**Beanstalk 란?**

Java, .NET, PHP, Node.js, Python, Ruby, Go Docker 를 사용하여 Apache, Nginx, Passenger, IIS 와 같은 서버에 개발된 웹 Application 및 서비스를 간편하게 배포하고 관리 할 수 있는 서비스 이다.

**Beanstalk 장점**

**-빠르고 간편한 시작**

Eclipse나 Visual Studio와 같은 IDE 를 통해 Application을 업로드 하기만 하면 Elastic Beanstalk가 용량 프로비저닝, 로드 밸런싱, Auto Scaling, Application 모니터링에 대한 배포를 자동으로 처리한다. 인프라나 관련 리소스를 직접 구성할 필요가 전혀 없기 때문에 시간이 매우 단축 된다.

**-개발자의 생산성 향상**

인프라를 프로비저닝(**Provisioning)** 하고 운영할 뿐만 아니라 사용자 대신 Application 스택을 관리해 주기 때문에 시간을 따로 들일 필요가 없다. 개발자는 서버, 데이터베이스, 로드 밸런서, 방화벽, 네트워크등을 관리하고 구성하는데 시간을 들이는 대신 코드 작성에 더 집중할 수 있다.

**-적절한 규모 유지**

손쉽게 조정할 수 있는 Auto Scaling 설정을 사용하여 Application의 특정한 필요에 따라 자동으로 확장/축소 할 수 있다. Elastic Beanstalk를 사용하면 Application 비용을 최소화하면서 높은 워크로드나 트래픽을 처리할 수 있다.

**-완벽한 리소스 제어**

Amazon EC2 인스턴스, RDS, VPC 유형 등의 AWS 리소스를 Application에 가장 적합한 리소스로 자유롭게 선택할 수 있다. 또한 해당 리소스를 손쉽게 제어할 수 있다.

**Beanstalk 요금**

AWS Elastic Beanstalk 에 대한 추가 비용은 없고 Application을 저장하고 실행하기 위해 생성한 AWS 리소스 (ex: EC2 인스턴스, S3 버킷 등) 에 대해서만 비용을 지불하면 된다.

**Beanstalk 기능**

* 새로운 Application 버전을 쉽게 배포 및 이전 버전으로 롤백 할 수 있음
* 평균 CPU 사용률, 요청 개수, 평균 지연 시간 등 내장된 CloudWatch 모니터링 메트릭에 엑세스할 수 있음
* Application 상태가 바뀌거나 추가 또는 제거되면 Amazon Simple Notification Service(Amazon SNS)를 통해 이메일 알림을 받을 수 있음
* Application 서버에 로그인하지 않고도 서버 로그 파일에 엑세스할 수 있음
* Application의 CPU와 메모리 요구 사항에 일치하는 가장 적합한 Amazon EC2 인스턴스 유형을 선택할 수 있음
* 즉각적이고 직접적인 문제 해결을 위해 Amazon EC2 인스턴스에 대한 로그인 액세스 활성화
* Application을 둘 이상의 가용 영역에서 실행하여 안정성을 빠르게 향상
* 로드 밸런서에서 HTTPS 프로토콜을 사용하여 Application 보안 강화
* 메모리 캐싱 서비스와 같은 다른 Application 구성 요소를 Amazon EC2에서 병렬로 실행
* Auto Scaling 설정을 조정하여 환경에서 인스턴스를 추가하거나 제거하는 시기를 결정할 때 사용하는 메트릭과 임계값 제어

**Beanstalk 지원 언어와 개발 스택 (2015년 5월 기준)**

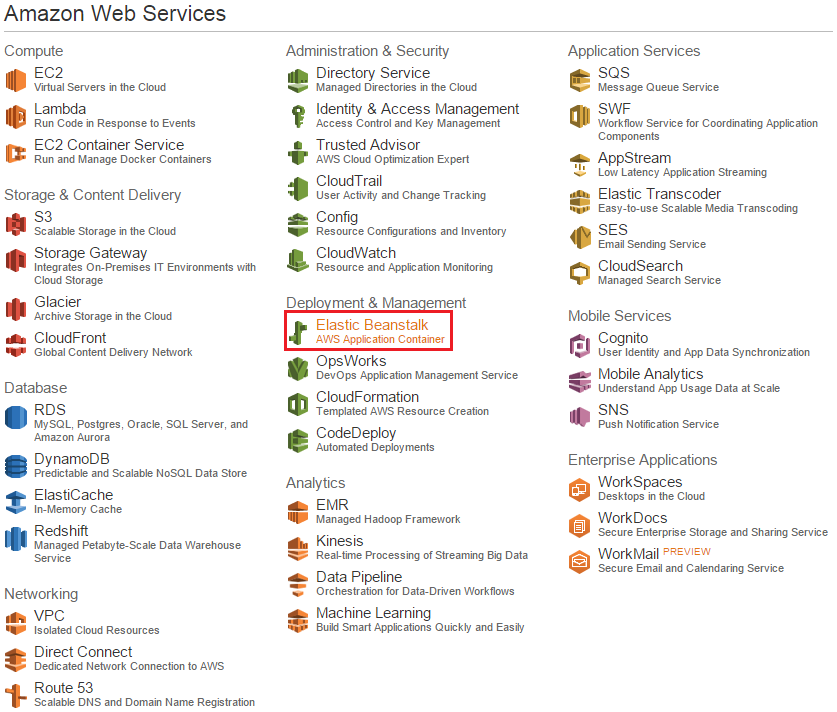
* Java8 Application용 Apache Tomcat8
* PHP Application용 Apache HTTP Server
* Python Application용 Apache HTTP Server
* Node.js Application용 Apache HTTP Server 또는 Nginx
* Ruby Application용 Passenger
* .NET Application용 Microsoft IIS 8.5

참고 자료 (Beanstalk) : <http://aws.amazon.com/ko/elasticbeanstalk>

**Amazon Web Services에서 Beanstalk 사용하기**

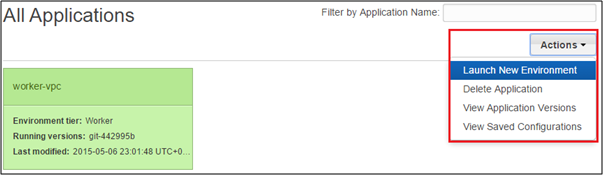
**1. Elastic Beanstalk 선택**

Amazon Web Services 페이지에서 Deployment & Management의 Elastic Beanstalk를 선택한다.



[그림 1] Amazon Web Services

Elastic Beanstalk 항목에 들어가 보면, 현재까지 생성 된 모든 Application들의 정보를 볼 수 있다. **Actions Drill Down** 버튼을 통해, 해당 Application에 Web Server, Worker를 추가 하고, Application 삭제/ 롤백 작업등도 할 수 있다. 해당 기능들은 뒤에서 설명 하겠다.

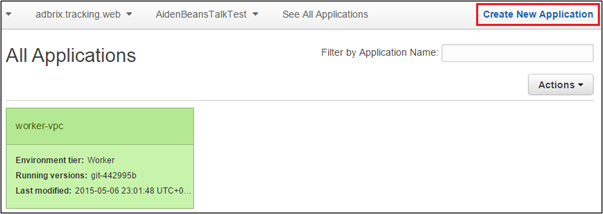


[그림 1] Elastic Beanstalk

**2. 새로운 Application 생성**

**1) Application 생성**

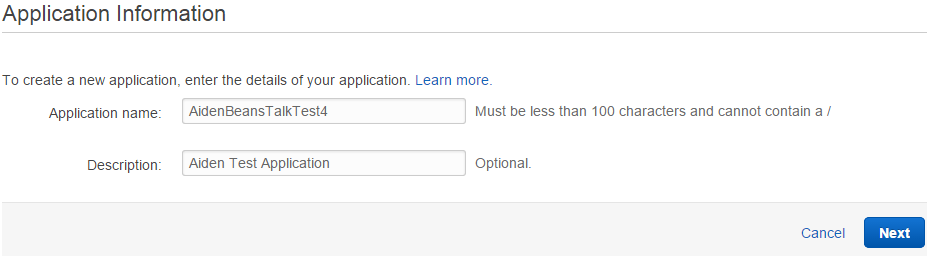
상단에 **Create New Application** 버튼을 클릭한다.



[그림 2-1] Create New Application

**2) Application 정보 설정**

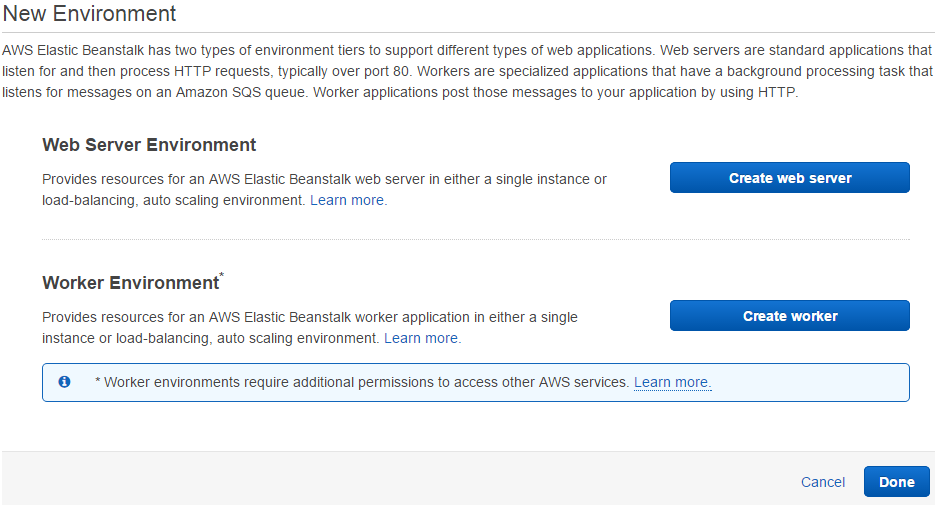
Application 이름 (100자 제한), Application 설명 (옵션) 등을 입력해 준다.



[그림 2-2] Application Information

**3) Application 종류 선택하기**

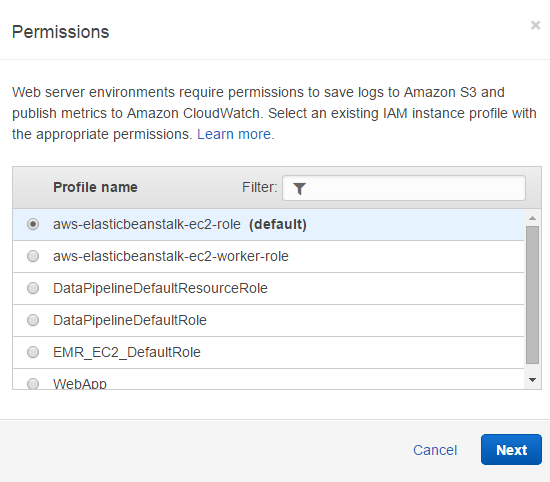
현재 Beanstalk의 Application 타입은 2가지를 지원하며, 우리가 흔히 사용하는 Web Server Application 을 사용하려면 **Web Server Environment**를 선택해 주고, 백그라운드 단에서 Amazon SQS queue(메시지가 유실되거나 손상되지 않도록 사용 됨)를 이용한 작업이 필요하면 **Worker Environment**를 선택 하면 된다. (Azure의 Worker Role 에서 Queue를 이용하는 것과 비슷한 맥락인 것 같음.)



[그림 2-3] Application Types

참고 자료 (SQS queue) : <http://aws.amazon.com/ko/sqs>

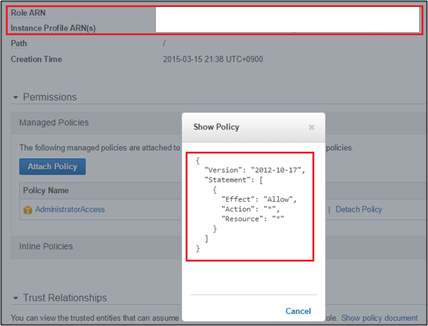
**Create web server**나 **Create worker**를 선택 하게 되면, 다음과 같이 Permissions 화면을 볼 수 있는데, 간단하게 생각하면 Application에서 리소스 접근할 수 있는 권한을 설정해 준다고 생각하면 될 것 같다. 해당 권한에 대해 몇 개만 살펴 보겠다.



[그림 2-3] Permissions

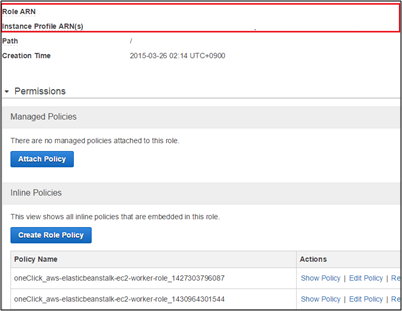
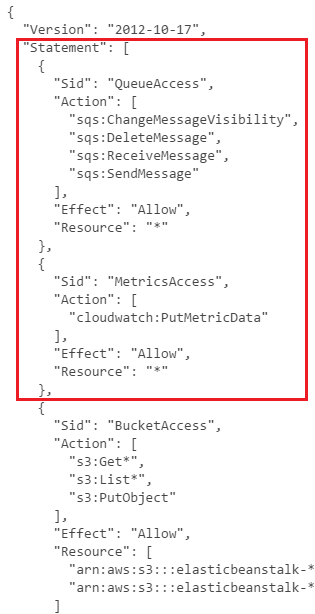
**aws-elasticbeanstalk-ec2-role (default)** 권한을 세부적으로 살펴 보면 Policy Name에

**AdministratorAccess**가 설정되어 있으며 해당 권한은 전체 리소스에 대해서 자유롭게 사용할 수 있도록 되어 있다.



[그림 2-3] aws-elasticbeanstalk-ec2-role

반면 **aws-elasticbeanstalk-ec2-worker-role** 에는 용도에 맞도록 세부 적인 권한들이 설정 되어 있는 것을 볼 수 있다.

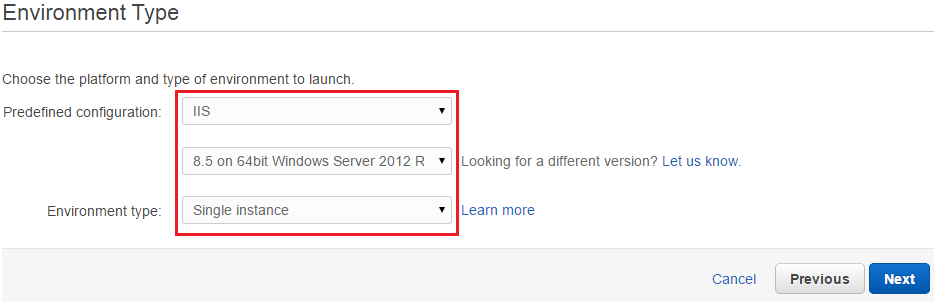
[그림 2-3] aws-elasticbeanstalk-ec2-worker-role

**4) 환경타입 선택하기**

개발에 필요한 Platform과 사용할 OS (Elastic Beanstalk는 기본적으로 Amazon Linux AMI 또는 Windows Server 2012 R2 AMI를 사용한다. )를 선택 한다.

**Environment Type** 으로는 **Single instance**와 **Load balancing, auto scaling** 을 고를 수 있는데, 생성 후에도 Type 을 변경할 수 있다.

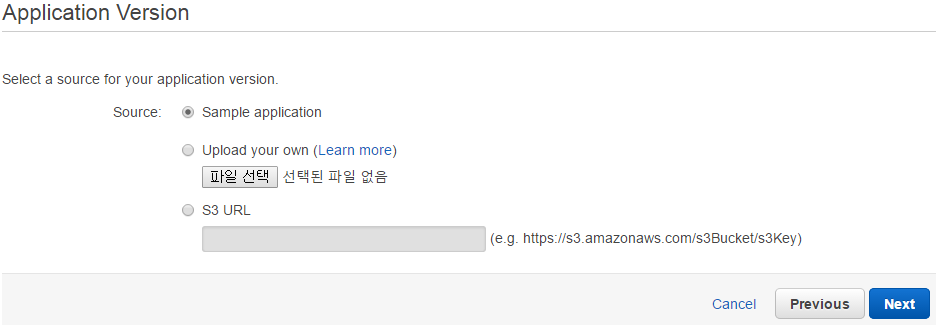
(ex : 적은 과금을 위해 TEST 개발 중일 경우 **Single instance**를 사용하다가 실제 서비스 오픈 후에는 **Load balancing, auto scaling** 으로 전환.하는 경우)



[그림 2-4] Environment Type

**5) Application 선택**

자신의 Application이 있다면 ZIP 파일 형태로 업로드 하면 된다.

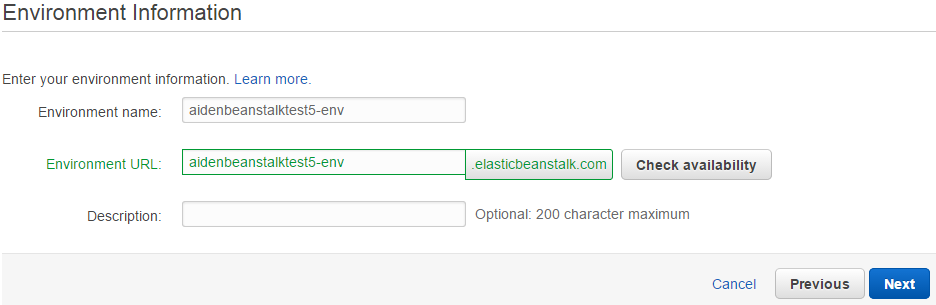


[그림 2-5] Application Version

**6) 환경 정보 설정**

**Environment name** 은 생성 될, EC2 인스턴스 이름으로 사용되며, 4 자에서 23 사이의 값으로 중복되지 않은 값으로 입력 해야 된다.

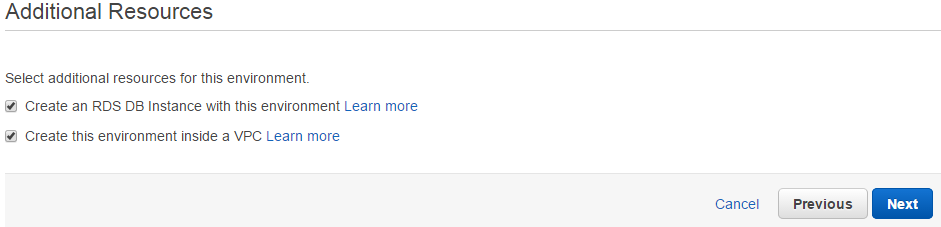
**Environment URL** 은 Deploy 완료 후에, 해당 Application에 유니크 한, 도메인이 되며, 중복되지 않은 값으로 입력 해야 된다.



[그림 2-6] Environment Information

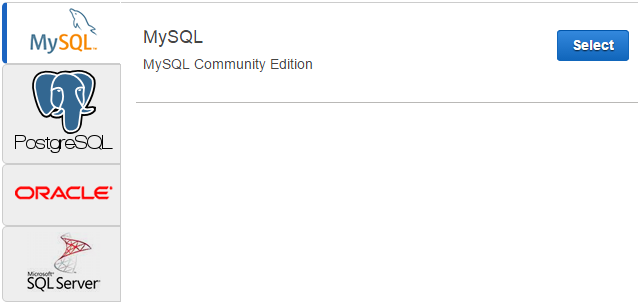
**7) 리소스 추가**

해당 Application에 새로운 **RDS(Relational Database Service) DB**나 **VPC**(Virtual Private Cloud) 를 추가 하고자 하는 경우 체크를 해 준다.



[그림 2-7] Environment Information

RDS는 관계형 데이터베이스를 더욱 쉽게 설치, 운영 및 확장할 수 있는 웹 서비스라고 생각하면 된다. Elastic Beanstalk 에서 사용하는 RDS의 종류에는 **MySQL, PostgreSQL, ORACLE, SQL Server** 등이 있다.



[그림 2-7] RDS 종류

VPC 는 자체 데이터 센터에서 운영하는 기존 네트워크와 매우 유사한 가상 네트워크로, AWS의 확장 가능한 인프라를 이용할 수 있다는 이점이 있다.

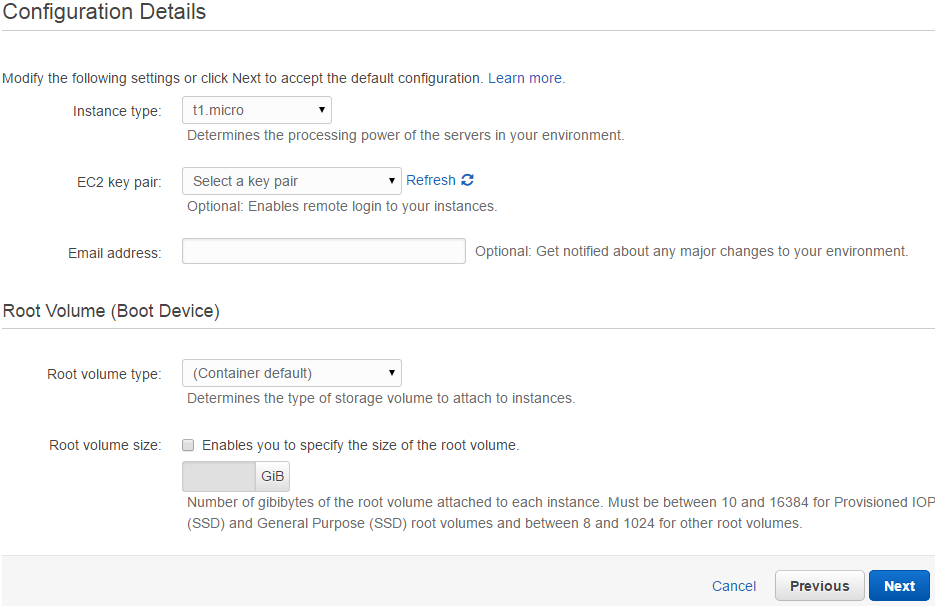
(RDS와 VPC의 자세한 내용은 아래 링크 참고)

참고 자료 (RDS) : <http://docs.aws.amazon.com/ko_kr/AmazonRDS/latest/UserGuide/Welcome.html>

참고 자료 (VPC) :

<http://docs.aws.amazon.com/ko_kr/AmazonVPC/latest/GettingStartedGuide/ExerciseOverview.html>

**8) 세부 설정**



[그림 2-8] Configuration Details

**Instance type** 은 CPU, 메모리, 스토리지 및 네트워킹 용량의 다양한 조합으로 구성되며, Application 성능에 따라 적합한 리소스 조합을 선택할 수 있는 유연성을 제공한다. 자신의 Application 성능에 따라서 사양을 늘리거나 줄여가면서 성능을 측정하는 것이 좋다.

참고 자료 (Instance type) : <http://aws.amazon.com/ko/ec2/instance-types>

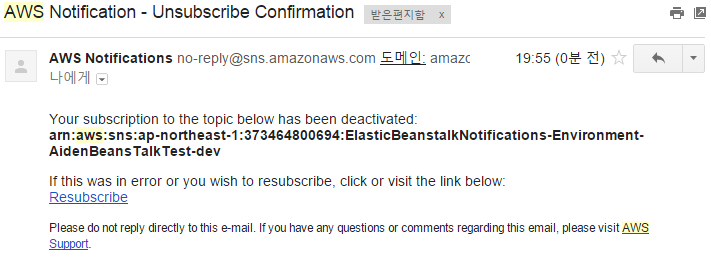
**EC2 key pair** 는 해당 인스턴스에 접근하기 위한 정보로 기존에 있는 EC2 Key를 사용 하거나 새로 생성할 수 있다. 새로 생성 시에는 **EC2 Dashboard > Resources (Key Pairs) > Create Key Pair** 에서 생성할 수 있으며, 생성 후 해당 PEM(Base64로 인코딩 된 인증서) 파일이 다운된다.

이 인증서를 가지고 SSH, RDP 등으로 해당 인스턴스에 접근할 수 있다.



[그림 2-8] Create Key Pair

**Email address (옵션)** 를 설정하면 인스턴스의 주요 상태가 변경되었을 경우, 해당 메일로 알려준다.



[그림 2-8] AWS Notifications Mail

**Root volume type** 에는 Magnetic, General Purpose (SSD), Provisioned IOPS (SSD) 세 가지가 있으며, 세 개의 볼륨 유형은 성능 특성과 비용이 각각 다르기 때문에 Application 요구 사항에 맞도록 선택해야 한다. Root volume size 는 최대 Magnetic 1TB, General Purpose, Provisioned IOPS는 16TB까지 설정할 수 있다.

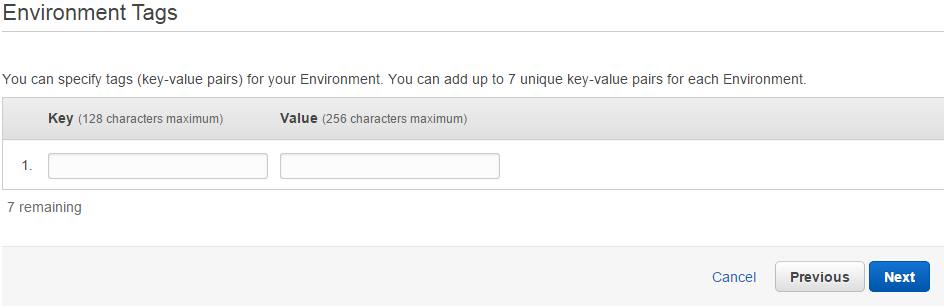


[그림 2-8] Amazone 볼륨 유형

참고 자료 (Amazon EBS Product Details) : <http://aws.amazon.com/ko/ebs/details>

**9) 태그 설정**

Key, Value 형태로 되어 있으며 고유의 메타데이터를 태그의 형태로 각 리소스에 배정하면 인스턴스, 이미지 및 기타 Amazon EC2 리소스를 쉽게 관리할 수 있다.



[그림 2-9] Environment Tags

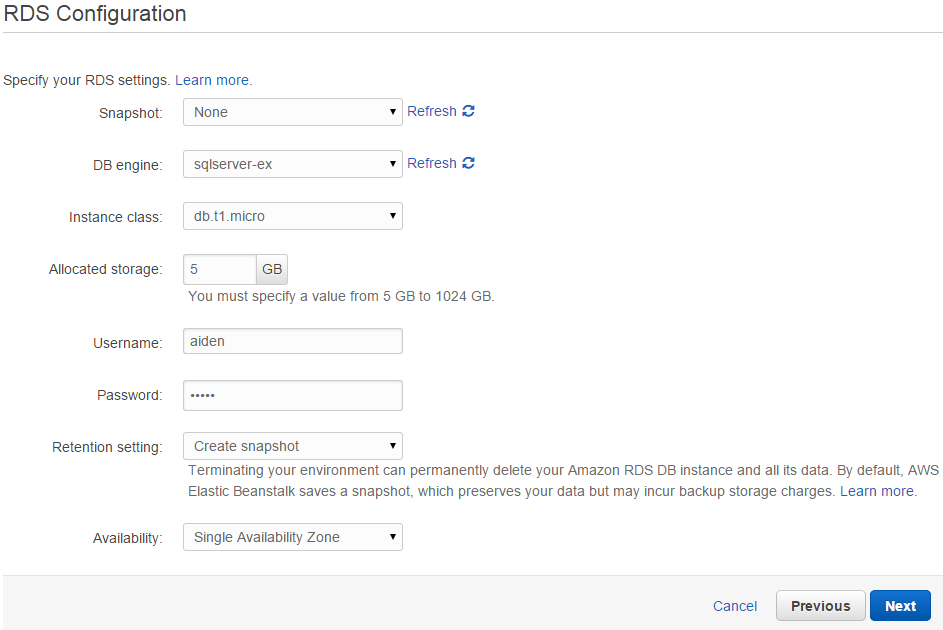
* 리소스당 최대 태그 수 7개 + 인스턴스 생성 후 3개 총 10개
* 최대 키 길이 – 유니코드 문자 127자
* 최대 값 길이 – 유니코드 문자 255자
* 태그 키와 값은 대/소문자 구분
* 태그 이름이나 값에 aws 라는 접두사 사용 금지. 해당 단어는 AWS 용으로 예약되어 있음.

참고 자료(Tag) : <http://docs.aws.amazon.com/ko_kr/AWSEC2/latest/UserGuide/Using_Tags.html>

**10) RDS 설정 (옵션)**

**7) 리소스 추가** 에서 **Create an RDB DB Instance with this environment**를 체크 한 경우

**RDS Configuration** 을 설정할 수 있다.



[그림 2-10] RDS Configuration

**Snapshot** 은 사용자가 시작한 DB 인스턴스 백업공간이다. 원하는 경우 언제나 DB 스냅샷에서 새 DB 인스턴스를 생성할 수 있다.

**DB engine** 종류에는 크게 MySQL, PostgreSQL, ORACLE, SQL Server가 있으며 세부적으로는 자신이 사용할 DB engine 을 선택하면 된다.

**Instance class** Amazon RDS는 다음과 같은 데이터 베이스 엔진용 DB 인스턴스 클래스를 지원한다.



[그림 2-10] DB Instance class

**Allocated storage** 5G부터 1024 GB 사이의 DB 용량을 산정해 준다.

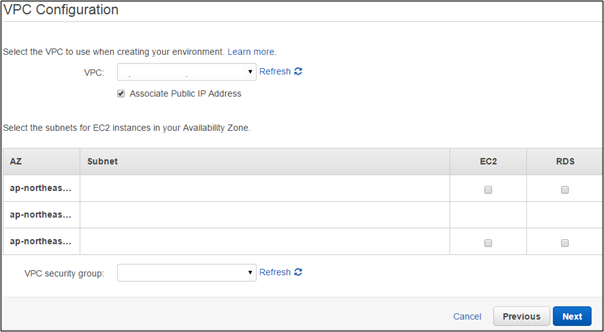
**Availability** Multiple Availability Zones (해당 기능 좀더 확인 필요)

**11) VPC 설정 (옵션)**

**7) 리소스 추가** 에서 **Create this environment inside a VPC**를 체크 한 경우VPC Configuration 을 설정 할 수 있다.

앞서 설명했듯이VPC를 사용하면 AWS의 확장 가능한 인프라를 이용할 수 있다는 이점이 있다.

또한 Amazon VPC를 사용하여 자체 가상 네트워크에 DB 인스턴스를 격리하고, 업계 표준의 암호화된 IPsec VPN을 사용하여 기존 IT 인프라에 연결할 수 있다.



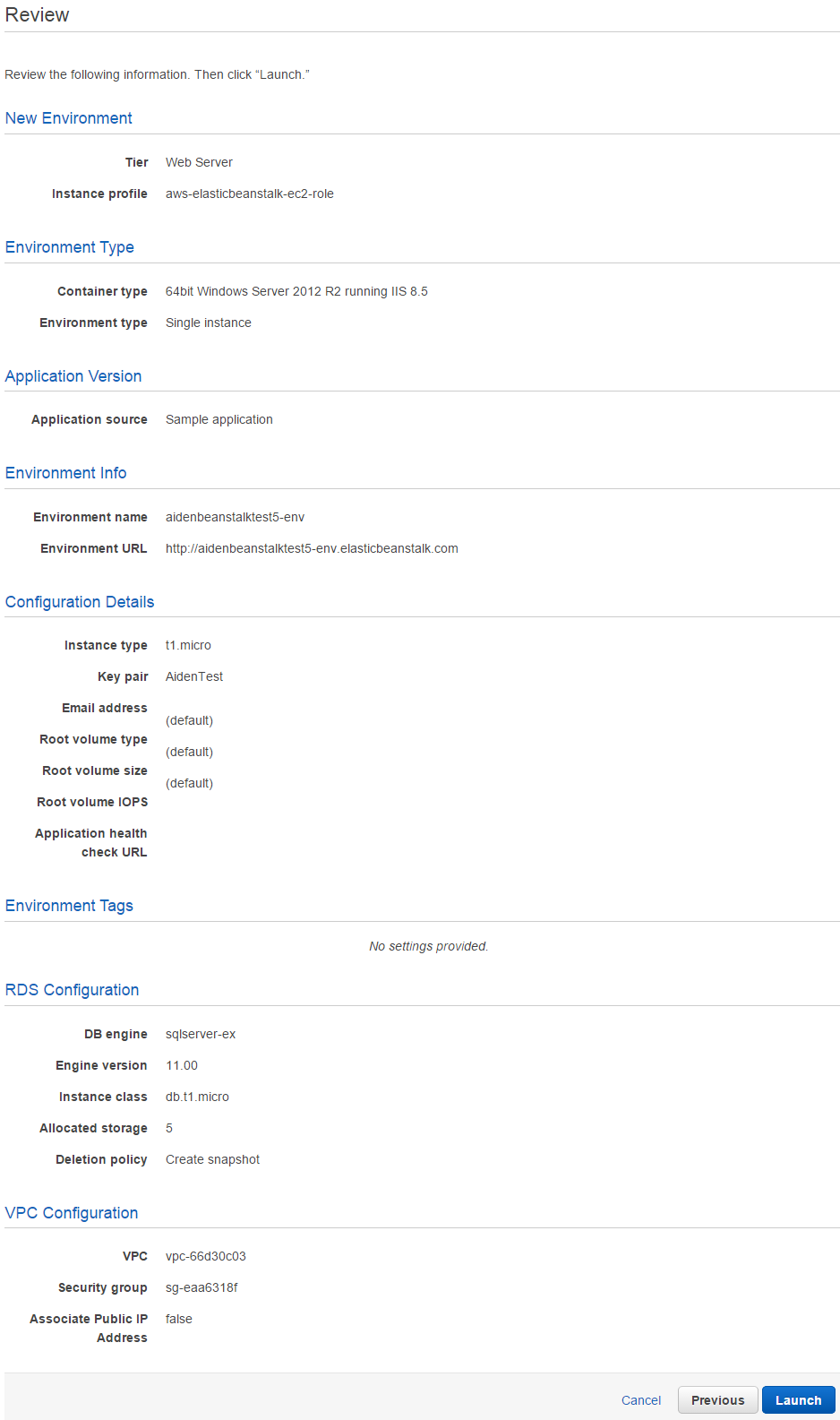
[그림 2-11] VPC Configuration

Amazon RDS 다중 AZ 배포는 데이터베이스(DB) 인스턴스에 개선된 가용성 및 내구성을 제공하므로, 프로덕션 데이터베이스 워크로드에 적합하다. 다중 AZ DB 인스턴스를 프로비저닝하면 Amazon RDS는 기본 DB 인스턴스를 자동 생성하고 동시에 다른 가용 영역(AZ)에 있는 예비 인스턴스에 데이터를 복제한다. 각 AZ는 물리적으로 분리된 자체 독립 인프라에서 실행되며 높은 안정성을 제공하도록 설계 되어 있으며, 인프라 장애가 발생한 경우(예: 인스턴스 충돌, 스토리지 장애 또는 네트워크 중단) Amazon RDS가 자동으로 failover를 수행하므로 failover가 완료된 후 데이터베이스 작동을 다시 시작할 수 있다. 그렇기 때문에 EC2 와 RDS를 선택할 때, 서로 다른 AZ의 Subnets 으로 선택 하여 준다

.

**12) Review**

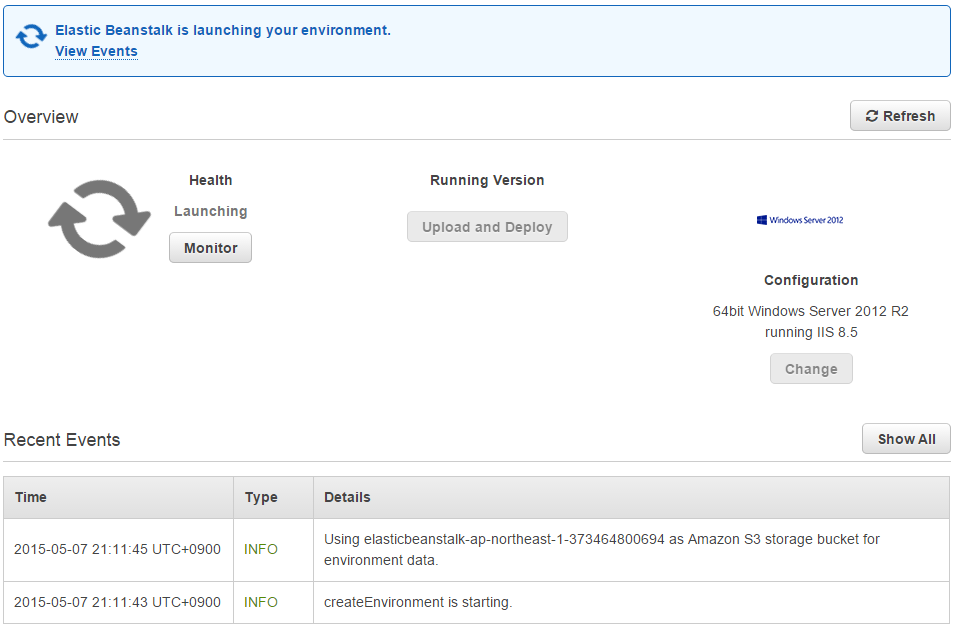
모든 설정이 끝나면 최종적으로 **Review** 화면이 보여지게 되는데, 확인 후 Launch를 누르면 관련 리소스가 생성이 된다.



[그림 2-12] Review

해당 인스턴스 및 리소스 생성이 완료 될 때까지 기다린다.

( 짧게는 몇 분부터 길게는 몇 십 분이 걸리는 시간이 소요된다. )



[그림 2-12] Elastic Beanstalk Dashboard



정상적인 Application 생성 비 정상적인 Application 생성

**Elastic Beanstalk > All Applications** 항목에 정상적으로 (초록색) Application이 생성되면 생성이 완료 된 것이다.

**3. Elastic Beanstalk 기능**

생성 완료 후, 버전 롤백, CloudWatch 모니터링, 이메일 알림 등 Beanstalk 기능들을 사용할 수 있으며 환경 설정도 변경할 수 있다.

**3.1 Actions**

**Elastic Beanstalk** 항목에 들어가 보면, 현재까지 생성 된 모든 Application들의 정보를 볼 수 있는데 **Actions Drill Down** 버튼을 통해, 해당 Application에 Web Server, Worker를 추가 하고, Application 삭제/ 롤백 작업등도 할 수 있다.



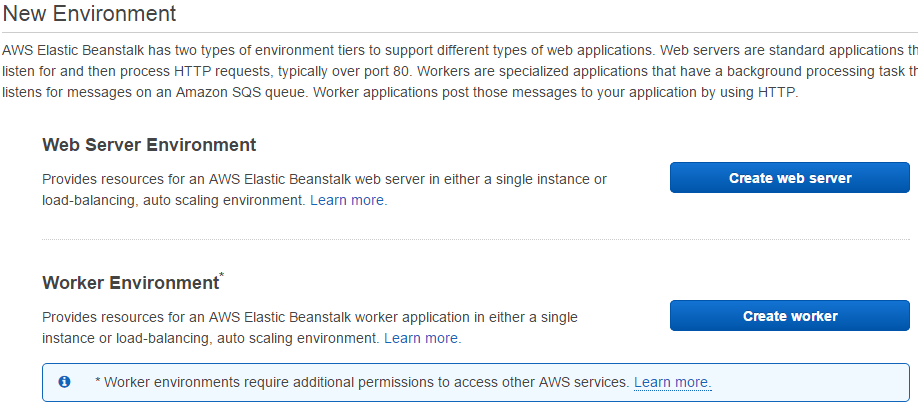
[그림 3-1] Elastic Beanstalk Actions

**1) Launch New Environment**

진행 과정은 위에서 다루었던

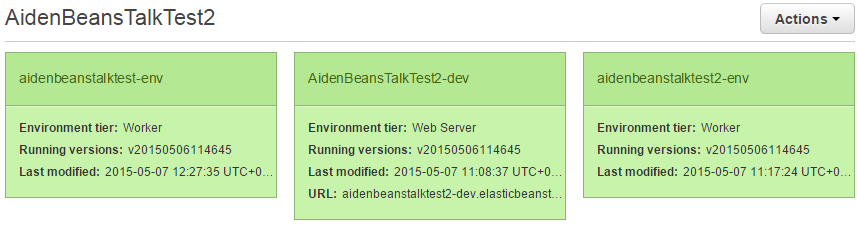
**Amazon Web Services에서 Beanstalk 사용하기 > 2. 새로운 Application 생성**

에서 했던 방식과 동일 하며 새로운 Web Server 와 Worker 을 추가할 수 있다.



[그림 3-1-1] New Environment

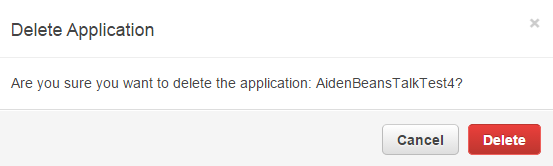
정상적으로 진행이 완료 되었다면, 해당 Application에 여러 개의 인스턴스가 생성된 것을 확인할 수 있을 것이다.



[그림 3-1] Applications

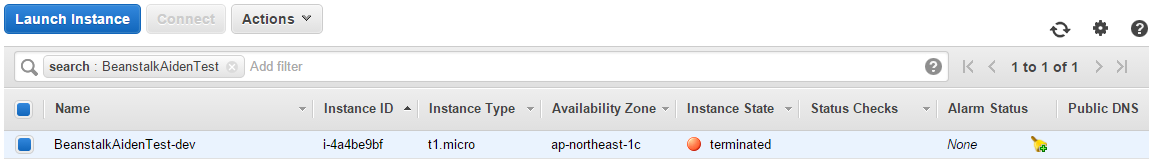
**2) Delete Application**

더 이상 사용하지 않거나, 생성과정에서 비 정상적으로 생성되었다면, Delete Application을 이용하여 삭제를 할 수 있다.



[그림 3-1-2] Delete Application

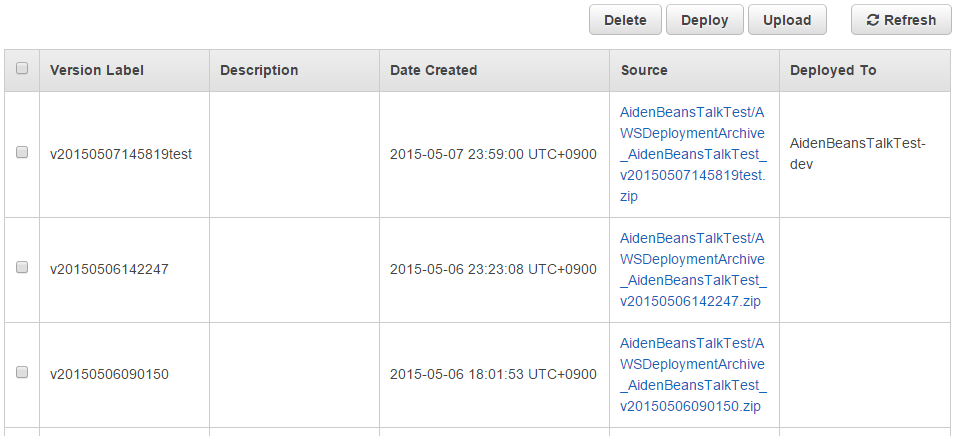
정상적으로 삭제 되었다면 해당 Application의 인스턴스 상태는 **terminated** 가 되고, 시간이 지나면 자동으로 삭제 된다.



[그림 3-1-2] Instance State (terminated)

**3) View Application Versions**

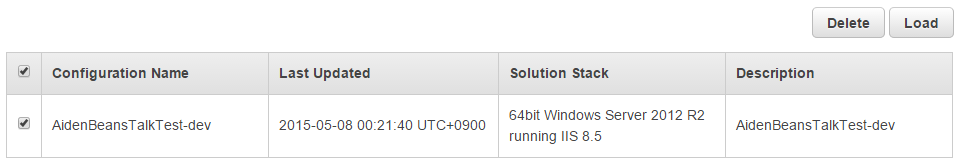
View Application Versions 에서는 사용자가 Deploy 한 버전을 시간 순으로 관리하고 있으며, 하나의 Version Label 을 선택해서 Deploy를 하면 해당 버전으로 롤백을 할 수 있다.



[그림 3-1-3] Application Versions

**4) View Saved Configurations**

현재 Application 에 설정되어 있는 환경 설정을 저장할 수도 있고, 다시 불러올 수도 있다. 단 물리적 환경 설정이 변경 되었을 경우에는, 기존 인스턴스를 삭제 하고, 다시 인스턴스를 생성하기 때문에 시간이 좀 더 걸린다.



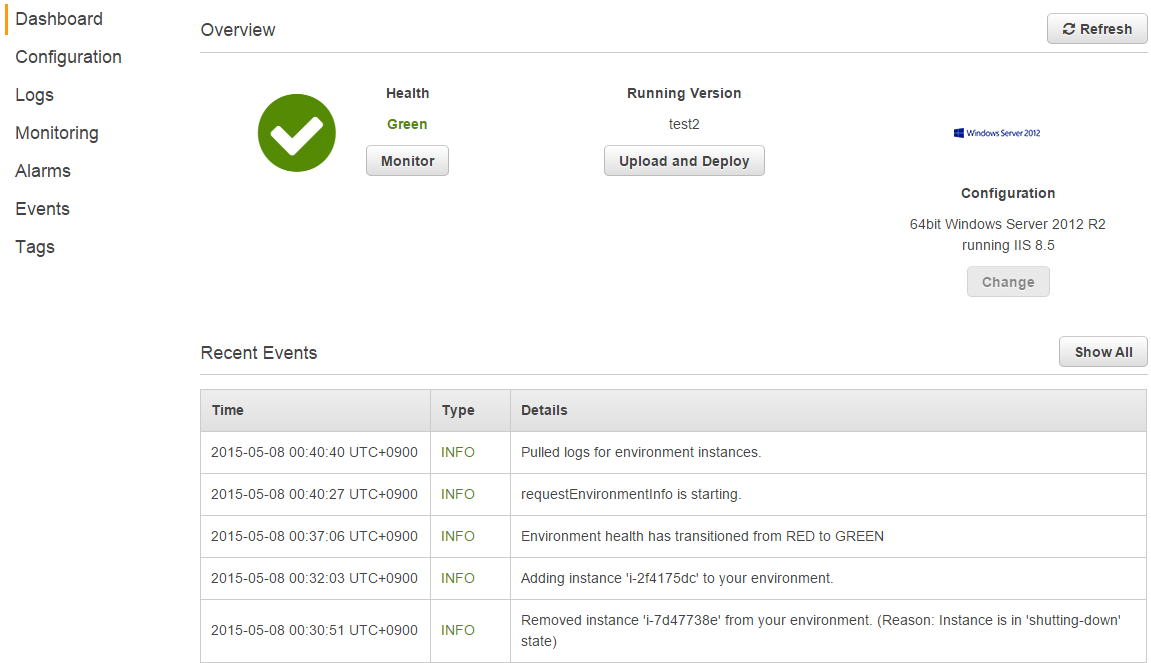
[그림 3-1-4] Saved Configurations

**3.2 Side Menu**

Application 내부로 들어 가면 Dashboard, Configuration, Monitoring 등을 확인할 수 있는데, 주요 기능들만 살펴 보겠다.

**1) Dashboard**

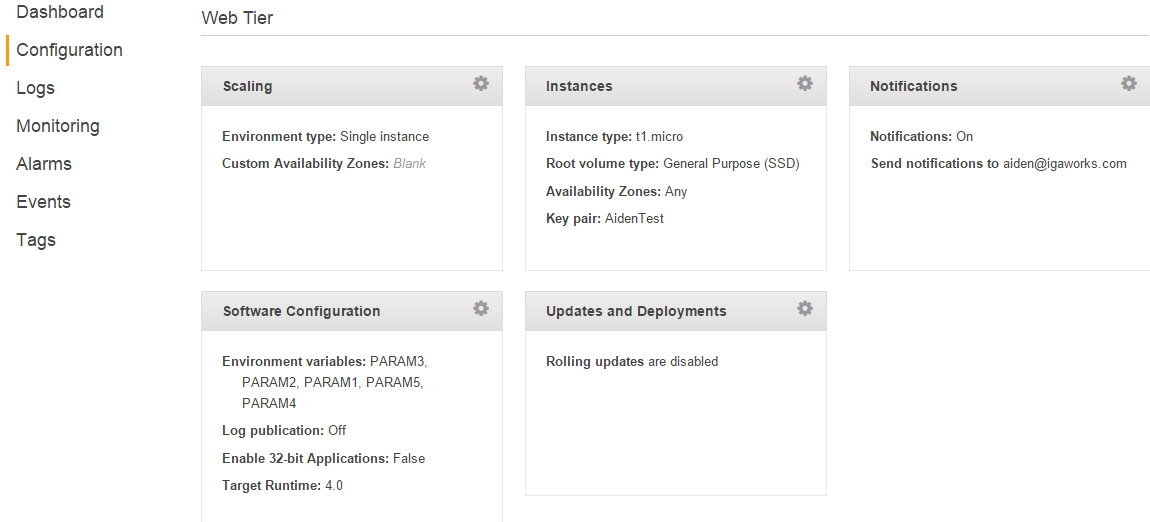
Dashboard 에서는 현재 Application 상태나, Deploy 된 버전, 그리고 Event Timeline 을 한눈에 볼 수 있다. 특히 Event Timeline 은 환경설정을 변경하거나 새로운 Application을 Deploy 할 때, 경과 상태를 알 수 있기 때문에 중요하게 사용된다.



[그림 3-2-1] Dashboard

**2) Configuration**

앞 서 Application을 생성할 때, 셋팅 했던 내용들을 확인/변경 할 수 있다.



[그림 3-2-2] Configuration

**Scaling** Environment type (Single instance, Load balancing)을 변경하거나 Availability Zone을 옮길 경우 사용한다.

**Instances** Instance type, 권한, EC2 security groups, Volume Type, Volume Size 등을 변경할 경우 사용한다.

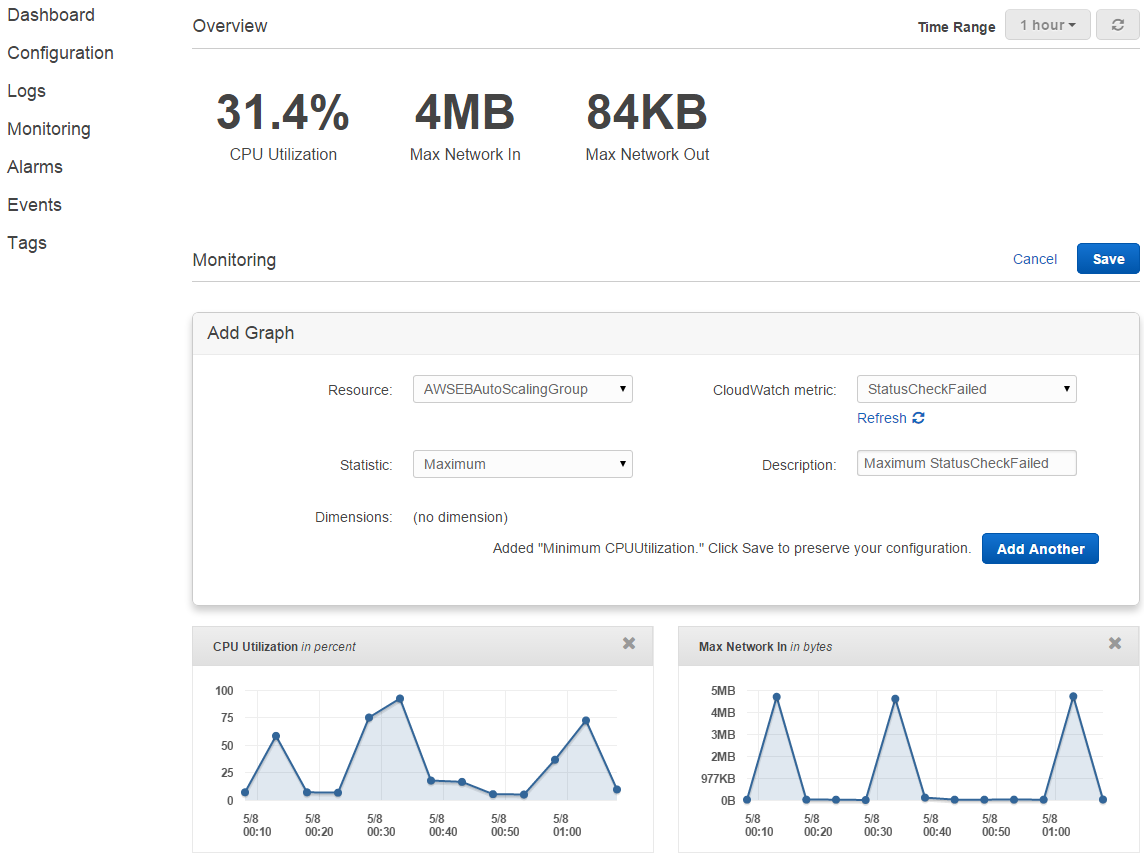
**Notifications** Email 을 등록 하면, Application의 상태가 변경될 때마다, 알림을 보내준다.

**Software Configuration** .NET, JAVA 버전, 32Bit application 사용 여부, Web Server 환경 변수 등을 설정 할 때 사용한다.

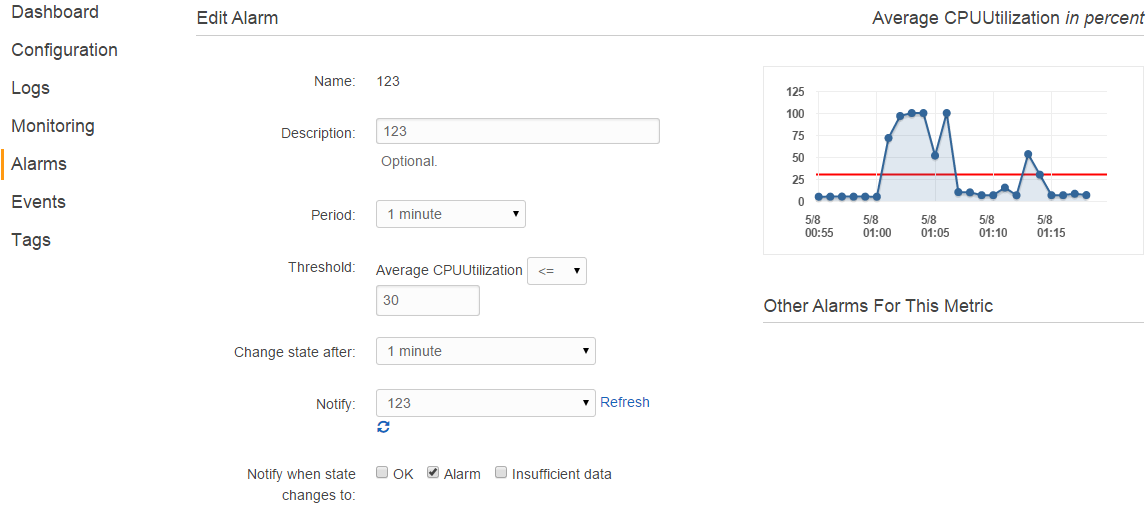
**Updates and Deployments ( 좀더 확인 필요 )**

**3) Monitoring / Alarms**

Dashboard 에서는 현재 Application에서 사용하고 있는 CPU 사용량, 디스크 사용량, 네트워크 사용량 등을 확인 할 수 있으며, 해당 항목 별로 임계치를 설정해서 임계치 초과 시, 알람을 받을 수 있다.

****

[그림 3-2-3] Monitoring



[그림 3-2-3] Edit Alarm

**IDE(Visual Studio)에서 Beanstalk 사용하기**

Elastic Beanstalk는 Eclise나 Visual Studio 같은 IDE 환경에서도 툴킷을 설치해서 사용할 수 있다. 여기서는 Visual Studio 기준으로 Beanstalk를 다루어 보겠다.

**1.AWS Toolkit for Visual Studio 설치**

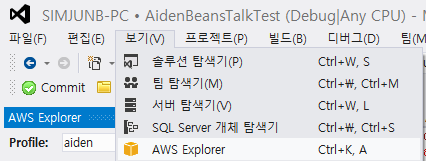
<http://aws.amazon.com/ko/visualstudio> 해당 경로 가면 AWS 관련 Toolkit 을 다운로드 받을 수 있다. 해당 파일을 다운 후, 설치해 준다.



[그림 1] AWS Toolkit for Visual Studio

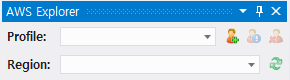
**2. AWS Toolkit for Visual Studio 설치**

정상적으로 설치가 완료 되면, Visual Studio의 **보기** 메뉴에 **AWS Explorer** 라는 항목이 생긴다.



[그림 2] AWS Explorer

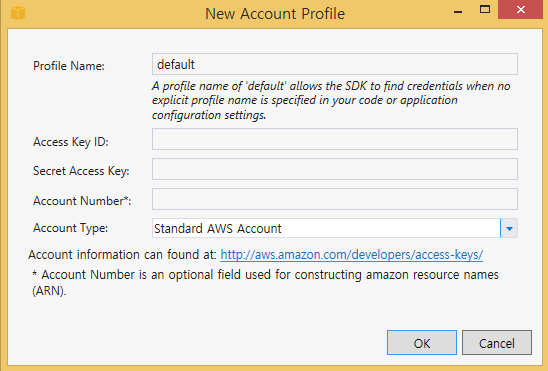
**AWS Explorer**를 선택 하면 다음과 같은 창이 활성화 된다.



[그림 2] AWS Explorer

**3. 사용자 Profile 등록**

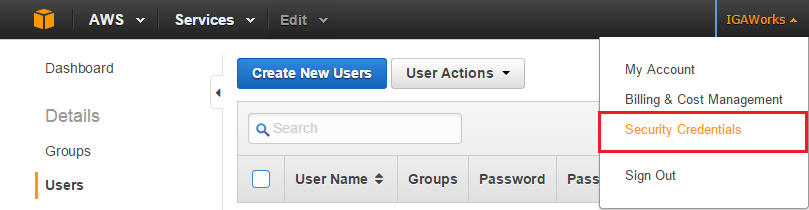
활성화 된 AWS Explorer 창에서 아이콘을 클릭하면, New Account Profile 에 대한 설정 창이 나온다.



[그림 3] New Account Profile

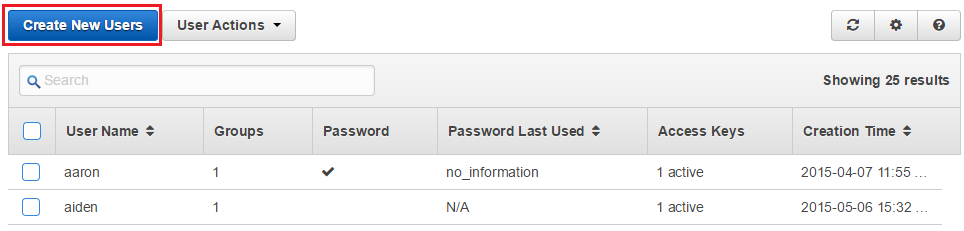
기존에 사용하고 있는 Profile 정보가 있으면, 해당 정보를 입력 하면 되고, 없다면 새로운 유저를 생성 후 등록하면 된다. 여기서는 유저를 새로 생성해서 등록해 보겠다.

AWS Management Console 에 접속 후, **Security Credentials** 항목을 클릭한다.



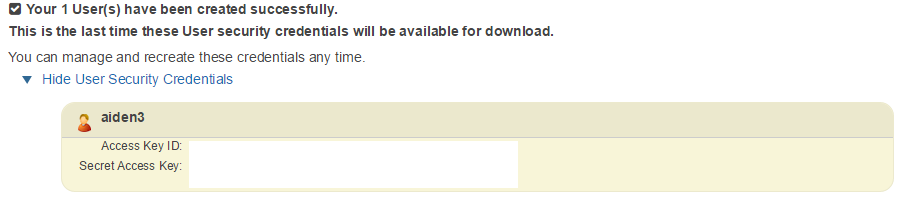
[그림 3] Security Credentials

Security Credentials 클릭하면, 해당 Users 정보가 나오는데 **Create New Users** 버튼을 통해서 새로운 유저 생성을 하면 된다.



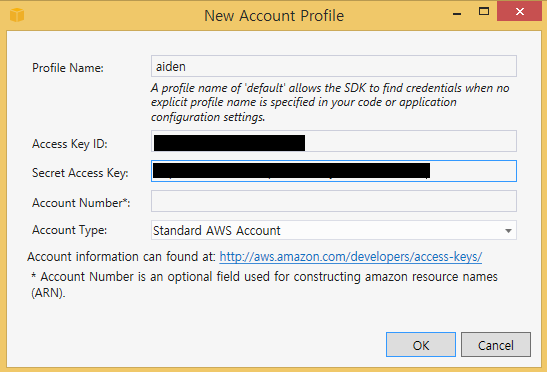
[그림 3] Security Credentials

정상적으로 유저가 생성 되면, 해당 Access Key ID와 Secret Access Key 가 발급된다.



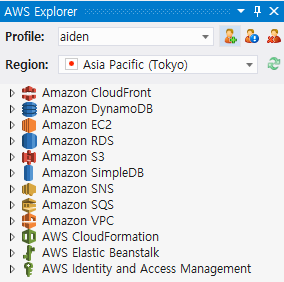
[그림 3] Profile Info

발급된 정보를 New Account Profile 에 입력 후 OK를 누른다.



[그림 3] New Account Profile

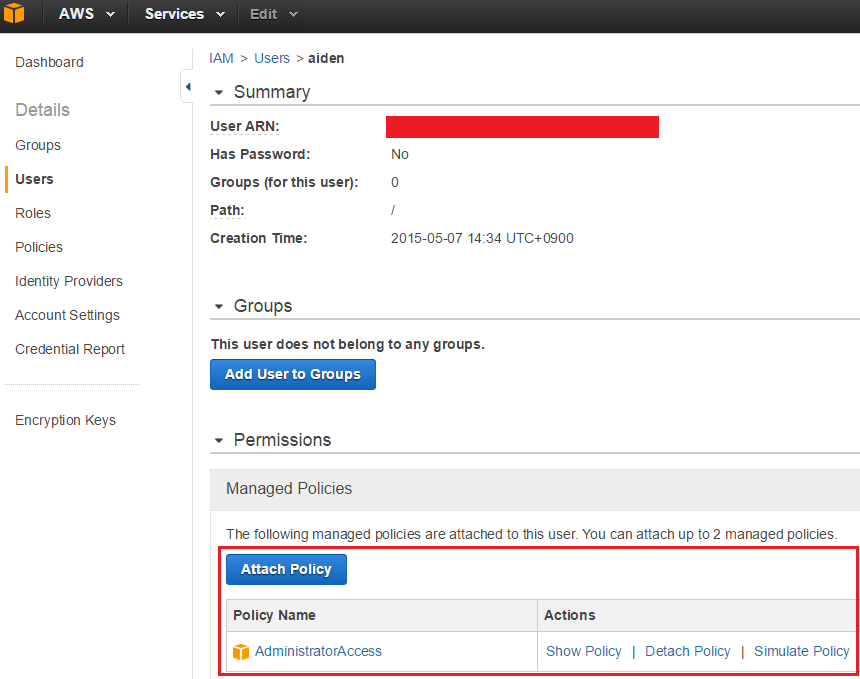
정상적으로 등록 되었다면, **AWS Explorer** 창에서 다음과 같이 Region 정보와 Amazon 리소스 들이 나타날 것이다.



[그림 3] AWS Explorer

한 가지 주의할 점은 생성 된, User에 권한이 없으면 정상적으로 Application을 생성할 수 없기 때문에, **Security Credentials > Users > 유저선택** 에서 **Attach Policy** 를 선택해서

**AdministratorAccess** 권한을 주어야 한다.

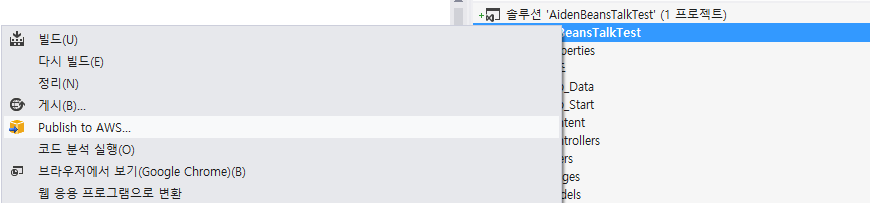


[그림 3] Attach Policy

**4. IDE를 이용한 새로운 Application 생성 및 배포**

**1) Publish to AWS**

배포하고자 하는 프로젝트에서 오른쪽 마우스를 클릭해서 **Publish to AWS**를 선택한다.

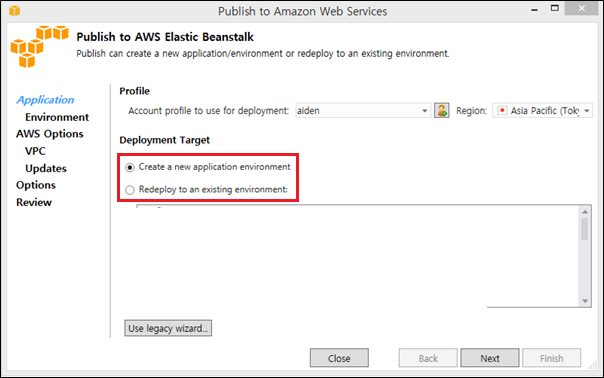


[그림 4-1] Publish to AWS

**2) Publish to AWS Elastic Beanstalk**

새로운 Application 환경을 만들려면 **Create a new application environment** 를 선택하고, 기존 Application 환경에 Deploy 하려면 **Redeploy to an existing environment** 를 선택하면 된다.

여기서는 Create a new application environment를 이용하여 진행을 하겠다.



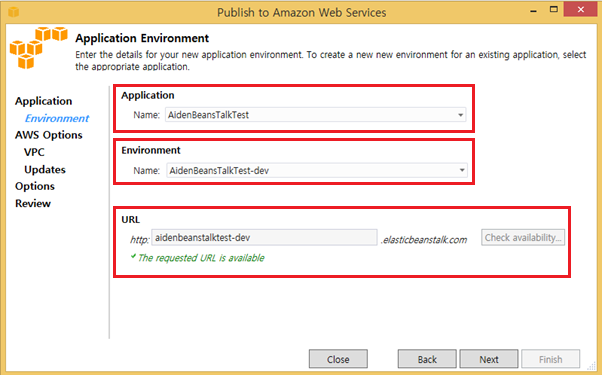
[그림 4-2] Publish to Amazon Web Service

**3) Application Environment**

**Application Name** 은 해당 Application 이름으로 사용되며, 100 자 이내 입력 해야 된다.

**Environment Name** 은 EC2 인스턴스를 생성할, 인스턴스 이름으로 사용되며, 4 자에서 23 사이의 값으로 중복되지 않은 값으로 입력 해야 된다.

**URL** 은 Deploy 완료 후에, 해당 Application에 유니크 한, 도메인이 되며, 중복되지 않은 값으로 입력 해야 된다.



[그림 4-3] Application Environment

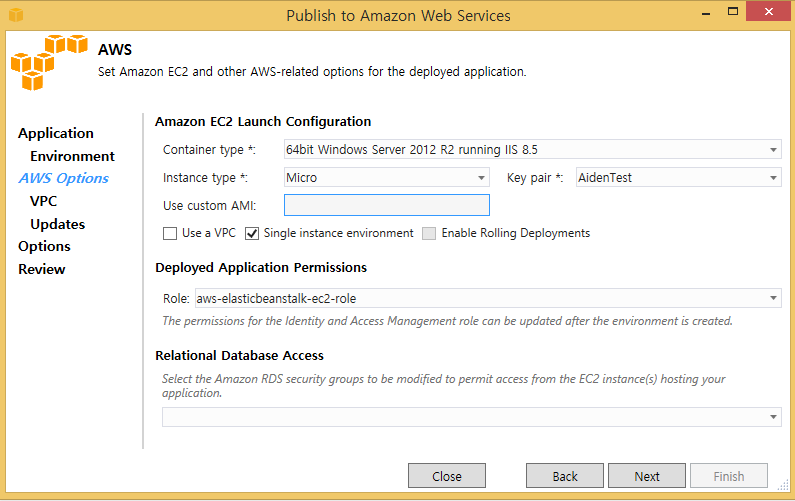
**4) AWS Options**

Amazon EC2 인스턴스 와 관련 리소스 옵션을 설정 한다.

**Container type** 은 자신이 사용할 OS 종류와 IIS 버전에 맞게 선택한다.

**Instance type** 은 CPU, 메모리, 스토리지 및 네트워킹 용량의 다양한 조합으로 구성되며, Application에 따라 적합한 리소스 조합을 선택할 수 있는 유연성을 제공한다. 자신의 Application 성능에 따라서 사양을 늘리거나 줄여가면서 성능을 측정하는 것이 좋다.

참고 자료 (Instance type) : <http://aws.amazon.com/ko/ec2/instance-types>



[그림 4-4] AW Options

**key pair** 는 해당 인스턴스에 접근하기 위한 정보로 기존에 있는 EC2 Key를 선택 하거나 새로 생성할 수 있다. 새로 생성 시에는 **EC2 Dashboard > Resources (Key Pairs) > Create Key Pair** 에서 생성할 수 있으며, 생성 후 해당 PEM(Base64로 인코딩 된 인증서) 파일이 다운된다. 이 인증서를 가지고 SSH, RDP 등으로 생성 된, 인스턴스에 접근할 수 있다.

**Use a VPC** 는 VPC를 사용할 경우 체크를 해 주면 된다.

**Single instance environment** 를 체크해 주면, **Singel instance**로 해제 하면 **Load balancing, auto scaling** 모드로 설정된다.

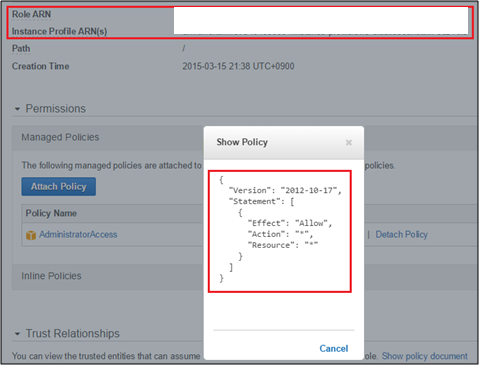
**Enable Rolling Deployments** (좀 더 확인 필요)

**Deployed Application Permissions Role**

생각하면 Application에서 리소스 접근할 수 있는 권한을 설정해 준다고 생각하면 될 것 같다. 해당 권한에 대해 몇 개만 살펴 보겠다.

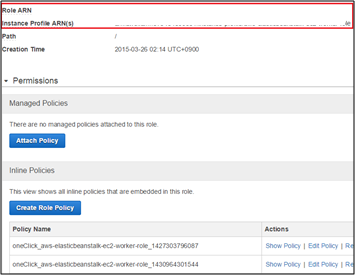
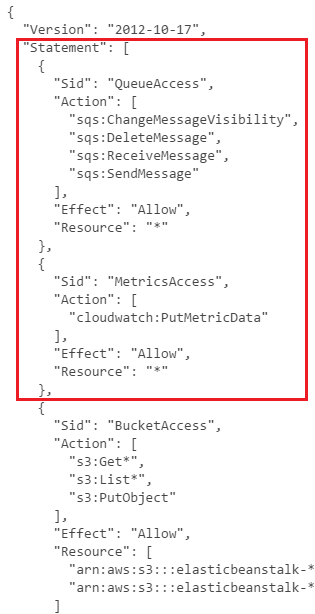
**aws-elasticbeanstalk-ec2-role (default)** 권한을 세부적으로 살펴 보면 Policy Name에

AdministratorAccess가 설정되어 있으며 해당 권한은 전체 리소스에 대해서 자유롭게 사용할 수 있도록 되어 있다.



[그림 4-4] aws-elasticbeanstalk-ec2-role

반면 **aws-elasticbeanstalk-ec2-worker-role** 에는 용도에 맞도록 세부 적인 권한들이 설정 되어 있는 것을 볼 수 있다.

[그림 4-4] aws-elasticbeanstalk-ec2-worker-role

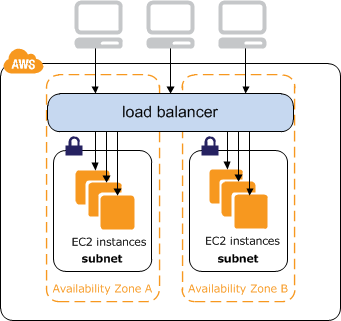
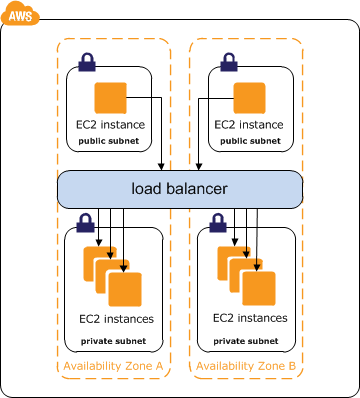
**Relational Database Access** RDS sercurity group을 선택해 준다.

**5) VPC Options (옵션)**

**4) AWS Options** 에서 **Use a VPC Check Box**를 선택 했다면 다음 항목을 설정할 수 있다.

**VPC** \*: 자신이 사용하고자 하는 VPC를 선택 해 준다.

**ELB Scheme** \* : Public 는 외부에 노출 된 형태 이고, Internal 은 내부를 통해 접근할 수 있는 형태 이다.

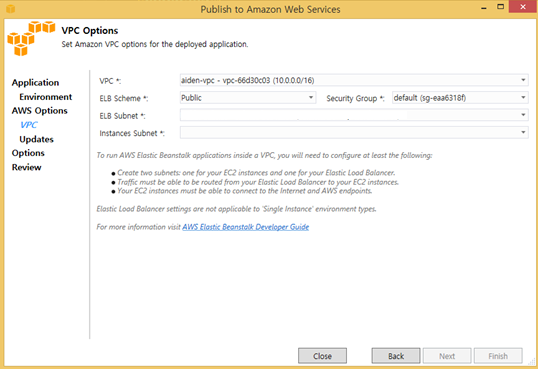
 

Public Internal

**Security Group** \*: 해당 VPC가 포함 될, Security Group 을 선택해 준다.

**Elb Subnet** \*: Load Balance 가 위치할 Subnet 을 지정해 준다.

**Instances Subnet** \*: Instances 가 위치할 Subnet 을 지정해 준다.

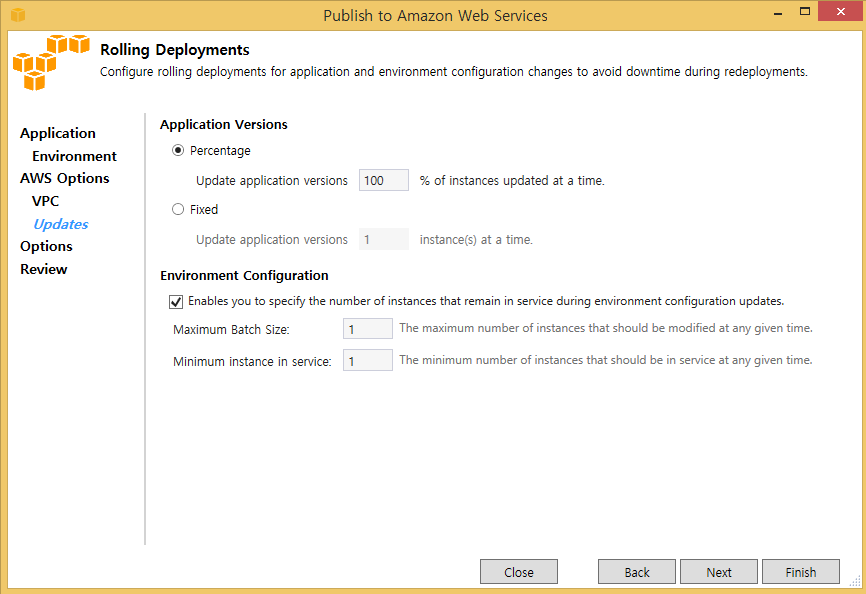


[그림 4-5] VPC Options

참고 자료 (VPC) :

http://docs.aws.amazon.com/ko\_kr/AmazonVPC/latest/GettingStartedGuide/ExerciseOverview.html

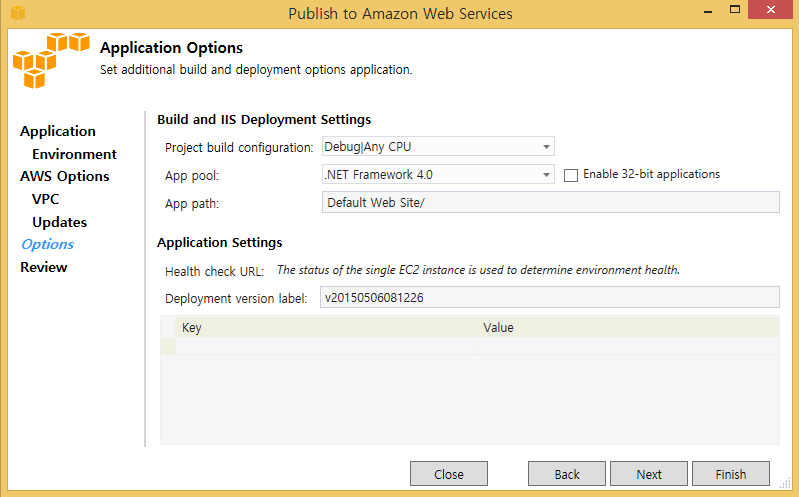
**6) Rolling Deployments** (해당 부분 좀 더 확인 필요)



[그림 4-6] Rolling Deployments

**7) Application Options**

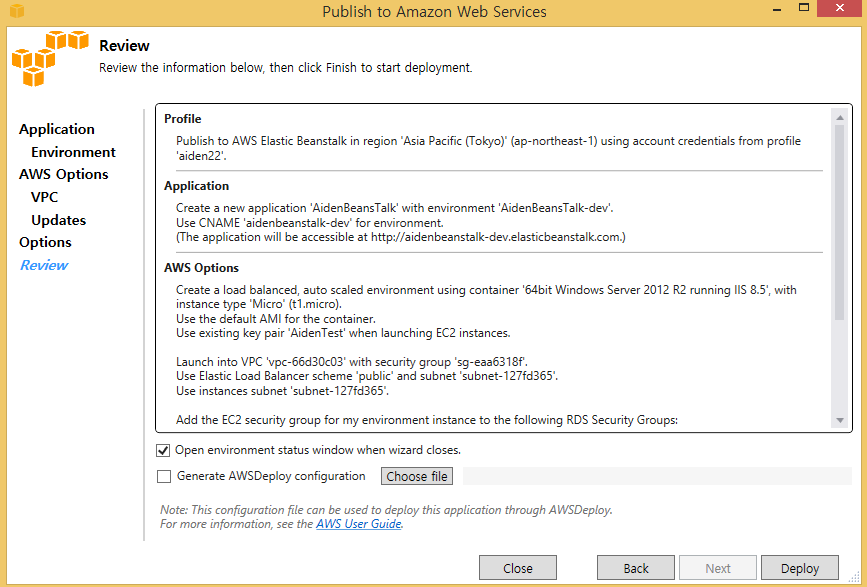
Application Options 에서는 해당 .NET Framework 버전이나 32-bit Application 사용 여부, Deployment version, Web Server 환경 변수 등을 설정 할 때 사용한다.



[그림 4-7] Application Options

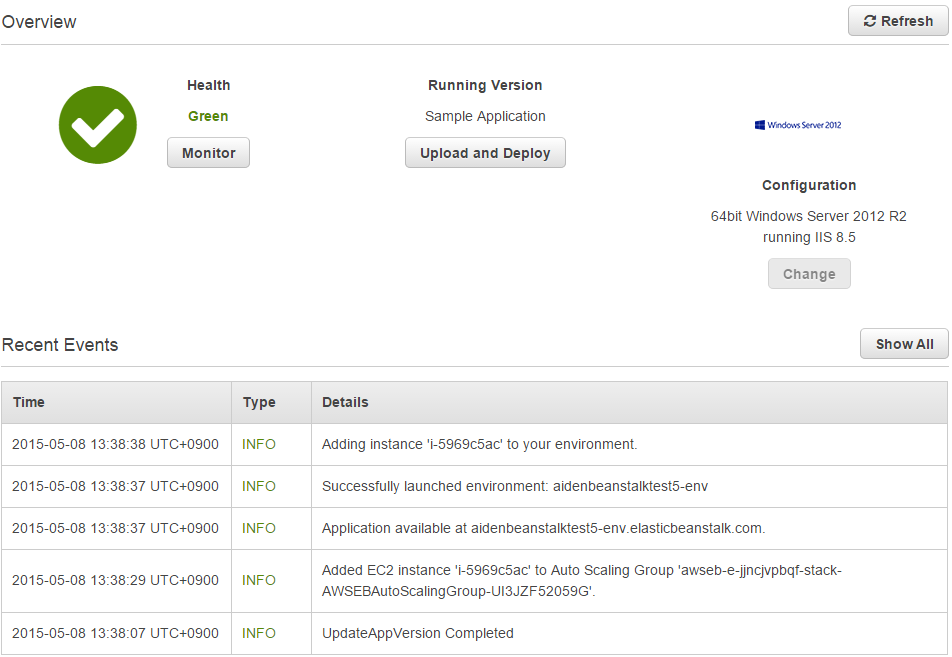
**8) Review**

모든 설정이 끝나면 최종적으로 Review 화면이 보여지게 되는데, 확인 후 Deploy를 누르면 관련 리소스가 생성이 된다.



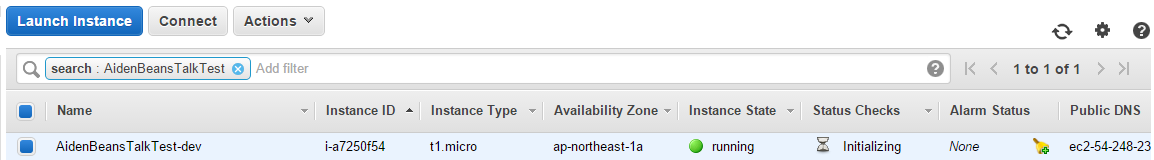
[그림 4-8] Review

**Elastic Beanstalk Dashboard**를 모니터링 하면서 정상적으로 배포되나 확인화면 된다.



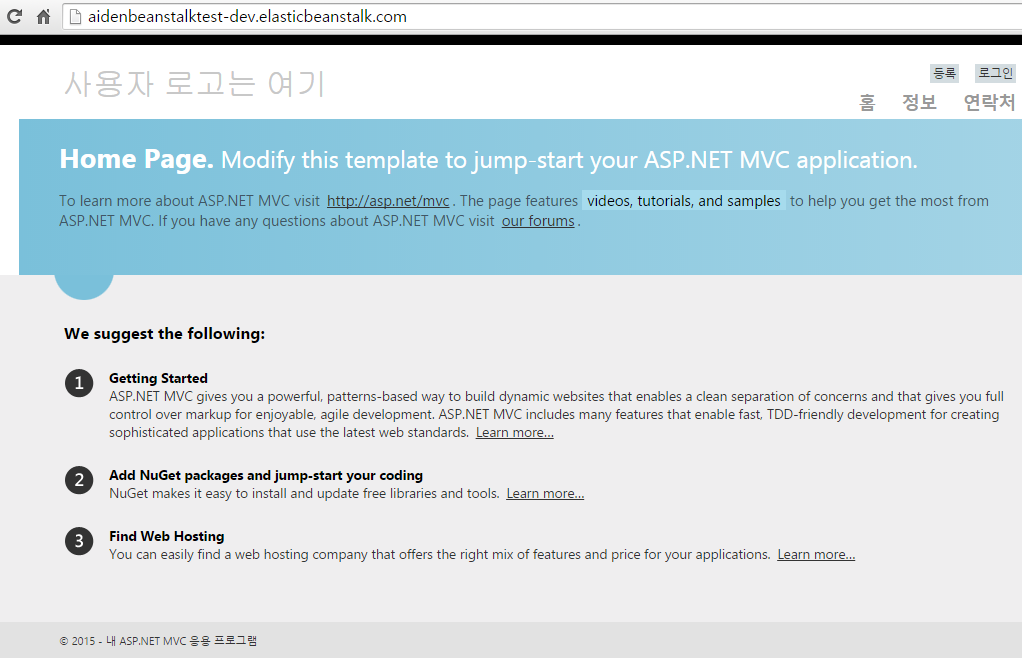
[그림 4-8] Elastic Beanstalk Dashboard

해당 Application의 인스턴스도 정상적으로 생성되었나 확인해 본다.



[그림 4-8] Instance State (running)

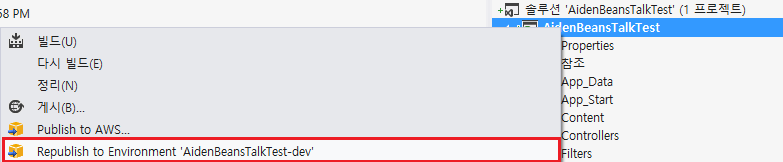
정상적으로 배포 후, 해당 URL로 접속되면 Deploy 된, Application 화면을 볼 수 있다.



[그림 4-8] Home Page

**9) Re-publish to AWS Elastic Beanstalk**

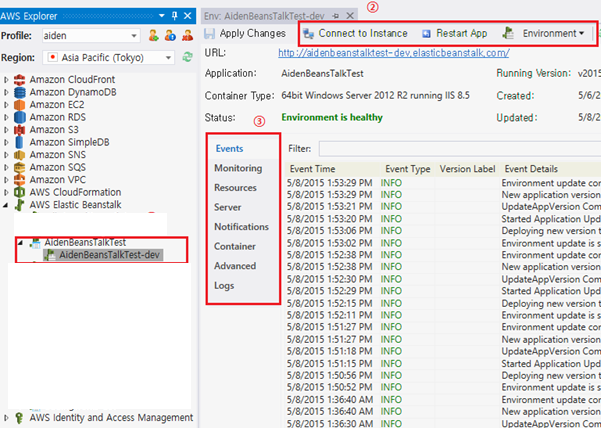
정상적으로 Deploy 완료 후, 프로젝트를 재 배포를 할 경우에는 프로젝트에서 오른쪽 마우스를 클릭해서 Republish to Environment ‘Application 명’ 을 선택 하면 된다.



[그림 4-9] Re-publish

**5. Elastic Beanstalk 기능**

Visual Studio 에서도 AWS Explorer를 통해 여러 가지 Elastic Beanstalk 기능들을 사용할 수 있다.



[그림 5] Elastic Beanstalk 기능들

1) AWS Explorer 에서 생성한 Application의 인스턴스를 클릭하면 왼쪽에 보이는 View Status 창이 활성화 된다.

2) 해당 인스턴스 (Web Server, Worker) 를 원격으로 접속하거나, 인스턴스 재 시작, Type 변경을 할 수 있다.

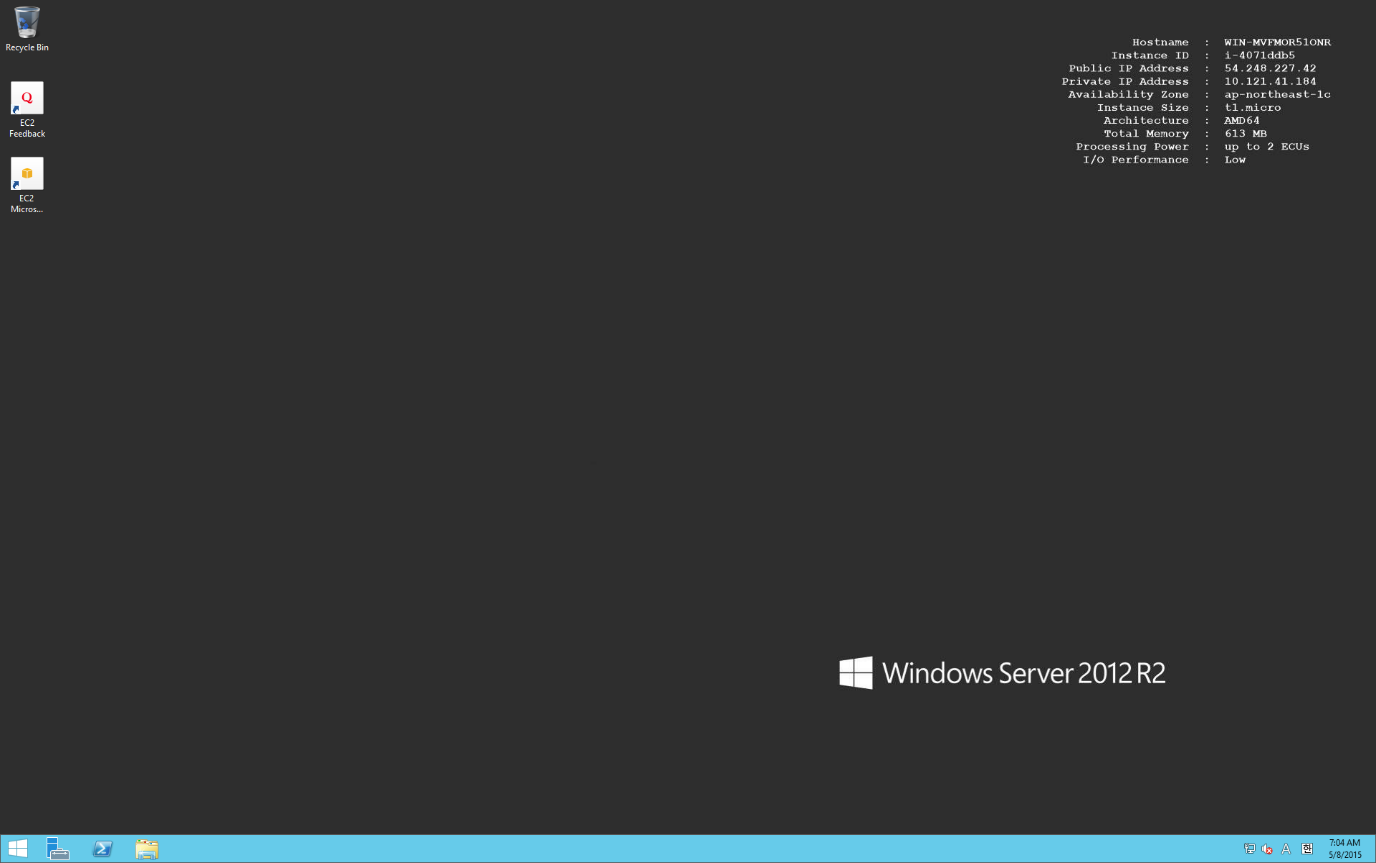
3) 해당 인스턴스의 모니터링, 환경설정 변경, 알람 설정, Events 메시지 등을 볼 수 있다.

**5.1 View Status Tab**



**1) Connect to Instance**

해당 PC에서 설정한 Key Pair와 Application에서 사용 중인 key Pair 가 동일하면 별도의 인증을 거치지 않고, 해당 인스턴스로 접속할 수 있다. 만약 동일하지 않다면, Private Key를 가지고 인증을 하면 된다.



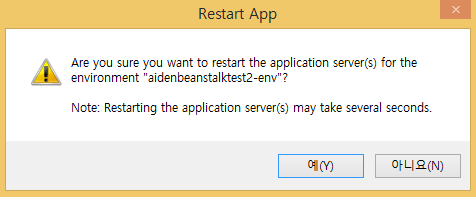
[그림 5-1-1] RDP를 이용한 Windows 원격



[그림 5-1-1] putty를 이용한 Linux 원격

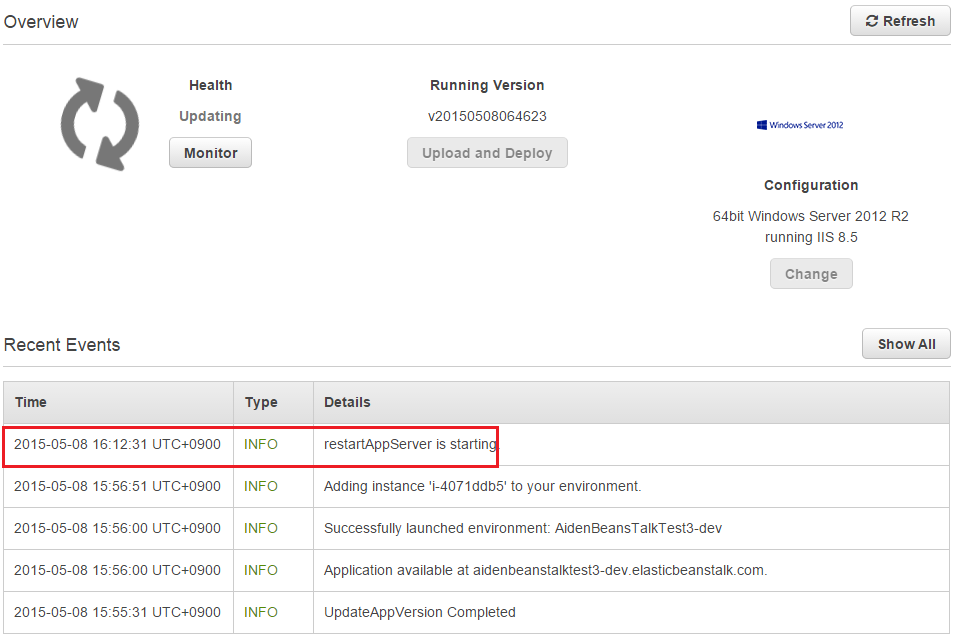
**2) Restart App**

AWS Management Console을 통하지 않고도 IDE 에서 직접 해당 Application을 재 기동 할 수 있다.



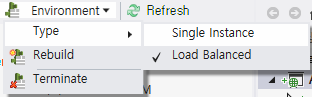
[그림 5-1-2] Restart App

**Elastic Beanstalk Dashboard** 에서 확인해 보면, **restartAppServer is starting** 이 되었다는 것을 알 수 있다.



[그림 5-1-2] Recent Events

**3) Environment**



[그림 5-1-3] Environment

**Type** 해당 인스턴스의 Type을 Single Instance 나 Load Balanced로 변경할 수 있다.

**Rebuild** 해당 인스턴스를 다시 생성할 수 있다. (변경된 환경 변수 재 적용)

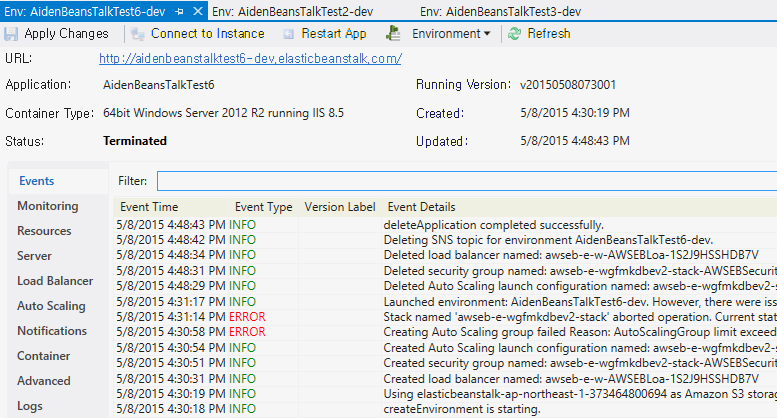
**Terminate** 해당 인스턴스를 삭제 할 수 있다.

**5.2 View Status Side Menu**

**1) Events**

새로운 Application 생성 상태나, 프로젝트 Deploy 상태 등을 확인 할 수 있다.

**AWS Management Console** 의 **Elastic Beanstalk Dashboard** 상에 Recent Events 와 동일한 메시지를 출력한다.

****

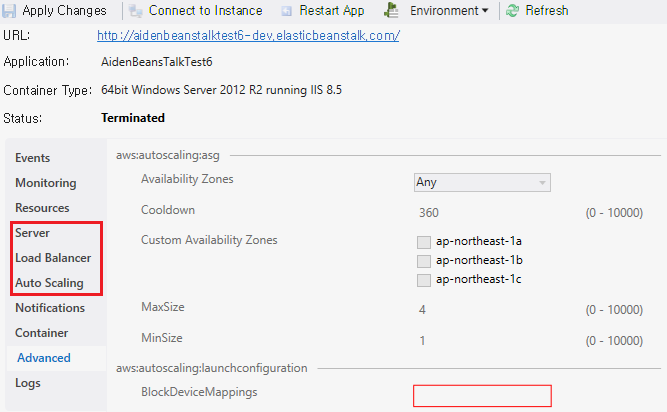
[그림 5-2-1] Restart App

**2) Monitoring**

**AWS Management Console** 의 **Monitoring** 과 성격이 같다.

**3) Server, Load Balancer, Auto Scaling**

Application 생성 시, 설정했던 부분을 변경 후, 재 적용할 수 있다. 물리적인 부분이 변경되면 인스턴스가 삭제 되고 다시 만들어 지기 때문에 시간이 좀 걸릴 수도 있다.

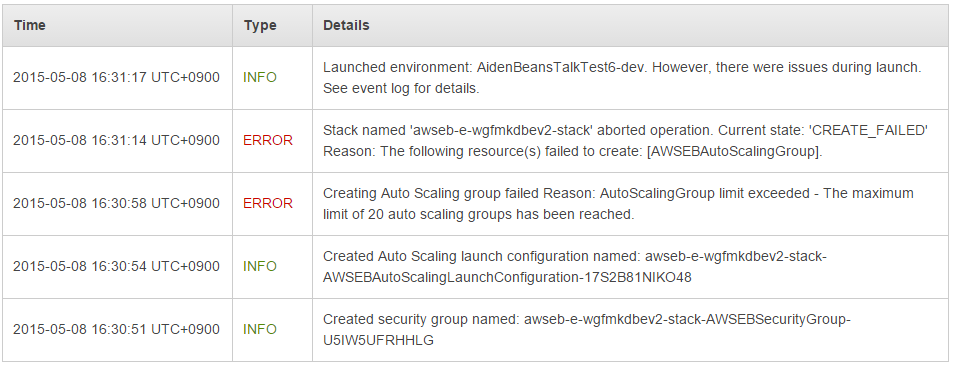
****

**[그림 5-2-3] Restart App**

**6. 주의사항**

Application을 생성할 때, Load balancing, auto scaling으로 생성할 때, ERROR가 뜨면서 실패 하는 경우가 있다. 현재 사내에서 사용하고 있는 AWS Load balancing, auto scaling 은

Default limit로 설정되어 있기 때문에, 개수가 이를 넘어가는 경우 발생한다. 차 후에 Load balancing, auto scaling 기능을 많이 사용하게 된다면, 해당 제한 개수를 늘려야 될 것 같다.



참고 자료(AWS Service Limits) :

http://docs.aws.amazon.com/general/latest/gr/aws\_service\_limits.html