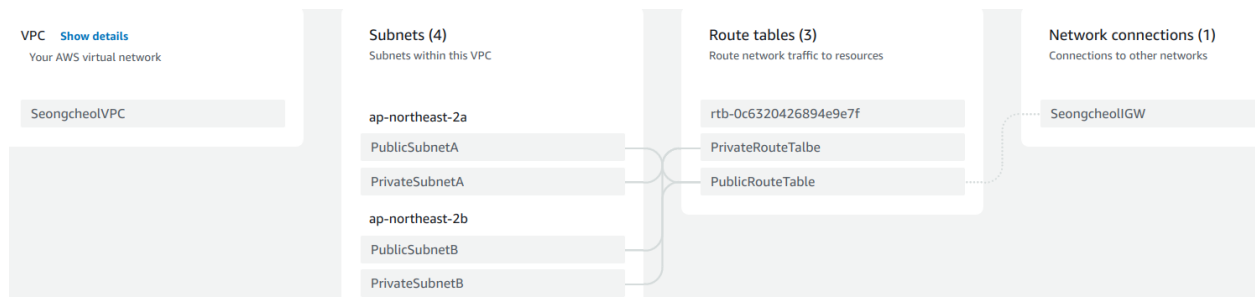


# 상세 구현 과정

## Network



### VPC를 만들었다.

- VPC 안에 서브넷들을 만들었다.
  - PublicSubnet 2개와 그에 속한 PrivateSubnet 2개를 만들었다
  - 이때고가용성 확보를 위해 서로 다른 서브넷에 두었다.

# rtb-0670725800b13cecb / PublicRouteTable

You can now check network connectivity with Reachability Analyzer

**Details** [Info](#)

Route table ID rtb-0670725800b13cecb	Main No	Expl 2 sul
VPC <a href="#">vpc-027d96c194fce3b9</a>   SeongcheolVPC	Owner ID 144018601164	

[Routes](#) | [Subnet associations](#) | [Edge associations](#) | [Route propagation](#) | [Tags](#)

**Routes (2)**

Destination	Target
0.0.0.0/0	<a href="#">igw-081dfc7c7d6d50289</a>
10.0.0.0/16	local

## RouteTable을 만들었다.

- PublicRouteTable, PrivateRouteTable을 따로 두었다.
- PublicRouteTable에만 IGW를 달아두었다.
  - 모든 IP에서의 접근은 IGW로 향하게 한다 → 인터넷 연결 허용
  - 10.0.0.0/16 에서의 접근은 local로 향하게 한다 → 10.0.0.0/16 은 VPC 전체의 IP 주소이다. VPC에서의 접근은 VPC로 향하게 한다는 말이니까, 이것은 곧 VPC 내부의 요소들끼리 모두 연결시켜 준다는 것이다.

## Details

Internet gateway ID

igw-081dfc7c7d6d50289

State

Attached

VPC ID

vpc-027d96c194fcfe3b9 | SeongcheolVPC

## IGW 이다.

- 내가 만든 VPC와 연결이 되어있다.
- VPC의 PublicRouteTable와 연결을 해두었고, PublicRouteTable은 PublicSubnet과 연결이 되어있으므로 PublicSubnet 위에 인스턴스를 띄우면 인터넷 연결이 된다.

VPC를 만들고 그것과 연결된 IGW를 만들었다. VPC 안에 퍼블릭 서브넷을 만들고, 퍼블릭 서브넷안에 인스턴스를 만들었다. 퍼블릭 서브넷을 관리하는 라우팅 테이블을 만들고, 그 테이블에서 0.0.0.0/0 → IGW를 해준다면 인스턴스가 인터넷 연결 가능해진다.

- Q.퍼블릭 라우팅 테이블에서 0.0.0.0/0 의 타겟을 IGW로 연결한다면 퍼블릭 라우팅 테이블에서 관리하는 요소들 뿐만 아니라 해당 VPC 내 모든 요소들이 인터넷 연결이 가능해진다?

- A. Yes. 그러나 보안상 이유로 일부 요소들은 인터넷에 연결하지 않는것이 일반적.

이를 위해서 프라이빗 서브넷과 퍼블릭 서브넷을 나누어 관리하고 프라이빗 서브넷에는 NAT 게이트웨이를 배치하여 인터넷에 연결함. ⇒ 이것은 비용 발생함.

(Private을 사용하는것이 좋지만, 비용이 발생하기 때문에 프로젝트를 진행하는 동안은 Public에 모두 올려 사용합니다)

## 인스턴스

<input type="checkbox"/>	Name ▾	Instance ID
<input type="checkbox"/>	JSC-Redis	i-03ecca04aca283e55
<input type="checkbox"/>	JSC_MySQL	i-0b6ea3fa118a7675a
<input type="checkbox"/>	JSC_WAS_AMI	i-03966a857934a681c

## WAS

- Web Application Server 역할
- MySQL 모듈 연동, Redis 모듈 연동됨.
- 스크립트로 작성 (초기 세팅 스크립트)

```
#!/bin/bash

echo "====="
echo
echo "Apache install"
yum install -y httpd

echo "====="
echo
echo "PHP install"
amazon-linux-extras install -y php8.0

echo "====="
echo
echo "Apache-PHP-FPM Connect"

function fpm() {
    local before="proxy:unix:/run/php-fpm/www.sock|fcgi://localhost"
    local after="proxy:fcgi://127.0.0.1:9000"
    sed -i "s#$before#$after#g" /etc/httpd/conf.d/php.conf
}
fpm
sed -i "s#listen = /run/php-fpm/www.sock#listen = 9000#g" /etc/php-fpm.d/www.conf

echo "====="
echo
echo "Apache Start"
systemctl start httpd
systemctl enable httpd
systemctl status httpd

echo "====="
echo
echo "Test PHP Script"
cat <<EOF > /var/www/html/index.php
<?php
phpinfo();
```

```

?>
EOF

echo "=====
echo
echo "PHP-FPM Start"
systemctl start php-fpm
systemctl enable php-fpm
systemctl status php-fpm

echo "=====
echo
echo "APRM install & change config file Finish"

echo "=====
echo
echo "PHP REIDS Connect"
mkdir /phpredis
cd /phpredis
yum install -y php-devel
yum install -y gcc
wget https://github.com/nicolasff/phpredis/zipball/master -O phpredis.zip
unzip phpredis.zip
if [ -e phpredis-phpredis-* ]
then
    cd phpredis-phpredis-*
    phpize
    ./configure
    make && make install
else
    echo "Not Found"
    break
fi
cat <<EOF > /etc/php.d/predis.ini
extension=redis.so
EOF
php -m | grep -i redis
php -i | grep -i redis | grep -i version
sleep 3
function prc_1() {
    local before="php_value[session.save_handler] = files"
    local after="php_value[session.save_handler] = redis"
    sed -i "s#$before#$after#g" /etc/php-fpm.d/www.conf
}
prc_1

function prc_2() {
    local before="php_value[session.save_path] = /var/lib/php/session"
    local after="php_value[session.save_path] = tcp://127.0.0.1:6379"
    sed -i "s#$before#$after#g" /etc/php-fpm.d/www.conf
}
prc_2

systemctl restart php-fpm
systemctl status php-fpm

echo "=====
echo
echo "mysql install"
yum install -y mysql

```

```
systemctl start mysqld
systemctl enable mysqld
systemctl status mysqld

echo "=====
echo
echo "reboot"
reboot
```

## **/var/www/html/test.php**

```
<html>

<head>

    <title>Using Redis Server with PHP and MySQL</title>

</head>

<body>

    <h1 align = 'center'>Employees' Register</h1>

    <table align = 'center' border = '2'>

    <?php

        try {

            $data_source = '';

            $redis = new Redis();

            $redis->connect('$redis_ip', 6379);

            $sql = 'select

                employee_id,

                first_name,

                last_name

            from employees
```

```

';

$cache_key = md5($sql);

if ($redis->exists($cache_key)) {

    $data_source = "Data from Redis Server";

    $data = unserialize($redis->get($cache_key));

} else {

    $data_source = 'Data from MySQL Database';

    $db_name      = '$mysql_db';
    $db_user      = '$mysql_user';
    $db_password  = '$mysql_pwd';
    $db_host      = '$mysql_ip';

    $pdo = new PDO('mysql:host=' . $db_host . '; dbname=' . $db_name, $db_user, $db_password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $stmt = $pdo->prepare($sql);
    $stmt->execute();
    $data = [];

    while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
        $data[] = $row;
    }

    $redis->set($cache_key, serialize($data));
    $redis->expire($cache_key, 10);

```

```

    }

    echo "<tr><td colspan = '3' align = 'center'><h2>$data_source</h2></td></tr>";

    echo "<tr><th>Employee Id</th><th>First Name</th><th>Last Name</th></tr>";

    foreach ($data as $record) {

        echo '<tr>';

        echo '<td>' . $record['employee_id'] . '</td>';

        echo '<td>' . $record['first_name'] . '</td>';

        echo '<td>' . $record['last_name'] . '</td>';

        echo '</tr>';

    }

    } catch (PDOException $e) {

        echo 'Database error. ' . $e->getMessage();

    }

?>

</table>

</body>

</html>

```

- 초기 접근시 MySQL 에서 Data를 가져오고 Redis에 캐시로 올려줌.
- Redis는 NoSQL이기 때문에 직렬화해서 Redis에 저장하고, 화면에 보여주기 위해서는 다시 역직렬화를 해서 MySQL에 저장된 원본 형태로 되돌려준다.
- MySQL 인스턴스와 Redis 인스턴스도 이렇게 스크립트로 설치하여 생성하였음.

WAS 접근하기.



## Employees' Register

Data from Redis Server		
Employee Id	First Name	Last Name
1	JOHN	ROE
2	SMITH	DOE
3	RICHARD	MAJOR

접근한 기록이 있기 때문에 Redis 화면을 보여준다.

### ALB 생성하기

미리 만들어둔 보안그룹을 사용해서 ALB를 생성한다.

#### VPC 정보

대상에 대한 Virtual Private Cloud(VPC)를 선택합니다. 인터넷 게이트웨이가 있는 VPC만 선택할 수 있습니다. 로드 밸런서가 생성된 후에는 선택한 \

성철VPC  
vpc-027d96c194fcfe3b9  
IPv4: 10.0.0.0/16

#### 매핑 정보

최소 2개의 가용 영역과 영역당 하나의 서브넷을 선택합니다. 로드 밸런서는 이러한 가용 영역의 대상으로만 트래픽을 라우팅합니다. 로드 밸런서 또는

##### ☒ ap-북동쪽-2a(apne2-az1)

서브넷

서브넷-0b0a62b7646b9c4ec

퍼블릭서브넷A ▼

IPv4 설정

AWS에서 할당

##### ☒ ap-northeast-2b(apne2-az2)

서브넷

서브넷-0aab16c1a8dadde40

퍼블릭서브넷B ▼

IPv4 설정

AWS에서 할당

대상그룹을 설정한다.

▼ 경청자 HTTP:80

규약	포트	기본 작업	정보
HTTP ▼	: 80 1-65535	를 향해서	JSC-WAS-TG 대상 유형: 인스턴스, IPv4 HTTP ▼

[대상 그룹 만들기](#)

---

**리스너 태그 - 선택 사항**  
리스너에 태그를 추가하는 것을 고려하십시오. 태그를 사용하면 AWS 리소스를 분류하여 보다 쉽게 관리할 수 있습니다.

**리스너 태그 추가**  
최대 50개의 태그를 더 추가할 수 있습니다.

---

▼ 경청자 HTTPS:443





규약	포트	기본 작업	정보
HTTPS ▼	: 443 1-65535	를 향해서	JSC-WAS-TG 대상 유형: 인스턴스, IPv4 HTTP ▼

HTTP, HTTPS 로 들어오는 요청들은 타겟 그룹에 보내준다.

이때, 타겟그룹이란 ALB가 관리하는 인스턴스들을 뜻한다.

- Q.ALB의 타겟그룹에 속한 인스턴스들은 ALB가 관리하는 서브넷상에 위치해야 하나?
  - A.대상 그룹에 속한 인스턴스는 ALB가 관리하는 서브넷 뿐만 아니라, VPC 내에 위치한 모든 서브넷에서도 선택할 수 있습니다. 대상 그룹을 생성할 때 서브넷을 지정하면 해당 서브넷 내의 인스턴스만 대상 그룹에 추가됩니다. 그러나 대상 그룹에 속한 인스턴스가 위치한 서브넷이 ALB가 관리하는 서브넷과 다른 경우, 이를 연결하는 VPC 내에서 인터넷 게이트웨이 또는 NAT 게이트웨이를 통해 ALB와 통신할 수 있도록 구성해야 합니다.


## JSC-ALB

<b>▼ Details</b>  <a href="#">arn:aws:elasticloadbalancing:ap-northeast-2:144018601164:loadbalancer/app/JSC-ALB/abf13d76887c3b8a</a>			
Load balancer type Application	DNS name  JSC-ALB-535056351.ap-northeast-2.elb.amazonaws.com (A Record)	Status  Active	VPC <a href="#">vpc-027d96c194fce3b9</a> 
IP address type IPv4	Scheme Internet-facing	Availability Zones <a href="#">subnet-0b0a62b7646b9c4ec</a> ap-northeast-2a (apne2-az1) <a href="#">subnet-0aab16c1a8dadde40</a> ap-northeast-2b (apne2-az2)	Hosted zone ZWKZPGTI48KDX
Date created April 24, 2023, 16:14 (UTC+09:00)			

ALB를 성공적으로 생성하였지만, 아직은 타겟 그룹안에 어떠한 인스턴스도 존재하지 않는다.

**ASG** 와 연동하여 쓰기 위해서 AMI를 만들어야 한다.

## AutoScaling , AMI , Launch Templates

Amazon Machine Images (AMIs) (1) <a href="#">Info</a>				
Owned by me ▼		 Find AMI by attribute or tag		
<input type="checkbox"/>	Name ▼	AMI ID ▼	AMI name ▼	Source
<input type="checkbox"/>	-	<a href="#">ami-0731473c68fd1e338</a>	JSC-WAS-AMI	144018601164/JSC-WAS-AMI

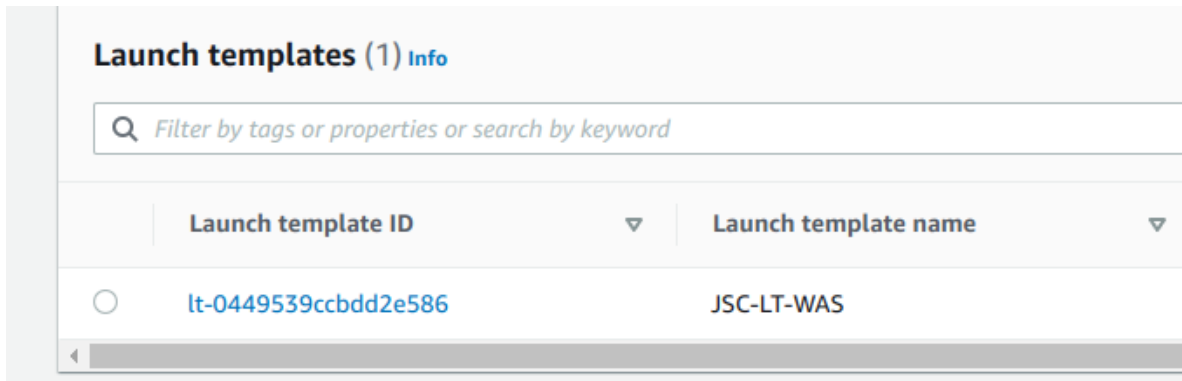
AutoScaling 하기 위해 WAS로 쓰이는 인스턴스를 AMI로 만들었다!

( AMI는 인스턴스를 생성하기 위한 이미지!)

AutoScaling하기 위해서는 Launch Templates가 필요하다

### Launch Templates

- AMI를 사용한다
- 버전 관리가 편하다
- ASG와 묶여 쓰인다.



런치 템플릿 생성!

WAS 이미지를 선택하여 템플릿을 생성하였다.

### Auto Scaling Group 만들기

## Name

Auto Scaling group name

Enter a name to identify the group.

Must be unique to this account in the current Region and no more than 255 characters.

## Launch template [Info](#)

[Switch to launch configuration](#)

**Launch template**

Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.

▼

↻

[Create a launch template](#)

**Version**

▼

↻

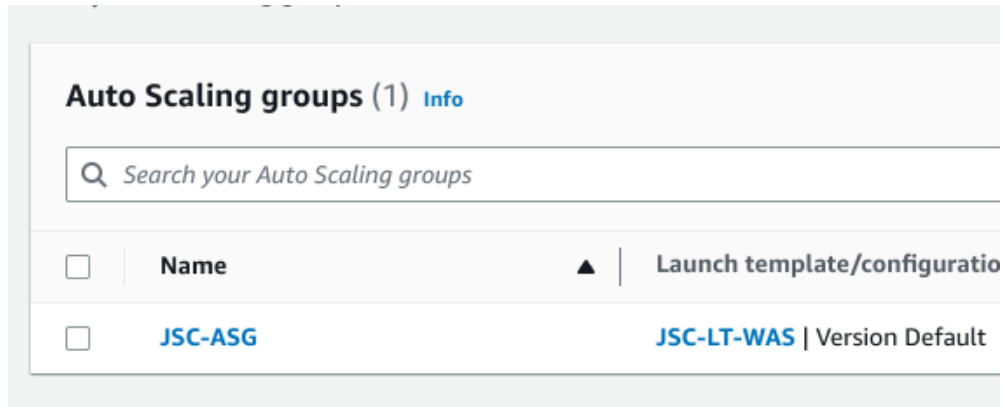
[Create a launch template version](#)

<p>Description</p> <p>JSC-LT-WAS</p>	<p>Launch template</p> <p><a href="#">JSC-LT-WAS</a></p> <p>lt-0449539ccbdd2e586</p>	<p>Instance type</p> <p>t2.micro</p>
<p>AMI ID</p> <p>ami-0731473c68fd1e338</p>	<p>Security groups</p> <p>-</p>	<p>Request Spot Instances</p> <p>No</p>
<p>Key pair name</p> <p>JSC-KeyPair</p>	<p>Security group IDs</p> <p><a href="#">sg-0592b3a3e87a66d61</a></p>	

Auto Scaling Group을 만들 때 VPC와 서브넷을 설정한다

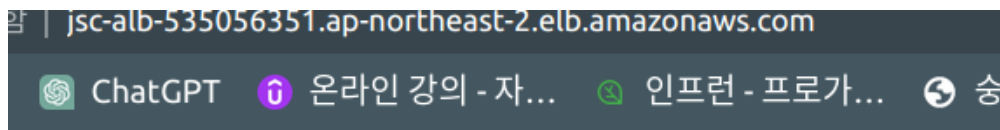
- ASG의 인스턴스가 위치할 서브넷을 선택하는 의미
- 런치 템플릿에 지정해둔 AMI를 통해 Scale in,out 시 어떤 서브넷에서 할 것인지를 정하는 것이다.
- 고가용성을 위해 서로 다른 가용영역에 위치한 서브넷을 선택!

## ASG 생성 완료



이제 IP가 아닌 ALB의 도메인을 통해 접근할 수 있다.

<http://jsc-alb-535056351.ap-northeast-2.elb.amazonaws.com/>



PHP 버전 8.0.28

체계

Linux

현재 ASG를 통해서 인스턴스 두개가 실행중이다.

## PHP 버전 8.0.28

체계	Linux ip-10-0-0-40.ap-northeast-2.compute.internal 5.10.176-17:49:06 UTC 2023 x86_64
빌드 날짜	2023년 3월 28일 17:43:43
시스템 구조	리눅스

## PHP 버전 8.0.28

체계	Linux ip-10-0-0-52.ap-northeast-2.compute.internal 5.10.176-17:49:06 UTC 2023 x86_64
빌드 날짜	2023년 3월 28일 17:43:43
시스템 구조	리눅스

서로 다른 IP를 갖고 있지만, 동일한 도메인으로 접속이 된다!

**To be continued...**