

しばらくお待ちください

本講義は、AIリテラシー・制作実習の授業です
開始(9:20～)となっております

- ・ 出席フォームから登録していますか？
- ・ 設定からスピーカーテストでしっかり音声聞こえるか？

本講義では、カメラ・マイク(ヘッドセット)があることが
望ましいです

AWS Academy へようこそ！

IT分野 AIテクノロジーコース
岡田 直己

サーバー構成



(1)招待を送っています

stメールに招待を送っています

You've been invited to participate in the course, AWS Academy Learner Lab - Foundation Services [16367]. Course role: Student

Name: 岡田 直己

Email: naokiokada@kobedenshi.ac.jp

Get Started



(2)ID,パスワード設定

ID : st.kobedenshi.ac.jpのメール

PASS:招待時設定する

※既に入っている人は不要

- ・ アクセスには以下のログインから利用可能

<https://awsacademy.instructure.com/login/canvas>

(3)ダッシュボード

こちらに各講座が並びます→classroomみたいなもの
こちらに入る

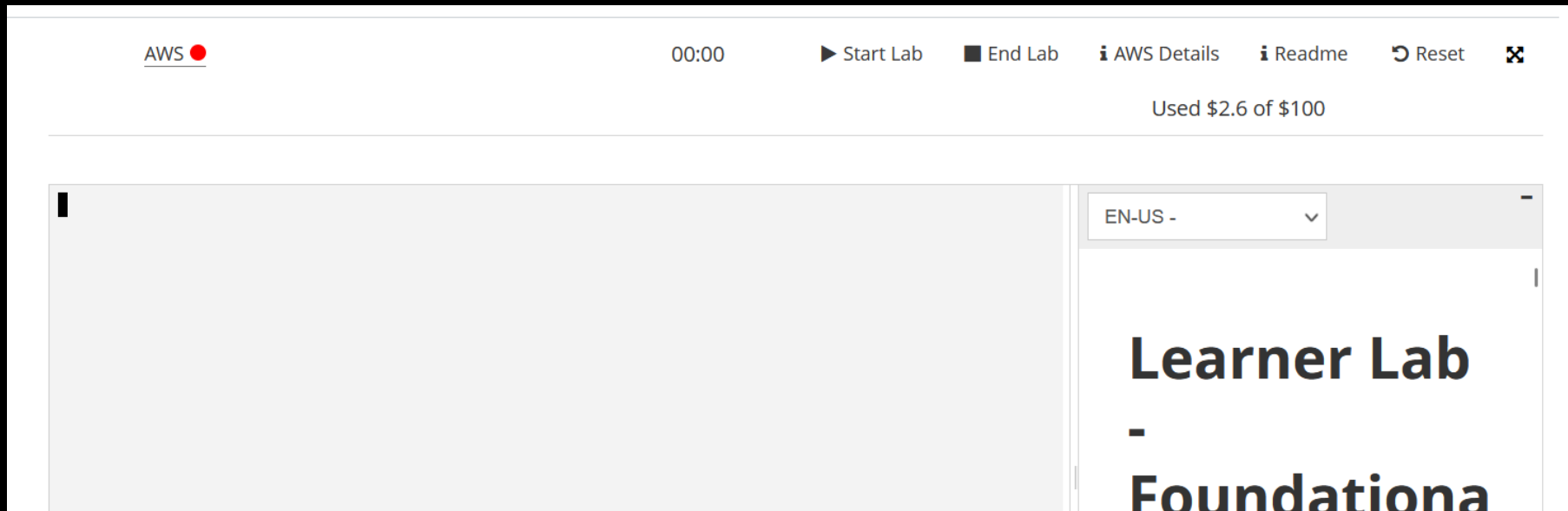


(4)Learner Labの起動

モジュール へ移動し

Learner Lab-Foundational Services

を起動する→アンケートみたいなものがあれば適当に答える



(5)Leaner Lab説明

Start Labで開始,EndLabで終了

赤：セッション切れ・・・使えない

黄：準備中、緑：利用可能

Reset：中身が全部消える(使わない)

Used 利用したクレジット

The screenshot displays the AWS Leaner Lab interface. At the top, there is a navigation bar with the following elements: 'AWS' with a green status indicator, a timer showing '04:00', and buttons for 'Start Lab' (play icon), 'End Lab' (stop icon), 'AWS Details' (info icon), 'Readme' (document icon), 'Reset' (refresh icon), and a close button (X icon). Below the navigation bar, it shows 'Used \$2.6 of \$100'. The main area features a terminal window with the prompt 'ddd_v1_w_75R_1493272@runweb65142:~\$' and a details panel on the right. The details panel contains the following information:

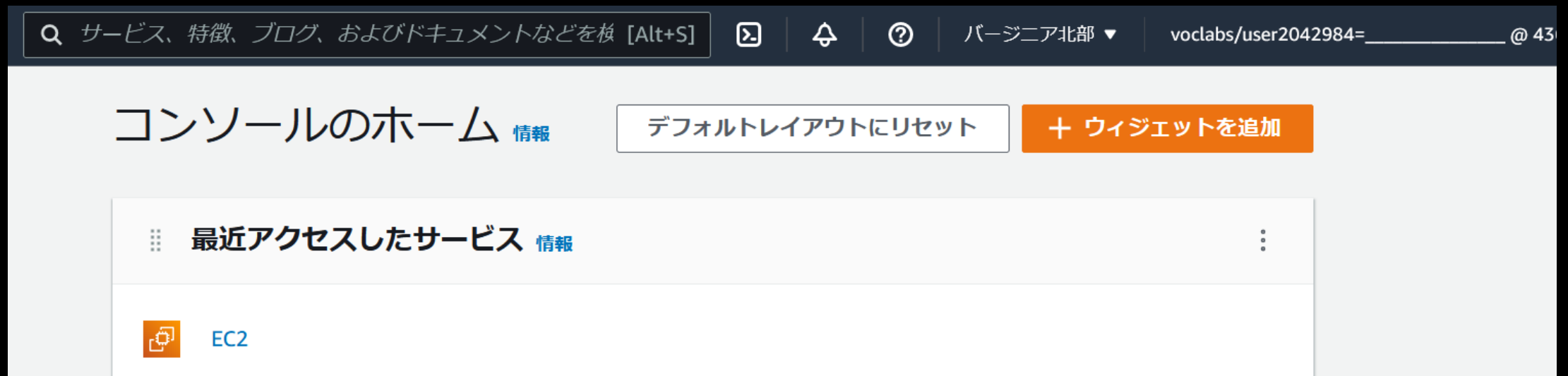
- Region: us-east-1
- Lab ID: arn:aws:cloudformation:us-east-1:436298545860:stack/c51097a7/abee9760-4c23-11ed-8f66-0ec41172962f
- Creation Time:

(6) マネジメントコンソールへ

AWS  をクリックしてManagement Consoleへ移動する

※一度、個人的にAWS使っていたらログアウトしてから再度移動

リージョンは、バージニア北部しか利用できない



EC2インスタンス(1)

検索からec2と入力して、EC2画面へ移動する
インスタンスを開いて、インスタンス作成画面へ移動

The screenshot shows the AWS Management Console interface. The top navigation bar includes the AWS logo, a 'サービス' (Services) menu, a search bar, and the current region 'バージニア北部' (us-east-1). The left sidebar contains a 'New EC2 Experience' notification and a navigation menu with categories like 'EC2 ダッシュボード' (EC2 Dashboard) and 'インスタンス' (Instances). The main content area is titled 'リソース' (Resources) and shows a summary of EC2 resources in the '米国東部 (バージニア北部)' (us-east-1) region. A table lists the following resources:

Resource	Count
インスタンス (実行中)	0
Elastic IP	0
インスタンス	0
キーペア	1
スナップショット	0
セキュリティグループ	1
プレースメントグループ	0
ボリューム	0
ロードバランサー	0
専有ホスト	0

At the bottom of the console, there is a notification banner for the 'AWS Launch Wizard for SQL Server', which states that it can simplify the deployment of Microsoft SQL Server on Amazon EC2.

EC2インスタンス(2)

インスタンスを起動でインスタンスを作成する

インスタンス 情報

🔄

接続

インスタンスの状態 ▼

アクション ▼

インスタンスを起動 ▼

🔍 Find インスタンス by attribute or tag (case-sensitive)

< 1 > ⚙️

Name ▼	インスタンス ID	インスタンス... ▼	インスタンス... ▼	ステータスチェ...	アラームの状態	アペ
インスタンスなし このリージョンにはインスタンスがありません。 <div>インスタンスを起動</div>						

インスタンスを選択

=

⚙️ ×

EC2インスタンス起動(1)

名前とタグ：名前：MyWebServer1

クイックスタート：AmazonLinux

Amazonマシンイメージ：Amazon Linux2 AMI
(Kernel 5.10・・・)

アーキテクチャ：64ビット(x86)

インスタンスタイプ：t2.micro

キーペア：新しいキーペアの作成

キーペア名：mykey1、キーペアのタイプ：RSA

プライベートキーファイル形式：.pem →キーペア作成ボタン

※ダウンロードファイルは絶対に失くさずに

EC2インスタンス起動(2)

ネットワーク設定

ファイヤーウォール(セキュリティグループ)

セキュリティグループを作成する

☒ からのSSHトラフィックを許可する

任意の場所

☒ インターネットからのHTTPSトラフィックを許可する

☒ インターネットからのHTTPトラフィックを許可する

EC2インスタンス起動(3)

ストレージを設定：30GiB、gp2を選択

上記以外は、変更しない

インスタンスを起動ボタンを押す

成功が出たら、EC2
インスタンス画面に戻る

状態が**実行中**に変わるまで待つ



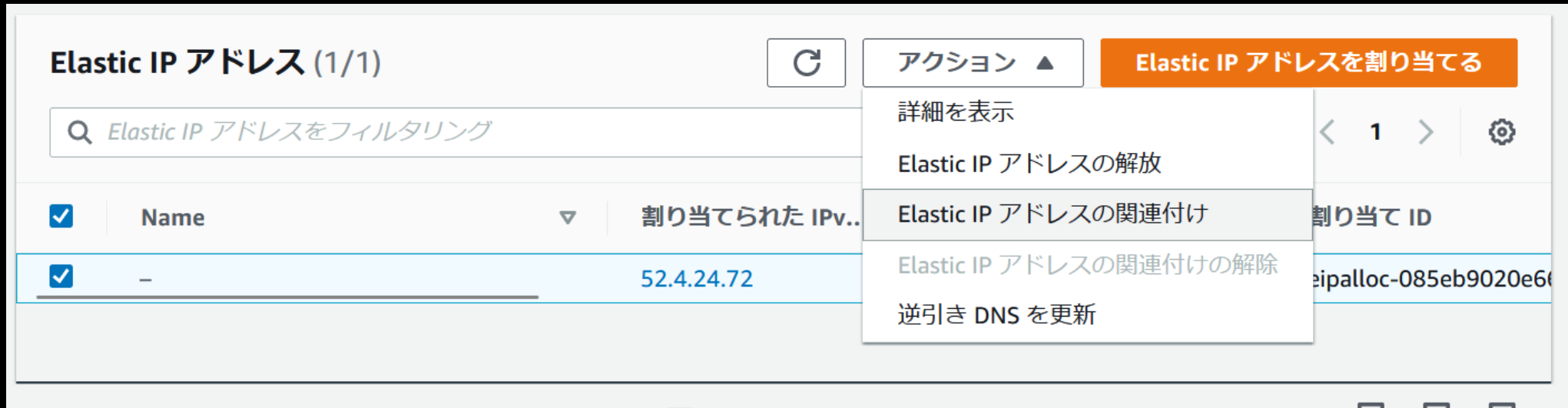
Elastic IP(1)

Elastic IPからElastic IPアドレスを割り当てる
特に中身は変更せず、割り当てボタンを押す

The screenshot displays the AWS Management Console interface for Elastic IP addresses. On the left, a navigation menu is visible with the following items: **▼ ネットワーク & セキュリティ**, **セキュリティグループ**, **Elastic IP**, **プレースメントグループ**, **キーペア**, and **ネットワークインターフェイス**. The main content area shows a top bar with a refresh button, an **アクション ▼** dropdown, and a prominent orange button labeled **Elastic IP アドレスを割り当てる**. Below this is a table with a header row containing **▼**, **タイプ**, **▼**, and **割り当て ID ▼**. The table body is currently empty. At the bottom right, there are two buttons: **キャンセル** and **割り当て**.

Elastic IP(2)

追加されたIPを☑選択して、アクションElastic IP
アドレスの関連付けを行う



The screenshot shows the AWS Elastic IP console interface. At the top, there's a header "Elastic IP アドレス (1/1)" with a refresh button. Below it is a search bar with the placeholder text "Elastic IP アドレスをフィルタリング". A table lists the Elastic IP addresses. The first row is selected, showing a checked checkbox, a name "-", and an assigned IP address "52.4.24.72". To the right of the table, an "アクション" (Actions) dropdown menu is open, displaying several options: "詳細を表示", "Elastic IP アドレスの解放", "Elastic IP アドレスの関連付け" (which is highlighted), "Elastic IP アドレスの関連付けの解除", and "逆引き DNS を更新". An orange button "Elastic IP アドレスを割り当てる" is also visible.

<input checked="" type="checkbox"/>	Name	割り当てられた IPv...
<input checked="" type="checkbox"/>	-	52.4.24.72

アクション ▲

- 詳細を表示
- Elastic IP アドレスの解放
- Elastic IP アドレスの関連付け
- Elastic IP アドレスの関連付けの解除
- 逆引き DNS を更新

Elastic IP アドレスを割り当てる

< 1 > ⚙

割り当て ID

ipalloc-085eb9020e6


Elastic IP(3)

Elastic IP アドレス: 52.4.24.72

リソースタイプ

Elastic IP アドレスを関連付けるリソースのタイプを選択します。

- ☒ インスタンス
☐ ネットワークインターフェイス

 すでに Elastic IP アドレスが関連付けられているインスタンスに Elastic IP アドレスを関連付けると、前に関連付けられていた Elastic IP アドレスの関連付けが解除されますが、アカウントへの割り当ては維持されます。
[詳細はこちら](#)

インスタンス

Q | インスタンスタイプを選択します



i-0d29897285b68fb99 (MyWebServer1) - running

プライベート IP アドレス

Elastic IP アドレスを関連付けるプライベート IP アドレスです。

i-0d29897285b68fb99 (MyWebServer1) - running

Q プライベート IP アドレスを選択します

起動しているインスタンス(MyWebServer1)を選択して
関連付けるボタンを押す

セキュリティグループ(1)

セキュリティグループからdefault以外のセキュリティグループのsg-xxxxxxxxxxのリンクをクリックする

AMI カタログ

▼ Elastic Block Store

ボリューム

スナップショット

ライフサイクルマネージャー

▼ ネットワーク & セキュリティ

セキュリティグループ

Elastic IP

セキュリティグループ (1/2) 情報

アクション ▼ セキュリティグループを CSV にエクスポート ▼ セキュリティグループを作成


Q セキュリティグループをフィルタリング

	Name ▼	セキュリティグルー... ▼	セキュリティグルー... ▼	VPC ID
<input type="checkbox"/>	-	sg-085b2ca5bb754e034	default	vpc-086ee66792e66d59a
<input checked="" type="checkbox"/>	-	sg-0b567214fd6910f47	launch-wizard-1	vpc-086ee66792e66d59a

セキュリティグループ(2)

インバウンドルール→インバウンドルールを編集


インバウンドルール | アウトバウンドルール | タグ


 Reachability Analyzerでネットワーク接続を確認できるようになりました


Reachability Analyzerの実行

×

インバウンドルール (4)

 タグを管理 インバウンドのルールを編集

 セキュリティグループのルールをフィルタリング

< 1 > 

<input type="checkbox"/>	Name ▾	セキュリティグルー... ▾	IP バージョン ▾	タイプ ▾	プロトコル
--------------------------	--------	----------------	------------	-------	-------

セキュリティグループ(3)

タイプ：カスタムTCP,ポート範囲：apps.pyで
起動するポート,ソース：0.0.0.0/0を選択→
ルールを保存

sgr-089e394c931655a51	HTTPS ▼	TCP	443	カスタム ▼	Q		削除
					0.0.0.0/0 X		
sgr-06c4aee383a415674	カスタム TCP ▼	TCP	8888	カスタム ▼	Q		削除
					0.0.0.0/0 X		
sgr-0dc9fe9a5d0ebdb66	HTTP ▼	TCP	80	カスタム ▼	Q		削除
					0.0.0.0/0 X		
sgr-07fdecf8fcc0d1e4c	SSH ▼	TCP	22	カスタム ▼	Q		削除
					0.0.0.0/0 X		

ルールを追加

キャンセル 変更をプレビュー **ルールを保存**

インスタンス確認

※実際は、MyWebServer1です
インスタンスIDのリンクをクリックする
Elastic IP アドレスの欄をコピーする

New EC2 Experience
Tell us what you think

EC2 ダッシュボード
EC2 グローバルレビュー
イベント
タグ
制限

▼ インスタンス
インスタンス New
インスタンスタイプ

インスタンスを起動

Find インスタンス by attribute or tag (case-sensitive)

<input type="checkbox"/>	Name ▼	インスタンス ID
<input type="checkbox"/>	MyWebServer2	i-02c7e088a6bf16d26
<input type="checkbox"/>	MyWebServer5	i-031b6abd7b9b06399

インスタンスを選択

Elastic IP アドレス
[パブリック IP]

AWS Compute Optimizer の検出結果
レコメンデーションについては、AWS Comp
ute Optimizer に最適化を推奨

SSHで接続(Mac)

(1)ダウンロードしたpemファイルを~/.sshへ
移動する

(2)Chmod 600 /Users/username/.ssh/xxxx.pem
で権限を変更

※/Users/usernameは自分のログインID合わせる

(3)sshで接続する

ssh -i /Users/username/.ssh/xxxx.pem

ec2-user@Elastic ipアドレス

※改行入れず、pemとec2-userの間はスペース

SSHで接続(Windows)

(1)TeraTermをインストール

(2)起動して、ホスト：Elastic IP、サービス：SSH
でOK

(3)ユーザー名：ec2-user

パスワード：なし

認証方式：RSA/DSA・・・鍵を使う

ダウンロードしたpemを指定

※無くさないように任意のディレクトリ格納

Amazon Linux 2 設定

ここからAmazon Linux 2の設定を行います
SSH接続できれば



<https://aws.amazon.com/amazon-linux-2/>

```
5 package(s) needed for security, out of 8 available
Run "sudo yum update" to apply all updates.
```

と表示されている
ec2-userで操作し、インストールはsudoで
rootになって操作する

PostgreSQL11(1)

以後、単語とオプションの間は、半角スペースがある

#パッケージ更新

```
sudo yum -y update
```

#postgresql11インストール

```
amazon-linux-extras install postgresql11
```

```
sudo amazon-linux-extras enable postgresql11
```

#以下、改行無しで

```
sudo yum install postgresql-server postgresql-  
contrib postgresql-devel
```

PostgreSQL11(2)

#postgresユーザーにスイッチ

sudo su - postgres

#DB初期化

initdb --locale=C --encoding=UTF8

#postgresユーザーを抜ける

exit

#自動起動設定

sudo systemctl enable postgresql.service

sudo systemctl start postgresql.service

PostgreSQL11(3)

#DB接続ユーザー作成

```
createuser -d -P -U postgres book_user
```

#パスワードは、ローカル環境と同じにする

#空のDBを作る

```
createdb -U book_user book_data
```

Python環境

#バージョン確認

python3 --version

#ec2-userのホームディレクトリで作業

#作業ディレクトリを作る

mkdir setup

cd setup

#pipが無いのでインストール -O : 大文字オー

curl -O <https://bootstrap.pypa.io/get-pip.py>

python3 get-pip.py --user

pip --version #バージョン表示があればOK

Nginx(1)

Webサーバー：Nginxをインストール

#レポジトリファイルを作る

```
sudo vi /etc/yum.repos.d/nginx.repo
```

—— (内容は以下)

```
[nginx]
```

```
name=nginx repo
```

```
baseurl=http://nginx.org/packages/mainline/centos/7/$basearch/
```

```
gpgcheck=0
```

```
enabled=1
```

—— (ここまで)

Nginx(2)

#インストール

```
sudo yum -y install nginx
```

インストール後の確認

```
nginx -V
```

#自動起動設定

```
sudo systemctl enable nginx.service
```

```
sudo systemctl start nginx.service
```

```
sudo systemctl state nginx.service
```

この時点で . . .

Webサーバーをインストールしたので
ブラウザからElastic ipのアドレスにアクセスすると

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Uwsgi

NginxとPythonのプログラムをつなぐ役割をします

#uwsgiインストール

```
sudo yum -y groupinstall "Development Tools"
```

```
sudo yum install python3-devel
```

```
pip install uwsgi
```

#ディレクトリを作る

```
sudo mkdir -p /var/www/uwsgi/
```

```
sudo chown ec2-user:ec2-user -R /var/www/
```


DBのバックアップ

- ①ローカルPCでpgAdmin4を起動する
- ②book_dataを右クリックして、バックアップする
ファイル名 : book_data.dump.sql
形式 : 平文(plain text)
エンコーディング : UTF8
- ③②で作ったファイルをプロジェクトルート/sql
ディレクトリに移動する
- ④git でpushする

デプロイ(1)

#完成プロジェクトをclone

#/home/ec2-user/bottle-bookとして展開

git clone <--自分のgitリポジトリ→/bottle-book.git

#バックアップからデータ復元(改行無し)

psql -U book_user -f book_data.dump.sql

book_dat

#requirements.txtを編集

#Beaker,bottle,Jinja2,psycopg2,SQLAlchemy

#以外は削除

vi requirements.txt

デプロイ(2)

#pipでインストール

```
pip install -r requirements.txt
```

#nginxの設定

```
cd /etc/nginx/conf.d/
```

```
sudo mv default.conf default.confbak
```

```
sudo vi uwsgi.conf
```

デプロイ(3)

#nginxの設定ファイルの中身

```
server {  
    listen      80;  
  
    location / {  
        include uwsgi_params;  
        uwsgi_pass unix:///tmp/uwsgi.sock;  
    }  
}  
#設定ファイルここまで
```

デプロイ(4)

```
cd /home/ec2-user/bottle-book  
#uwsgiの設定ファイル、以下で保存する
```

```
vi apps.ini
```

```
[uwsgi]  
socket      = /tmp/uwsgi.sock  
chdir       = /home/ec2-user/bottle-book/  
master      = true  
processes   = 5  
file        = apps.py  
chmod-socket = 666  
vacuum      = true  
die-on-term = true  
logto       = /var/www/uwsgi/apps.log
```

この時点で起動確認

```
ugsgi --ini apps.ini
```

#以下の起動メッセージが出たら

```
[uWSGI] getting INI configuration from  
apps.ini
```

ブラウザからElastic ipにアクセスすると
サービスがアプリが利用できるはず・・・

Ctrl+Cで起動を停止する

デプロイ(5)

#uwsgi自動起動設定,サービスとして登録

sudo vi /etc/systemd/system/uwsgi.service

[Unit]

Description = uWSGI

After = syslog.target

[Service]

User = ec2-user

ExecStart = /home/ec2-user/.local/bin/uwsgi --ini /home/ec2-user/bottle-book/apps.ini

Restart=always

KillSignal=SIGQUIT

Type=notify

StandardError=syslog

NotifyAccess=all

[Install]

WantedBy=multi-user.target

デプロイ(6)

```
sudo systemctl enable uwsgi.service  
sudo systemctl start uwsgi.service
```

これでデプロイ(アプリケーション配置)は完了です
これ以後、サーバー(インスタンス)を起動すれば
アプリが使える状態になります

この後は,こんなことも必要(1)

- ・ドメインの取得

Elastic ipでアクセスしているので

ドメイン取得してアクセスすることもできます

(例)hoge hoge.comにブラウザからアクセスなど

①無料のドメインを取得する

②route53というawsのサービスに登録する

この後は,こんなことも必要(2)

- SSL取得

通常のアクセスをしていると、ID,パスワードなど送信した情報がインターネット上で覗かれることもあります。SSL(httpsでアクセス)で通信を暗号化します

①Let's Encryptというサービスで暗号キーを取得する

②サーバー上でLet'sEncryptの自動更新を行う
時間が経つと、証明書の賞味期限が切れる

AWS Academyは

- ・ 今年度(2023/3/31)まで有効です
以後は残念ながら使えません
- ・ セッションが切れるとサービスが落ちるので
通常のサーバーではありません
- ・ もし、常設のサーバーが必要であれば
AWSを自分でアカウント作って試して下さい
(これぐらいの構成だと2ヵ月ぐらいは無料枠で行ける)
有料でも600円@月ぐらい
- ・ ElasticIPは使わない場合は、必ず削除
逆にお金が掛かります