

BELAJAR DENGAN JENIUS AMAZON IAM (IDENTITY ACCESS MANAGEMENT)





“Menulis sebagai Janji Bakti, menuju tanah Air Jaya Sakti”

Gun Gun Febrianza
Bandung, 14 Maret 2020

First Published 13 May 2020,

Under License Attribution-NonCommercial-ShareAlike 4.0 International.

Open Library Indonesia



Sebuah konsep Perpustakaan Digital Terbuka untuk membantu mempermudah siapapun untuk mengakses ilmu pengetahuan. OpenLibrary.id adalah sebuah gerakan dan konsep pemikiran yang penulis usung sebagai wadah tempat untuk mengabdikan kepada masyarakat melalui kontribusi karya tulis. Karya tulis yang diharapkan dapat membantu agar minat baca jutaan pemuda-pemudi di Indonesia terus meningkat. Sebab penulis percaya **dengan membaca peluang keberhasilan hidup seseorang kedepannya akan menjadi lebih besar dan membaca dapat membawa kita ketempat yang tidak pernah kita sangka-sangka yaitu tempat yang lebih baik dari sebelumnya.**

Penulis sadar gerakan ini memerlukan penulis-penulis lainnya agar tujuannya bisa tercapai dan jangkauan manfaatnya bisa lebih luas lagi. Semakin banyak penulis dari berbagai bidang keilmuan akan semakin berwarna manfaat hasil karya tulis yang bisa diberikan untuk masyarakat. Maka dari itu penulis secara terbuka mengundang siapapun yang ingin bergabung menjadi penulis di gerakan *Indonesia Open Library*, agar bisa bertemu dan saling bersilaturahmi.

Orang boleh pandai setinggi langit, tapi selama ia tidak menulis maka ia akan hilang dalam masyarakat dan sejarah

- Pramoedya Ananta Toer –

Untuk teman-teman ku, rekan-rekan sebangsaku, apapun kepercayaan kalian, penulis meminta doa dari rekan-rekan supaya selalu diberi kesehatan, keselamatan dan keberkahan dalam hidup.

Agar tetap bisa menulis dan berkarya bersama sama.

Dari Penulis yang bercita-cita ingin bisa menjadi Guru di negeri ini.

Table of Contents

Contents

Open Library Indonesia.....	3
<i>Table of Contents</i>	5
<i>Account Preparation</i>	10
<i>Amazon Web Service</i>	10
<i>Chapter 1</i>	11
<i>AWS CLI</i>	11
<i>Subchapter 1 – AWS CLI V1 & V2</i>	11
1. <i>Command Line Interface (CLI)</i>	11
<i>Linux Shell</i>	11
<i>Windows Command Line</i>	12
<i>Remote</i>	12
2. <i>AWS CLI V2</i>	13
<i>Install AWS CLI V2 on Linux</i>	13
<i>Install AWS CLI V2 on MacOS</i>	13
<i>Install AWS CLI V2 on Windows</i>	14
3. <i>AWS CLI V1</i>	15
<i>Install AWS CLI</i>	15

Upgrade AWS CLI	15
Verify AWS CLI.....	15
Chapter 2.....	17
Identity Access & Management.....	17
Subchapter 1 – AWS IAM	17
1. IAM Standard Security	18
Users.....	18
Groups	18
Permissions	18
Roles.....	18
2. Setup Custom Sign-in Link	20
3. Setup Multi-Factor Authentication	21
4. Create IAM User.....	24
Set User Details	24
AWS Access Type	24
Set user Group	25
Set Permission.....	26
Review Account	27
Download User Credentials.....	27
5. Administrator Access Policy.....	30
6. Manage Password Policy	32

<i>Minimum Password Length Rule</i>	<i>33</i>
<i>Uppercase & Lowercase Letter Rule.....</i>	<i>33</i>
<i>Number Character Rule.....</i>	<i>33</i>
<i>Non-alphanumeric Character Rule.....</i>	<i>33</i>
<i>Password Expiration Rule</i>	<i>33</i>
<i>Change Password Rule.....</i>	<i>33</i>
<i>Password Re-use Rule</i>	<i>34</i>
7. Users Summary	35
<i>User Permissions</i>	<i>35</i>
<i>User Groups</i>	<i>36</i>
<i>Multiple Group.....</i>	<i>36</i>
<i>Remove User from Group.....</i>	<i>37</i>
<i>Security Credentials</i>	<i>37</i>
<i>Manage Console Password.....</i>	<i>38</i>
<i>Manage MFA Device</i>	<i>39</i>
<i>Create New Access Key.....</i>	<i>39</i>
<i>Active & Inactive Access Key.....</i>	<i>40</i>
<i>Delete Access Key</i>	<i>41</i>
<i>Delete User.....</i>	<i>41</i>
8. Create IAM Role	42
<i>Role EC2 to S3 Integration</i>	<i>43</i>

<i>Amazon S3 Full Access</i>	44
<i>Subchapter 2 – AWS IAM SDK</i>	47
1. <i>Create IAM User</i>	47
<i>Check Created User</i>	49
2. <i>Create Login Profile</i>	51
<i>Check Login Profile</i>	52
<i>Change Password</i>	54
3. <i>Delete Login Profile</i>	55
<i>Check Login Profile</i>	56
4. <i>Get User</i>	57
5. <i>List User</i>	59
6. <i>Update User</i>	61
<i>Check Update User</i>	62
7. <i>Delete IAM User</i>	63
<i>Check Deleted User</i>	64
8. <i>Create Group</i>	65
<i>Check Created Group</i>	66
9. <i>Add User to Group</i>	67
<i>Check User & Group</i>	68
10. <i>Remove User From Group</i>	70
<i>Check User & Group</i>	71

11. <i>List Group</i>	73
12. <i>Delete Group</i>	75
<i>Check Deleted Group</i>	76
Belajar Dengan Jenius AWS S3 & Node.js	77
Tentang Penulis	78

Account Preparation

Amazon Web Service

Sebelum menggunakan layanan **Amazon Web Services** pastikan anda membuat akun AWS terlebih dahulu. Kunjungi Halaman AWS [Click here](#).

Untuk tutorial bagaimana cara mendaftarkan akun di AWS, silahkan baca tutorial [Click here](#).

AWS menyediakan layanan yang sangat luas seperti layanan komputasi (*compute*), penyimpanan (*storage*), *databases*, *networking*, *analytics*, *machine learning* dan *artificial intelligence (AI)*, *Internet of Things (IoT)*, *security* dan masih banyak lagi. Layanan yang memberikan **non-stop solution** untuk menggunakan *product AWS*.

Chapter 1

AWS CLI

Subchapter 1 – AWS CLI V1 & V2

*If you double the number of experiments you do per year
you're going to double your inventiveness.*

— Jeff Bezos

Subchapter 1 – Objectives

- Mengetahui AWS Command-line Interface (CLI)
 - Belajar Install AWS CLI V2 di OS Linux
 - Belajar Install AWS CLI V2 di OS MacOS
 - Belajar Install AWS CLI V2 di OS Windows
 - Belajar Install AWS CLI V1
-

AWS Command Line Interface (CLI) adalah sebuah *tool* untuk memanajemen seluruh layanan AWS yang ingin kita gunakan. Melalui *command line* kita bisa mengendalikan berbagai layanan AWS sekaligus baik secara manual atau **automation** melalui **script**.

1. Command Line Interface (CLI)

Linux Shell

Pada sistem operasi **linux** atau **macOS** kita dapat berinteraksi dengan **AWS CLI** menggunakan program seperti **bash**, **zsh** dan **fish** untuk mengeksekusi suatu perintah.

Windows Command Line

Pada sistem operasi ***windows*** kita dapat berinteraksi dengan ***AWS CLI*** menggunakan program ***windows command-prompt (CMD)*** atau melalui ***powershell***.

Remote

Mengeksekusi perintah pada mesin ***Amazon Elastic Compute Cloud (Amazon EC2)*** secara ***remote*** melalui terminal yang disediakan dalam program ***PuTTY*** dan ***SSH*** atau melalui ***AWS System Manager***.

2. AWS CLI V2

Saat ini **AWS CLI** sedang dalam tahap upgrade dengan menyediakan **AWS CLI** Versi ke 2, versi pertama tidak akan dikembangkan lagi namun masih tetap dapat digunakan. Saat ini versi ke 2 disarankan tidak digunakan untuk **production**, **AWS CLI** Versi ke 2 disediakan khusus untuk uji coba.

Pada **AWS CLI** Versi ke 2, program sudah tidak lagi menggunakan *dependencies* dari *software package* lainnya. Sebelumnya pada versi pertama terdapat *dependency* yaitu keharusan untuk instalasi *python* terlebih dahulu.

Pada **AWS CLI** Versi ke 2 sudah disediakan dukungan untuk **Linux Distribution** seperti **CentOS, Fedora, Ubuntu, Amazon Linux 1**, dan **Amazon Linux 2**. Untuk **MacOS**, **AWS CLI** Versi ke 2 sudah disediakan dukungan untuk **High Sierra (10.13), Mojave (10.14)**, dan **Catalina (10.15)**.

Install AWS CLI V2 on Linux

Eksekusi perintah di bawah ini :

```
$ curl "https://d1vhwvl2y92vvt.cloudfront.net/awscli-exe-linux-x86_64.zip" -o  
"awscliv2.zip"  
  
unzip awscliv2.zip
```

Kemudian :

```
$ sudo ./aws/install
```

Install AWS CLI V2 on MacOS

Eksekusi perintah di bawah ini :

```
$ curl "https://d1vhwvl2y92vvt.cloudfront.net/awscli-exe-macos.zip" -o "awscliv2.zip"
```

```
unzip awscliv2.zip
```

Kemudian :

```
$ sudo ./aws/install
```

Install AWS CLI V2 on Windows

Pada sistem Operasi **Windows**, **download MSI Installer** untuk **AWS CLI V2** disini :

<https://d1vhw12y92vvt.cloudfront.net/AWSCLIV2.msi>

atau anda dapat mengunjungi halaman di bawah ini :

<https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-windows.html>

Setelah selesai download, lakukan instalasi kemudian eksekusi perintah di bawah ini :

```
C:\> aws2 --version
```

```
aws-cli/2.0.0dev3 Python/3.7.5 Windows/10 botocore/2.0.0dev2
```

3. AWS CLI V1

Sebelum melakukan instalasi **AWS CLI**, pastikan sistem operasi anda sudah melakukan instalasi bahasa pemrograman *python* versi 3.7 ke atas, kita akan melakukan instalasi **AWS CLI** melalui *python package manager* yang bernama *pip*.

Install AWS CLI

Untuk memulai instalasi eksekusi perintah di bawah ini :

```
pip install awscli
```

Upgrade AWS CLI

Jika sebelumnya anda sudah memiliki *AWS CLI*, direkomendasikan untuk melakukan upgrade ke versi yang terakhir untuk alasan keamanan. Untuk *upgrade AWS CLI*, eksekusi perintah di bawah ini :

```
pip install --upgrade awscli
```

Verify AWS CLI

Lakukan verifikasi untuk memastikan *AWS CLI* sudah terpasang dalam sistem operasi anda :

```
aws --version
```

```
aws-cli/1.18.56 Python/3.8.2 Windows/10 botocore/1.16.6
```

Eksekusi perintah di atas untuk mendapatkan informasi versi *AWS CLI* yang kita miliki.

Saat menggunakan *AWS CLI*, kita memerlukan *AWS Credentials* untuk melakukan ***authentication*** pada layanan **AWS**.

Terdapat beberapa cara diantaranya adalah :

1. ***Environment Credentials*** berupa :
AWS_ACCESS_KEY_ID dan **AWS_SECRET_KEY**
2. ***The Shared Credentials File***
3. ***IAM Roles***, jika anda menggunakan *AWS CLI* di sebuah mesin **EC2** maka kita tidak perlu lagi mengatur *credential files* di *production*.

Chapter 2

Identity Access & Management

Subchapter 1 – AWS IAM

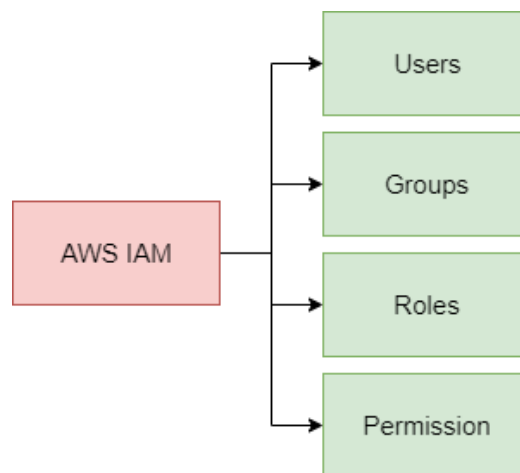
*If you don't understand the details of your business
you are going to fail.*

— Jeff Bezos

Subchapter 1 – Objectives

- Mengetahui AWS IAM
 - Belajar Membuat **IAM User**
 - Belajar Mengkonfigurasi *User Credential*
 - Belajar Membuat **IAM Role**
-

AWS IAM (*Identity and Access Management*) adalah layanan untuk manajemen user dan level akses untuk mengamankan *AWS Resources*. IAM juga mendukung konsep standar keamanan seperti *users, groups, roles* dan *permissions*.



Gambar 1 Standard Security Conceptual

1. IAM Standard Security

Users

Users adalah seorang pengguna atau orang-orang yang ingin mengelola layanan AWS.

Groups

Groups adalah sekumpulan *users*, sebuah *users* dapat ditambahkan ke dalam sebuah *group* yang telah kita klasifikasi. Setiap *user* akan memiliki atau mewarisi *permission* yang diberikan kepada suatu *group*.

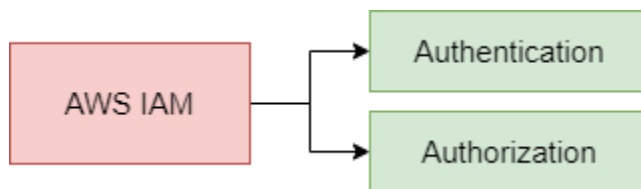
Permissions

Permissions adalah sekumpulan dokumen yang seringkali disebut dengan *policy documents*. Dokumen dengan format *JSON* ini digunakan untuk menentukan *permission* pada sebuah *user*, *group*, atau *role*.

Roles

Roles digunakan oleh suatu *AWS Services* untuk akses *AWS Services* lainnya.

Dengan *IAM* kita dapat melakukan kontrol siapa saja yang melakukan *authentication* (*signed in*) dan memiliki izin *authorization* pada *AWS Resources*.



Gambar 2 Authentication & Authorization

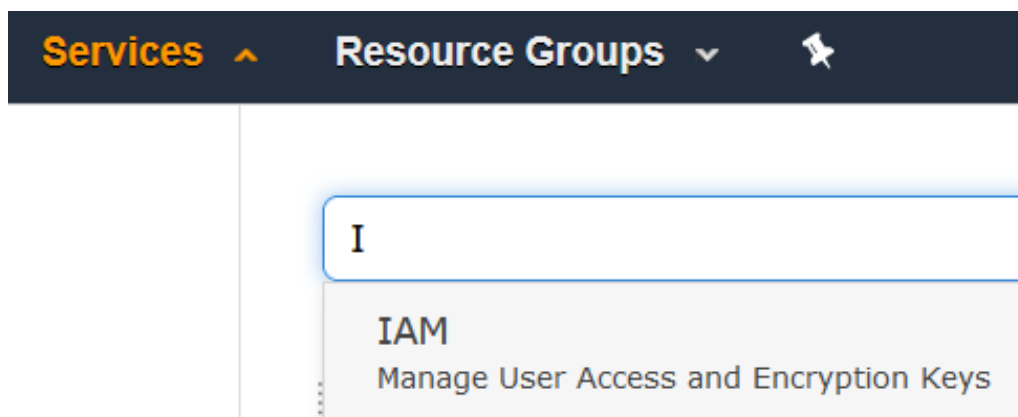
Saat pertama kali kita membuat *AWS Account*, kita menggunakan sebuah *Root Account* yang memiliki akses pada seluruh layanan *AWS Resources*.

Untuk memulai aktivitas sangat tidak disarankan menggunakan **AWS Root Account**, karena akun tersebut memiliki kapabilitas untuk membuat dan menghapus **IAM users**, mengubah *billing*, menghapus *account* dan segala aksi dalam akun AWS anda.

Untuk memulai belajar AWS IAM, masuk ke halaman **AWS Management Console** :

<https://aws.amazon.com/console/>

Kemudian cari **IAM** dalam kolom **services** :



Gambar 3 IAM Service

2. Setup Custom Sign-in Link

Pada halaman **IAM Console**, anda akan melihat alamat untuk melakukan **sign-in** masih **default**. Kita dapat mengubahnya menggunakan nama yang kita miliki sebagai alias.

Welcome to Identity and Access Management

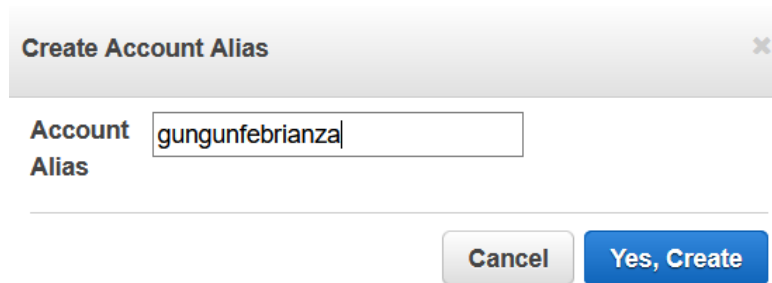
IAM users sign-in link:

<https://790699697882.signin.aws.amazon.com/console> 

| [Customize](#)

Gambar 4 IAM User Sign-In

Klik link **customize** :



The image shows a 'Create Account Alias' dialog box. It has a title bar with 'Create Account Alias' and a close button (X). Below the title bar, there is a label 'Account Alias' and a text input field containing 'gungunfebrianza'. At the bottom of the dialog, there are two buttons: 'Cancel' and 'Yes, Create'.

Gambar 5 Create Sign-in Alias

Jika berhasil maka **link IAM Users sign-in** akan berubah :

Welcome to Identity and Access Management

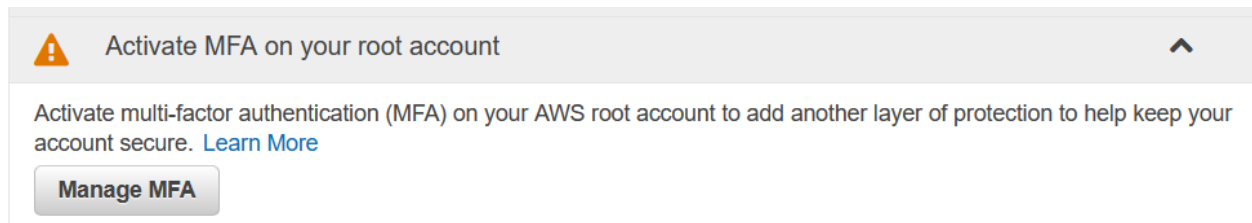
IAM users sign-in link:

<https://gungunfebrianza.signin.aws.amazon.com/console> 

Gambar 6 New User Sign-in

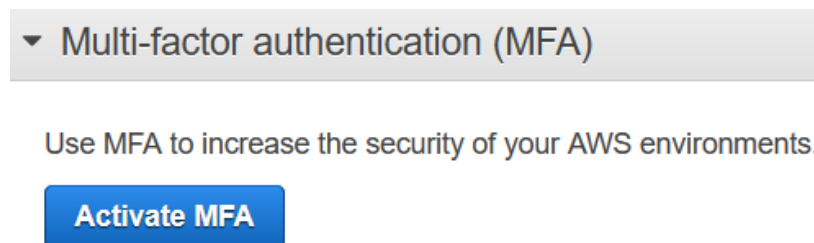
3. Setup Multi-Factor Authentication

Untuk memastikan akun **root** pada **AWS** kita aman, konfigurasi **Multi-factor authentication (MFA)** harus dilakukan. Pada menu **MFA** klik tombol **Manage MFA** seperti pada gambar di bawah ini :



Gambar 7 Manage MFA

Pilih menu **Multi-factor authentication (MFA)** kemudian klik tombol **Activate MFA** seperti pada gambar di bawah ini :



Gambar 8 Activate MFA

Selanjutnya akan muncul sebuah dialog **Manage MFA Device**, pada kesempatan kali ini kita akan menggunakan **Virtual MFA Device** seperti pada gambar di bawah ini :

Manage MFA device

Choose the type of MFA device to assign:

☒ **Virtual MFA device**
Authenticator app installed on your mobile device or computer

☐ **U2F security key**
YubiKey or any other compliant U2F device

☐ **Other hardware MFA device**
Gemalto token

For more information about supported MFA devices, see [AWS Multi-Factor Authentication](#)

Cancel


Continue

Gambar 9 Virtual MFA Device

Klik **Continue** untuk melanjutkan. Pastikan anda telah memasang aplikasi **Google Authenticator** pada **smartphone**, selanjutnya **scan QR Code** di bawah ini menggunakan aplikasi **Google Authenticator** :

Set up virtual MFA device

2. Use your virtual MFA app and your device's camera to scan the QR code



Alternatively, you can type the secret key. [Show secret key](#)

3. Type two consecutive MFA codes below

MFA code 1

MFA code 2


Cancel

Previous

Assign MFA

Gambar 10 Setup Virtual MFA Device

Isi kolom **MFA Code** 1 dan 2 sesuai dengan **token** yang tampil pada aplikasi **Google Authenticator**. Selain itu klik **link Show secret key** agar anda dapat melihat **secret key** untuk menyimpannya :

Alternatively, you can type the secret key. [Hide secret key](#) 

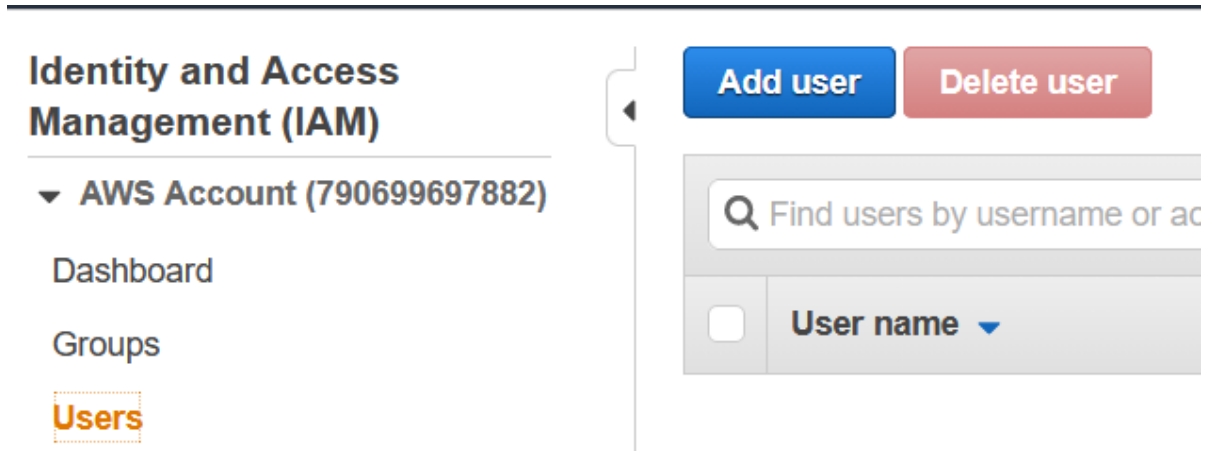
NOAJ6VSB5LFILCNJOK

Gambar 11 Secret Key

Jika sudah tekan tombol **Assign MFA**.

4. Create IAM User

Pilih menu **Users** dan klik tombol **Add User** :



Gambar 12 Add IAM User

Set User Details

Pada kolom **Set user details** masukan nama **username** yang anda ingin gunakan :

Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name*

[+ Add another user](#)

Gambar 13 Set User Details

AWS Access Type

Pada kolom **Select AWS access type**, checklist **Programmatic access** seperti gambar di bawah ini :

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

- Access type*** ☒ **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.
- ☒ **AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

Gambar 14 Setup AWS Access Type

Selanjutnya pilih **Autogenerated password** agar **AWS** memberikan **random password** secara otomatis dan **checkbox Require password reset** agar user membuat **password** yang baru setelah **sign-in** :

Console password* ☒ Autogenerated password
☐ Custom password

Require password reset ☒ User must create a new password at next sign-in

Gambar 15 Setup Password User

Kemudian Klik tombol **Next: Permission** di pojok kanan bawah.

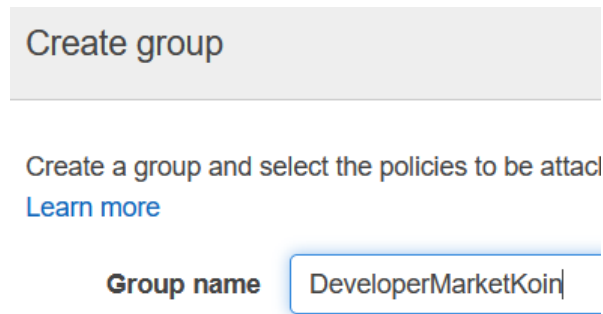
Set user Group

Untuk lebih mudah dalam memanajemen **user**, kita harus membuat **group** untuk mengelompokan **user**. Pada **menu Add user to group** pilih **Create group** :

Add user to group

Gambar 16 Create Group

Selanjutnya kita harus menentukan nama grup yang akan kita buat untuk menyimpan user :



Create group

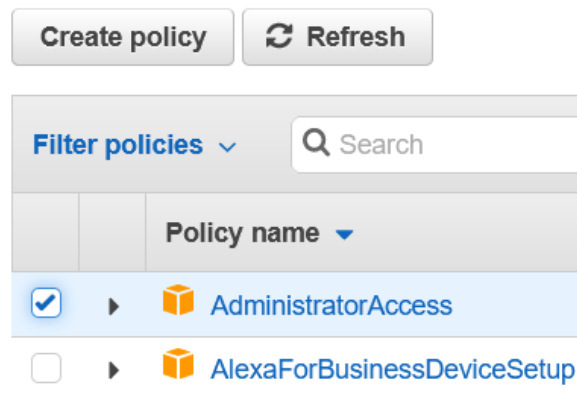
Create a group and select the policies to be attached
[Learn more](#)

Group name DeveloperMarketKoin

Gambar 17 Create Group Name

Set Permission

Setelah itu kita perlu menambahkan **permission** pada **group** yang akan berlaku untuk setiap **user** di dalamnya. Pada grup **DeveloperMarketKoin** penulis memerlukan **AdministratorAccess** agar memiliki akses penuh :



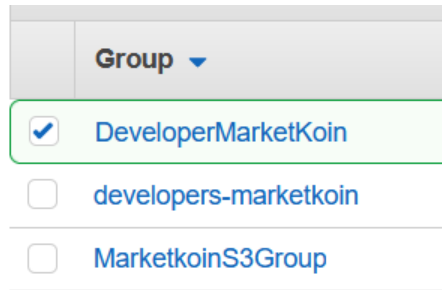
Create policy Refresh

Filter policies Search

	Policy name
<input checked="" type="checkbox"/>	AdministratorAccess
<input type="checkbox"/>	AlexaForBusinessDeviceSetup

Gambar 18 Add Policy AdministratorAccess

Kemudian klik tombol **Create Group** di pojok kanan bawah, selanjutnya kita dapat memilih **group** yang telah kita buat sebelumnya untuk menyimpan **user** :



Group ▼	
<input checked="" type="checkbox"/>	DeveloperMarketKoin
<input type="checkbox"/>	developers-marketkoin
<input type="checkbox"/>	MarketkoinS3Group

Gambar 19 Group Created

Review Account

Klik tombol **Next: Tags** dipojok kanan bawah dan skip untuk melanjutkan **review**.

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

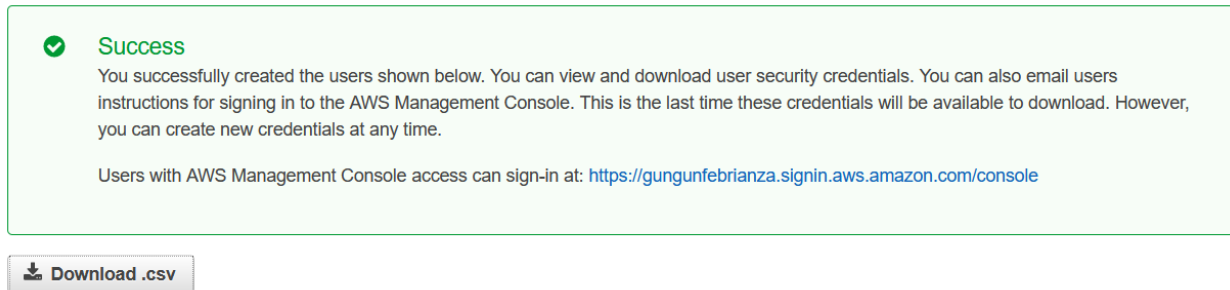
User details

User name	gungunfebrianza
AWS access type	Programmatic access and AWS Management Console access
Console password type	Autogenerated
Require password reset	Yes
Permissions boundary	Permissions boundary is not set

Gambar 20 Review Create User

Download User Credentials

Klik tombol **Create user** dipojok kanan bawah, selanjutnya informasi sukses akan ditampilkan :



Gambar 21 Download Security Credentials

Klik **Download .csv**, di dalamnya terdapat informasi :

1. **Username**
2. **Password**
3. **Access Key**
4. **Secret Access Key**
5. **Console Login Link**

Informasi juga ditampilkan pada tabel di bawah ini :

	User	Access key ID	Secret access key	Password	Email login instructions
▼	gungunfebri...	AKIA3QGGK743NG6RDHWN	***** Show	***** Show	Send email ↗
<div><div>✓ Created user gungunfebrianza</div><div>✓ Added user gungunfebrianza to group DeveloperMarketKoin</div><div>✓ Created access key for user gungunfebrianza</div><div>✓ Created login profile for user gungunfebrianza</div></div>					

Gambar 22 User Credentials

Pastikan kita mencatat dan menyimpan ditempat yang aman dan lakukan manajemen **backup** untuk mengantisipasi lupa atau kehilangan. Hal ini dikarenakan kita tidak akan lagi mendapatkan kedua data di atas.

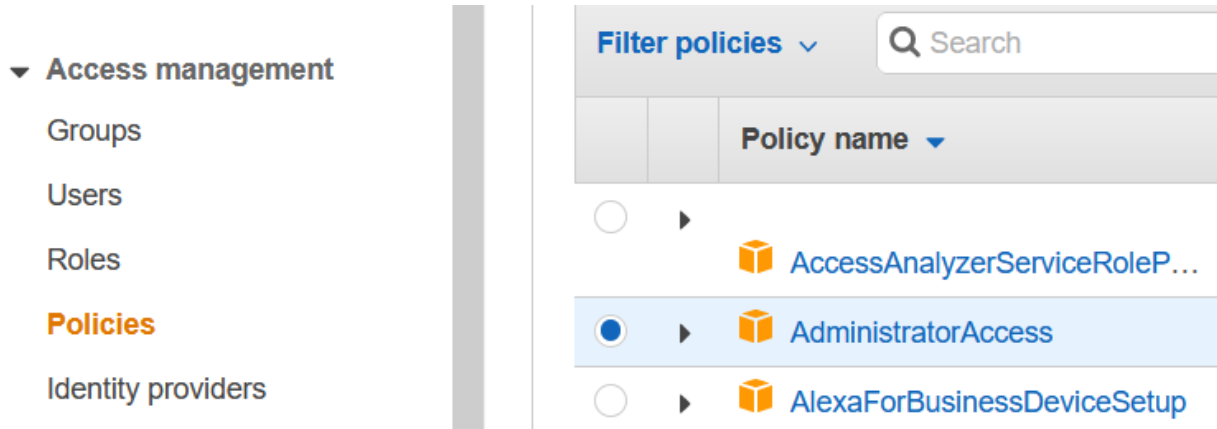
Pada **Tab Menu Users** kita akan langsung melihat **user** yang telah kita buat seperti pada gambar di bawah ini :

<input type="text" value="Find users by username or access key"/>		
<input type="checkbox"/>	User name ▼	Groups
<input type="checkbox"/>	gungunfebrianza	DeveloperMarketKoin

Gambar 23 New User Created

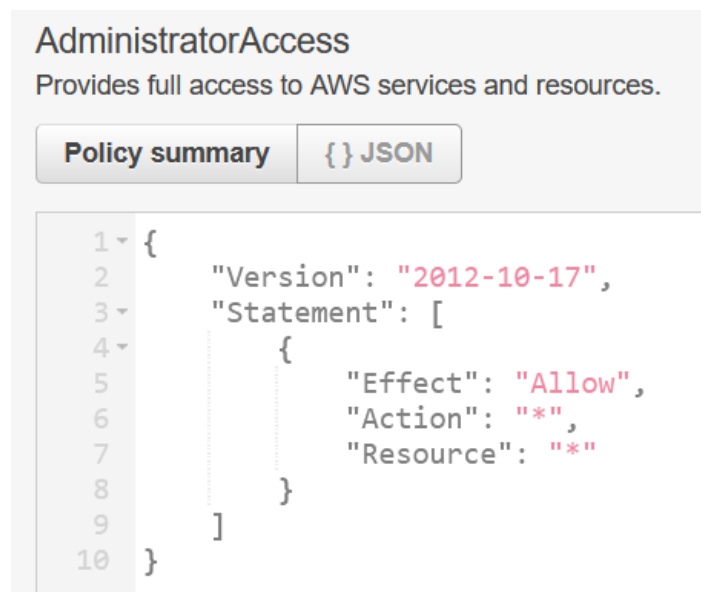
5. Administrator Access Policy

Klik **Tab Menu Policies** di sebelah kiri, kemudian klik **AdministratorAccess Policy** :



Gambar 24 AdministratorAccess Policy

Inilah yang dimaksud dengan **Permissions** dan **Policy document** yang telah kita bahas sebelumnya di **IAM Standard Security**. Di dalam **AdministratorAccess Policy** kita dapat melihat terdapat sebuah kode **JSON** :



Gambar 25 Administrator Access Policy

Di dalam sebuah **policy document** terdapat versi dari **policy** tersebut, **statement** yang terdiri dari sebuah **object** dalam **array**, **object** tersebut tersusun dari sekumpulan **key/value pair** dan memiliki **properties** :

1. **Effect**
2. **Action**
3. **Resource**

Effect Allow artinya **policy document** ini diizinkan untuk digunakan, **Action** dan **resource** dengan **wildcard *** artinya kita diizinkan untuk melakukan apa saja pada seluruh **resources** yang tersedia dalam **AWS**.

6. Manage Password Policy

Pemanfaat **password policy** di atur agar setiap **IAM users** yang kita buat memiliki **password** yang aman. Untuk itu manajemen **password** yang baik diperlukan dengan cara menekan tombol **Manage Password Policy** :

Use a password policy to require your IAM users to create strong passwords and to rotate their passwords regularly.
[Learn More](#)

Manage Password Policy

Gambar 26 Manage Password Policy

Setelah itu anda akan melihat menu **Modify password policy** seperti pada gambar di bawah ini :

Modify password policy

A password policy is a set of rules that define complexity requirements and mandatory rotation periods for your IAM users' passwords. [Learn more](#)

Select your account password policy requirements:

- ☒ Enforce minimum password length

characters
- ☒ Require at least one uppercase letter from Latin alphabet (A-Z)
- ☒ Require at least one lowercase letter from Latin alphabet (a-z)
- ☒ Require at least one number
- ☐ Require at least one non-alphanumeric character (!@#\$%^&*()_+-=[]{}|')
- ☐ Enable password expiration
- ☐ Password expiration requires administrator reset
- ☒ Allow users to change their own password
- ☐ Prevent password reuse

Gambar 27 Modify Password Policy

Minimum Password Length Rule

Di gunakan untuk menentukan minimum panjang karakter ***password*** yang harus dipatuhi oleh setiap ***IAM User***.

Uppercase & Lowercase Letter Rule

Di gunakan untuk menentukan apakah ***password*** wajib mengandung ***uppercase & lowercase*** atau tidak sama sekali yang harus dipatuhi oleh setiap ***IAM User***.

Number Character Rule

Di gunakan untuk menentukan apakah ***password*** wajib mengandung ***number*** atau tidak sama sekali yang harus dipatuhi oleh setiap ***IAM User***.

Non-alphanumeric Character Rule

Di gunakan untuk menentukan apakah ***password*** wajib mengandung ***non-alphanumeric*** atau tidak sama sekali yang harus dipatuhi oleh setiap ***IAM User***.

Password Expiration Rule

Di gunakan untuk menentukan apakah ***password*** dapat mengalami ***expiration*** atau tidak dan apakah ***expiration*** memerlukan ***reset password*** terlebih dahulu dari ***administrator*** untuk setiap ***IAM User***.

Change Password Rule

Di gunakan untuk menentukan apakah ***password*** dapat diubah atau tidak oleh setiap ***IAM User***.

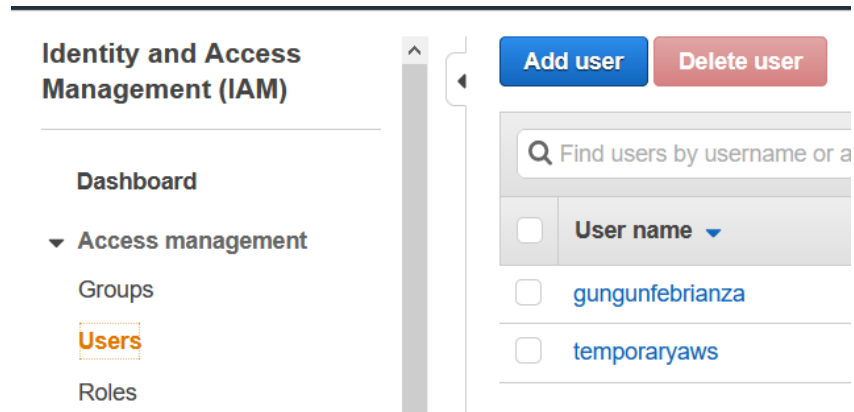
Password Re-use Rule

Di gunakan untuk menentukan apakah ***password*** lama dapat digunakan kembali jika kita sedang mengubah ***password*** untuk setiap ***IAM User***.

Klik tombol ***Save Changes*** di pojok kanan bawah.

7. Users Summary

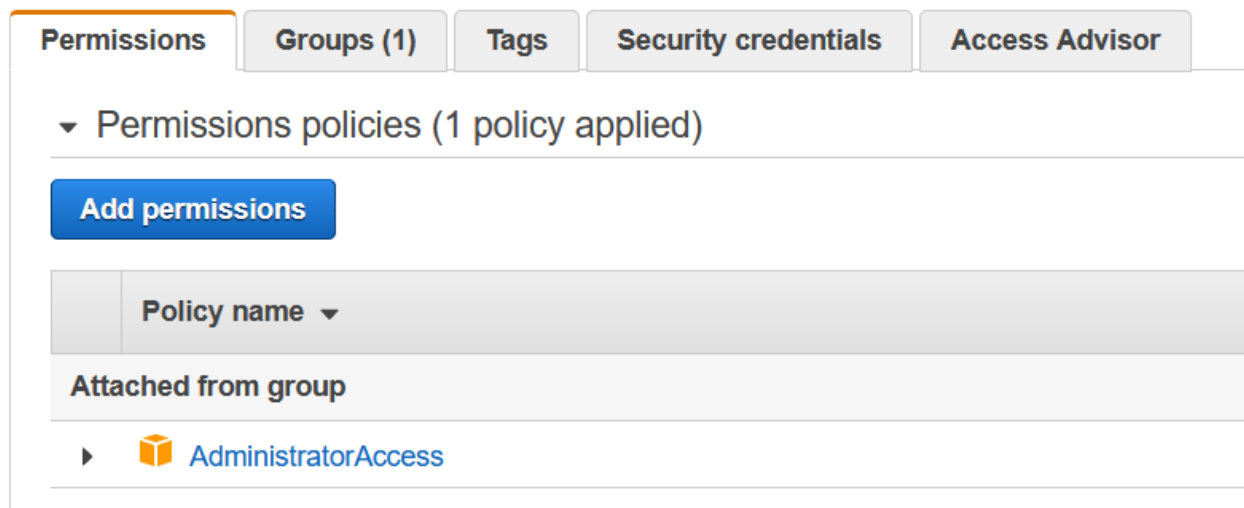
Untuk mengetahui **summary user** yang telah kita buat sebelumnya, pada **Tab Menu Users** klik **username** yang telah kita buat :



Gambar 28 Get User Summary

User Permissions

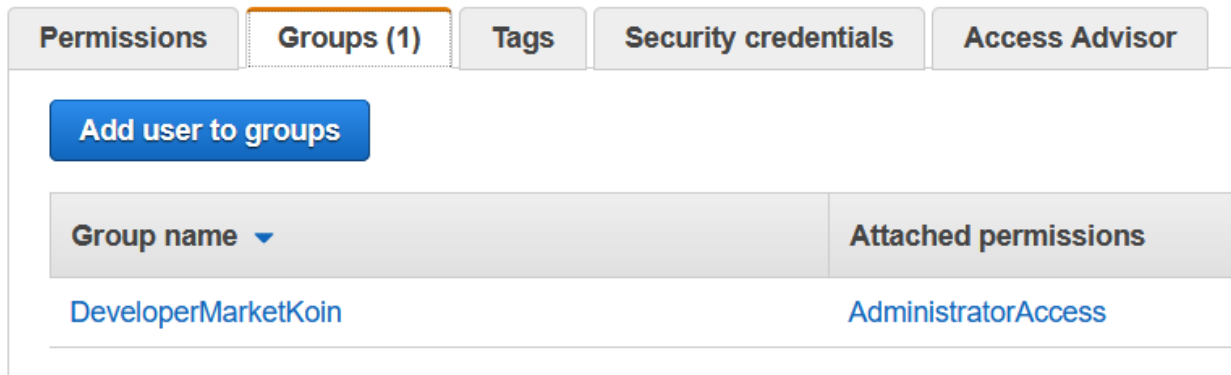
Pada menu di bawah ini kita dapat melihat terdapat **permission** yang dimiliki oleh user **gungunfebrianza**, **permission AdministratorAccess Policy** yang di dapatkan berasal dari **group**.



Gambar 29 Permissions Summary

User Groups

Selanjutnya pada **Tab Groups** kita dapat melihat bahwa **user gungunfebrianza** beda di dalam **group DeveloperMarketkoin**.

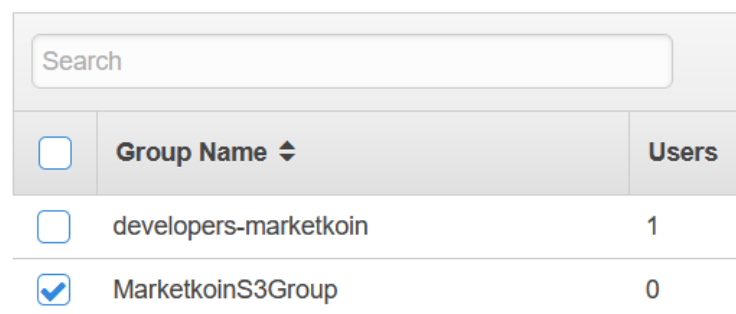


Gambar 30 User Group Summary

Multiple Group

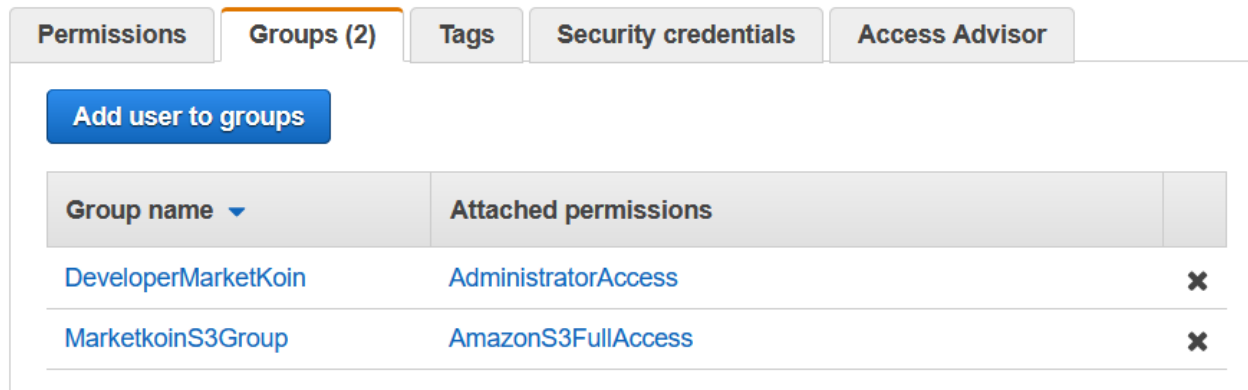
Untuk menambahkan **user** agar memiliki atau bergabung ke dalam beberapa **group** sekaligus, klik tombol **Add user to groups**. Pilih **Group** yang ingin di tambahkan misal **MarketkoinS3Group** :

Select groups that user **gungunfebrianza** will be added to.



Gambar 31 Add to Group

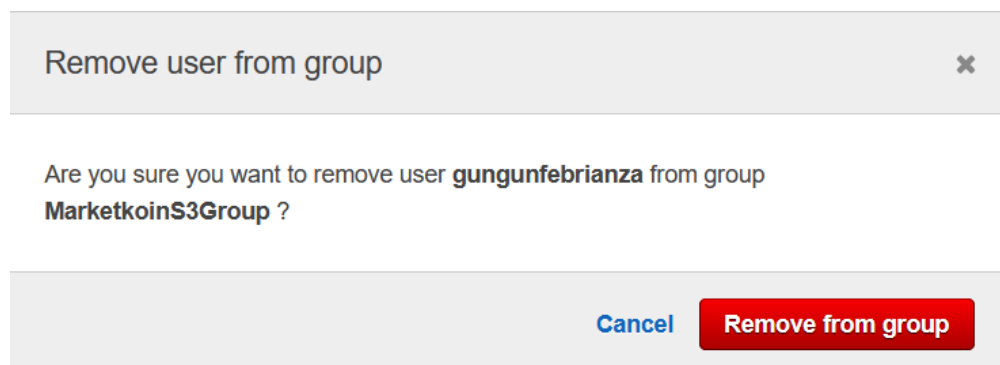
Klik tombol **Add to Groups** di pojok kanan bawah. Maka kita akan melihat bahwa **user gungunfebrianza** sekarang sudah tergabung dalam grup lebih dari 1 :



Gambar 32 Multi Group

Remove User from Group

Untuk mengeluarkan **user gungunfebrianza** dari sebuah **group** cukup klik tanda centang X, maka terdapat **dialog** konfirmasi seperti pada gambar di bawah ini :



Gambar 33 Remove User From Group

Klik **Remove from group** untuk melanjutkan.

Security Credentials

Pada kolom **security credentials** kita dapat melihat :

1. **Console Sign-in Link**
2. **Console Password Management**
3. **MFA Device Management**

4. Signing Certificate

Sign-in credentials

Summary	• Console sign-in link: https://gungunfebrianza.signin.aws.amazon.com/console
Console password	Enabled (never signed in) Manage
Assigned MFA device	Not assigned Manage
Signing certificates	None

Gambar 34 Sign-in Credential

Manage Console Password

Jika kita klik **link Manage** pada **Console password** maka akan muncul **dialog** seperti pada gambar di bawah ini :

Manage console access

×

Manage gungunfebrianza's AWS console access and password.

Console access

☒ Enable
☐ Disable
Disabling will remove pre-existing password.

Set password*

☒ Keep existing password
☐ Autogenerated password
☐ Custom password

Require password reset

☐ User must create a new password at next sign-in

Cancel

Apply

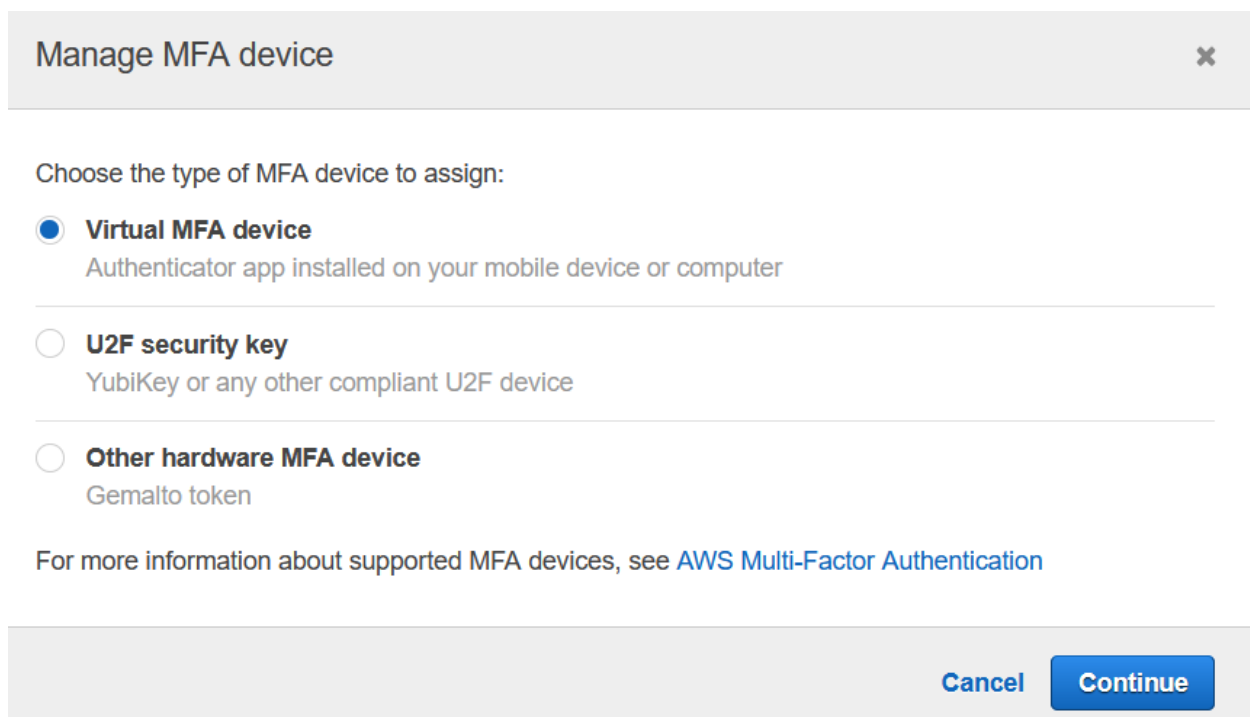
Gambar 35 Manage Console Password

Pengaturan **Console access** untuk mengatur apakah **user gungunfebrianza** boleh masuk melalui **console** atau tidak. **Set Password** untuk mengatur **password** apakah kita akan mengubahnya secara **random**, manual atau tetap mempertahankan **password** yang ada.

Pengaturan **Require password reset** digunakan agar **user gungunfebrianza** membuat **password** baru lagi jika login ke dalam **console**.

Manage MFA Device

Penerapan **Multi-factor Authentication (MFA)** pada **user gungunfebrianza** juga dapat dilakukan jika kita ingin manajemen pengaturan **Assigned MFA Device**. Caranya masih sama seperti pada pengaturan **MFA** yang sudah dilakukan sebelumnya.



Manage MFA device

Choose the type of MFA device to assign:

☒ **Virtual MFA device**
Authenticator app installed on your mobile device or computer

☐ **U2F security key**
YubiKey or any other compliant U2F device

☐ **Other hardware MFA device**
Gemalto token

For more information about supported MFA devices, see [AWS Multi-Factor Authentication](#)

Cancel Continue

Gambar 36 Manage MFA Device

Create New Access Key

Kita juga dapat membuat **Access keys** yang baru untuk **user gungunfebrianza**, untuk melakukannya klik **Create access key** :

Access keys

Use access keys to make secure REST or HTTP Query protocol requests to AWS service APIs. For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation. [Learn more](#)

Create access key

Gambar 37 Create Access Key

Jika berhasil maka akan muncul dialog seperti pada gambar di bawah ini :

Create access key ✕

✓ **Success**

This is the **only** time that the secret access keys can be viewed or downloaded. You cannot recover them later. However, you can create new access keys at any time.

Download .csv file

Access key ID	Secret access key
AKIA3QGGK743NG5ZHG5ON	***** Show

Close

Gambar 38 Create Access Key Success

Active & Inactive Access Key

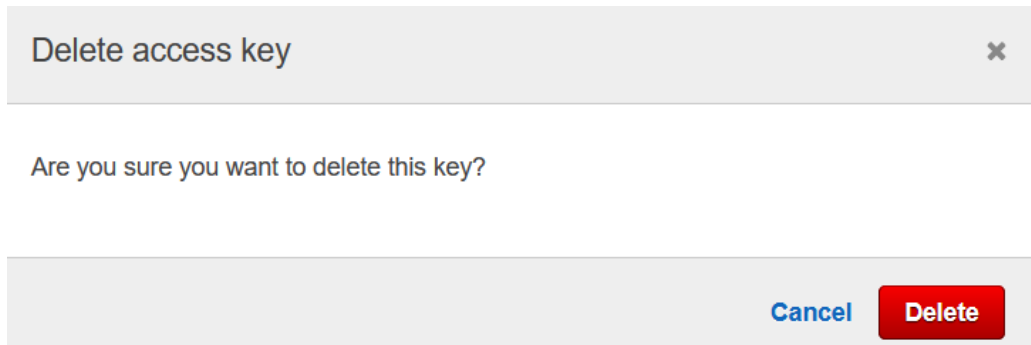
Untuk melakukan konfigurasi **Access key** agar bisa kita aktifkan atau kita non-aktifkan, klik label **Make Inactive** dan **Make Active** seperti pada gambar di bawah ini :

Access key ID	Created	Last used	Status	
AKIA3QGGK743NG6RDHWNY	2020-05-11 09:07 UTC+0700	N/A	Active Make inactive	✕
AKIA3QGGK743NG5ZHG5ON	2020-05-11 10:33 UTC+0700	N/A	Inactive Make active	✕

Gambar 39 Active & Inactive Access Key

Delete Access Key

Untuk menghapus sebuah access key klik tanda X sampai muncul dialog konfirmasi seperti pada gambar di bawah ini :



Gambar 40 Delete Access Key

Delete User

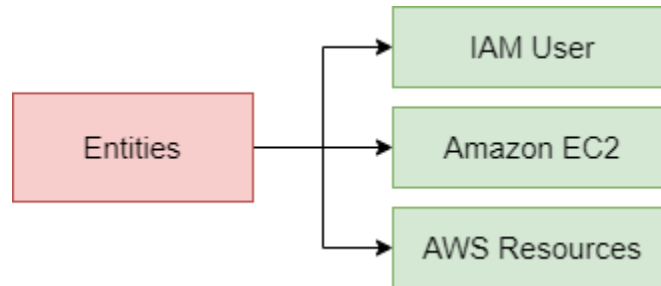
Untuk menghapus **user gungunfebrianza**, pada pojok kanan atas terdapat menu **Delete user**. Menu tersebut digunakan jika kita ingin menghapus **user gungunfebrianza**.



Gambar 41 Delete User

8. Create IAM Role

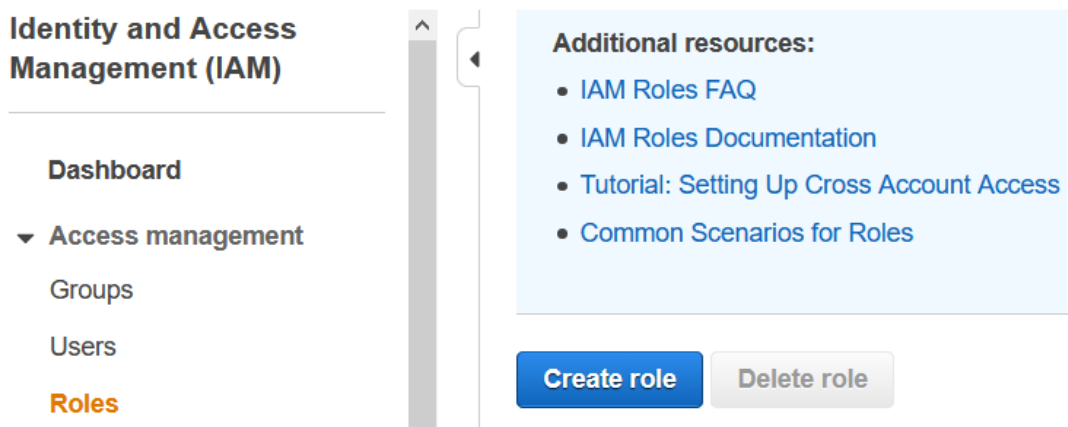
Kita membuat **IAM Roles** agar bisa menyediakan izin dan jalur yang aman untuk setiap entitas yang kita percaya. Entitas yang dimaksud adalah :



Gambar 42 Role Entities

1. **IAM User** dalam akun AWS yang lain.
2. Sebuah **application** yang berjalan di mesin EC2 kemudian membutuhkan akses pada **AWS Resources**.
3. Sebuah **Amazon Service** yang membutuhkan akses pada **AWS resources** lainnya dalam akun yang kita miliki agar bisa menyediakan layanan tertentu.

Untuk membuat **IAM Roles** klik tombol **Create Role** :



Gambar 43 IAM Roles

Role EC2 to S3 Integration

Pada studi kasus kita hari ini, Kita ingin bisa mengelola **Amazon S3** melalui mesin **Amazon EC2** yang kita buat. Pilih **EC2** pada kolom **Common use cases** :

Choose a use case

Common use cases

EC2

Allows EC2 instances to call AWS services on your behalf.

Lambda

Allows Lambda functions to call AWS services on your behalf.

Gambar 44 EC2 Case

Kemudian klik tombol **Next: Permissions**. Pada kolom Review terdapat **Role name*** isi seperti pada gambar di bawah ini :

Review

Provide the required information below and review this role before you create it.

Role name*

S3FullAccess

Use alphanumeric and '+=, @-_' characters. Maximum 64 characters.

Role description

Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and '+=, @-_' characters.

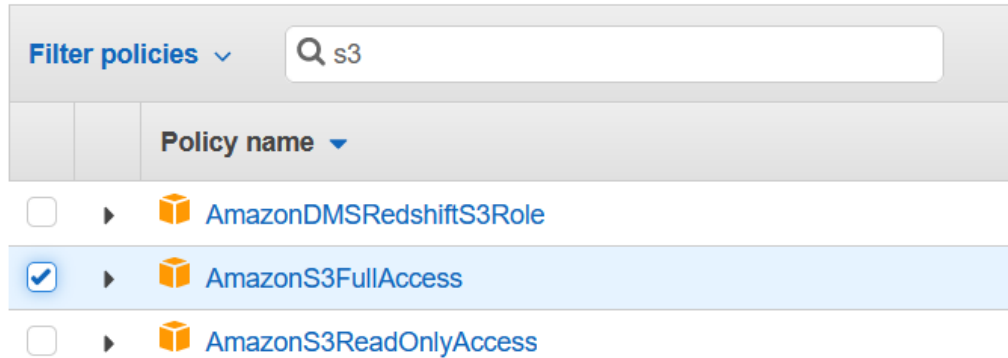
Gambar 45 Review Role

Pada **Role description** sudah disediakan **description default** seperti pada gambar di atas, namun kedepannya anda dapat mengubahnya penjelasan sesuai dengan **role** yang ingin anda buat.

Klik tombol **Create role** di pojok kanan bawah.

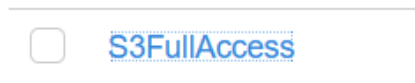
Amazon S3 Full Access

Pada kolom **Filter policies** ketik **S3**, lalu pilih **AmazonS3FullAccess** :



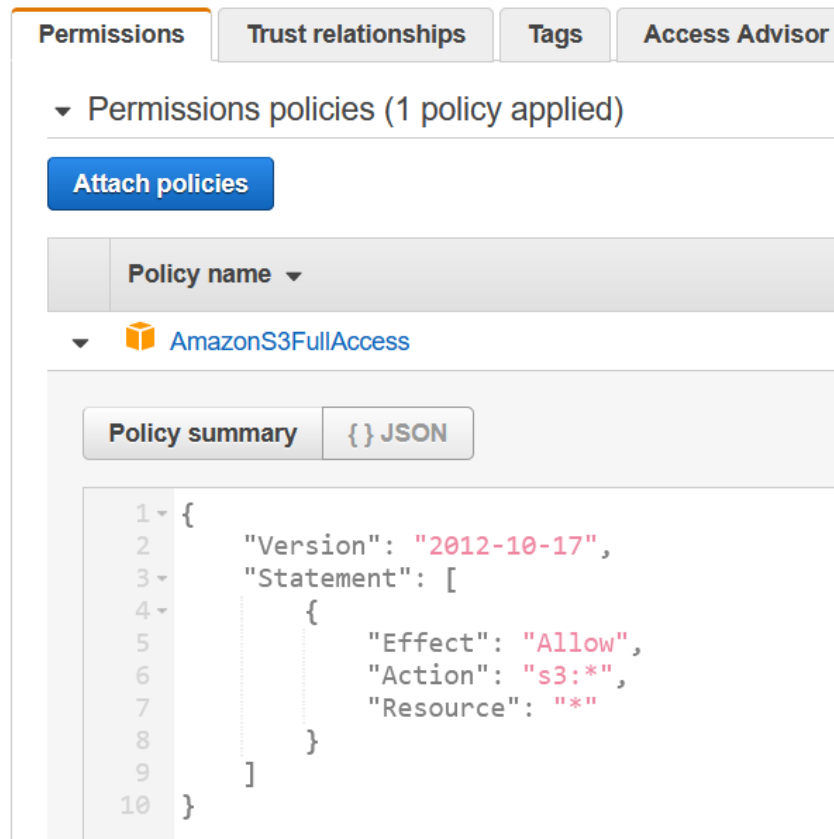
Gambar 46 Add AmazonS3FullAccess Policy

Kita membutuhkan *policy* **AmazonS3FullAccess** agar kita memiliki akses ke seluruh *bucket* mengeksekusi **Simple Storage Service (S3)**. Jika berhasil maka pada kolom Role anda akan melihat :



Gambar 47 Role S3FullAccess

Jika kita klik *role* tersebut pada **Tab Permissions** kita bisa expand **Policy Summary** **AmazonS3FullAccess**, perhatikan gambar di bawah ini :



Gambar 48 S3 Polisy Summary

Policy Statement di atas adalah kita menggunakan :

1. **Version element** tahun 2012 yang menentukan aturan *syntax* untuk memproses *policy statement*.
2. **Statement element** adalah komponen utama dalam *policy statement*, elemen ini dapat memiliki satu *statement* tunggal atau sekumpulan individual *statement* dalam sebuah *array*.
Setiap individual *statement* harus dibungkus didalam (). Pada *policy statement* di atas terdapat satu *statement* saja dengan tanda warna hijau.
3. **Effect element** bersifat *required* digunakan untuk mengizinkan atau menolak suatu akses terhadap suatu *resources* secara eksplisit.

4. **Action element** menjelaskan aksi apa saja yang diizinkan atau ditolak. Setiap layanan AWS memiliki sekumpulan aksi yang bisa diizinkan untuk dieksekusi atau dibatasi.
5. Pada *policy statement* di atas kita mengizinkan AWS S3 untuk bisa melakukan pemanggilan seluruh fungsi pada AWS S3.
6. **Resource element** menjelaskan keseluruhan *statement* yang dibuat dalam *policy statement* dapat digunakan pada *resource* mana saja.
Dapat digunakan untuk sebuah user tertentu atau layanan AWS tertentu. Simbol * artinya kita dapat menggunakannya di seluruh *resources*.

Subchapter 2 – AWS IAM SDK

*The best customer service is if the customer doesn't need to call you,
doesn't need to talk to you*

— Jeff Bezos

Subchapter 2 – Objectives

- Berinteraksi dengan Amazon IAM Menggunakan Javascript SDK
 - Membuat, Mendapatkan, Memodifikasi & Menghapus User
 - Membuat, Mendapatkan, Memodifikasi & Menghapus Group
 - Membuat, Mendapatkan, Memodifikasi & Menghapus Role
-

Pada **subchapter** 2 kita akan mempelajari bagaimana cara berinteraksi dengan **AWS IAM** menggunakan **Javascript SDK**.

1. Create IAM User

Jika kita ingin membuat sebuah **user** dalam **Amazon IAM** buatlah **file javascript** dengan nama **create-user.js**.

Pasang **module** **aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:  
'default' });  
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** yang akan kita gunakan untuk menyimpan nama **user** :

```
var params = {  
  UserName: "Maudy Ayunda"  
};
```

Selanjutnya **object iam** harus memanggil **method createUser()** :

```
iam.createUser(params, function(err, data) {  
  if (err) console.log(err, err.stack); // an error occurred  
  else     console.log(data);           // successful response  
});
```

Parameter pertama digunakan untuk menyimpan **argument params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node create-user  
{  
  ResponseMetadata: { RequestId: '50c9d30e-8c2c-424f-a046-ce98d00ff969' },  
  User: {  
    Path: '/',  
    UserName: 'MaudyAyunda',  
    UserId: 'AIDA3QGK743NKMLUYVWHL',  
    Arn: 'arn:aws:iam::790699697882:user/MaudyAyunda',  
    CreateDate: 2020-05-13T03:46:58.000Z,  
    Tags: []  
  }  
}
```

Gambar 49 Create User Response

Check Created User

Jika kita periksa dalam **IAM Console** maka **user** yang telah kita buat akan tersedia disana :

<input type="checkbox"/>	User name ▼	Groups
<input type="checkbox"/>	gungunfebrianza	DeveloperMarketK MarketkoinS3Grou
<input type="checkbox"/>	MaudyAyunda	None

Gambar 50 New User

Klik **user MaudyAyunda**, kita akan memeriksa **Tab Security Credentials** :


Summary

User ARN	arn:aws:iam::790699697882:user/MaudyAyunda 
Path	/
Creation time	2020-05-13 10:46 UTC+0700
<div>PermissionsGroupsTagsSecurity credentialsAccess Advisor</div>	
Sign-in credentials	

Gambar 51 Security Credentials

Pada informasi **sign-in credentials** kita akan melihat bahwa **console password** milik **user MaudyAyunda** masih **disabled** untuk itu kita harus membuatnya menjadi **enable**.

Sign-in credentials

Summary	<ul style="list-style-type: none">• User does not have console management access
Console password	Disabled Manage
Assigned MFA device	Not assigned Manage
Signing certificates	None 

Gambar 52 Sign-in Credentials

Untuk mengaktifkannya kita perlu memanggil ***method Create login Profile***.

2. Create Login Profile

Jika kita ingin membuat sebuah **user** dalam **Amazon IAM** dapat melakukan **login** pada **IAM Console** buatlah **file javascript** dengan nama **create-login-profile.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** yang akan kita gunakan untuk :

1. Menyimpan **Password**,
2. Pengaturan **PasswordResetRequired** agar user mengubah **password** setelah melakukan **login** untuk pertama kalinya.
3. Dan Username user yang akan diberi **login profile** :

```
var params = {
  Password: "Xc3Esx@R*6v34",
  PasswordResetRequired: true,
  UserName: "MaudyAyunda"
};
```

Selanjutnya **object iam** harus memanggil **method createLoginProfile()** :

```
iam.createLoginProfile(params, function(err, data) {  
    if (err) console.log(err, err.stack); // an error occurred  
    else     console.log(data);           // successful response  
});
```

Parameter pertama digunakan untuk menyimpan **argument params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :



```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node create-login-profile  
{  
  ResponseMetadata: { RequestId: 'fbbb990c-eb15-44f1-8fb4-31c66122a4ca' },  
  LoginProfile: {  
    Username: 'MaudyAyunda',  
    CreateDate: 2020-05-13T03:53:13.000Z,  
    PasswordResetRequired: true  
  }  
}
```

Gambar 53 Create Login Profile Response

Check Login Profile

Jika kita periksa dalam **IAM Console** dalam **Tab Security Credentials**, maka **user MaudyAyunda** memiliki pengaturan **Console Password Enabled** seperti pada gambar di bawah ini :

Sign-in credentials

Summary	• Console sign-in link: https://gungunfebrianza.signin.aws.amazon.com/console 
Console password	Enabled (never signed in) Manage
Assigned MFA device	Not assigned Manage
Signing certificates	None 

Gambar 54 Console Password Enable

Pada **Summary** terdapat **URL** untuk **user MaudyAyunda** melakukan **sign-in**, berikan **link** tersebut kepada **user MaudyAyunda**. Saat pertama kali **user MaudyAyunda** mengunjungi **link** tersebut masukan **username** dan **password** yang telah dibuat :

Sign in as IAM user

Account ID (12 digits) or account alias

IAM user name

Password

Sign in

Gambar 55 New User Sign-in

Change Password

Saat **login** berhasil maudy harus memasukan **password** baru untuk mengganti **password** lama :

AWS account 790699697882

IAM user name MaudyAyunda

Old password

New password

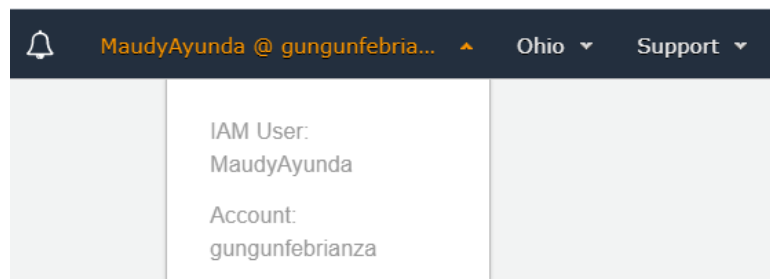
Retype new password

[Confirm password change](#)

[Sign in using root user email](#)

Gambar 56 Change Password

Di pojok kanan atas terdapat informasi **IAM User MaudyAyunda** sebagai penanda :



Gambar 57 User MaudyAyunda

3. Delete Login Profile

Jika kita ingin menghapus **Login Profile** untuk sebuah **user** dalam **Amazon IAM**, buatlah **file javascript** dengan nama **delete-login-profile.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** yang akan kita gunakan untuk menyimpan nama user:

```
var params = {
  UserName: "MaudyAyunda"
};
```

Selanjutnya **object iam** harus memanggil **method deleteLoginProfile()** :

```
iam.deleteLoginProfile(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

Parameter pertama digunakan untuk menyimpan **argument params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :


```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node delete-login-profile
{
  ResponseMetadata: { RequestId: '77e11873-47db-46c9-8325-c17b2596ee0d' }
}
```

Gambar 58 Delete Login Profile Response

Check Deleted Login Profile

Jika kita periksa dalam **IAM Console** dalam **Tab Security Credentials**, maka **user MaudyAyunda** memiliki pengaturan **Console Password Disabled** seperti pada gambar di bawah ini :

Sign-in credentials

Summary	• User does not have console management access
Console password	Disabled Manage
Assigned MFA device	Not assigned Manage
Signing certificates	None 

Gambar 59 Disabled Login Password

4. Get IAM User

Jika kita ingin mendapatkan informasi sebuah **user** dalam sebuah akun yang kita miliki dalam **Amazon IAM**, buatlah **file javascript** dengan nama **get-user.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** dengan **object** kosong.

```
var params = {
  UserName: "MaudyFebrianza",
};
```

Selanjutnya **object iam** harus memanggil **method getUser()** :

```
iam.getUser(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

Parameter pertama digunakan untuk menyimpan **argument params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node get-user
{
  ResponseMetadata: { RequestId: '89d59215-9ded-44d2-bfac-4db0bbd49457' },
  User: {
    Path: '/',
    UserName: 'MaudyFebrianza',
    UserId: 'AIDA3QGK743NKQDRDK2Q0',
    Arn: 'arn:aws:iam::790699697882:user/MaudyFebrianza',
    CreateDate: 2020-05-13T04:44:03.000Z,
    Tags: []
  }
}
```

Gambar 60 Get User Response

5. List User

Jika kita ingin mendapatkan seluruh **user** dalam sebuah akun yang kita miliki dalam **Amazon IAM**, buatlah **file javascript** dengan nama **list-user.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** dengan **object** kosong.

```
var params = {
};
```

Selanjutnya **object iam** harus memanggil **method listUsers()** :

```
iam.listUsers(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

Parameter pertama digunakan untuk menyimpan **argument** **params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node list-user
{
  ResponseMetadata: { RequestId: '3291655d-1bba-4714-a11b-3a9d4d6038a8' },
  Users: [
    {
      Path: '/',
      UserName: 'gungunfebrianza',
      UserId: 'AIDA3QGK743NJLPSAF77R',
      Arn: 'arn:aws:iam::790699697882:user/gungunfebrianza',
      CreateDate: 2020-05-11T02:07:28.000Z,
      Tags: []
    },
    {
      Path: '/',
      UserName: 'temporaryaws',
      UserId: 'AIDA3QGK743NPQI762NW7',
      Arn: 'arn:aws:iam::790699697882:user/temporaryaws',
      CreateDate: 2020-03-23T02:31:59.000Z,
      Tags: []
    }
  ],
  IsTruncated: false
}
```

Gambar 61 List User Response

6. Update User

Jika kita ingin melakukan **update** pada sebuah **user**, yaitu **update** untuk mengubah sebuah **username** sebuah **user** dalam **Amazon IAM** buatlah **file javascript** dengan nama **update-user.js**.

Pasang **module** **aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** yang akan kita gunakan untuk menyimpan nama **user** :

```
var params = {
  NewUserName: "MaudyFebrianza",
  UserName: "MaudyAyunda",
};
```

Selanjutnya **object iam** harus memanggil **method** **createUser()** :

```
iam.updateUser(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

```
});
```

Parameter pertama digunakan untuk menyimpan **argument** **params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node update-user  
{  
  ResponseMetadata: { RequestId: '58b5a913-4659-4167-8414-f5f414b61cba' }  
}
```

Check Updated User

Jika kita periksa **user** **MaudyAyunda** dalam **IAM Console** maka **user** yang telah kita **update** akan berubah disana :

<input type="checkbox"/>	User name ▾	Groups
<input type="checkbox"/>	gungunfebrianza	DeveloperMarketKoin and MarketkoinS3Group
<input type="checkbox"/>	MaudyFebrianza	None
<input type="checkbox"/>	temporaryaws	developers-marketkoin

Gambar 62 Update User Success

7. Delete IAM User

Jika kita ingin menghapus sebuah **user** dalam **Amazon IAM**, buatlah **file javascript** dengan nama **delete-user.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** yang akan kita gunakan untuk menyimpan nama **user**:

```
var params = {
  UserName: "MaudyAyunda"
};
```

Selanjutnya **object iam** harus memanggil **method deleteUser()** :

```
iam.deleteUser(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

Parameter pertama digunakan untuk menyimpan **argument params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node delete-user
{
  ResponseMetadata: { RequestId: 'f2e712b8-269f-46e8-bad9-1e05196058ff' }
}
```

Gambar 63 Delete User Response

Check Deleted User

Jika kita periksa **user MaudyAyunda** dalam **IAM Console** maka **user** yang telah kita hapus tidak akan tersedia disana :

Find users by username or access key		
<input type="checkbox"/>	User name ▼	Groups
<input type="checkbox"/>	gungunfebrianza	DeveloperMarketKoin , MarketkoinS3Group , and 1 more
<input type="checkbox"/>	temporaryaws	developers-marketkoin

Gambar 64 Check Deleted User

8. Create Group

Jika kita ingin membuat sebuah **group** dalam **Amazon IAM** buatlah **file javascript** dengan nama **create-group.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** yang akan kita gunakan untuk menyimpan nama grup :

```
var params = {
  GroupName: "TesterAppX"
};
```

Selanjutnya **object iam** harus memanggil **method createGroup()** :

```
iam.createGroup(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

Parameter pertama digunakan untuk menyimpan **argument params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node create-group
{
  ResponseMetadata: { RequestId: 'c8fb3ae3-d8e0-4c1e-b9b1-e9e7d86f2e84' },
  Group: {
    Path: '/',
    GroupName: 'TesterAppX',
    GroupId: 'AGPA3QGK743NNL2NZXIFD',
    Arn: 'arn:aws:iam::790699697882:group/TesterAppX',
    CreateDate: 2020-05-13T03:09:15.000Z
  }
}
```

Gambar 65 Create Group Response

Check Created Group

Jika kita periksa dalam **IAM Console** maka **group** yang telah kita buat akan tersedia disana :

<input type="checkbox"/>	Group Name ↕	Users	Inline Policy	Creation Time ↕
<input type="checkbox"/>	DeveloperMarketKoin	1		2020-05-11 09:02 UTC+0700
<input type="checkbox"/>	developers-marketkoin	1		2019-02-10 00:35 UTC+0700
<input type="checkbox"/>	MarketkoinS3Group	1		2019-11-19 09:49 UTC+0700
<input checked="" type="checkbox"/>	TesterAppX	0		2020-05-13 10:09 UTC+0700

Gambar 66 Group List

9. Add User to Group

Jika kita ingin membuat sebuah **user** bergabung dalam sebuah **group** dalam **Amazon IAM** buatlah **file javascript** dengan nama **add-user-to-group.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** yang akan kita gunakan untuk menyimpan user kedalam sebuah grup :

```
var params = {
  GroupName: "TesterAppX",
  UserName: "gungunfebrianza"
};
```

Selanjutnya **object iam** harus memanggil **method addUserToGroup()** :

```
iam.addUserToGroup(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

```
});
```

Parameter pertama digunakan untuk menyimpan **argument params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node add-user-to-group  
{  
  ResponseMetadata: { RequestId: '8085826e-551c-4fb4-a6e8-ccec625c3bfb' }  
}
```

Gambar 67 Add User to Group Response

Check Added User

Jika kita cek dalam **IAM Console** maka **user** yang telah kita buat akan tersedia dalam **group TesterAppX** :


<input type="checkbox"/>	Group Name ↕	Users
<input type="checkbox"/>	DeveloperMarketKoin	1
<input type="checkbox"/>	developers-marketkoin	1
<input type="checkbox"/>	MarketkoinS3Group	1
<input type="checkbox"/>	TesterAppX	1

Gambar 68 TesterAppX User

Jika kita klik **group TesterAppX** maka kita akan menemukan bahwa **user gungunfebrianza** berhasil masuk kedalam **group TesterAppX**.

UsersPermissionsAccess Advisor

This view shows all users in this group: 1 User

User	Actions
 gungunfebrianza	Remove User from Group

Gambar 69 User Gun Gun Febrianza

10. Remove User From Group

Jika kita ingin membuat sebuah **user** dicabut dari dalam sebuah **group** dalam **Amazon IAM** buatlah **file javascript** dengan nama **remove-user-from-group.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** yang akan kita gunakan untuk mencabut **user** kedalam sebuah grup :

```
var params = {
  GroupName: "TesterAppX",
  UserName: "gungunfebrianza"
};
```

Selanjutnya **object iam** harus memanggil **method removeUserFromGroup()** :

```
iam.removeUserFromGroup(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
```

```
});
```

Parameter pertama digunakan untuk menyimpan **argument params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node remove-user-from-group  
{  
  ResponseMetadata: { RequestId: '1666a9c6-bf34-4cee-8d35-e5d849212fb6' }  
}
```

Gambar 70 Remove User From Group

Check Removed User

Jika kita cek dalam **IAM Console** maka **user** yang telah kita buat akan tersedia dalam **group TesterAppX** :

<input type="checkbox"/>	Group Name ↕	Users
<input type="checkbox"/>	DeveloperMarketKoin	1
<input type="checkbox"/>	developers-marketkoin	1
<input type="checkbox"/>	MarketkoinS3Group	1
<input type="checkbox"/>	TesterAppX	0

Gambar 71 TesterAppX User

Jika kita klik **group TesterAppX** maka kita akan menemukan bahwa **user gungunfebrianza** berhasil masuk kedalam **group TesterAppX**.

11. List Group

Jika kita ingin mendapatkan seluruh **group** dalam sebuah akun yang kita miliki dalam **Amazon IAM**, buatlah **file javascript** dengan nama **list-group.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** dengan **object** kosong.

```
var params = {
};
```

Selanjutnya **object iam** harus memanggil **method listGroups()** :

```
iam.listGroups(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

Parameter pertama digunakan untuk menyimpan **argument** `params` yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
{
  Path: '/',
  GroupName: 'MarketkoinS3Group',
  GroupId: 'AGPA3Q GK743NFV7LMJ32I',
  Arn: 'arn:aws:iam::790699697882:group/MarketkoinS3Group',
  CreateDate: 2019-11-19T02:49:00.000Z
},
{
  Path: '/',
  GroupName: 'TesterAppX',
  GroupId: 'AGPA3Q GK743NNL2NZXIFD',
  Arn: 'arn:aws:iam::790699697882:group/TesterAppX',
  CreateDate: 2020-05-13T03:09:15.000Z
}
],
IsTruncated: false
```

Gambar 72 Response List Group

12. Delete Group

Jika kita ingin menghapus sebuah **group** dalam **Amazon IAM**, buatlah **file javascript** dengan nama **delete-group.js**.

Pasang **module aws-sdk** :

```
const AWS = require('aws-sdk');
```

Gunakan **Shared File Credentials** untuk mengamankan akun **IAM User** yang kita buat :

```
const credentials = new AWS.SharedIniFileCredentials({ profile:
'default' });
AWS.config.credentials = credentials;
```

Buat **Object IAM** :

```
var iam = new AWS.IAM({apiVersion: '2010-05-08'});
```

Buat **parameter** yang akan kita gunakan untuk menyimpan nama **user**:

```
var params = {
  GroupName: "TesterAppX"
};
```

Selanjutnya **object iam** harus memanggil **method deleteGroup()** :

```
iam.deleteGroup(params, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
```

Parameter pertama digunakan untuk menyimpan **argument params** yang telah kita sebelumnya dan **parameter** kedua kita menggunakan **function** untuk menampilkan pesan berhasil atau **error**.

Jika berhasil maka kita akan mendapatkan respon seperti di bawah ini :

```
C:\Users\Gun Gun Febrianza\Pictures\Belajar-Dengan-Jenius-AWS-IAM\src>node delete-group  
{  
  ResponseMetadata: { RequestId: '5de426da-755d-44fc-a4c9-a3778cba4206' }  
}
```

Gambar 73 Delete Group Response

Check Deleted Group

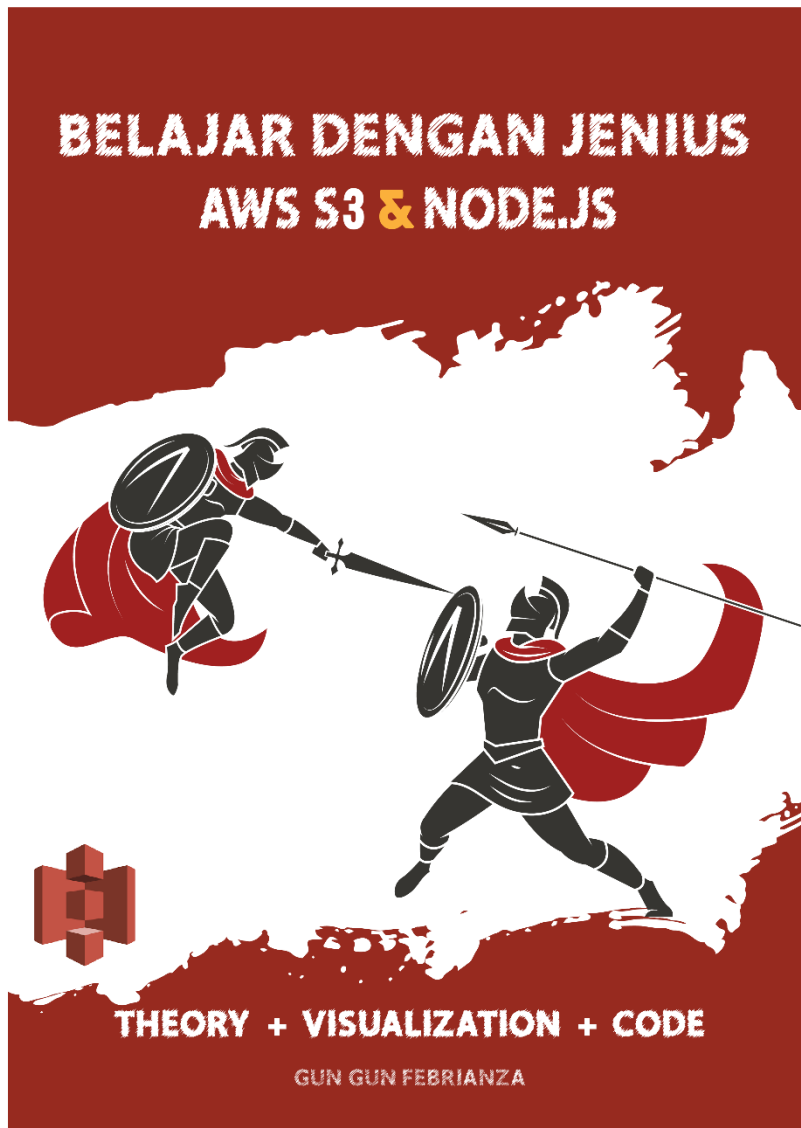
Jika kita periksa **group TesterAppX** dalam **IAM Console** maka **groups** yang telah kita hapus tidak akan tersedia disana :

<input type="checkbox"/>	Group Name ↕	Users
<input type="checkbox"/>	DeveloperMarketKoin	1
<input type="checkbox"/>	developers-marketkoin	1
<input type="checkbox"/>	MarketkoinS3Group	1

Gambar 74 Check Deleted Group

Belajar Dengan Jenius AWS S3 & Node.js

Setelah anda mempelajari buku ini anda selanjutnya dapat memprogram Amazon S3 Menggunakan Node.js, tunggu buku ini launching ya :



Tentang Penulis



Penulis adalah Mahasiswa lulusan Universitas Komputer Indonesia (UNIKOM Bandung). Semenjak masuk ke bangku kuliah sudah memiliki *habit* membuat karya tulis di bidang pemrograman, *habit* ini mengantarkan penulis untuk membangun skripsi di sektor **Compiler Construction** dengan judul "Kompiler untuk pemrograman dalam Bahasa Indonesia".

Skripsi yang fokus membuat bahasa pemrograman berbahasa Indonesia, dengan paradigma *object-oriented programming*.

Penulis adalah *Founder* sekaligus (*CTO*) *Market Koin Indonesia*, sebuah *platform Trading Engine* tempat masyarakat dapat membeli dan menjual *bitcoins*, *ethereum* dan *alternative cryptocurrency* lainnya.

Dari tahun 2017 penulis sudah mendapatkan investasi dan pembiayaan *equity financing* dari pengusaha-pengusaha di Eropa untuk mengembangkan *platform* Market Koin & Blockchain.

Riset-riset yang sedang penulis kembangkan adalah teknologi *Cross-border Payment*, *Cryptocurrency Arbitrage System*, *High-Frequency Trading (HFT) Engine*, dan *platform* terbaru yang sedang dikembangkan adalah **Lightning Bank**.

Sebuah teknologi yang penulis kembangkan untuk membantu perbankan di *Denmark* dan Indonesia agar bisa bertransaksi secara *instant* dan biaya transaksi di bawah 3% sesuai target *Sustainable Development Goals (SDG) United Nation*.

Penulis juga aktif dalam kegiatan literasi finansial untuk masyarakat dan pengembangan Industri Maritim Indonesia, tempat penulis membangun usaha di sektor Industri Udang (*vannamei*).