好奇心の赴くままに

やりたいことを書いています。

2019-12-22

AWS(EC2)上にSpringBootアプリをデプロイする

EC2 RDS SpringBoot

Herokuといった選択肢もあるけど、せっかくならAWS上に作ったWeb アプリ（ここではSpringBootアプリ）をデプロイしたいよね。ということで、AWSを触ったことのない人でもAWS環境を構築して、デプロイできるように手順を書いてみました。用語とか、設定値の説明はほとんど書いていないです。とりあえず手を動かしながら、覚えたい人向け。

目次

- 目次
- 開発環境
- 前提
- 今回のゴール
- 手順
  - AWSインフラ構築
  - ざっくりとした用語説明
    - VPC、サブネットの作成
    - インターネットゲートウェイ作成及びVPCとの関連付け
    - ルートテーブルの作成及びサブネットとの関連付け
    - EC2の作成
    - ElasticIPの作成及び関連付け

        - RDSのセキュリティグループの作成
        - RDSの作成
    - EC2に接続後、各種インストール
    - ローカル環境でSpringBootアプリをMavenビルド
    - EC2上にデプロイ
    - EC2に接続後、アプリ起動
    - ブラウザからアクセス
- 参考
- 雑感

開発環境

Window10

Java1.8

Maven

Spring Boot 2.0.4

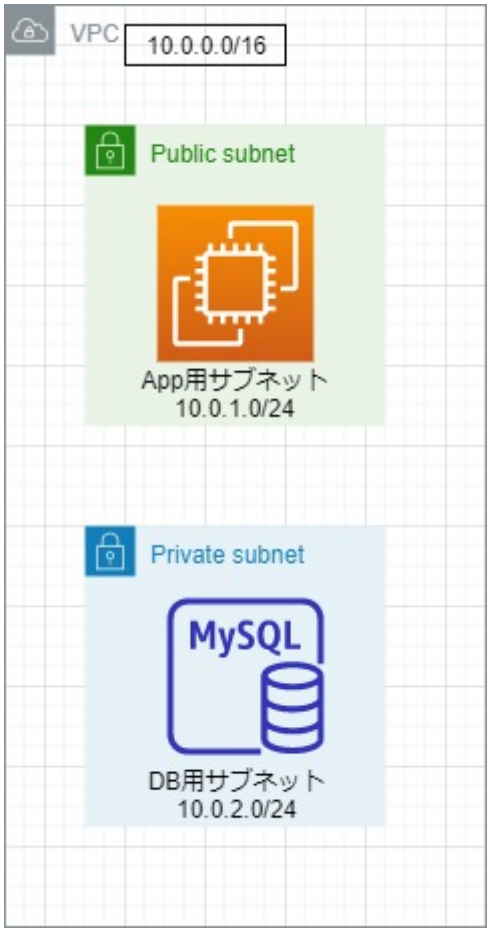
前提

- RDB(MySQL)を利用したSpringBootアプリを作成済み
- mysql-connector-javaをインストール済み
  ※ない場合はpom.xmlに以下を追加してください。

```
<dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <scope>runtime</scope>
</dependency>
```

今回のゴール

以下AWS環境にSpring Bootアプリをデプロイする



手順

AWSインフラ構築

ざっくりとした用語説明

項目	説明
EC2	仮想サーバー
RDS	DBサーバ
VPC	EC2やRDSを立てるための大枠のアドレス空間
サブネット	VPCをさらに分割したアドレス空間

項目	説明
ルートテーブル	各サブネットのパケットの宛先を見て行先を決めるテーブル
パブリックサブネット	0.0.0.0/0 (デフォルトゲートウェイへの通信) がインターネットゲートウェイに流れるような設定
プライベートサブネット	上記以外のサブネット
インターネットゲートウェイ	VPC、インターネット間の通信を実現するためのもの
Elastic IP	静的な公開IPアドレス

VPC、サブネットの作成

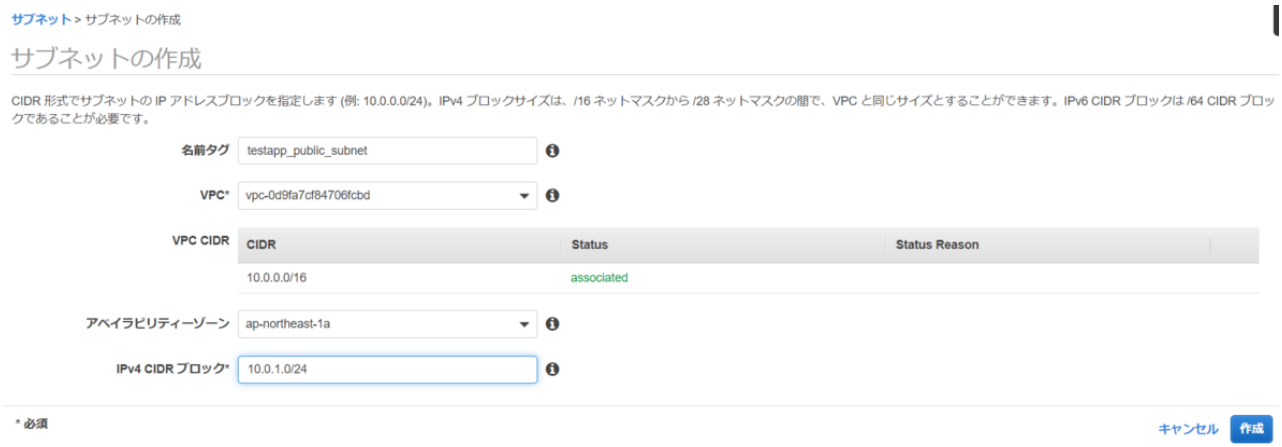
左上の[サービス]-[VPC]を選択後、[VPCの作成]より以下項目を入力して [作成]ボタンを押下

項目	設定値
名前タグ	testapp_vpc(適当に設定)
IPv4 CIDR ブロック	10.0.0.0/16

VPC作成後、左上ペインより、 [サブネット]-[サブネットの作成]より以下項目を入力して[作成]ボタンを押下

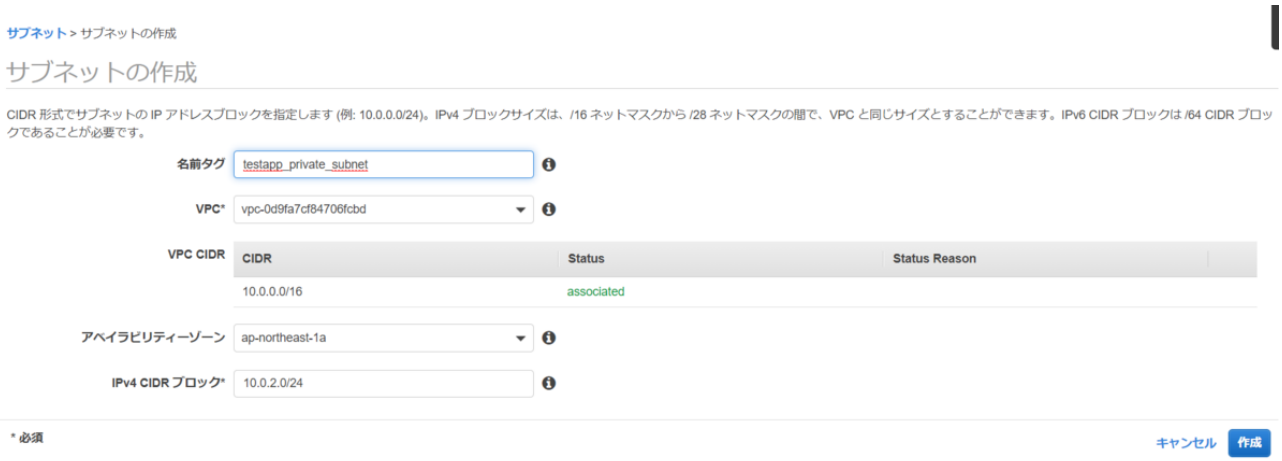
項目	設定値
名前タグ	testapp_public_subnet(適当に設定)

項目	設定値
VPC	testapp_vpc(先ほど作成したものを選択)
IPv4 CIDR ブロック	10.0.0.0/16
アベイラビリティゾーン	ap-northeast-1a
IPv4 CIDR ブロック	10.0.1.0/24



同じようにして以下プライベートサブネットも作成

項目	設定値
名前タグ	testapp_private_subnet(適当に設定)
VPC	testapp_vpc(先ほど作成したものを選択)
IPv4 CIDR ブロック	10.0.0.0/16
アベイラビリティゾーン	ap-northeast-1a
IPv4 CIDR ブロック	10.0.2.0/24



インターネットゲートウェイ作成及びVPCとの関連付け

左ペインより[インターネットゲートウェイ]-[インターネットゲートウェイの作成]より 以下項目を入力して[作成]ボタンを押下

項目	設定値
名前タグ	testapp_igw(適当に設定)

作成後、作成したインターネットゲートウェイを選択し、[アクション]-[VPCにアタッチ]より作成したVPCを選び、アタッチボタンを押下

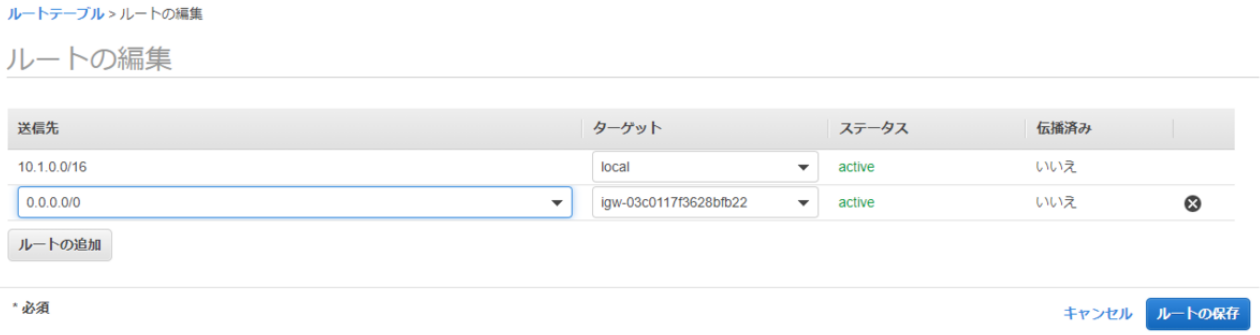
ルートテーブルの作成及びサブネットとの関連付け

左ペインより[ルートテーブル]-[ルートテーブルの作成]より 以下項目を入力して[作成]ボタンを押下

項目	設定値
名前タグ	testapp_igw(適当に設定)
VPC	testapp_vpc（先ほど作成したものを選択）

作成後、作成済みのルートテーブルを選択し、下のルートタブから [ルートの編集]を押下し、以下のルートを追加

送信先	ターゲット
0.0.0.0/0	testapp_igw（先ほど作成したインターネットゲートウェイを選択）



ルート追加後、作成済みのルートテーブルを選択し、[アクション]-[サブネット関連付けの編集]より
作成した公開サブネット(testapp_public_subnet)を選択し、保存ボタンを押下

EC2の作成

左上の[サービス]-[EC2]を選択後、左ペインより[インスタンス]-[インスタンスの作成]より以下項目を入力して[作成]ボタンを押下

項目	設定値
AMI	Amazon Linux2
インスタンスタイプ	t2.micro
ネットワーク	testapp_vpc(作成したVPCを選択)

項目	設定値
サブネット	testapp_public subnet

ステップ6:セキュリティグループの設定において以下でセキュリティグループを新規作成する

タイプ	プロトコル	ポート範囲	ソース	用途
SSH	TCP	22	マイIP	サーバ接続用
カスタムTCP	TCP	(SpringBootの接続ポート)	0.0.0.0/0	ブラウザからのアクセス用

ElasticIPの作成及び関連付け

左ペインより[ElasticIP]-[ElasticIPアドレスの割り当て]より [割り当て]ボタンを押下して、ElasticIPを取得する。

取得後、取得したElasticIPを選択し、[アクション]-[ElasticIPアドレスの割り当て] より作成したEC2インスタンスを選択後、[関連付ける]ボタンを押下

RDSのセキュリティグループの作成

左ペインより[セキュリティグループ]-[セキュリティグループの作成]より 以下項目を入力して、[作成]ボタンを押下。

項目	設定値
セキュリティグループ名	testapp_rdssg
説明	testapp_rdssg
VPC	testapp_vpc（作成したVPC)

タイプ	プロトコル	ポート範囲	ソース	用途
MySQL/Aurora	3306	(EC2のプライベートIP)	DB接続用	

RDSの作成

左上の[サービス]-[RDS]を選択後、[VPCの作成]より以下項目を入力して [作成]ボタンを押下

項目	設定値
パラメータグループファミリー	MySQL5.7
グループ名	testapp-prgr（適当に設定）
説明	testapp-prgr（適当に設定）

作成後、[パラメータグループアクション]-[編集]より以下パラメータを変更（日本語の文字化けを防ぐため）

パラメータ	説明	設定値
character_set_client	クライアントの送信する文字コード	utf-8
character_set_connection	文字コードの情報がない時の文字コード	utf-8
character_set_datebase	参照しているデータベースの文字コード	utf-8
character_set_results	クライアントへ送信する文字コード	utf-8

パラメータ	説明	設定値
character_set_server	既定の文字コード	utf-8

以下構成でRDSを作成する

項目	設定値
DBエンジン	MySQL
バージョン	5.7.22
テンプレート	無料利用枠
DB インスタンス識別子	testapp-rds（適当に設定）
マスターユーザー名	*******（適当に設定）
マスターパスワード	*******（適当に設定）
インスタンスタイプ	t2.micro
VPC	testapp_vpc
最初のデータベース名	*******（適当に設定）
セキュリティグループ	testapp_rdssg
パラメータグループ	testapp_prgr

EC2に接続後、各種インストール

Powershellを開き、EC2にSSH接続する

```
> ssh -i  "(キーペアのパス)" ec2-user@(公開IP)  # EC2にSSH接続
$ sudo yum update # yumリポジトリアップデート
$ yum install java-1.8.0-openjdk.x86_64   # SpringBootを動かすためのJavaをインストール
$ yum install mysql                                    # MySQLクライアントをインストール
```

RDSのエンドポイントを確認後、RDSに接続確認

```
> mysql -h (RDSのエンドポイント ) -P 3306 -u (マスターユーザ名) -p
> 先ほど設定したマスターパスワード入力
```

環境変数を設定して、SpringBootアプリの接続情報を上書きする

※ 公式リファレンスを見るとapplication.propertiesよりも環境変数のほうが読み込み優先度が高いことがわかる

Spring Boot Reference Documentation

```
$ vim ~/.bach_profile  # 環境変数の設定
export SPRING_DATASOURCE_URL=jdbc:mysql://(RDSのエンドポイント):3306/（作成したDB名)
export SPRING_DATASOURCE_USERNAME=（設定したマスターユーザー）
export SPRING_DATASOURCE_PASSWORD=（設定したマスターユーザーのパスワード）
export SPRING_DATASOURCE_DRIVER_CLASS_NAME=com.mysql.jdbc.Driver

$ source ~/.bash_profile  # 反映
```

接続確認後、切断

ローカル環境でSpringBootアプリをMavenビルド

```
> cd (SpringBootアプリのパス)
> mvnw package
```

EC2上にデプロイ

EC2にsftp接続後、デプロイ

```
> cd (ビルド後のjarファイルのあるフォルダパス)
> sftp -i  "(キーペアのパス)" ec2-user@(公開IP)
sftp > put （jarファイル)
```

EC2に接続後、アプリ起動

```
> ssh -i  "(キーペアのパス)" ec2-user@(公開IP)  # EC2にSSH接続
$ java -jar ~/（jarファイル)
```

ブラウザからアクセス

http:///(公開IPアドレス):(SpringBootの起動ポート番号)/にアクセスして DB接続できているかを確認する

参考

AWS EC2上で Spring Bootアプリ起動 - 闘うITエンジニアの覚え書き

Spring Boot Reference Documentation

Spring-Bootの設定プロパティと環境変数 - Qiita

雑感

AWSコンソール上でやるのはやっぱりしんどいし、手順も面倒くさい。あと、Paasの方が楽やん。

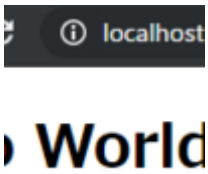
kamada-math 3年前

関連記事

2020-12-02
mac EC2にWindows10環境からSSH・VNC接続してみた
「AWS Late Night Week1」にて発表された何かと話題!?のmac EC2…

2019-12-29
JenkinsMasterサーバからJenkinsSlaveサーバにSSH接続してジョブを実行する

以下の本を進めている中でdockercomposeでJenkinsMasterとJensk…



2019-12-08

JenkinsとGithubの連携(SpringBoot)

目次 目次 開発環境 Jenkins環境 今回のゴール 手順 Spring boo…

コメントを書く

« JenkinsMasterサーバからJenkinsSlaveサーバ…

JenkinsとGithubの連携(SpringBoot) »

プロフィール

kamada-math

某SIerに生息しているエンジニア

読者になる 3

このブログについて

検索

記事を検索

最新記事

Try EnvoyのGetting Started with EnvoyでEnvoyに入門してみる

SDKMANでsdk list javaした際に「INTERNET NOT REACHABLE!」エラーが出た時の対処法

2022年の目標宣言

2021年の振り返り

atcoder-cli,online-judge-toolsを入れてAtCoderのローカル自動テスト環境を構築した

月別アーカイブ

- 2022 (4)
  - 2022 / 9 (1)
  - 2022 / 6 (1)
  - 2022 / 1 (2)
- 2021 (2)
- 2020 (7)
- 2019 (9)

カテゴリー

雑記 (2)

EC2 (2)

Jenkins (2)

RDS (1)

SpringBoot (2)

CentOS (1)

IAM (1)

linux (1)