












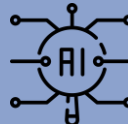



# AWS Rekognition Service 개념

## ➔ AWS의 딥러닝 기반 지능형 Service

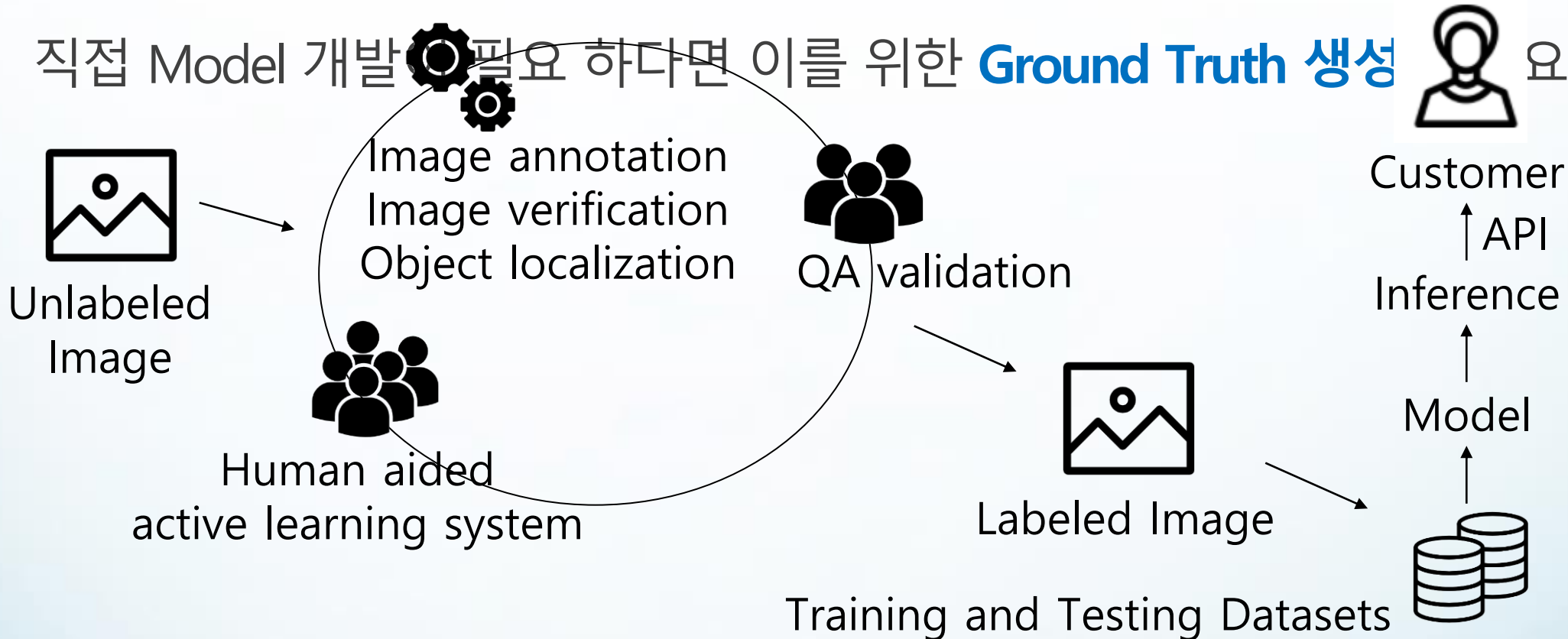
	AI Services	 Amazon Rekognition	 Amazon Polly (TTS)	 Amazon Lex (Chatbot)
	AI Platforms	 Amazon ML	 Amazon EMR (bigdata platform)	 Amazon Spark
	AI Frameworks	 Amazon Deep Learning AMI		
		 Apache MXNET	 Tensorflow	 TORCH
	AI Infra.	 Amazon GPUs, CPUs, Lambda, IoT, Greengrass		

# AWS Rekognition Service 개념

## ➔ AWS의 딥러닝 기반 지능형 Service

- Deep Learning 플랫폼을 직접 운용하는 데는 **많은 전문 인력과 인프라 자원**이 필요

- 직접 Model 개발  필요 하다면 이를 위한 **Ground Truth 생성**  요



# AWS Rekognition Service 개념

## ➡ AWS의 딥러닝 기반 지능형 Service

- Amazon의 컴퓨터 비전 과학자들이 Prime Photos에서 매일 수십억 개의 이미지들을 분석할 목적으로 개발하여 높은 성능이 검증
- **기계 학습 전문지식과 딥러닝 지식이 없이도 사용 가능한 Service**
- 완전 관리형, AWS Service 통합, 증명된 확장성, 안전성, 저렴한 비용

# AWS Rekognition Service 개념

## ➔ AWS의 딥러닝 기반 지능형 Service

### ● AWS Rekognition

- 이미지/동영상을 시각적 분석 기능을 쉽게 제공

#### Rekognition Image

- 수많은 이미지를 검색, 확인 및 구성 가능
- 객체, 장면 및 얼굴을 감지하는 이미지 인식 Service  
텍스트를 추출하고, 유명인사를 인식하며, 이미지에서 부적절한 콘텐츠를 식별 가능, 얼굴을 검색하고 비교 가능, 나이대도 예측 가능

#### Rekognition Video

- 저장된 동영상 또는 실시간 스트림 동영상에서 동작 기반 컨텍스트를 추출하고 이를 분석 가능
- 활동 탐지, 프레임 내 사람의 움직임 이해, aws s3에 저장된 동영상과 실시간 동영상 스트림에서 객체, 유명인사 및 부적절한 콘텐츠 인식

# AWS Rekognition Service 개념

## ➡ AWS의 딥러닝 기반 지능형 Service

### ● **Rekognition Image**

- JPEG 및 PNG 이미지 형식을 지원함. 이미지는 S3객체 또는 바이트 배열로 인식
- S3 객체로 전달 시 최대 15MB, 이미지 바이트 배열로 전달 시 최대 5MB의 이미지 파일 크기를 지원
- 분석할 수 있는 최소크기는 가로 또는 세로 화소가 최소 45픽셀이어야 함

# AWS Rekognition Service 개념

## ➡ AWS의 딥러닝 기반 지능형 Service

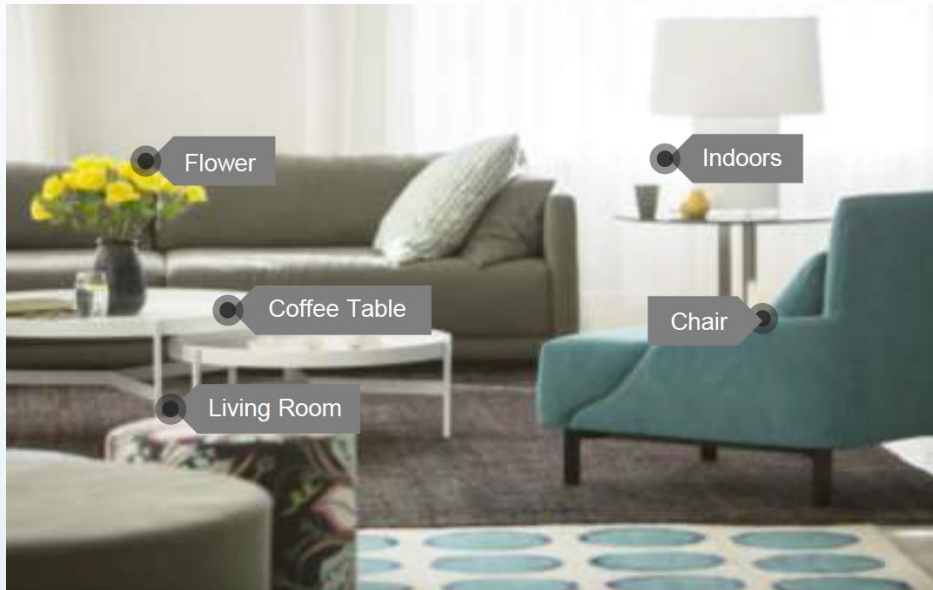
### ● **Rekognition Video**

- H.264 코덱을 사용하여 인코딩 해야 함. 지원되는 파일 형식은 MPEG-4 및 MOV 임. MPEG-4 및 MOV 형식 동영상 파일이 작동하지 않는 경우 동영상을 인코딩하는 데 사용된 코덱이 H.264인지 확인하고 사용
- S3 파일로 전달될 때 최대 8GB 파일과 최대 2시간 동영상을 지원

# AWS Rekognition Service 개념

## ➡ AWS의 딥러닝 기반 지능형 Service

- 객체 및 장면을 인식
  - 신뢰도 점수(confidence scores)와 함께 제공
- 객체 및 장면 인식을 이용하여 많은 양의 이미지 라이브러리로부터 검색, 필터링 등의 기능을 손쉽게 추가가능

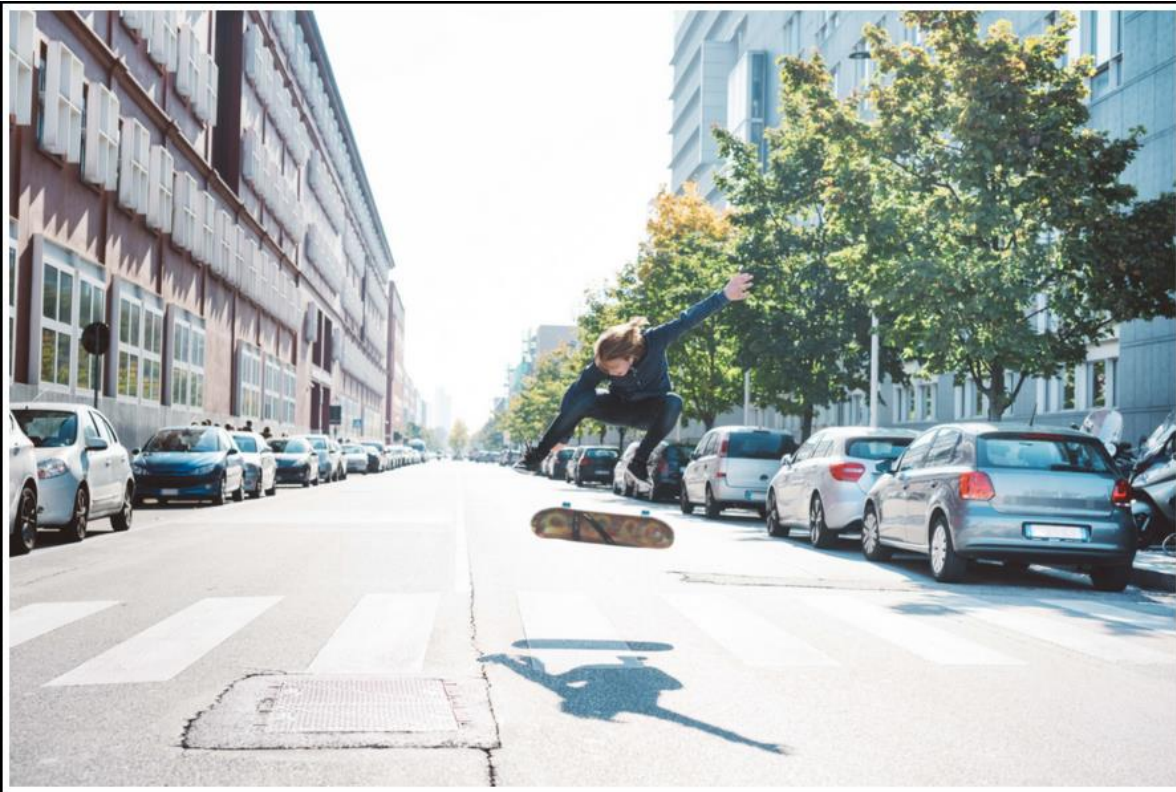




# AWS Rekognition Service 개념

## ➔ AWS의 딥러닝 기반 지능형 Service

- 객체 및 장면 인식



▼ Labels   Confidence	
Skateboard	99.2%
Sport	99.2%
People	99.2%
Person	99.2%
Human	99.2%
Parking	97.4%
<a href="#">Show more</a>	
▶ Request	
▶ Response	



# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API



DetectLabels

Image : base64-encoded bytes  
혹은 S3 objects

```
{  
  "Confidence": 94.62968444824219,  
  "Name": "adventure"  
},  
{  
  "Confidence": 94.62968444824219,  
  "Name": "boat"  
},  
{  
  "Confidence": 94.62968444824219,  
  "Name": "rafting"  
},
```

# AWS Rekognition Service API 활용 방법

## AWS Rekognition API

```
{  
  "contentString":{  
    "Attributes":[  
      "ALL"  
    ],  
    "Image":{  
      "S3Object":{  
        "Bucket":"console-sample-  
images",  
        "Name":"skateboard.jpg"  
      }  
    }  
  }  
}
```

Request

```
{ "Labels":[ { "Confidence":99.25359344482422, "Name":"Skateboard" }, {  
"Confidence":99.25359344482422, "Name":"Sport" }, { "Confidence":99.24723052978516,  
"Name":"People" }, { "Confidence":99.24723052978516, "Name":"Person" }, {  
"Confidence":99.23908233642578, "Name":"Human" }, { "Confidence":97.42484283447266,  
"Name":"Parking" }, { "Confidence":97.42484283447266, "Name":"Parking Lot" }, {  
"Confidence":91.53300476074219, "Name":"Automobile" }, {  
"Confidence":91.53300476074219, "Name":"Car" }, { "Confidence":91.53300476074219,  
"Name":"Vehicle" }, { "Confidence":76.85114288330078, "Name":"Intersection" }, {  
"Confidence":76.85114288330078, "Name":"Road" }, { "Confidence":76.21503448486328,  
"Name":"Boardwalk" }, { "Confidence":76.21503448486328, "Name":"Path" }, {  
"Confidence":76.21503448486328, "Name":"Pavement" }, {  
"Confidence":76.21503448486328, "Name":"Sidewalk" }, {  
"Confidence":76.21503448486328, "Name":"Walkway" }, {  
"Confidence":66.71541595458984, "Name":"Building" }, { "Confidence":62.04711151123047,  
"Name":"Coupe" }, { "Confidence":62.04711151123047, "Name":"Sports Car" }, {  
"Confidence":61.98909378051758, "Name":"City" }, { "Confidence":61.98909378051758,  
"Name":"Downtown" }, { "Confidence":61.98909378051758, "Name":"Urban" }, {  
"Confidence":60.978023529052734, "Name":"Neighborhood" }, {  
"Confidence":60.978023529052734, "Name":"Town" }, { "Confidence":59.22066116333008,  
"Name":"Sedan" }, { "Confidence":56.48063278198242, "Name":"Street" }, {  
"Confidence":54.235477447509766, "Name":"Housing" }, {  
"Confidence":53.85226058959961, "Name":"Metropolis" }, {  
"Confidence":52.001792907714844, "Name":"Office Building" }, {  
"Confidence":51.325313568115234, "Name":"Suv" }, { "Confidence":51.26075744628906,  
"Name":"Apartment Building" }, { "Confidence":51.26075744628906, "Name":"High Rise" },  
{ "Confidence":50.68067932128906, "Name":"Pedestrian" }, {  
"Confidence":50.59548568725586, "Name":"Freeway" }, {  
"Confidence":50.568580627441406, "Name":"Bumper" } ] }
```

# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- AWS-SDK를 통한 AWS Rekognition API 사용
- AWS SDK for Java를 사용하려면 다음 조건이 충족되어야 함
  - Java 개발 환경
  - AWS 계정 및 액세스 키
  - 환경에서 또는 공유된(AWS CLI 및 기타 SDK에 의해) 자격 증명 파일을 사용하여 설정된 AWS 자격 증명(액세스 키)

# AWS Rekognition Service API 활용 방법

## AWS Rekognition API

- AWS SDK
  - Java 1을 위한 SDK (AWS SDK for Java)
  - <https://github.com/aws/aws-sdk-java>
  - Java 2를 위한 SDK (AWS SDK for Java 2)
  - <https://github.com/aws/aws-sdk-java-v2>

# AWS Rekognition Service API 활용 방법

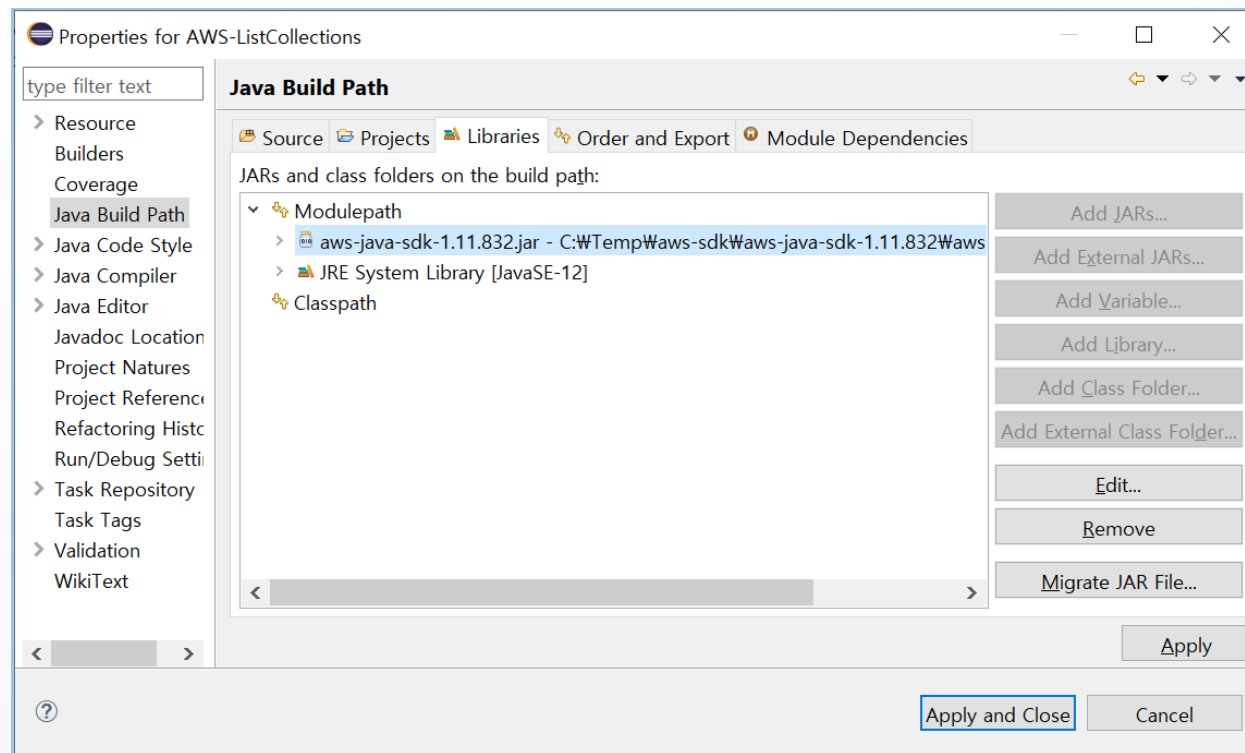
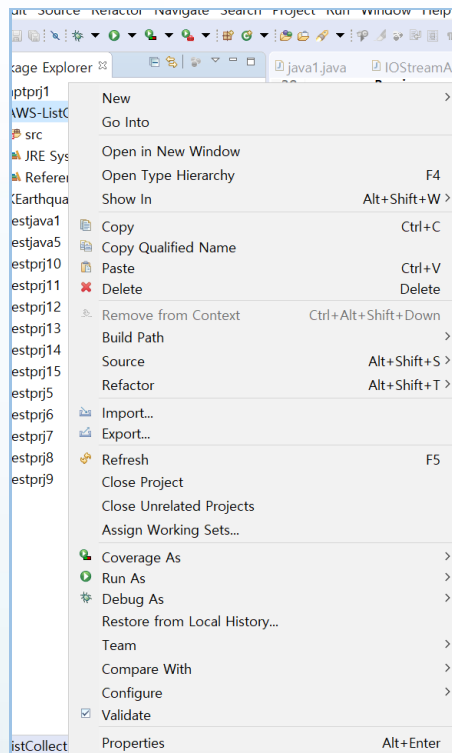
## ➡ AWS Rekognition API

- AWS-SDK를 통한 AWS Rekognition API 사용
- Java 1을 위한 SDK (AWS SDK for Java)
  - <https://github.com/aws/aws-sdk-java> 또는 <https://sdk-for-java.amazonwebservices.com/latest/aws-java-sdk.zip>에서 SDK를 다운로드

# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

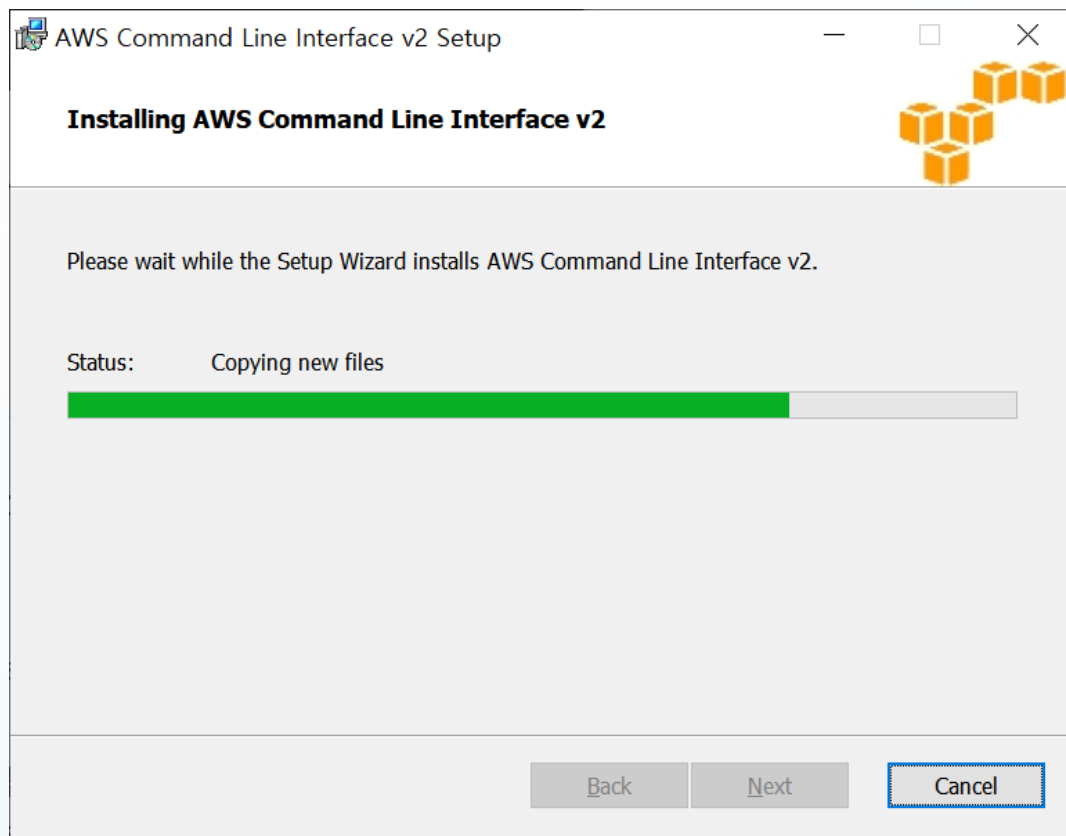
- Java 1을 위한 SDK (AWS SDK for Java)
  - SDK를 다운로드 한 후 압축 해제 후 aws-java-sdk-1.xx.jar 파일을 CLASSPATH에 추가 (또는, eclipse projec의 build path에 추가)



# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- Java 2을 위한 SDK (AWS SDK for Java 2)
  - Apache Maven 활용 방법 소개





# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

- AWS-SDK를 통한 Rekognition API 접근 : 사용자 추가 및 권한 설정(정책 필터를 통한 검색)

The screenshot shows the 'Add user' wizard in the AWS IAM console. Step 1 is active, showing the 'Set user details' section. The 'User name' field is filled with 'rek\_chaemin'. Below it, there is a link '자세히 알아보기' (Learn more). The 'AWS access type' section is also visible, with 'Programmatic access' selected. The 'Permissions' section is partially visible at the bottom.

aws 서비스 리소스 그룹

Chae, Min 글로벌 지원

### 사용자 추가

1 2 3 4 5

#### 사용자 세부 정보 설정

동일한 액세스 유형 및 권한을 사용하여 한 번에 여러 사용자를 추가할 수 있습니다. [자세히 알아보기](#)

사용자 이름\* rek\_chaemin

[다른 사용자 추가](#)

#### AWS 액세스 유형 선택

해당 사용자가 AWS에 액세스하는 방법을 선택합니다. 마지막 단계에서는 액세스 키와 자동 생성된 비밀번호가 제공됩니다. [자세히 알아보기](#)

액세스 유형\* ☒ 프로그래밍 방식 액세스  
AWS API, CLI, SDK 및 기타 개발 도구에 대해 액세스 키 ID 및 비밀 액세스 키(를) 할  
성화합니다.

☒ AWS Management Console 액세스  
사용자가 AWS Management Console에 로그인할 수 있도록 허용하는 비밀번호(를)  
활성화합니다.

\* 필수

취소 다음: 권한

의견 한국어

© 2008 - 2020, Amazon Web Services, Inc. 또는 계열사. All rights reserved. 개인 정보 보호 정책 이용 약관

The screenshot shows the 'Add user' wizard in the AWS IAM console, Step 2: Set permissions. The 'Permissions' section is active, showing a list of permissions. The 'AmazonRekognitionFullAccess' policy is selected. Below it, there are links for 'AmazonRekognitionReadOnlyAccess' and 'AmazonRekognitionServiceRole'. The 'Permissions' section is partially visible at the bottom.

Chae, Min 글로벌 지원

1 2 3 4 5

권한 복

[기존 정책 직접 연결](#)

5 결과 표시

유형	사용 용도
AWS 관리형	없음
AWS 관리형	없음
AWS 관리형	없음
AWS 관리형	없음

취소 이전 다음: 태그

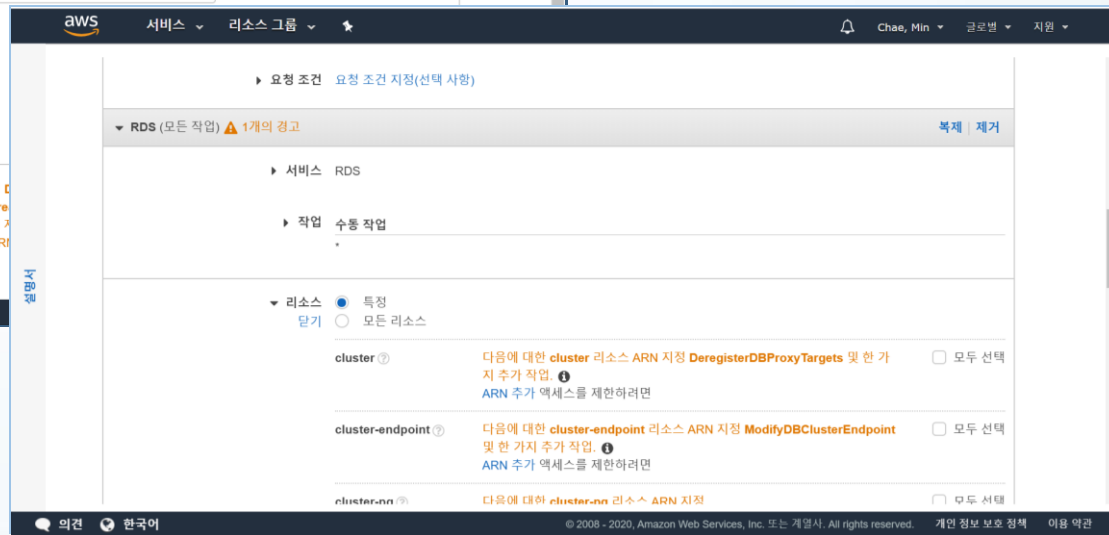
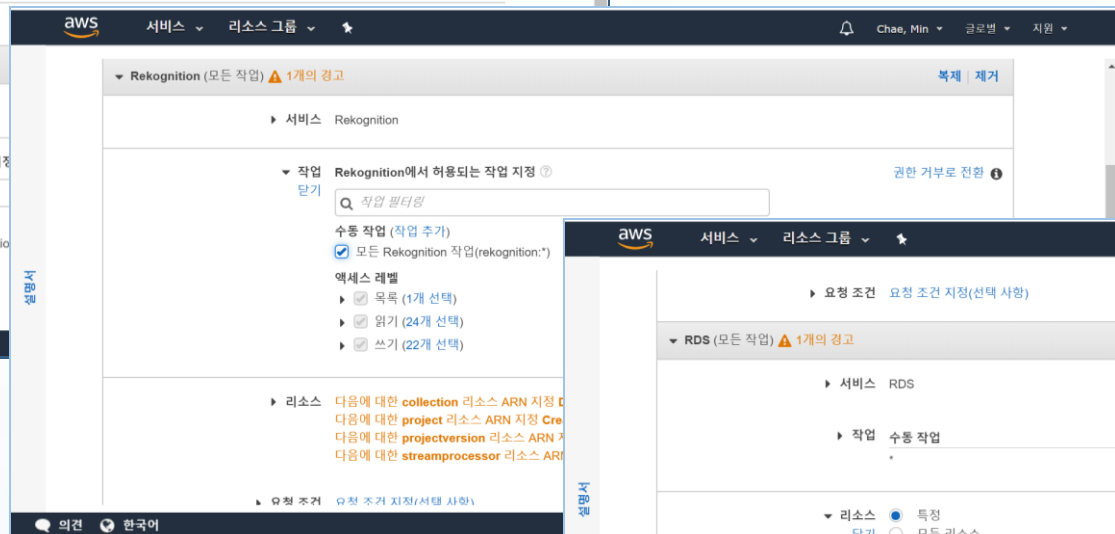
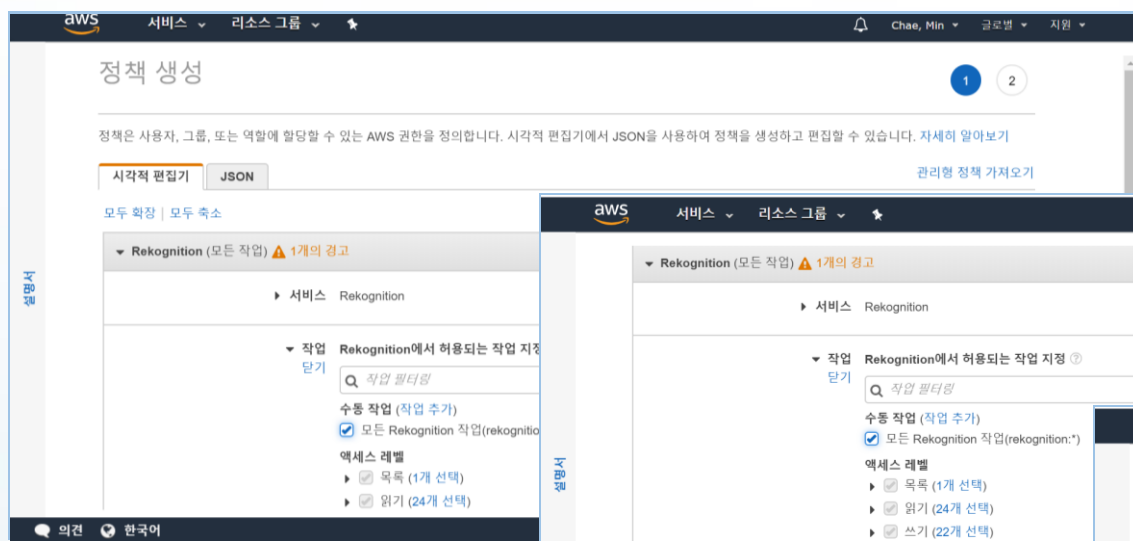
의견 한국어

© 2008 - 2020, Amazon Web Services, Inc. 또는 계열사. All rights reserved. 개인 정보 보호 정책 이용 약관

# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

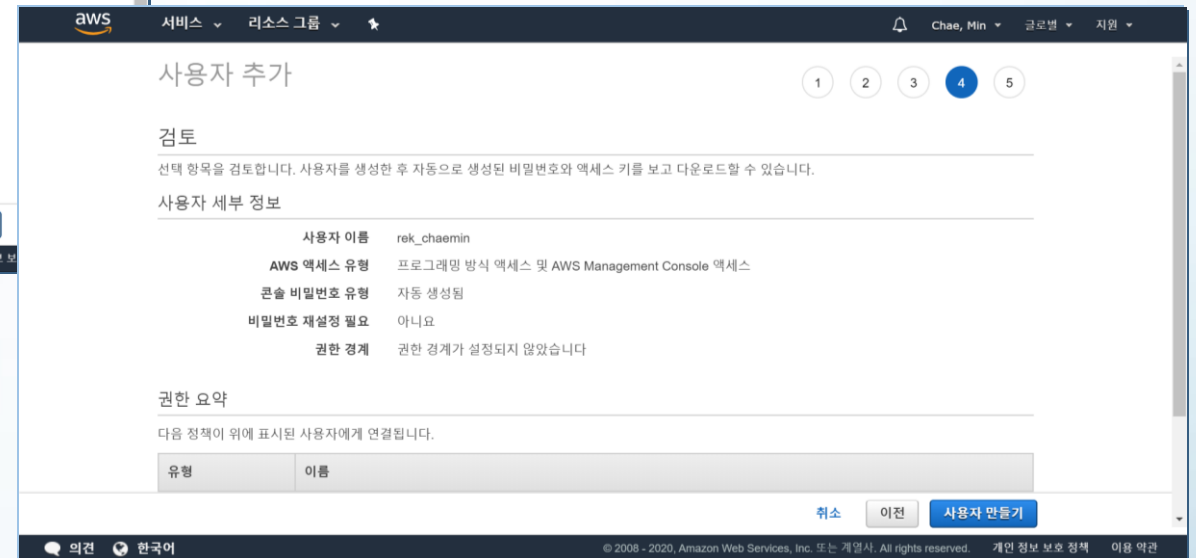
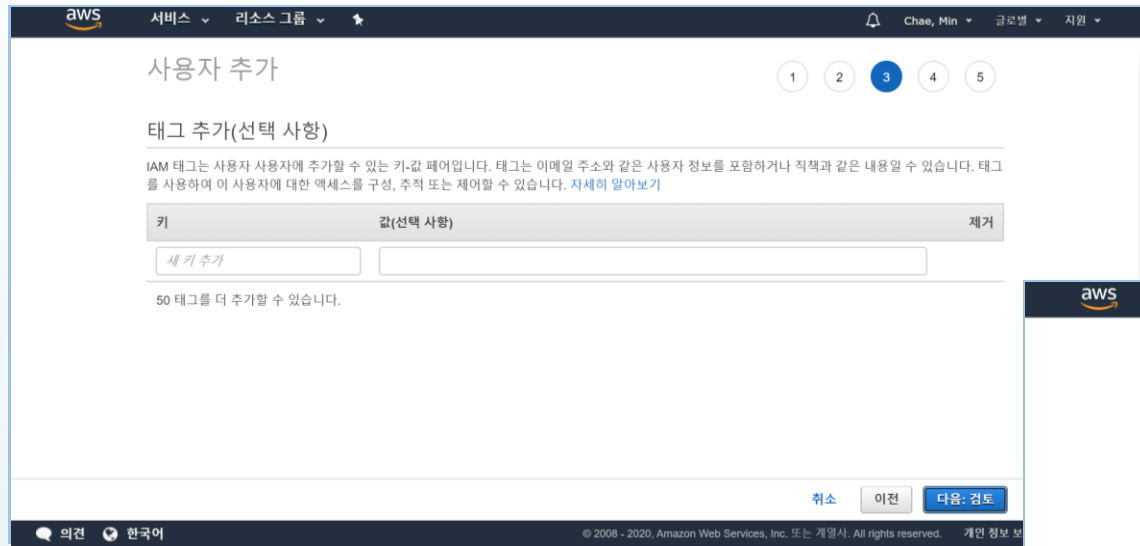
- AWS-SDK를 통한 Rekognition API 접근 : 권한 설정, 정책 생성도 가능



# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

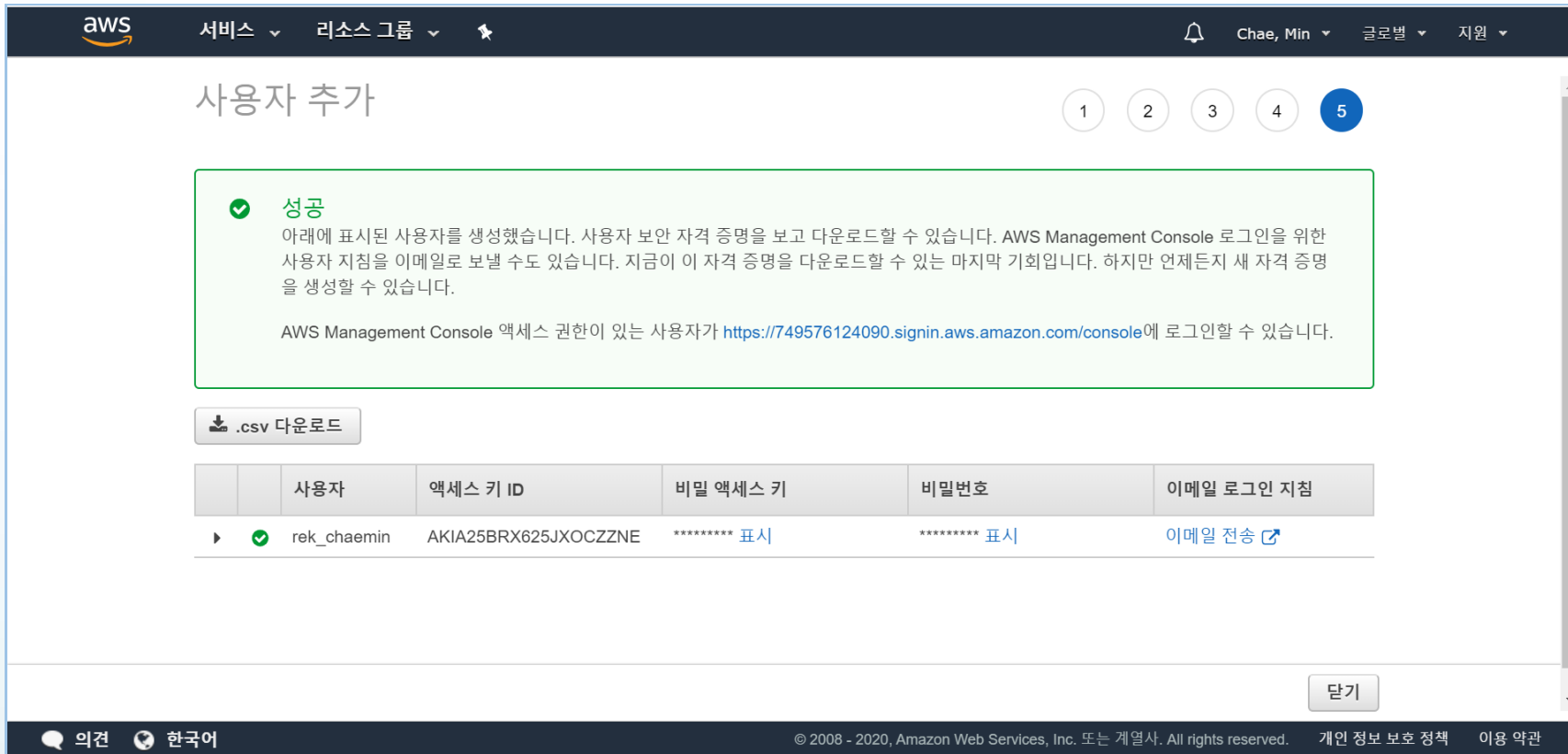
- AWS-SDK를 통한 Rekognition API 접근 : 사용자 추가 및 권한 설정(정책 필터를 통한 검색)



# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- AWS-SDK를 통한 Rekognition API 접근 : **생성된 사용자와 access key, secret access key 등 정보**



The screenshot shows the AWS IAM console 'Add User' page. A success message indicates that the user 'rek\_chaemin' has been created. The user's details are listed in a table below the message.

**성공**  
아래에 표시된 사용자를 생성했습니다. 사용자 보안 자격 증명을 보고 다운로드할 수 있습니다. AWS Management Console 로그인을 위한 사용자 지침을 이메일로 보낼 수도 있습니다. 지금이 이 자격 증명을 다운로드할 수 있는 마지막 기회입니다. 하지만 언제든지 새 자격 증명을 생성할 수 있습니다.

AWS Management Console 액세스 권한이 있는 사용자가 <https://749576124090.signin.aws.amazon.com/console>에 로그인할 수 있습니다.

[.csv 다운로드](#)

	사용자	액세스 키 ID	비밀 액세스 키	비밀번호	이메일 로그인 지침
▶	✓ rek_chaemin	AKIA25BRX625JXOCZZNE	***** 표시	***** 표시	<a href="#">이메일 전송</a>

닫기

의견 한국어 © 2008 - 2020, Amazon Web Services, Inc. 또는 계열사. All rights reserved. 개인정보 보호 정책 이용 약관

# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- AWS-SDK를 통한 Rekognition API 접근 : [AWS CLI를 통한 AWS SDK API](#)

1단계: AWS 계정 설정 및 IAM 사용자 만들기

2단계: AWS CLI 및 AWS SDK 설정

3단계: AWS CLI 및 AWS SDK API를 사용하여 시작하기

4단계: Amazon Rekognition 콘솔 사용 시작하기

# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- Collection 개념
  - AWS Rekognition의 기본 리소스
  - 생성되는 각각의 컬렉션에는 고유의 Amazon 리소스 이름(ARN)이 부여
  - 컬렉션에 얼굴을 저장하므로 'photo-collection'라는 이름의 Collection 생성
- `aws rekognition detect-labels --image`  
`"{W"S3ObjectW":{W"BucketW":W"photo-`  
`collectionW",W"NameW":W"photo.jpgW"}}" --region us-west-2`

# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

- AWS-SDK를 통한 Rekognition API 접근 : [AWS CLI를 통한 AWS SDK API](#)

### 활용

- Access Key, Secret Access Key, default region, default output format

```
관리자: 명령 프롬프트 - aws configure
Microsoft Windows [Version 10.0.18363.959]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\cutec>aws
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

    aws help
    aws <command> help
    aws <command> <subcommand> help
aws: error: the following arguments are required: command

C:\Users\cutec>aws --version
aws-cli/2.0.36 Python/3.7.7 Windows/10 exe/AMD64

C:\Users\cutec>aws configure list
      Name                               Value                Type      Location
      ----                               -
      profile                            <not set>            None      None
      access_key                         <not set>            None      None
      secret_key                         <not set>            None      None
      region                             <not set>            None      None

C:\Users\cutec>aws configure
AWS Access Key ID [None]:
```

```
선택 관리자: 명령 프롬프트
Microsoft Windows [Version 10.0.18363.959]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\cutec>aws
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

    aws help
    aws <command> help
    aws <command> <subcommand> help
aws: error: the following arguments are required: command

C:\Users\cutec>aws --version
aws-cli/2.0.36 Python/3.7.7 Windows/10 exe/AMD64

C:\Users\cutec>aws configure list
      Name                               Value                Type      Location
      ----                               -
      profile                            <not set>            None      None
      access_key                         <not set>            None      None
      secret_key                         <not set>            None      None
      region                             <not set>            None      None

C:\Users\cutec>aws configure
AWS Access Key ID [None]: AKIA25BRX625B4YXAJNI
AWS Secret Access Key [None]: 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
Default region name [None]: US_EAST
Default output format [None]: JSON

C:\Users\cutec>
```



# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

- AWS-SDK를 통한 Rekognition API 접근 : **mvn** 을 통한 new project 생성

```
mvn -B archetype:generate -DgroupId=com.mycompany.app -DartifactId=my-app  
-DarchetypeArtifactId=maven-archetype-quickstart -DarchetypeVersion=1.4
```

- Pom.xml 파일 생성
- ./src 디렉토리와  
./target 디렉토리 생성

```
C:\Temp\aws-sdk\project\mvnproject\my-app>dir  
C 드라이브의 볼륨에는 이름이 없습니다.  
볼륨 일련 번호: 7A70-C0C1  
  
C:\Temp\aws-sdk\project\mvnproject\my-app 디렉터리  
  
2020-08-03 오후 08:28 <DIR> .  
2020-08-03 오후 08:28 <DIR> ..  
2020-08-03 오후 02:58 5,382 pom.xml  
2020-08-03 오후 02:46 <DIR> src  
2020-08-03 오후 02:58 <DIR> target  
1개 파일 5,382 바이트  
4개 디렉터리 56,116,899,840 바이트 남음
```

# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- AWS-SDK를 통한 Rekognition API 접근 : 기본 생성 pom.xml 파일 대비  
변경할 부분

```
<properties>  
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
  <java.version>1.8</java.version>  
  <maven.compiler.source>1.7</maven.compiler.source>  
  <maven.compiler.target>1.7</maven.compiler.target>  
</properties>
```

# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

```
<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.11</version>
    <scope>test</scope>
  </dependency>

  <dependency>
    <groupId>software.amazon.awssdk</groupId>
    <artifactId>bom</artifactId>
    <version>2.11.11</version>
    <type>pom</type>
    <scope>import</scope>
  </dependency>
  <dependency>
    <groupId>software.amazon.awssdk</groupId>
    <artifactId>dynamodb-enhanced</artifactId>
    <version>2.11.4-PREVIEW</version>
  </dependency>
  <dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter-api</artifactId>
    <version>5.4.2</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

```
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-engine</artifactId>
  <version>5.4.2</version>
  <scope>test</scope>
</dependency>
<!-- https://mvnrepository.com/artifact/org.junit.platform/junit-platform-commons -->
<dependency>
  <groupId>org.junit.platform</groupId>
  <artifactId>junit-platform-commons</artifactId>
  <version>1.4.0</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.junit.platform/junit-platform-launcher -->
<dependency>
  <groupId>org.junit.platform</groupId>
  <artifactId>junit-platform-launcher</artifactId>
  <version>1.4.0</version>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-log4j12</artifactId>
  <version>1.7.25</version>
</dependency>
<!-- https://mvnrepository.com/artifact/software.amazon.awssdk/rekognition -->
<dependency>
  <groupId>software.amazon.awssdk</groupId>
  <artifactId>rekognition</artifactId>
  <version>2.13.31</version>
</dependency>
</dependencies>
```

# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

```
<build>
  <pluginManagement> <!-- lock down plugins versions to
avoid using Maven defaults (may be moved to parent pom) --
  >
    <plugins>
      <!-- clean lifecycle, see
https://maven.apache.org/ref/current/maven-core/lifecycles.html#clean\_Lifecycle -->
      <plugin>
        <artifactId>maven-clean-plugin</artifactId>
        <version>3.1.0</version>
      </plugin>
      <!-- default lifecycle, jar packaging: see
https://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\_bindings\_for\_jar\_packaging -->
      <plugin>
        <artifactId>maven-resources-plugin</artifactId>
        <version>3.0.2</version>
      </plugin>
      <plugin>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.8.0</version>
      </plugin>
```

```
<plugin>
  <artifactId>maven-surefire-plugin</artifactId>
  <version>2.22.1</version>
</plugin>
<plugin>
  <artifactId>maven-jar-plugin</artifactId>
  <version>3.0.2</version>
</plugin>
<plugin>
  <artifactId>maven-install-plugin</artifactId>
  <version>2.5.2</version>
</plugin>
<plugin>
  <artifactId>maven-deploy-plugin</artifactId>
  <version>2.8.2</version>
</plugin>
<!-- site lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#site\_Lifecycle -->
<plugin>
  <artifactId>maven-site-plugin</artifactId>
  <version>3.7.1</version>
</plugin>
<plugin>
  <artifactId>maven-project-info-reports-plugin</artifactId>
  <version>3.0.0</version>
</plugin>

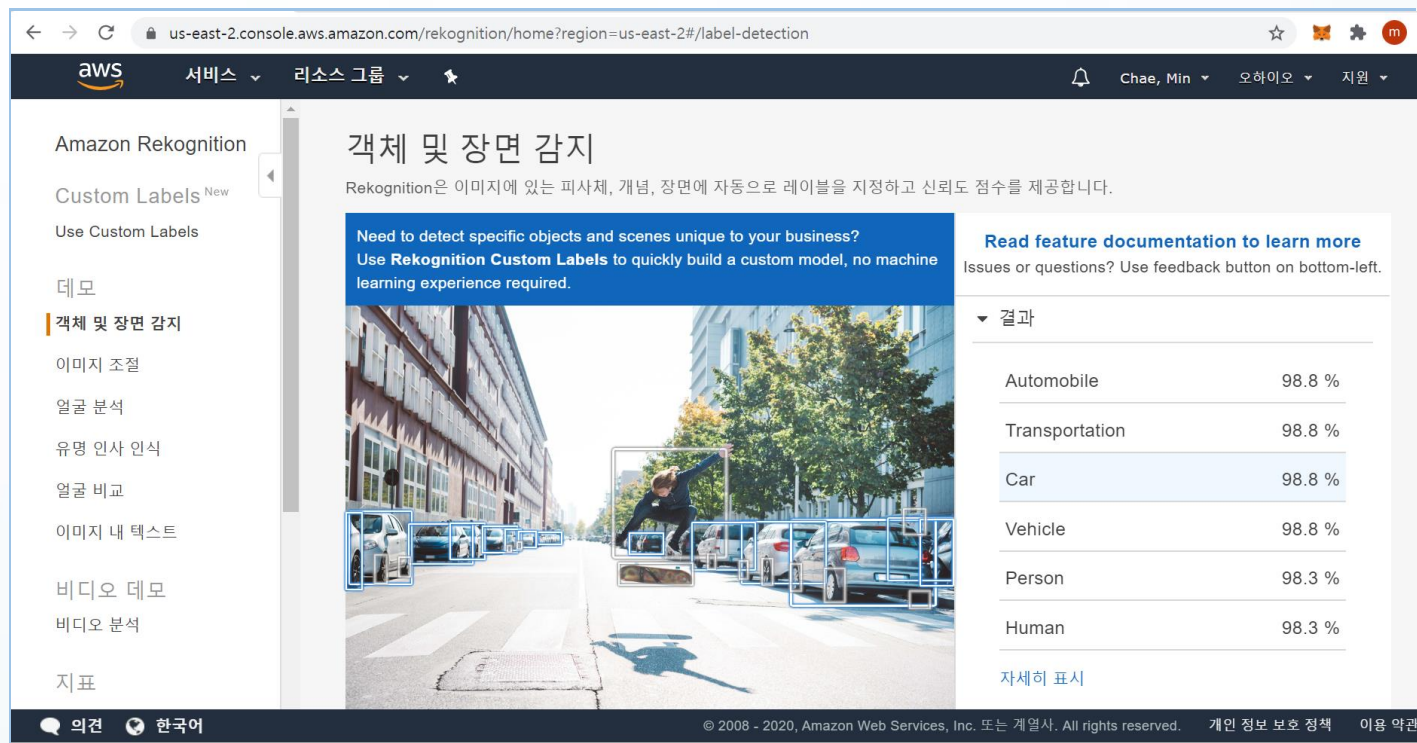
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.1</version>
  <configuration>
    <source>${java.version}</source>
    <target>${java.version}</target>
  </configuration>
</plugin>
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-surefire-plugin</artifactId>
  <version>2.22.1</version>
</plugin>

</plugins>
</pluginManagement>
</build>
```

# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

- AWS Rekognition 콘솔( <https://console.aws.amazon.com/rekognition/>) 접속
- 객체 및 장면 인식 선택
  - 이미지 업로드 – Upload
  - URL – Type the URL in the text box, and then choose Go



The screenshot displays the AWS Rekognition console interface. The left sidebar contains a navigation menu with options like 'Amazon Rekognition', 'Custom Labels', and '객체 및 장면 감지' (Object and Scene Detection). The main content area shows the '객체 및 장면 감지' page with a title '객체 및 장면 감지' and a description. Below this, there's a promotional banner for 'Rekognition Custom Labels'. The central part of the page features a large image of a street scene with bounding boxes around various objects. To the right of the image, a table lists the detected objects and their confidence scores.

결과	신뢰도
Automobile	98.8 %
Transportation	98.8 %
Car	98.8 %
Vehicle	98.8 %
Person	98.3 %
Human	98.3 %

자세히 표시

# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

### ● AWS SDK for PHP version

### AWS SDK for PHP

Developer Guide

Documentation - This Guide

Search

- [AWS SDK for PHP](#)
- [Getting Started](#)
  - [Prerequisites](#)
  - [Installing the SDK](#)
  - [Basic Use](#)

[AWS Documentation](#) » [AWS SDK for PHP](#) » [Developer Guide](#) » [Getting Started with the AWS SDK for PHP version 3](#) » Installing the AWS SDK for PHP Version 3

## Installing the AWS SDK for PHP Version 3

You can install the AWS SDK for PHP Version 3 by using:

- Composer
- The prepackaged phar in the SDK
- The ZIP file in the SDK

### Installing by Using Composer

On this page:

- Installing by Using Composer**
- [Installing by Using the Packaged Phar](#)
- [Installing by Using the ZIP file](#)

# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- AWS SDK for PHP version

```
public function awsImageRecognition($imageFileName)
{
    $returnData = array();

    try {
        $rekognitionClient = $this->getAWSClient( services: 'rekognition');

        $imageInfo = array();
        $imageInfo['Image']['S3Object']['Bucket'] = $this->awsS3Bucket;
        $imageInfo['Image']['S3Object']['Name'] = $imageFileName;
        $imageInfo['MaxLabels'] = 10;
        $imageInfo['MinConfidence'] = 80;

        $aryResults = $rekognitionClient->detectLabels($imageInfo);

        $returnData['result'] = true;
        $returnData['message'] = $aryResults;
    } catch (Exception $e) {
        $returnData['result'] = false;
        $returnData['message'] = $e->getMessage();
    }

    return $returnData;
}
```



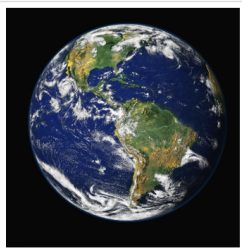
# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

- 실행 결과

### AWS Rekognition 을 활용한 이미지 분석 예제

이미지 업로드

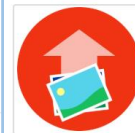


변경

S3에 이미지 업로드하기

### AWS Rekognition 을 활용한 이미지 분석 예제

이미지 업로드



이미지 선택

S3에 이미지 업로드하기

업로드 된 이미지들



#Astronomy #Earth #Globe  
#Outer Space #Planet #Space  
#Sphere #Universe



#Beach #Coast #Nature #Ocean  
#Outdoors #Sea #Water

#Dance #Dance Pose  
#Leisure Activities #Human #People  
#Person #Ballet

```
array(3) {  
  ["Labels"]=>  
  array(8) {  
    [0]=>  
    array(2) {  
      ["Name"]=>  
      string(9) "Astronomy"  
      ["Confidence"]=>  
      float(96.8987350464)  
    }  
    [1]=>  
    array(2) {  
      ["Name"]=>  
      string(5) "Earth"  
      ["Confidence"]=>  
      float(96.8987350464)  
    }  
    [2]=>  
    array(2) {  
      ["Name"]=>  
      string(5) "Globe"  
      ["Confidence"]=>  
      float(96.8987350464)  
    }  
    [3]=>  
    array(2) {  
      ["Name"]=>  
      string(11) "Outer Space"  
      ["Confidence"]=>  
      float(96.8987350464)  
    }  
  }  
}
```

# AWS Rekognition Service API 활용 방법

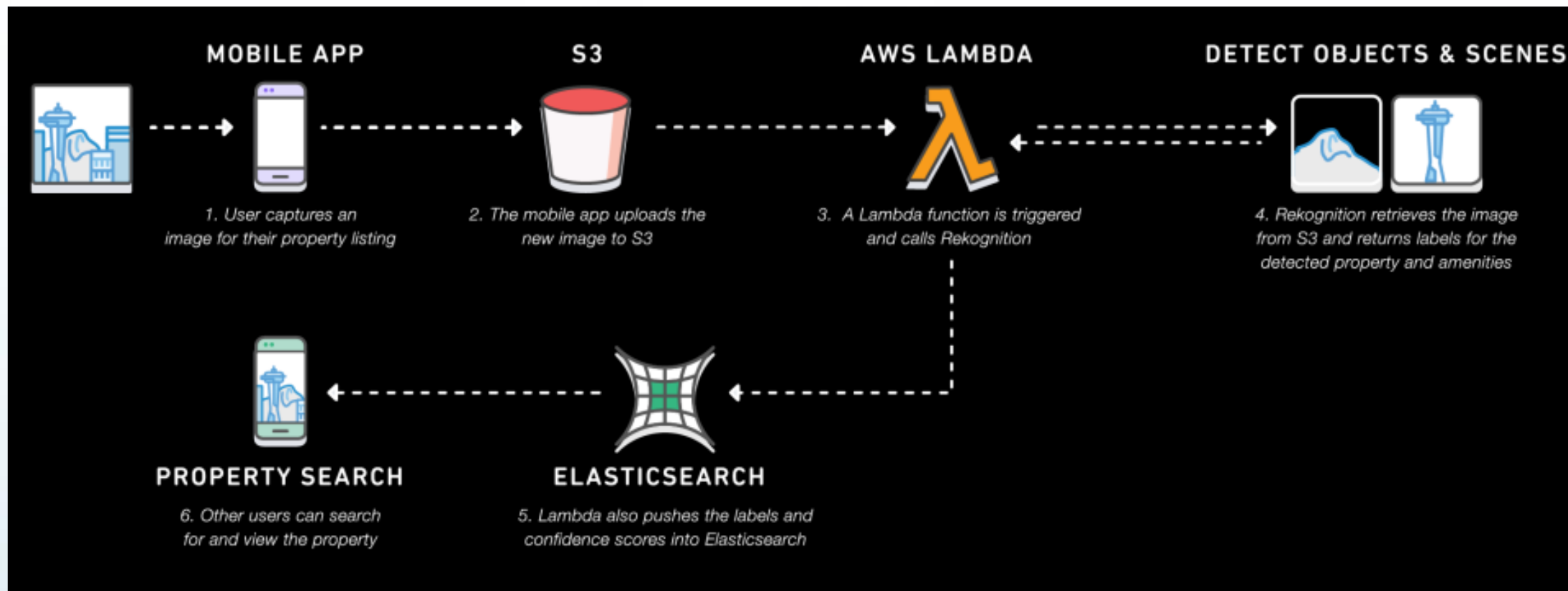
## ➡ AWS Rekognition API

- 객체 및 장면 인식 활용
  - 사진 공유 앱이나 Service : 자동으로 사진을 구분/분류/태깅하여 검색엔진 구현
  - ex) 지역, 이벤트, 결혼, 여행 등등
- 렌탈 관련 Service
  - 사진을 올리면 알아서 특징과 관련된 것을 탐지하여 쉽게 구현.
  - Airbnb와 같은 서비스를 구현 한다면 집주인은 사진만 업로드하면 자동
- 태깅
  - 여행 관련 Service : 다양한 사진으로부터 여행 관련 정보들을 구분하여 제공.
  - ex) 산, 바다, 도시 등등

# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- 객체 및 장면 인식 – 동적 이미지 검색 사용 사례



# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

- 다양한 차원의 얼굴 특성 분석

DetectFaces



# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API



```
[
{
  "BoundingBox": {
    "Height": 0.3449999988079071,
    "Left": 0.09666666388511658,
    "Top": 0.27166667580604553,
    "Width": 0.23000000417232513
  },
  "Confidence": 100,
  "Emotions": [
    {"Confidence": 99.1335220336914,
    "Type": "HAPPY" },
    {"Confidence": 3.3275485038757324,
    "Type": "CALM"},
    {"Confidence": 0.31517744064331055,
    "Type": "SAD"}
  ],
  "Eyeglasses": {"Confidence": 99.8050537109375,
  "Value": false},
  "EyesOpen": {"Confidence": 99.99979400634766,
  "Value": true},
  "Gender": {"Confidence": 100,
  "Value": "Female"}
```

DetectFaces

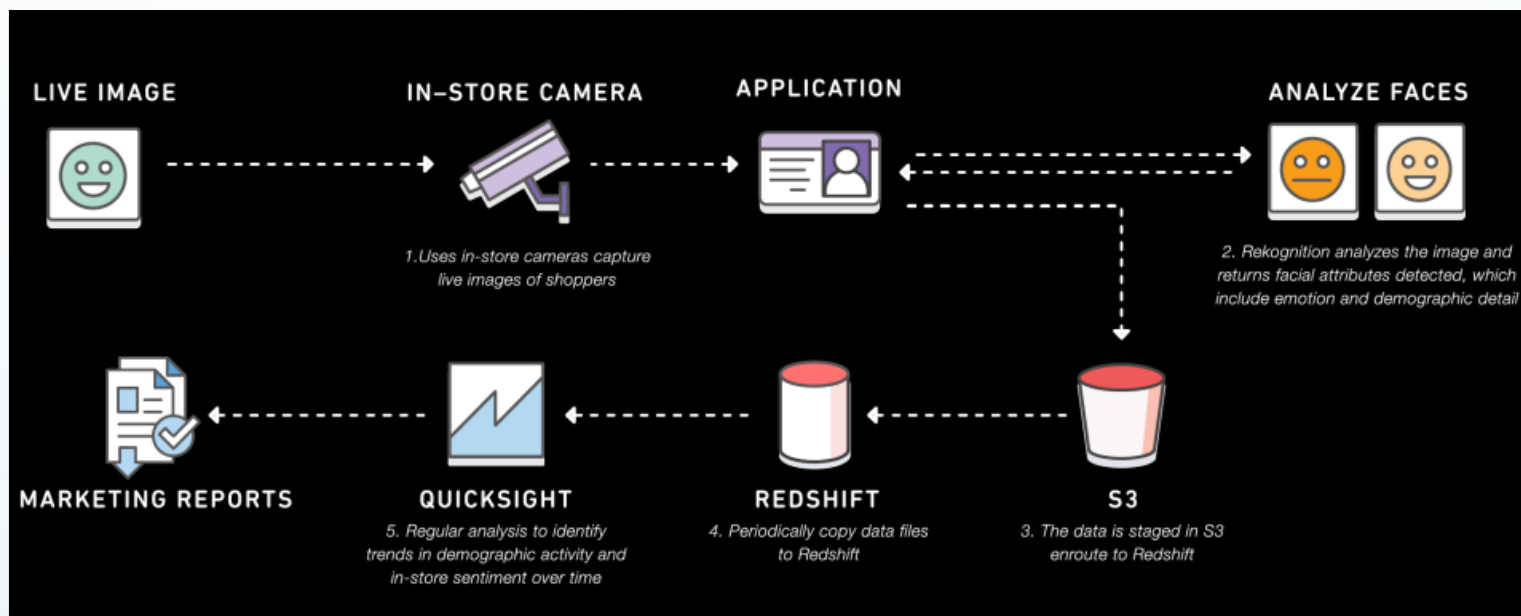




# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

사진 출력 Service	원하는 분위기의 또는 가장 잘 나온 사진을 추천
온라인 데이팅 Service	이상형에 가까운? 얼굴 추천?
리테일 비즈니스	고객의 움직임, 손님의 기분 등을 인식하여 비즈니스에 활용
디지털 광고	개인화된 광고 제공



# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- 이미지 상의 얼굴들을 비교하여 유사도(Similarity)를 측정 CompareFaces





# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API



```
{  
  "FaceMatches": [  
    {"Face": {"BoundingBox": {  
      "Height": 0.2683333456516266,  
      "Left": 0.5099999904632568,  
      "Top": 0.1783333271741867,  
      "Width": 0.17888888716697693},  
      "Confidence": 99.99845123291016},  
      "Similarity": 96  
    },  
    {"Face": {"BoundingBox": {  
      "Height": 0.2383333295583725,  
      "Left": 0.6233333349227905,  
      "Top": 0.3016666769981384,  
      "Width": 0.15888889133930206},  
      "Confidence": 99.71249389648438},  
      "Similarity": 0  
    }  
  ],  
  "SourceImageFace": {"BoundingBox": {  
    "Height": 0.23983436822891235,  
    "Left": 0.28333333134651184,  
    "Top": 0.351423978805542,  
    "Width": 0.1599999964237213},  
    "Confidence": 99.99344635009766}  
}
```

CompareFaces

# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- Rekognition API로 이미지 전달 방법
  - Image – `com.amazonaws.services.rekognition.model.Image` 패키지이며, 바이트 또는 S3객체로 입력 이미지를 제공함
    - Image에서 Bytes 속성을 지정하여 base64로 인코딩된 이미지 바이트를 전달함 (aws java sdk 사용시 인코딩 필요없음. 또한 s3객체 사용시에도)
    - 바이트 사용시 – 최대 5mb의 이미지 바이트 / 유형 : Base64로 인코딩 된 바이너리 데이터 개체 / 길이 제한 조건 : 최소 길이 1 최대길이 5242880
  - S3객체 사용시- S3객체를 이미지 소스로 식별 / 유형 : S3Object 객체

# AWS Rekognition Service API 활용 방법

## ➡ AWS Rekognition API

- S3 input 최대 이미지 사이즈 : 15MB
- non-s3 (Base64 encoded) API 호출 : 5MB 제한
- 최소 이미지의 해상도 : 80 pixel (가로 혹은 세로)
- 단일 컬렉션의 최대 얼굴 개수: 백만개
- search API에 의한 결과(일치한 얼굴) : 최대 4096개까지 반환
- 얼굴의 크기는 감지를 위해 이미지의 5% 이상을 차지해야 함

# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

```
import com.amazonaws.util.IOUtils;
// 위에는 여기서 사용한 패키지
@Test
public void test() throws FileNotFoundException, IOException {
    // 로컬 파일 경로
    String filePath = "C:/Users/11h11m/Desktop/zz/5054000478s.jpg";

    // 인풋스트림에 파일 추가
    ByteBuffer imageBytes;
    try (InputStream inputStream = new FileInputStream(new File(filePath))) {
        imageBytes = ByteBuffer.wrap(IOUtils.toByteArray(inputStream));
    }

    AmazonRekognition rekognitionClient = new AmazonRekognitionClient(
        new BasicAWSCredentials("accessKey", "secretKey"));

    DetectLabelsRequest request = new DetectLabelsRequest()
        .withImage(new Image() // 입력 이미지는 base64로 인코딩 된 바이트 또는 S3 객체입니다
            .withBytes(imageBytes))
        .withMaxLabels(10) // 서비스에서 응답으로 반환 할 최대 레이블 수입니다.
        .withMinConfidence(77F); // 레이블이 반환 할 최소 정확도 수준을 지정합니다.

    try {
        DetectLabelsResult result = rekognitionClient.detectLabels(request);
        List <Label> labels = result.getLabels(); // 레이블 뽑기

        System.out.println("파일 위치" + filePath);
        for (Label label: labels) {
            // getName 은 레이블 즉 객체 추출한 명 , getConfidence().toString()은 레이블의 정확도
            System.out.println(label.getName() + ": " + label.getConfidence().toString());
        }
    } catch (AmazonRekognitionException e) {
        e.printStackTrace();
    }
}
```

# AWS Rekognition Service API 활용 방법

## ➔ AWS Rekognition API

```
@Test
public void test() throws FileNotFoundException, IOException {
    // s3 버킷명 이랑 s3 객체 키값
    String bucket = "common01";
    String s3Key = "script02/12345678/img/lighthouse.jpg";

    // AmazonRekognition - Amazon Rekognition에 액세스하기 위한 인터페이스
    // AmazonRekognitionClient aws 계정 연동을 위한
    AmazonRekognition rekognitionClient = new AmazonRekognitionClient(
        new BasicAWSCredentials("accessKey", "secretKey"));

    // s3 버킷 리전과 rekognition image 작업 리전이 같아야하므로 엔드포인트
    rekognitionClient.setEndpoint("rekognition.ap-northeast-2.amazonaws.com");

    // AmazonWebServiceRequest를 상속 받고 있음
    DetectLabelsRequest request = new DetectLabelsRequest()
        .withImage(new Image() // 입력 이미지는 base64로 인코딩 된 바이트 또는 S3 객체입니다
            .withS3Object(new S3Object() // s3Object 를 사용 버킷명과 키값을 적용
                .withName(filePath).withBucket(bucket)))
        .withMaxLabels(10) // 서비스에서 응답으로 반환 할 최대 레이블 수입니다.
        .withMinConfidence(77F); // 레이블이 반환 할 최소 정확도 수준을 지정합니다.

    try {
        // AmazonWebServiceResult<ResponseMetadata> 상속 받고 있음
        // 입력으로 제공된 이미지 (JPEG 또는 PNG) 내의 실제 엔티티 인스턴스를 감지
        DetectLabelsResult result = rekognitionClient.detectLabels(request);
        List<Label> labels = result.getLabels(); // 레이블 뽑기

        for (Label label: labels) {
            // getName 은 레이블 즉 객체 추출한 명, getConfidence().toString()은 레이블의 정확도
            System.out.println(label.getName() + ": " + label.getConfidence().toString());
        }
    } catch (AmazonRekognitionException e) {
        e.printStackTrace();
    }
}
```