Elastic Beanstalk to manage AWS resources and permissions on your behalf. [Learn more](#)

Environment information

Choose the name, subdomain, and description for your environment. These cannot be changed later.

Application name PythonApp

Environment name Pythonapp-env

Domain .us-east-1.elasticbeanstalk.com

[Check availability](#)

csc571-python.us-east-1.elasticbeanstalk.com is available.

Description

- In the **Base configuration** section, choose the **Preconfigured platform**, and the following option in the drop-down list:
 - Choose a platform: **Preconfigured - Python**

Base configuration

Platform Preconfigured platform
Platforms published and maintained by AWS Elastic Beanstalk.
Choose a platform --
Beta
Corretto 11
Corretto 8
Generic
Docker
Multi-container Docker
Preconfigured
Elastic Beanstalk Packer Builder
Go
.NET (Windows/IIS)
Java
Node.js
Ruby
PHP
Python
Tomcat
Preconfigured - Docker
GlassFish
Go
 Upload your code

- In the **Application Code** section, select **Sample application**.

Application code Sample application

Get started right away with sample code.

Existing version

Application versions that you have uploaded for PythonApp.

-- Choose a version --

Upload your code

Upload a source bundle from your computer or copy one from Amazon S3.

ZIP or WAR

[Cancel](#)

[Configure more options](#)

[Create environment](#)

- Click **Create environment**.
- After a minute or so the “Creating <environment-name>” dialog appears, with the message “This will take a few minutes...”

i Creating Newphp-env
This will take a few minutes....

1:45pm Environment health has transitioned from Pending to Ok. Initialization completed 9 seconds ago and took 4 minutes.

1:43pm Added instance [i-037c397c9487d2d3a] to your environment.

1:42pm Waiting for EC2 instances to launch. This may take a few minutes.

1:41pm Environment health has transitioned to Pending. Initialization in progress (running for 16 seconds). There are no instances.

1:41pm Created EIP: 54.83.201.175

1:40pm Created security group named: awseb-e-ethjxs5uzy-stack-AWSEBSecurityGroup-1FCZQNMU89H8M

1:40pm Using elasticbeanstalk-us-east-1-231003652681 as Amazon S3 storage bucket for environment data.

1:40pm createEnvironment is starting.

Learn More

[Get started using Elastic Beanstalk](#)
[Modify the code](#)
[Create and connect to a database](#)
[Add a custom domain](#)

Featured

[Create your own custom platform](#)
[Command Line Interface \(v3\)](#)
[Installing the AWS EB CLI](#)
[EB CLI Command Reference](#)

You will need to wait for several minutes as your **Amazon Linux + Python 3.6** instance is created and launched. You will see several messages appear as the instance is being created and deployed. a *rotating wheel* next to the “**Monitor**” button. Once creation and launch are completed, you will see the wheel turn into a green round circle with a check mark in the middle.

Overview

Health Ok Causes	Running Version Sample Application Upload and Deploy	Platform Python 3.6 running on 64bit Amazon Linux/2.9.4 Change
--	---	--

Recent Events [Show All](#)

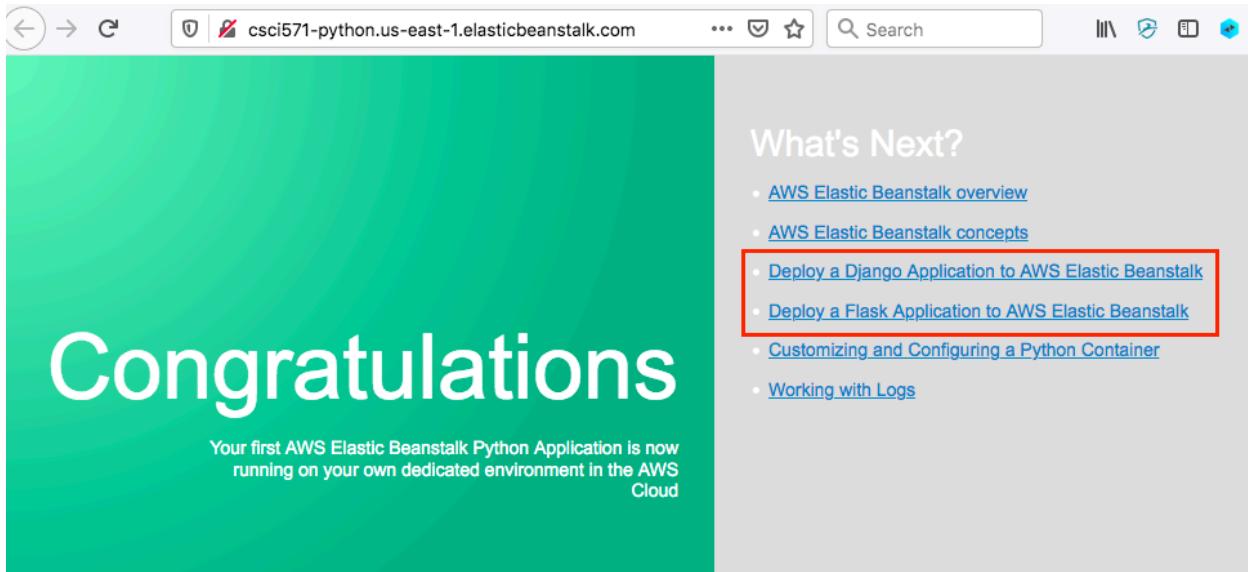
Time	Type	Details
2020-01-06 14:37:11 UTC-0800	INFO	Environment health has transitioned from Pending to Ok. Initialization completed 11 seconds ago and took 2 minutes.
2020-01-06 14:36:46 UTC-0800	INFO	Successfully launched environment: Pythonapp-env
2020-01-06 14:36:44 UTC-0800	INFO	Application available at csci571-python.us-east-1.elasticbeanstalk.com.
2020-01-06 14:36:11 UTC-0800	INFO	Added instance [i-0990b8c74ccda915e] to your environment.
2020-01-06 14:36:06 UTC-0800	INFO	Waiting for EC2 instances to launch. This may take a few minutes.

Python Instance Dashboard

Beside “<YourEnvironment>” subtitle there is a **URL** such as *YourAppName-env.elasticbeanstalk.com*.

The screenshot shows the AWS Elastic Beanstalk console. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, 'Resource Groups' dropdown, and user information ('Marco Papa', 'N. Virginia', 'Support'). Below the navigation is a tab bar with 'Elastic Beanstalk' (selected), 'NodeJS-APP', and 'PythonApp'. On the right of the tab bar is a 'Create New Application' button. The main content area shows 'All Applications > PythonApp > Pythonapp-env (Environment ID: e-5xm2drmbjk, URL: csci571-python.us-east-1.elasticbeanstalk.com)'. A red box highlights the URL 'csci571-python.us-east-1.elasticbeanstalk.com'. To the right of the URL is an 'Actions' dropdown.

Click on it. You should see the "*Congratulations*" page. If you see it as shown below, your application and environment have been created properly.



Python Sample Application

You have two options listed for deploying web apps in Python on AWS:

- **Flask** web application framework
- **Django** web application framework

We personally recommend that you use Flask, as we believe it is simpler to install and maintain. You are free to use either Python Web Framework, but we will support in Piazza only the Flask deployment.

5. Deploy your Python application

5.1 Installing Python

On **MacOS** we recommend you use the “brew” package manager to install Python and pip. Flask requires Python 2.7 (which is preloaded on every Mac) or Python 3.4 or newer. We personally recommend **Python 3.7** and **pip3**. In the latter case you should change your shell startup files to point to Python 3.7 instead of Python 2.7.

Note: steps for Installing **Python 3.7** can be found in section 2, “*Setting up a Python development environment*”, in the file entitled “*Homework #5 Google Cloud Platform (GCP) with Python*”, available at:

https://csci571.com/hw/hw5/HW5_Google_Python.pdf

On **Windows 10**, you can [install the Windows Subsystem for Linux](#) to get a Windows-integrated version of Ubuntu and Bash.

You can deploy your applications using the AWS Elastic Beanstalk console **Upload and Deploy** or the Elastic Beanstalk Command Line Interface (**EB CLI**).

5.2 Deploying a Flask Application to AWS Elastic Beanstalk using “Upload and Deploy”

This is the installation [that we recommend](#), as it uses the Sample Application environment set up in **section 5. Set up the Default Elastic Beanstalk Application**.

Windows ONLY: download and install **PowerShell**.

- a. Create a project folder:

```
$ mkdir eb-flask  
$ cd eb-flask
```

- b. Create an isolated Python environment:

```
$ python3 -m venv env  
$ source env/bin/activate
```

(the terminal prompt will add (env) to the terminal prompt)

- c. Install flask with pip install:

```
(env) $ pip install flask==1.0.2
```

- d. View installed libraries with pip freeze:

```
(env) $ pip freeze  
Click==7.0  
Flask==1.0.2  
itsdangerous==1.1.0  
Jinja2==2.10.3  
MarkupSafe==1.1.1  
Werkzeug==0.16.0
```

- e. Create requirement.txt

```
(env) $ pip freeze > requirements.txt
```

- f. Next, create an application that you'll deploy using Elastic Beanstalk Upload and Deploy. We'll create a "Hello World" RESTful web service.

- g. Create a new text file in this directory named application.py with the following contents:

- h. Download new sample code (RESTful app) from:

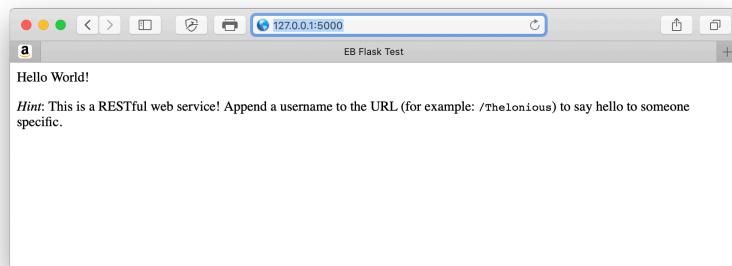
<https://csci571.com/hw/hw5/application.py>

save the file as application.py. Using application.py as the filename and providing a callable application object (the Flask object, in this case) allows Elastic Beanstalk to easily find your application's code.

- i. Run application.py locally with Python on port 5000:

```
(env) $ python application.py
 * Serving Flask app "application" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production
environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 305-600-227
```

- j. Test your application locally, by opening `http://127.0.0.1:5000/` in your web browser. You should see the application running, showing the index page:

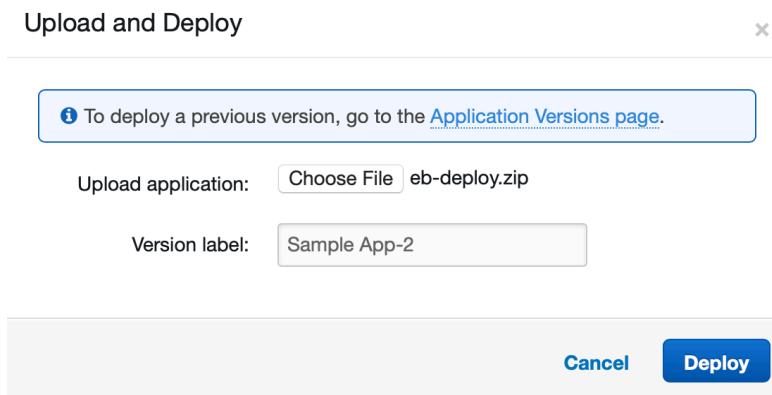


You can stop the web server and return to your virtual environment by typing **Ctrl+C**.

- k. You are ready now to upload and deploy. First of all, “zip” the two needed files, application.py and requirements.txt:

```
(env $ zip eb-deploy.zip application.py requirements.txt
  adding: application.py (deflated 48%)
  adding: requirements.txt (deflated 9%)
(env) $
```

- l. Now go to the AWS EB console, and click the **Upload and Deploy** button:



- m. Choose the eb-deploy.zip file from your desktop. Enter a unique Version label. Click **Deploy**. The AWS EB server will restart and update your environment.

Recent Events		
Time	Type	Details
2020-01-11 17:15:45 UTC-0800	INFO	Environment update completed successfully.
2020-01-11 17:15:45 UTC-0800	INFO	New application version was deployed to running EC2 instances.
2020-01-11 17:14:57 UTC-0800	INFO	Environment health has transitioned from Ok to Info. Application update in progress (running for 20 seconds).
2020-01-11 17:14:55 UTC-0800	INFO	Deploying new version to instance(s).
2020-01-11 17:14:15 UTC-0800	INFO	Environment update is starting.

- n. You are ready now to run the updated AWS “cloud” version of your app.



- o. Modify application.py for the next exercises, as appropriate.

5.3 Deploying a Flask Application to AWS Elastic Beanstalk using “EB CLI”

(Optional)

Click on the corresponding link in the sample application, or follow the tutorial at:

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create-deploy-python-flask.html>

The Tutorial above includes all of the following:

- Prerequisites
- Flask Framework installation
- Details on installing and configuring the EB CLI
- Set Up a Python Virtual Environment with Flask
- Create a Flask Application
- Run the application locally on your Mac or PC
- Deploy your site with the EB CLI
- Cleanup

Once you have created, deployed and tested the tutorial application, you will have the basic skeleton for a **RESTful web service**.

Additional Notes:

The instructions in this Tutorial creates a new Elastic Beanstalk environment and deploys using the EB CLI.

The Tutorial also uses the **us-east-2** region in step 1 of the section titled “To create an environment and deploy your flask application”. Since **AWS Educate Starter Accounts** are limited to use only the **us-east-1** region, that step must be changed to use the us-east-1 region as in:

```
$ eb init -p python-3.6 flask-tutorial --region us-east-1
```

Also, you will likely get an error such as “zlib not available” during the installation using EB CLI. As mentioned in:

<https://github.com/aws/aws-elastic-beanstalk-cli-setup/issues/23>

this can be fixed by running:

```
pip install virtualenv  
python ./scripts/ebcli_installer.py
```

instead of:

```
brew install awsebcli
```

or installing the EB CLI using Setup Scripts (as in the Tutorial).

5.4 Deploying a Django Application to AWS Elastic Beanstalk (**Optional**)

Click on the corresponding link in the sample application, or follow the tutorial available at:

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create-deploy-python-django.html>

Follow the steps listed in the tutorial.

6. Set up Exploring Your Instance (**Optional**)

If you want to explore your Instance and create your own domain-based URL with SSH control, you can add the following steps.

6.1 Get and Setup SSH

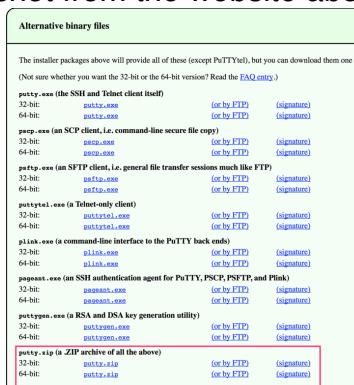
Once the Python app with SSH-enabled environment is running, you can get access using SSH. You can use SSH on a Mac running MacOS, or Putty when running on Windows.

On a Mac, SSH is built into MacOS and can be accessed through the **Terminal** app and there is no additional setup needed.

On a Windows PC, you will need to download the complete PuTTY distribution at:

<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

You should download the file **putty.zip** that contains all the binaries, including **PuTTYgen** as see in this snapshot from the website above:



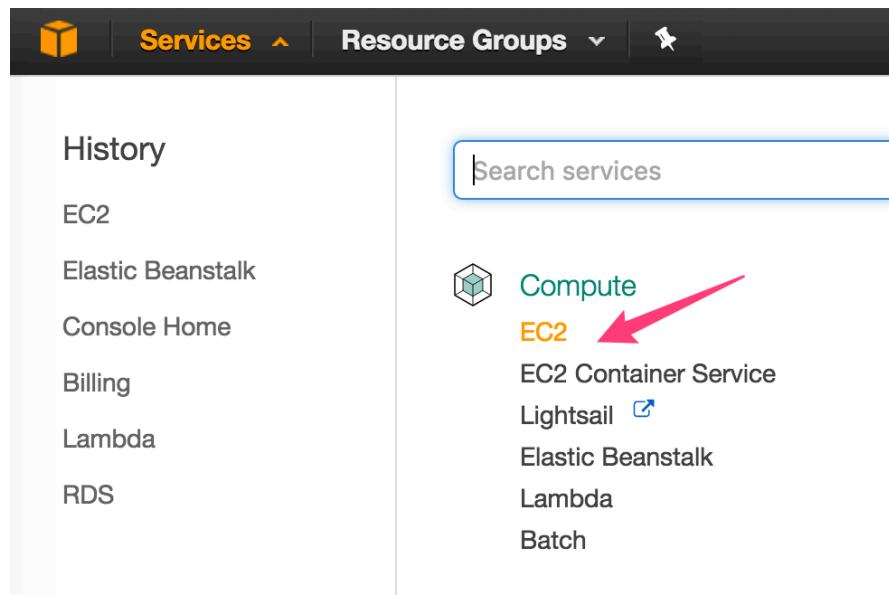
PuTTY needs additional setup as it needs to use a converted version of the private key. The instructions on how to perform such conversion are available here:

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html>

The major step is to use **PuTTYgen** to convert your private key format (.pem) generated by Amazon EC2 into the required PuTTY format (.ppk).

6.2 Create a Key Pair

- From the **Services** drop down, under the **Compute** section, select the **EC2**.



- Under **NETWORK AND SECURITY** select **Key Pairs**.
- Click on the button **Create Key Pair**.
- Enter a name like **pythonhosts** (you must have your own random name!) and click on **Create**.
- A download of your private key should start automatically. Save the key, like **pythonhosts.pem**, in an appropriate location.

6.2.1 Associate your Instance to the Key Pair

- You now need to associate your Instance with the just created key pair.
- Select the **Elastic Beanstalk** under **Services**.
- Select your environment but clicking anywhere in the “green” rectangle.
- Click on **Configuration** on the left menu.
- Click on the **Modify** button next to **Security**.

- Select the key pair you just created for the **EC2 key pair** field. Click **Refresh icon**.

The screenshot shows the 'Service role' configuration page. It includes fields for 'Service role' (set to 'arn:aws:iam::034721222574:role/aws-el'), 'Virtual machine permissions' (with 'EC2 key pair' set to 'pythonhosts' and 'IAM instance profile' set to 'aws-elasticbeanstalk-ec2-role'), and a bottom row with 'Cancel', 'Continue', and a large red-bordered 'Apply' button.

- Hit **Apply** and then **Confirm** and wait for several minutes for the configuration changes to take place. You may get INFO, WARN and sometimes **SEVERE** messages during this time. Wait until the update of the environment has completed, and **Health** is back to **Ok**.
- Go back to your EC2 instance (listed under **INSTANCES – Instances**) after some time and check under **Key Name**, you should now see your associated key pair.

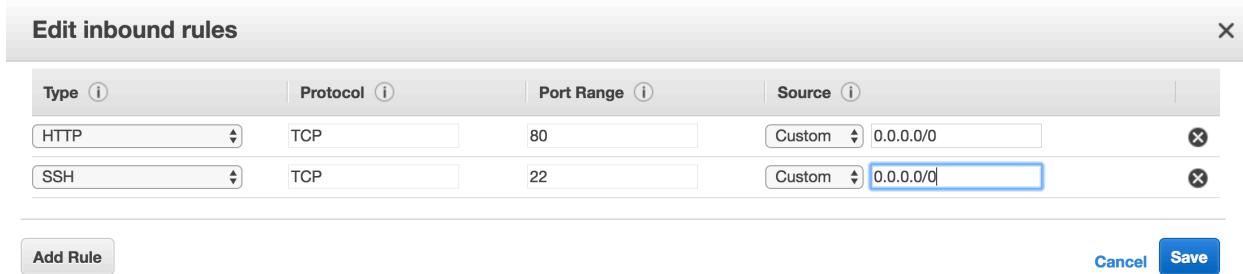
The screenshot shows the EC2 Instances page with three entries. The columns include Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, Public DNS (IPv4), IPv4 Public IP, IPv6 IPs, and Key Name. The 'Key Name' column for the first instance is highlighted with a red border and contains the value 'pythonhosts'.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 IPs	Key Name
Pythonapp-env	i-097bfa3c2e0bd4d17	t2.micro	us-east-1c	running	2/2 checks ...	None	ec2-3-233-231-125.co...	3.233.231.125	-	pythonhosts
Pythonapp-env	i-0990b6c74ccda915e	t2.micro	us-east-1c	terminated	None	None	ec2-3-233-231-125.co...	-	-	
NodejsApp-env	i-0ec03d138015b4afc	t2.micro	us-east-1c	running	2/2 checks ...	None	ec2-3-232-16-111.comp...	3.232.16.111	-	

6.3 Open port 22

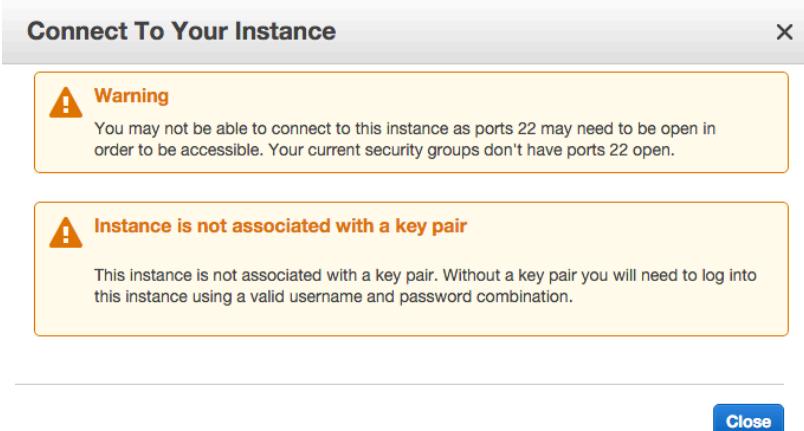
To open port 22, which is needed by SSH, follow these steps:

- In the EC2 Management Console, click on **Instances**.
- Under **NETWORK & SECURITY**, click on **Security Groups**.
- Select the security group (present as a link) configured for your instance.
- For the security group, edit (or verify) the "Inbound rules" (**Inbound** tab present on the bottom of the pane) by clicking the **Edit** button.
- If missing, add a new rule for Type = SSH, Protocol = TCP, Port Range = 22, Source = Custom 0.0.0.0/0. Click **Save**. If rule is already present, **do nothing**.



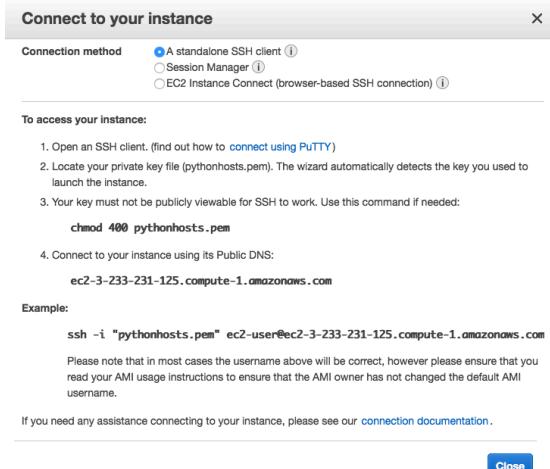
6.3.1 Errors when Connecting

If you fail to either open port 22 or associate your instance to a key pair, you will get an error popup when you try to **Connect to Your Instance** using EC2 Dashboard >> INSTANCES >> Instances >> select instance >> Connect, as show in the picture below.



6.4 Access your Linux Instance with SSH

- To see how to launch your SSH client go to **Services** and select **EC2**.
- Under the **INSTANCES** section in the navigation pane on the left, select **Instances**.
- Select your instance in the table (the check box turns **blue**) and select the **Connect** button next to Launch Instance.
- The **Connect to your instance** popup will display. Select the radio button **A standalone SSH client**. Notice the hyperlink “connect using PuTTY” (see section 7.4.2). See the snapshot below, showing Elastic IP connection string.



6.4.1 Mac running MacOS / ssh

Change the permission of pythonphosts.pem first:

```
chmod 400 pythonhosts.pem
```

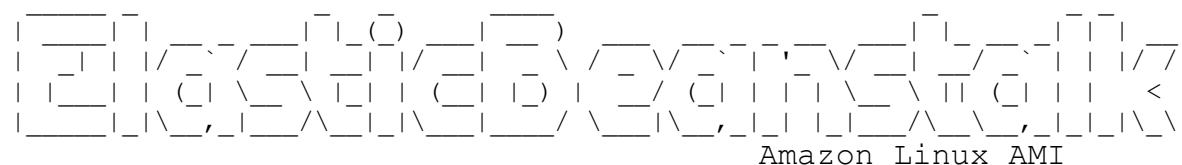
On a Mac you will need to enter a command like this one (when using **Public DNS**):

```
ssh -i "pythonhosts.pem" ec2-user@ec2-3-233-231-125.compute-1.amazonaws.com
```

type **yes**, when asked. Make sure that you are executing the ssh command in the same folder that contains the key. You should see output like this one (using **Public DNS**):

```
$ ssh -i "phphosts.pem" ec2-user@ec2-204-236-235-251.compute-1.amazonaws.com
```

Last login: Tue Oct 27 16:22:06 2015 from 159.83.115.214



This EC2 instance is managed by AWS Elastic Beanstalk. Changes made via SSH

WILL BE LOST if the instance is replaced by auto-scaling. For more information

on customizing your Elastic Beanstalk environment, see our documentation here:

<http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/customize-containers-ec2.html>

[ec2-user@ip-10-30-13-153 ~]\$

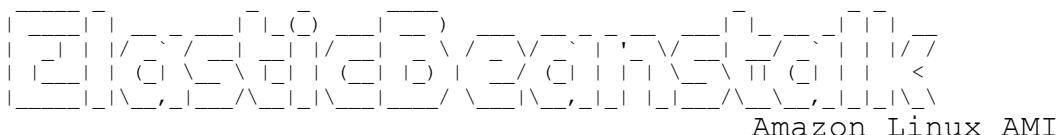
```
Mac-mini:Desktop marcopapa$ chmod 400 pythonhosts.pem
```

```
Mac-mini:Desktop marcopapa$ ssh -i "pythonhosts.pem" ec2-user@ec2-3-233-231-125.compute-1.amazonaws.com
```

The authenticity of host 'ec2-3-233-231-125.compute-1.amazonaws.com (3.233.231.125)' can't be established.
ECDSA key fingerprint is

SHA256:u0+G0tC6c9OJOixq3uGWW7S9u8wb74kC8MW15qZ3yHw.

```
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-3-233-231-125.compute-1.amazonaws.com,3.233.231.125' (ECDSA) to the list of known hosts.
```



This EC2 instance is managed by AWS Elastic Beanstalk. Changes made via SSH

WILL BE LOST if the instance is replaced by auto-scaling. For more information

on customizing your Elastic Beanstalk environment, see our documentation [here](#):

<http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/customize-containers-ec2.html>

```
[ec2-user@ip-172-31-19-89 ~]$
```

You can find more info here:

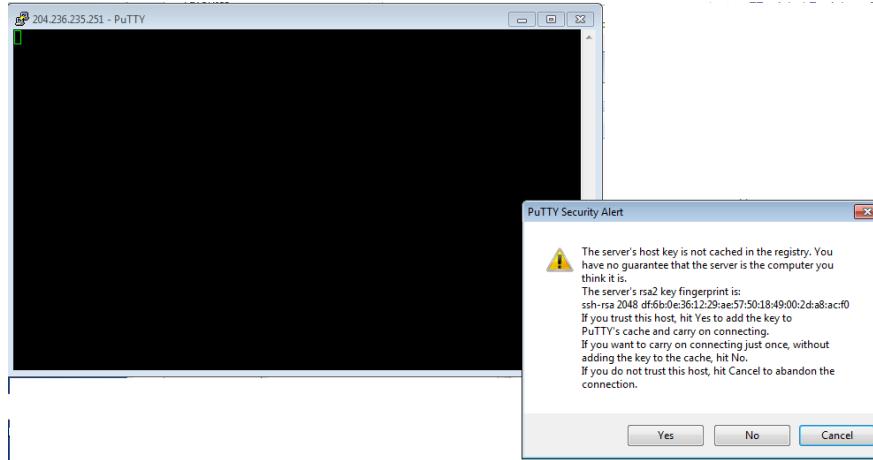
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html?console_help=true

6.4.2 PC running Windows / PuTTY

In the popup windows titled **Connect To Your Instance**, click on **Connect using PuTTY**. You will be redirected to the URI

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html>

Follow the steps under **Starting a PuTTY Session** to connect to your the Linux instance using PuTTY. The first time you connect by clicking **Open** to start the session, PuTTY displays a **PuTTY Security Alert** dialog box, as show in the following snapshot. Click the **Yes** button.



Once connected, PuTTY will open, and log you in, as shown in the next snapshot.

```
ec2-user@ip-10-30-13-153:~$ Using username "ec2-user".
Authenticating with public key "imported-openssh-key"
Last login: Mon Oct 26 22:04:08 2015 from 159.83.115.1
[ec2-user@ip-10-30-13-153 ~]$
```

As with SSH, you can either use your Public DNS or your Elastic IP to log in.

6.5 Explore

You can now explore your Instance. When you log in with SSH, your account home directory will be located at:

/home/ec2-user

That folder is empty and is not where your **Python** files are. Run 'ps ax', and you should see several instances of Apache **httpd** and **Python 2.7**:

```
[ec2-user@ip-172-31-19-89 ~]$ ps ax
```

```

PID TTY      STAT   TIME COMMAND
...
3005 ?        Ss      0:00 /usr/bin/python2.7
/usr/local/bin/supervisord --nodaemon -c /opt/python
...
1940 ?        S      0:00 /usr/sbin/httpd -D FOREGROUND
1941 ?        S1      0:00 /usr/sbin/httpd -D FOREGROUND
1942 ?        S1      0:00 /usr/sbin/httpd -D FOREGROUND
1944 ?        S1      0:00 /usr/sbin/httpd -D FOREGROUND
1945 ?        S1      0:00 /usr/sbin/httpd -D FOREGROUND
...
3325 ?        Ssl     2:07 /usr/bin/python2.7 /opt/aws/bin/cfn-
hup
...
[ec2-user@ip-172-31-19-89 ~]$

```

To see your mounted volumes, run 'df -h':

```

[ec2-user@ip-172-31-19-89 ~]$ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        483M   60K  483M   1% /dev
tmpfs          493M     0  493M   0% /dev/shm
/dev/xvda1      7.9G  1.6G  6.2G  21% /
[ec2-user@ip-172-31-19-89 ~]$

```

To see the Python (2.7 and 3.6) folders:

```

[ec2-user@ip-172-31-19-89 ~]$ ls /usr/bin/pyth*
/usr/bin/python           /usr/bin/python36
/usr/bin/python3.6m-x86_64-config
/usr/bin/python27          /usr/bin/python3.6
/usr/bin/python3-config
/usr/bin/python2.7          /usr/bin/python3.6-config
/usr/bin/python-config
/usr/bin/python2.7-config   /usr/bin/python3.6m
/usr/bin/python-config2
/usr/bin/python3            /usr/bin/python3.6m-config
[ec2-user@ip-172-31-19-89 ~]$

```

To see your Python files, located in the “bundle” folder:

```

[ec2-user@ip-172-31-19-89 ~]$ ls /opt/python
bin  bundle  current  etc  log  run
[ec2-user@ip-172-31-19-89 ~]$ ls /opt/python/bin
httpdlaunch
[ec2-user@ip-172-31-19-89 ~]$ ls -l /opt/python/bundle
total 4

```

```
drwxr-xr-x 3 root root 4096 Jan  6 23:53 2
[ec2-user@ip-172-31-19-89 ~]$ ls -l /opt/python/bundle/2
total 8
drwxr-xr-x 2 wsgi root 4096 Jan  6 23:53 app
-rw-r--r-- 1 root root 103 Jan  6 23:53 env
[ec2-user@ip-172-31-19-89 ~]$ ls -l /opt/python/bundle/2/app
total 12
-rw-r--r-- 1 wsgi root 5065 Apr  2 2015 application.py
-rw-r--r-- 1 wsgi root  84 Apr  2 2015 cron.yaml
[ec2-user@ip-172-31-19-89 ~]$ ls -l
/opt/python/bundle/2/app/application.py
-rw-r--r-- 1 wsgi root 5065 Apr  2 2015
/opt/python/bundle/2/app/application.py
```

To see the Python application file that creates the “sample application” HTML page:

```
[ec2-user@ip-172-31-19-89 ~]$ more
/opt/python/bundle/2/app/application.py
```

Note: the “bundle number” (2 above) will increase as you update and deploy new versions.

6.6 Additional Resources

An additional tutorial entitled “*Deploy a Python Web App*” on AWS, is available at:

<https://aws.amazon.com/getting-started/projects/deploy-python-application/>

This is a more complex application, that will incur some charges. We recommend deleting the environment once the app is tested.

Have fun exploring AWS!!