

# AWS Basic Server Production Part 3

## Detail Materi

**Indikator :**  
Dapat mengaplikasikan CDN menggunakan Amazon CloudFront, mengarahkan instance dan layanan pada AWS ke domain dan menggunakan Amazon RDS untuk database aplikasi

Konsep Dasar Amazon CloudFront, Route53 dan Amazon RDS

Konfigurasi CDN pada Amazon CloudFront

Konfigurasi Route53 dan Domain Provider

Membuat Database RDS dan Replica Database

Integrasi Database RDS ke Aplikasi Pesbuk

## Modul Sekolah DevOps Cilsy

Hak Cipta © 2020 **PT. Cilsy Fiolution Indonesia**

Hak Cipta dilindungi Undang-Undang. Dilarang memperbanyak atau memindahkan sebagian atau seluruh isi buku ini dalam bentuk apapun, baik secara elektronis maupun mekanis, termasuk mecopy, merekam atau dengan sistem penyimpanan lainnya, tanpa izin tertulis dari Penulis dan Penerbit.

Penulis : Adi Saputra, Irfan Herfiandana, Estu Fardani & Tresna Widiyaman  
Editor: Taufik Maulana, Muhammad Fakhri Abdillah, Rizal Rahman & Tresna

Widiyaman

**Revisi Batch 9**

Penerbit : **PT. Cilsy Fiolution Indonesia**

Web Site : <https://cilsyfiolution.com> , <https://devops.cilsy.id>

### Sanksi Pelanggaran Pasal 113 Undang-undang Nomor 28 Tahun 2014 tentang Hak Cipta

1. Setiap orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam pasal 9 ayat (1) huruf i untuk penggunaan secara komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan atau pidana denda paling banyak Rp100.000.000,00 (seratus juta rupiah).
2. Setiap orang yang dengan tanpa hak dan atau tanpa izin pencipta atau pemegang hak cipta melakukan pelanggaran hak ekonomi pencipta sebagaimana dimaksud dalam pasal 9 ayat (1) huruf c, huruf d, huruf f, dan atau huruf h, untuk penggunaan secara komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah)
3. Setiap orang yang dengan tanpa hak dan atau tanpa izin pencipta atau pemegang hak melakukan pelanggaran hak ekonomi pencipta sebagaimana dimaksud dalam pasal 9 ayat (1) huruf a, huruf b, huruf e, dan atau huruf g, untuk penggunaan secara komersial dipidana dengan pidana penjara paling lama 4 (empat) tahun dan atau pidana denda paling banyak Rp1.000.000.000,00 (satu miliar rupiah)
4. Setiap orang yang memenuhi unsur sebagaimana dimaksud pada ayat (3) yang dilakukan dalam bentuk pembajakan, dipidana dengan pidana penjara paling lama 10 (sepuluh) tahun dan atau pidana denda paling banyak Rp4.000.000.000,00 (empat miliar rupiah)



## Daftar Isi

Cover Modul Sekolah DevOps Cilsy.....	2
6. AWS Basic Server Production Part 3.....	5
Learning Outcomes.....	5
Outline Materi.....	5
6.1. Roadmap Pembelajaran.....	6
6.2. Pengenalan Elastic Beanstalk.....	7
6.2.1. Apa itu Elastic Beanstalk ?.....	7
6.2.2. Aplikasi yang didukung Elastic Beanstalk.....	9
6.2.3. Exercise.....	10
6.3. Pengenalan Amazon S3.....	10
6.3.1. Apa itu Amazon S3 ?.....	10
6.3.2. Fitur Amazon S3.....	11
6.4. Praktek Amazon S3.....	12
6.4.1. Membuat Bucket S3.....	12
6.4.2. Konfigurasi Public Access.....	18
6.4.3. Konfigurasi Hosting S3.....	22
6.4.4. Mounting Amazon S3 kedalam Instance.....	24
6.4.5. Exercise.....	27
6.5. Pengenalan Amazon CloudFront.....	27
6.5.1. Apa itu CDN ?.....	27
6.5.2. Macam-macam CDN.....	28
6.5.3. Apa itu Amazon CloudFront ?.....	29
6.6. Praktek Amazon CloudFront.....	30
6.6.1. Setup S3 Bucket.....	30
6.6.2. Setup Amazon CloudFront dengan S3.....	30
6.6.3. Setup Amazon CloudFront dengan Load Balancer.....	34
6.6.4. Exercise.....	37
6.7. Pengenalan Route53.....	38

6.7.1. Apa itu Route53 ? .....	38
6.8. Praktek Route53.....	39
6.8.1. Persiapan Domain.....	39
6.8.2. Konfigurasi Domain di Route53.....	40
6.8.3. Konfigurasi Domain Provider.....	42
6.8.4. Integrasi Instance ke Route53.....	45
6.8.5. Setup Subdomain dan Layanan AWS Lainnya.....	47
6.8.6. Exercise.....	50
6.9. Amazon RDS (Relational Database Service).....	50
6.9.1. Apa itu Amazon RDS ? .....	50
6.9.2. Fungsi dan kegunaan Amazon RDS.....	51
6.9.3. Exercise.....	52
6.10. Setup Amazon RDS.....	53
6.10.1. Launch DB Instance.....	53
6.10.2. Replikasi Amazon RDS.....	61
6.10.3. Exercise.....	65
6.11. Setup Web Aplikasi dengan RDS.....	65
6.11.1. Migration Database.....	65
6.11.2. Konfigurasi Web Aplikasi.....	66
6.11.3. Exercise.....	68
6.12. Summary.....	68

## 6.

# AWS Basic Server Production Part 3

## Learning Outcomes

Setelah selesai mempelajari bab ini, peserta mampu :

- 1.** Memahami Konsep Amazon CloudFront, Route53 dan Amazon RDS
- 2.** Konfigurasi CDN Amazon CloudFront dengan S3 dan Load Balancer
- 3.** Melakukan Konfigurasi Route53 dan Domain Provider
- 4.** Membuat Database RDS dan Replica Database
- 5.** Melakukan Integrasi Database RDS ke Aplikasi Pesbuk

## Outline Materi

- 1.** Pengenalan CDN dan Amazon CloudFront
- 2.** Konfigurasi Amazon CloudFront
- 3.** Pengenalan Route53
- 4.** Konfigurasi Route53 dan Domain Provider
- 5.** Memasangkan Domain ke Instance
- 6.** Pengenalan Amazon RDS
- 7.** Membuat Database RDS dan Replica Database
- 8.** Integrasi Database ke Aplikasi

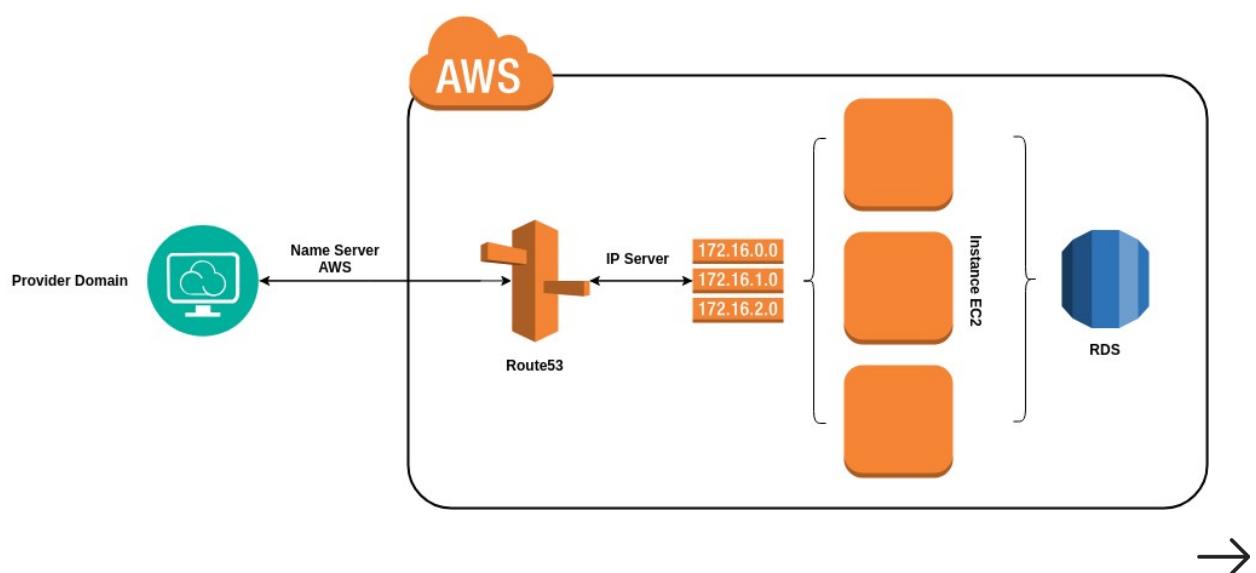
## 6.1. Roadmap Pembelajaran



*Roadmap Pembelajaran*

Jika kita lihat pada roadmap yang sudah kita bahas sebelumnya, kita kemarin sudah melewati **tahap pertama sampai dengan tahap ke-enam dan tujuh** yaitu **Elastic Beanstalk dan Juga Amazon S3**. Sekarang kita masuk **tahap ke-delapan** yaitu **Menggunakan CloudFront, Amazon Route53 dan juga Database RDS**.

Pada pembahasan kali ini, kita akan coba untuk mempelajari bagaimana membuat sebuah server yang menggunakan CDN agar akses bisa lebih cepat, mengarahkan domain ke instance yang kita miliki menggunakan route53 dan juga mengarahkan database aplikasi kita ke database dari Amazon RDS. Berikut merupakan topologi yang akan kita jalankan.



*Topologi yang akan dijalankan*

## 6.2. Pengenalan Elastic Beanstalk

### 6.2.1. Apa itu Elastic Beanstalk ?

**AWS Elastic Beanstalk** adalah sebuah *Platform As a Service* (PAaaS) yang memberikan kemudahan dan manfaat bagi developer untuk mengembangkan dan mengatur aplikasi di AWS. Seorang Developer dapat menggunakan aplikasi mereka di AWS tanpa Elastic Beanstalk akan tetapi mereka akan menghabiskan waktu untuk memilih dan menyatukan layanan dari beragam pilihan di dalam sistem lingkungan AWS tersebut.



*Logo Elastic Beanstalk*

Dengan Elastic Beanstalk, memungkinkan mereka dengan cepat menyebarluaskan dan mengelola aplikasi di AWS dan mengabstraksi layer infrastruktur dari developer. Meskipun begitu, developer sama sekali tidak dibatasi oleh Elastic Beanstalk. Mereka masih mendapatkan konfigurasi granular atas aplikasi mereka.

Jika kita analogikan, misalkan kita akan merakit sebuah komputer. Maka ada dua cara yang bisa kita lakukan untuk mencapainya, yakni :



*Ilustrasi analogi memilih komputer*

## 1. Pergi ke Gudang Komputer

Gudang komputer besar dan membingungkan dengan deretan demi deretan rak yang diletakkan di depan kalian. Kalian akan merasa kewalahan dengan banyaknya pilihan tetapi sangat perlu untuk memiliki semua bagian yang dibutuhkan untuk membangun komputer.

Kalian akan dengan hati-hati menavigasi di antara rak-rak untuk memilih CPU, SSD untuk penyimpanan, perangkat lunak sistem operasi dll. Lalu kalian mulai merakitnya bersama untuk mendapatkan produk akhir, yaitu komputer dambaan kalian.

Ini sama halnya dengan membangun aplikasi pada AWS tanpa Elastic Beanstalk. Kalian memilih seberapa kuat yang kalian inginkan untuk server EC2, jenis layanan penyimpanan apa dan seberapa besar ruang penyimpanannya, dll. Kemudian kalian mulai menyatukannya untuk membuat aplikasi aktif dan berjalan.

## 2. Pergi ke Toko Ritel Komputer

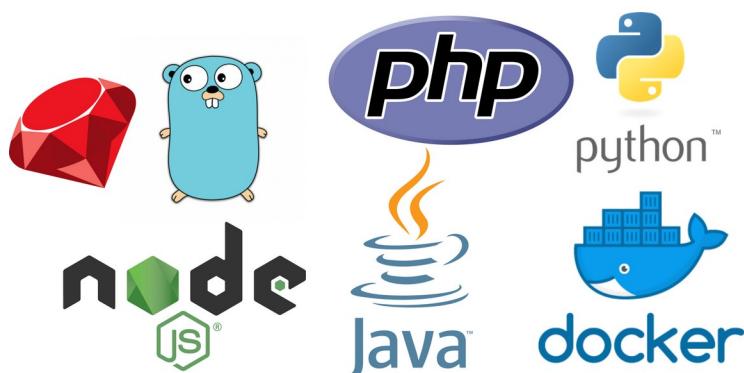
Toko ritel komputer memiliki bagian depan toko yang lebih ramah konsumen. Ini jauh lebih bersih dan ber-AC. Di sepanjang gang, kita bisa menemukan salesman sedang tersenyum pada kita, siap untuk menjawab pertanyaan apa pun.

Kalian memberi tahu pramuniaga kebutuhan komputasi (misalkan komputer Game) dan kalian berjalan keluar dengan komputer yang dapat memenuhi kebutuhan itu. Sesederhana itu. Kalian tidak perlu mengambil manual dan berkeliling mencari di rak-rak toko tersebut.

Dan jika kita tidak puas dengan spesifikasi yang biberikan, kita memiliki kebebasan untuk meningkatkan dan mengganti bagian tersebut. Seperti inilah kehidupan devops dengan Elastic Beanstalk.

### 6.2.2. Aplikasi yang didukung Elastic Beanstalk

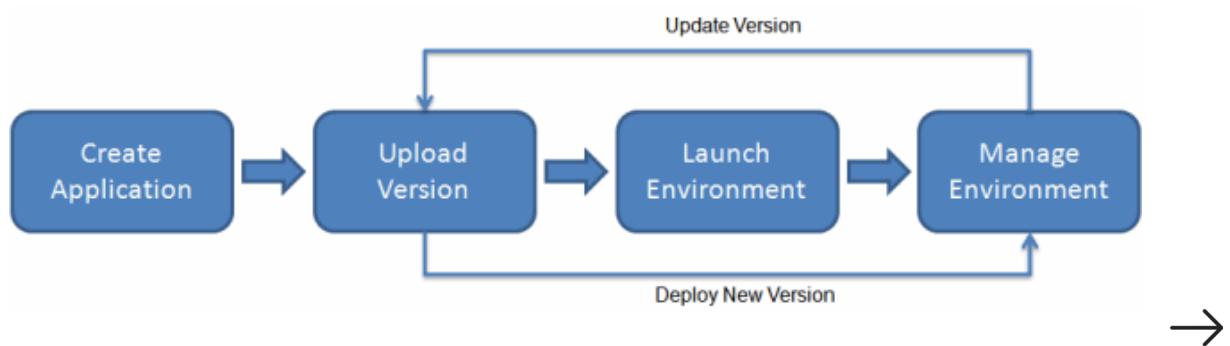
Elastic Beanstalk sangat ideal jika kita memiliki aplikasi web **PHP, Java, Python, Ruby, Node.js, .NET, Go, atau Docker**. Elastic Beanstalk menggunakan layanan AWS inti seperti *Amazon EC2*, *Amazon Elastic Container Service (Amazon ECS)*, *Auto Scaling* dan *Elastic Load Balancer* untuk mendukung aplikasi yang perlu skala besar dalam melayani jutaan pengguna.



Ilustrasi Aplikasi yang didukung Elastic Beanstalk

Untuk menggunakan **Elastic Beanstalk**, pertama kita perlu membuat aplikasi dan mengunggah versi aplikasi tersebut dalam bentuk bundel aplikasi (misalkan zip) ke Elastic Beanstalk, kemudian memberikan beberapa informasi tentang aplikasi tersebut.

Elastic Beanstalk akan secara otomatis meluncurkan environment dan membuat konfigurasi dari resource AWS yang diperlukan untuk menjalankan aplikasi kita. Setelah environment berhasil diluncurkan, selanjutnya kita dapat mengelola environment dan menyebarkan aplikasi dengan versi terbarunya. Berikut diagram ilustrasi dari alur kerja Elastic Beanstalk tersebut.



*Diagram alur kerja amazon beanstalk*

Setelah kita membuat dan menyebarkan aplikasi kita, semua informasi tentang aplikasi tersebut baik metrik, event, dan status environment akan bisa kita akses. Elastic Beanstalk sendiri memiliki dashboard yang nantinya akan memudahkan developer dan administrator untuk mengelola aplikasi mereka tanpa harus khawatir dengan infrastructure-nya.

### 6.2.3. Exercise

1. Apa yang menjadi kelebihan dari elastic beanstalk dibanding dengan EC2 ?

## 6.3. Pengenalan Amazon S3

### 6.3.1. Apa itu Amazon S3 ?

**Amazon Simple Storage Service (Amazon S3)** merupakan layanan dari AWS yang berfungsi untuk menyimpan data secara cloud layaknya google drive ataupun dropbox. Amazon S3 memiliki antarmuka layanan web sederhana yang dapat kita gunakan untuk menyimpan dan mengambil sejumlah data, kapan saja, dari mana saja.

**Amazon S3** memberikan akses kepada developer ke infrastruktur penyimpanan data yang sangat scalable, andal, cepat, dan murah yang digunakan Amazon untuk menjalankan jaringan globalnya sendiri dari situs web.

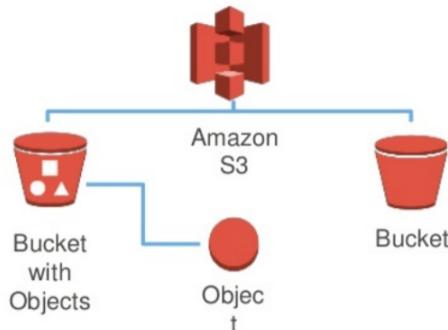


*Logo Amazon S3*

**Amazon S3** menggunakan system bucket, dimana ketika kita menyimpan atau menyetor sebuah data, kita akan menyimpannya kedalam sebuah ember (disebut bucket didalam S3). Kita bisa membuat banyak bucket didalam



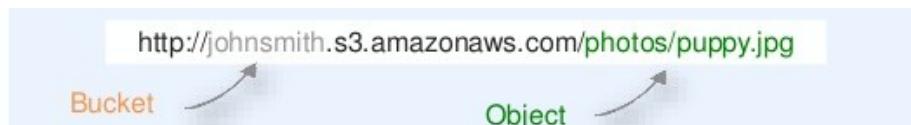
amazon S3 layaknya membuat sebuah hardisk baru. Kita bisa memiliki bucket hingga 100 buah.



*Ilustrasi penggunaan bucket di Amazon S3*

Setiap objek disimpan sebagai file dengan metadata yang disertakan dan diberikan nomor ID. Aplikasi menggunakan nomor ID ini untuk mengakses objek. S3 memungkinkan pelanggan untuk mengunggah, menyimpan, dan mengunduh hampir semua file atau objek yang berukuran hingga lima terabyte (TB), dengan ukuran tunggal terbesar dibatasi pada lima gigabytes (GB).

Salah satu keistimewaan lain dari amazon S3 adalah kita dapat membuat bucket menjadi sebuah hosting, dimana objek atau file yang ada pada bucket bisa kita panggil menggunakan alamat URL.



*Ilustrasi sebuah objek gambar yang diakses dengan alamat S3*

### 6.3.2. Fitur Amazon S3

Berikut merupakan beberapa fitur dan manfaat yang diberikan oleh layanan Amazon S3 sebagai berikut.

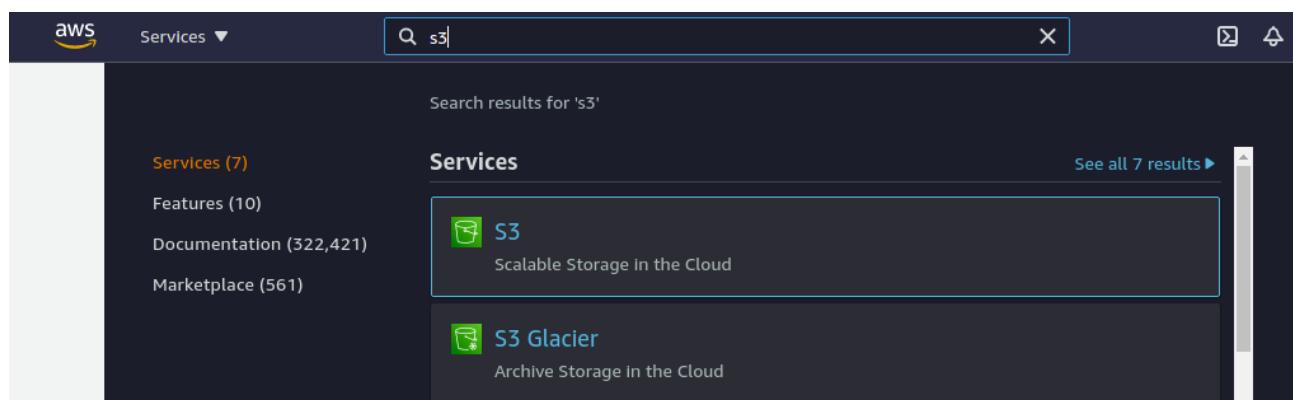
- Memungkinkan penyimpanan data dan objek tidak terbatas untuk sebagian besar tipe data dalam berbagai format. Satu set data yang disimpan, yang merupakan objek, berkisar dari 1 B hingga 5 TB.

- Menyediakan Reduced Redundancy Storage (RRS), yang mengurangi latensi dengan menyimpan data dalam bucket yang terpisah secara regional. Ini menghemat sumber daya dan memfasilitasi efisiensi aplikasi untuk pengguna di lokasi yang tersebar secara geografis.
- Otentikasi yang kuat memastikan keamanan data yang disimpan di wilayah.
- Menyediakan Transfer Antar Negara Representasi (REST) dan Protokol Layanan Web Access Object Sederhana (SOAP) yang dibangun untuk beroperasi dengan semua jenis toolkit pengembangan Web.

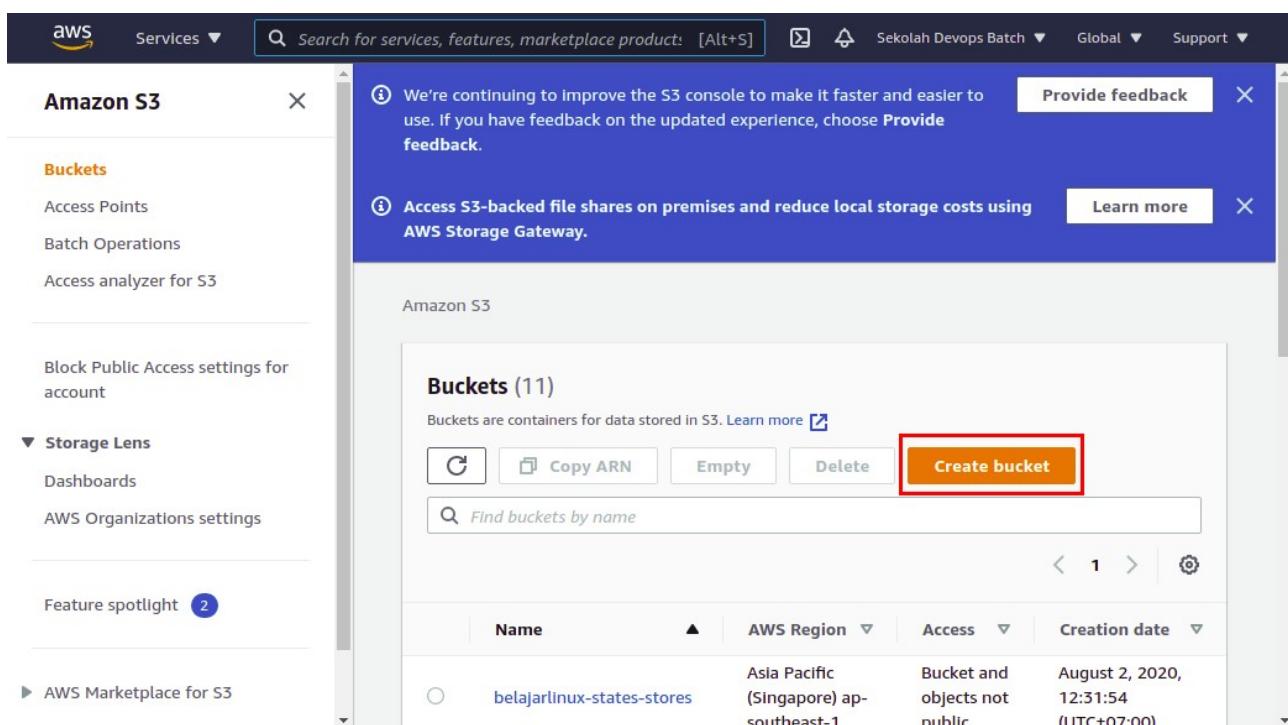
## 6.4. Praktek Amazon S3

### 6.4.1. Membuat Bucket S3

Pada bagian ini kita akan coba untuk membuat sebuah **bucket S3** baru, hal pertama yang harus dilakukan adalah search **S3** lalu pilih **S3** seperti berikut.

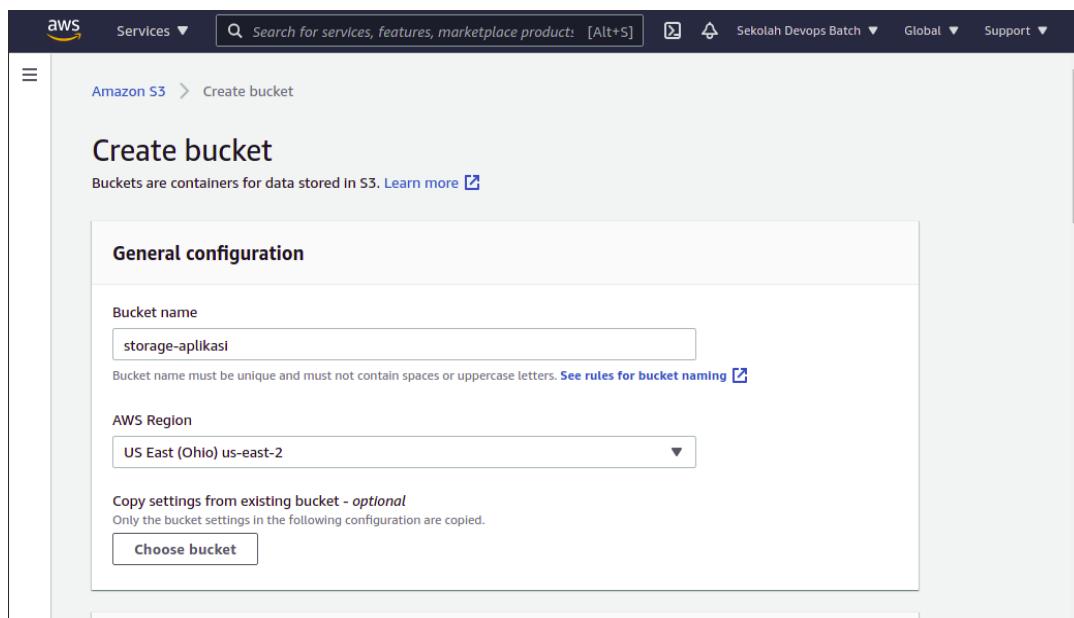


Kita akan diarahkan kedalam dashboard menu amazon S3 seperti berikut. Untuk membuat sebuah bucket baru, kita pilih tombol **Create bucket**.



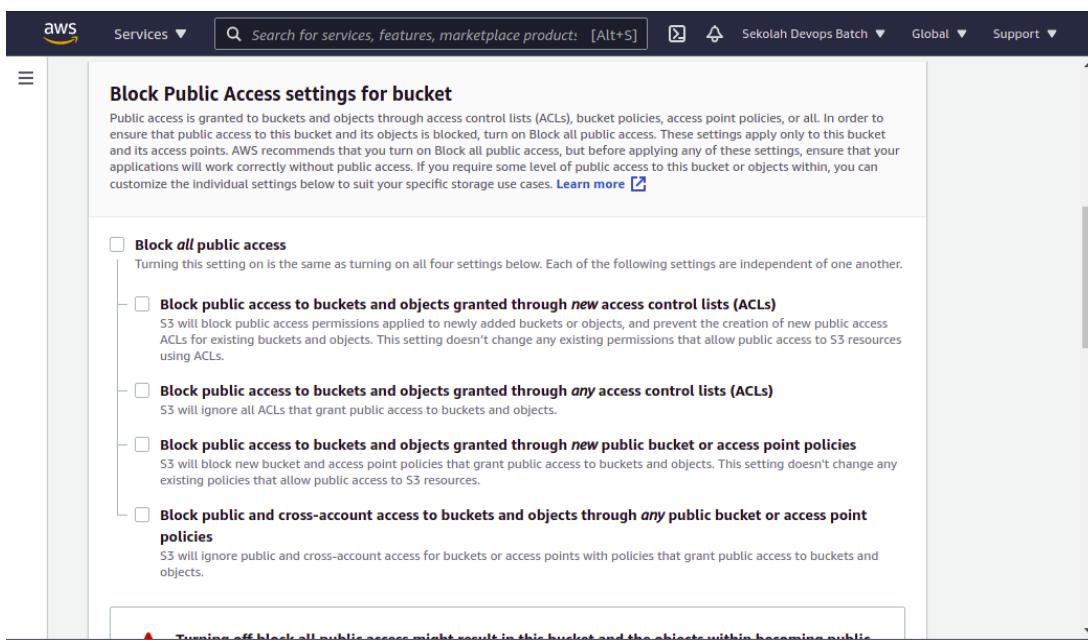
The screenshot shows the AWS S3 service page. On the left, there's a sidebar with options like 'Buckets', 'Access Points', 'Batch Operations', 'Access analyzer for S3', 'Block Public Access settings for account', 'Storage Lens' (with 'Dashboards' and 'AWS Organizations settings'), 'Feature spotlight', and 'AWS Marketplace for S3'. The main area is titled 'Amazon S3' and shows a message about improving the console. Below that, it says 'Access S3-backed file shares on premises and reduce local storage costs using AWS Storage Gateway.' A blue bar at the top has 'Provide feedback' and 'Learn more' buttons. The main content area is titled 'Buckets (11)' and shows a table with one row. The table columns are 'Name', 'AWS Region', 'Access', and 'Creation date'. The single row shows 'belajarlinux-states-stores', 'Asia Pacific (Singapore) ap-southeast-1', 'Bucket and objects not public', and 'August 2, 2020, 12:31:54 (UTC+07:00)'. A large orange 'Create bucket' button is highlighted with a red box.

Selanjutnya diarahkan untuk mengisi nama bucket dan region yang akan kita gunakan. Nama bucket ini juga nantinya akan menjadi alamat domain static dari bucket tersebut



The screenshot shows the 'Create bucket' wizard. The first step is 'General configuration'. It has fields for 'Bucket name' (containing 'storage-aplikasi') and 'AWS Region' (set to 'US East (Ohio) us-east-2'). Below these, there's a note about copying settings from an existing bucket and a 'Choose bucket' button. The URL in the browser is 'Amazon S3 > Create bucket'.

Selanjutnya scroll ke bawah untuk men-uncheck ***Block all public access*** agar object pada bucket bisa diakses secara publik.



Lalu check peringatan "***I acknowledge that ...***" untuk mengkonfirmasi bucket untuk dapat diakses secara publik seperti gambar berikut.

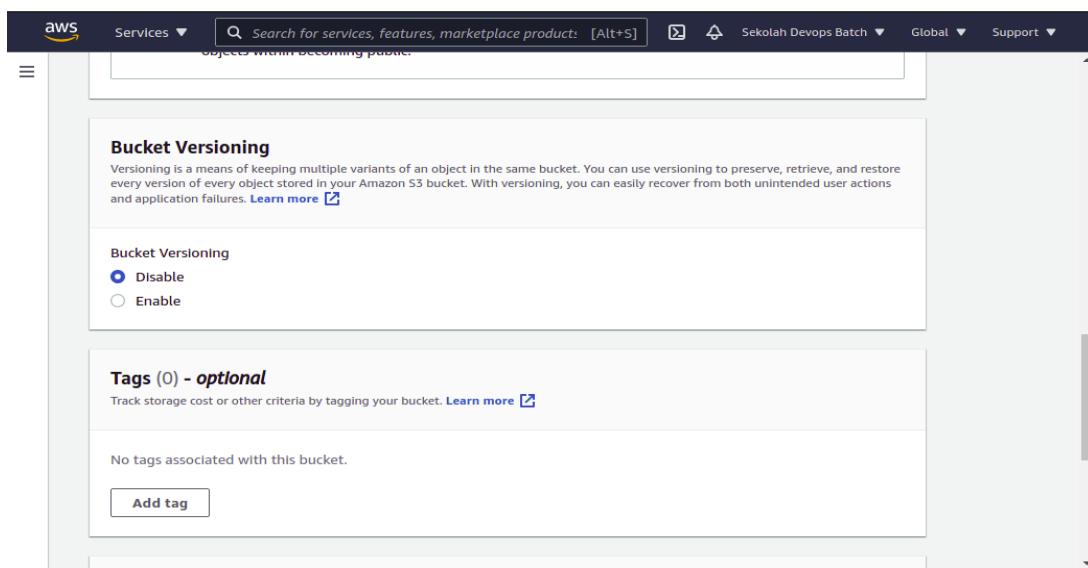


**Turning off block all public access might result in this bucket and the objects within becoming public**  
 AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

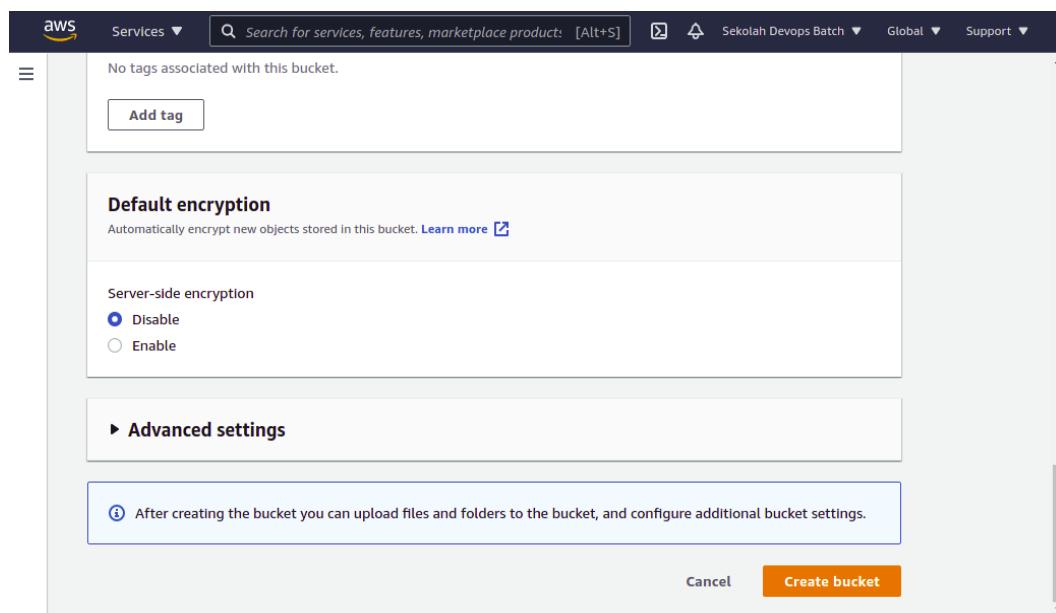
- I acknowledge that the current settings might result in this bucket and the objects within becoming public.

Jika discroll ke bawah lagi, ada bagian **Versioning** dan **Tags**. Pada bagian ini biarkan saja secara default. Versioning berguna untuk melihat perubahan yang terjadi pada objek pada bucket, konsepnya sama seperti **Git**. Untuk **Tags** berfungsi untuk memberi informasi label pada bucket, bisa berupa penamaan atau kebutuhan bucket tersebut. Tags ini bersifat optional yang berarti kita tidak diwajibkan untuk mengisinya.

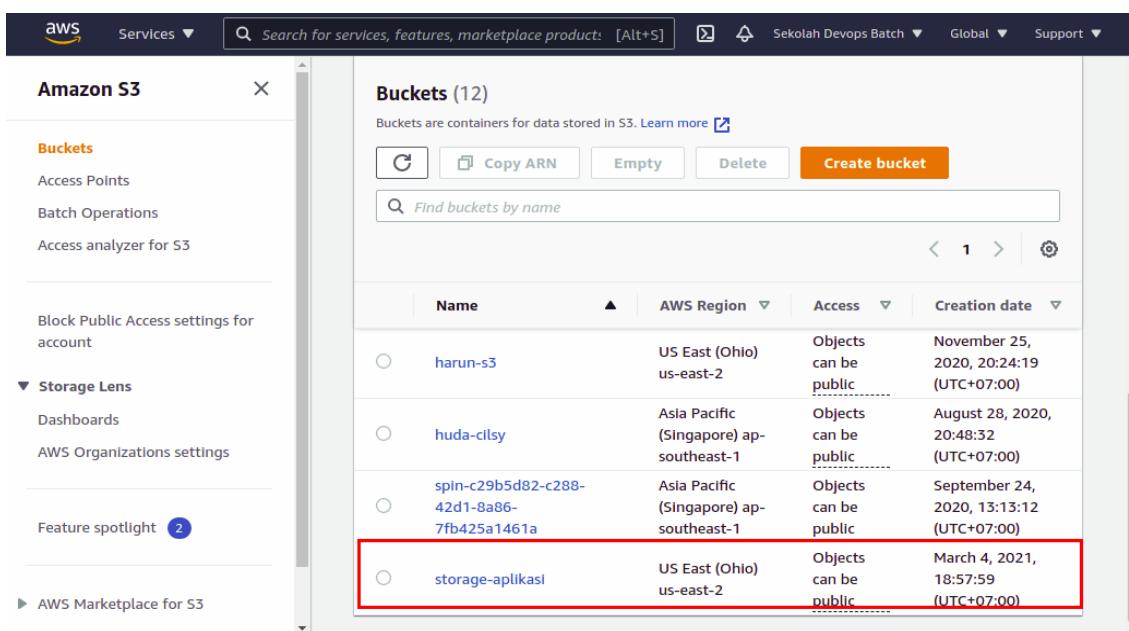




Pada bagian default encryption, kita biarkan seperti default. Lalu klik **Create Bucket**.



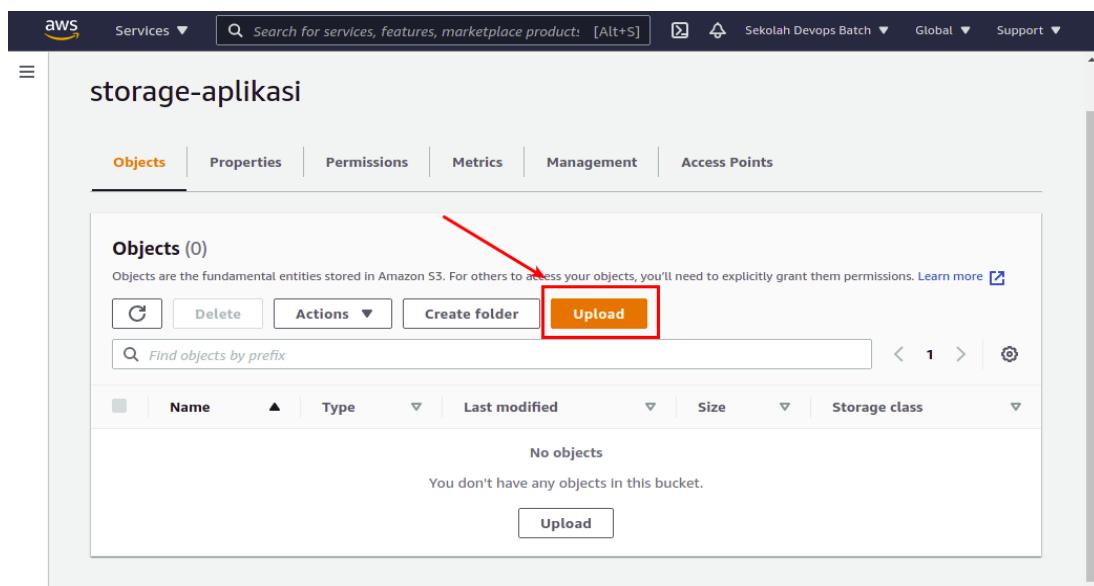
Setelah berhasil maka bisa kita lihat bucket yang sudah buat sebelumnya akan terlihat seperti dibawah ini.



The screenshot shows the AWS S3 Buckets page. On the left, there's a sidebar with options like 'Buckets', 'Access Points', 'Batch Operations', 'Access analyzer for S3', 'Block Public Access settings for account', 'Storage Lens', 'Dashboards', 'AWS Organizations settings', 'Feature spotlight' (with a '2' notification), and 'AWS Marketplace for S3'. The main area is titled 'Buckets (12)' and contains a table with columns: Name, AWS Region, Access, and Creation date. The table lists four buckets: 'harun-s3', 'huda-cilsy', 'spin-c29b5d82-c288-42d1-8a86-7fb425a1461a', and 'storage-aplikasi'. The 'storage-aplikasi' row is highlighted with a red box.

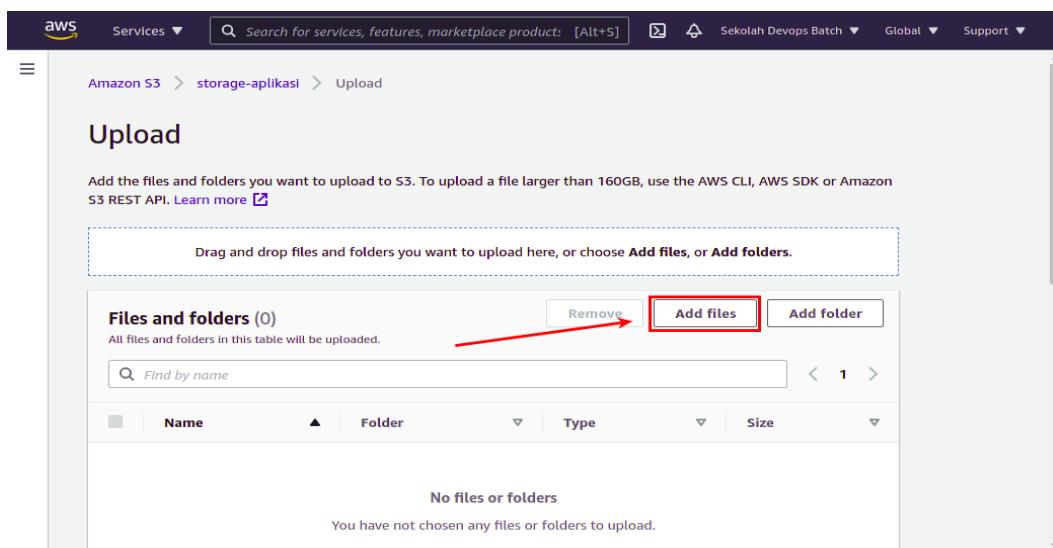
Name	AWS Region	Access	Creation date
harun-s3	US East (Ohio) us-east-2	Objects can be public	November 25, 2020, 20:24:19 (UTC+07:00)
huda-cilsy	Asia Pacific (Singapore) ap-southeast-1	Objects can be public	August 28, 2020, 20:48:32 (UTC+07:00)
spin-c29b5d82-c288-42d1-8a86-7fb425a1461a	Asia Pacific (Singapore) ap-southeast-1	Objects can be public	September 24, 2020, 13:13:12 (UTC+07:00)
<b>storage-aplikasi</b>	US East (Ohio) us-east-2	Objects can be public	March 4, 2021, 18:57:59 (UTC+07:00)

Selanjutnya pilih bucket yang sudah kita buat tadi, sampai masuk kedalam **dashboard bucket** tersebut. Kita akan coba upload sebuah file ke dalam bucket dengan memilih tombol **Upload**.

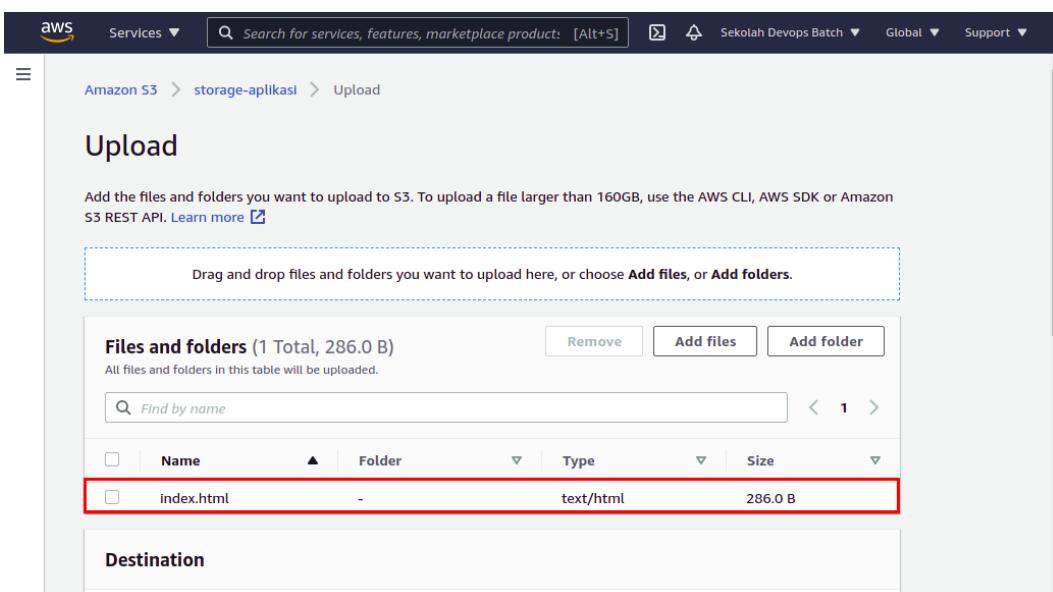


The screenshot shows the 'storage-aplikasi' bucket's objects page. At the top, there are tabs for 'Objects', 'Properties', 'Permissions', 'Metrics', 'Management', and 'Access Points'. The 'Objects' tab is selected. Below it, there's a section titled 'Objects (0)' with a note: 'Objects are the fundamental entities stored in Amazon S3. For others to access your objects, you'll need to explicitly grant them permissions.' There are buttons for 'Delete', 'Actions', 'Create folder', and a large orange 'Upload' button. A red arrow points from the 'Upload' button on the previous screen to this one. Below the 'Upload' button is a search bar and a table header with columns: Name, Type, Last modified, Size, and Storage class. The table body shows 'No objects' and a message: 'You don't have any objects in this bucket.' There is another 'Upload' button at the bottom of the table.

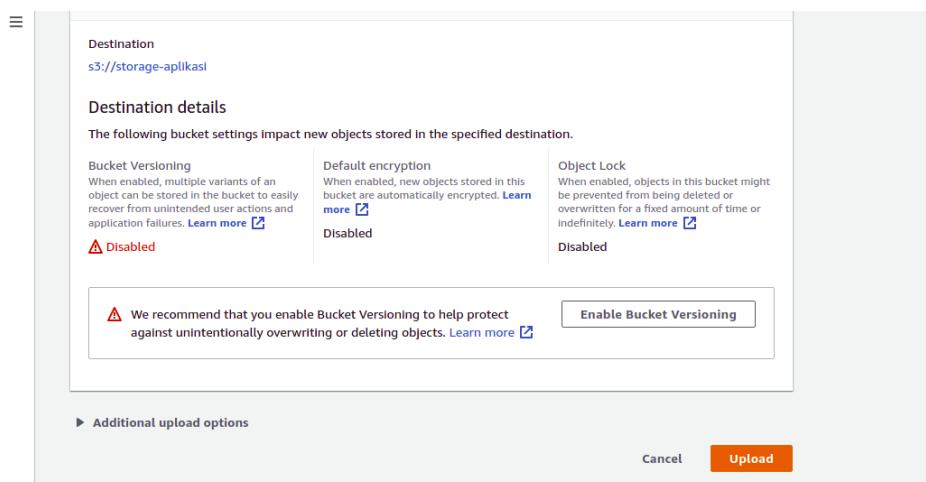
Setelah itu akan diarahkan ke laman upload, pilih file mana yang akan kita upload. Disini saya akan mencoba untuk upload sebuah file **index.html** ke dalam bucket tersebut. Klik **Add files**.



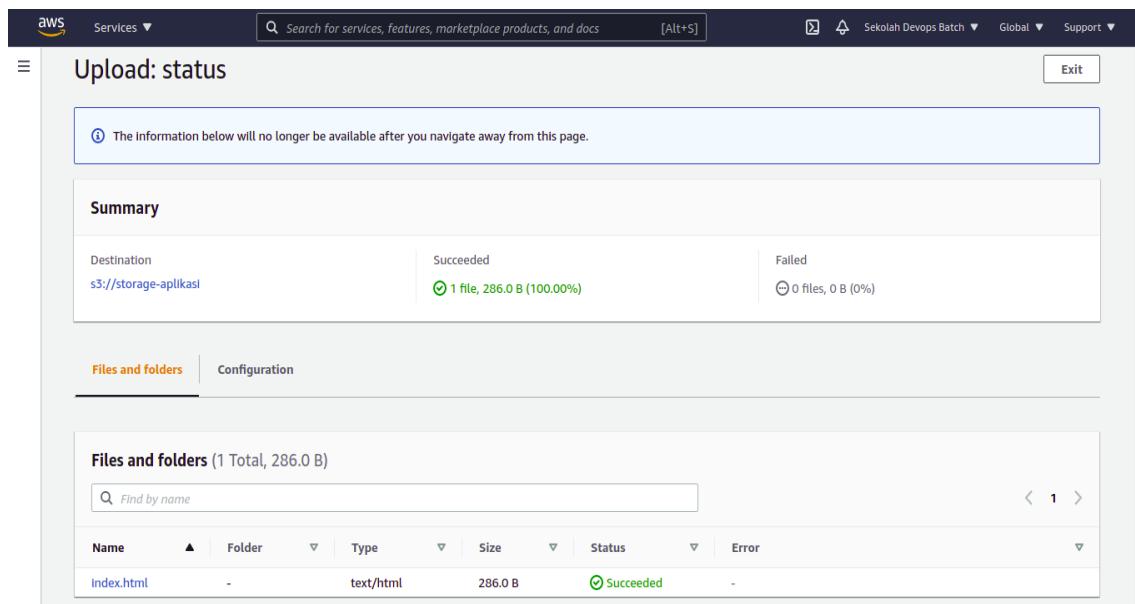
File yang sudah terupload, akan tampil seperti gambar berikut.



Namun file tersebut belum terupload ke **Bucket**, oleh karenanya kita scroll ke bawah untuk mengupload file tadi. Klik **Upload**



Apabila **success** maka akan muncul seperti dibawah ini, disertai dengan log bahwa file yang di upload success.



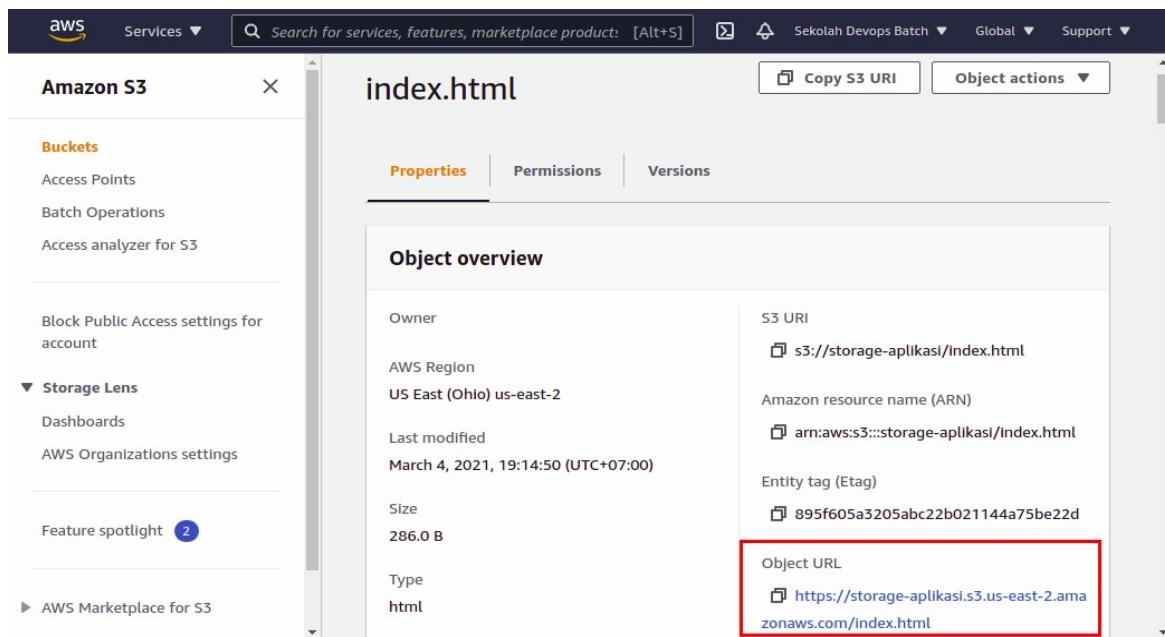
Dengan begini kita sudah berhasil membuat sebuah bucket dan melakukan upload file pada bucket tersebut. Kita bisa melakukan upload file secara banyak dengan menyeleksinya.

### 6.4.2. Konfigurasi Public Access

Pada dasarnya file yang sudah kita upload memiliki alamatnya sendiri, dan tidak dapat kita akses secara langsung apabila kita tuju alamatnya. Salah satunya adalah file index.html tadi. Kita bisa melihat alamat file tersebut

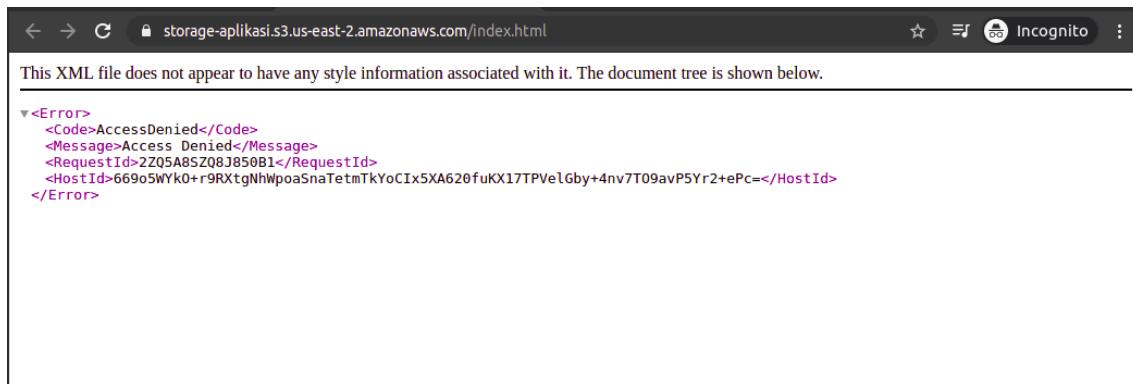


dengan memilih file tersebut dan akan muncul diarahkan menu memberikan informasi detail salah satunya url.



The screenshot shows the AWS S3 console interface. On the left, there's a sidebar with options like Buckets, Storage Lens, and Feature spotlight. The main area is titled "index.html" and shows the "Properties" tab selected. Under "Object overview", it lists details such as Owner, AWS Region (US East (Ohio) us-east-2), Last modified (March 4, 2021, 19:14:50 (UTC+07:00)), Size (286.0 B), Type (html), and S3 URI (s3://storage-aplikasi/index.html). A red box highlights the "Object URL" section, which contains the full URL: https://storage-aplikasi.s3.us-east-2.amazonaws.com/index.html.

Apabila kita coba akses URL tersebut, maka kita tidak akan bisa mengaksesnya dan akan muncul tampilan seperti dibawah ini.

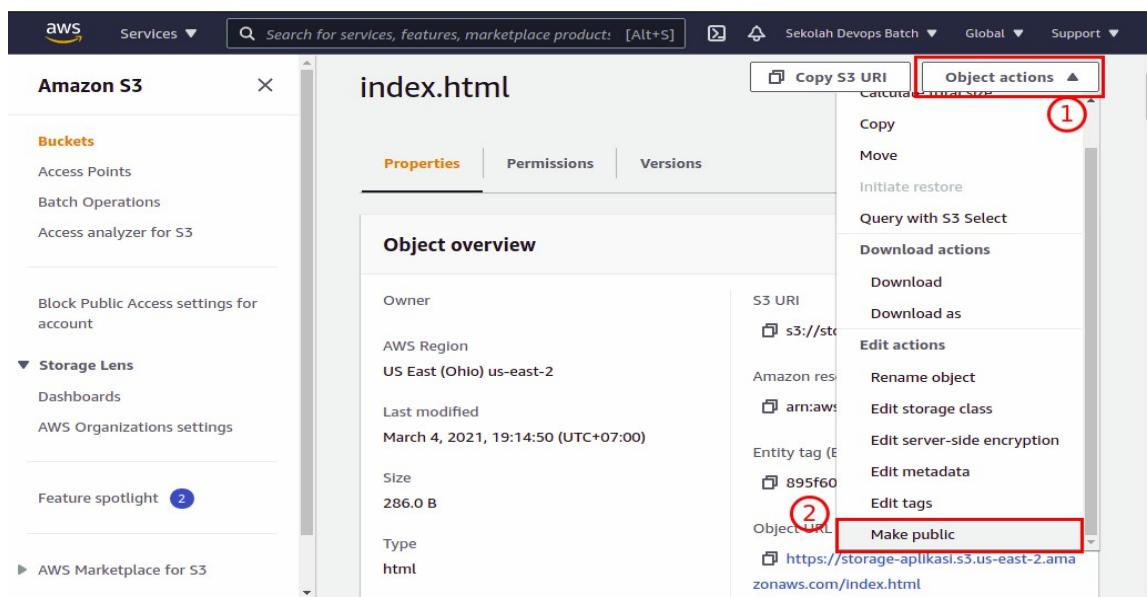


The screenshot shows a web browser window with the URL https://storage-aplikasi.s3.us-east-2.amazonaws.com/index.html. The page displays an "Access Denied" error message: "This XML file does not appear to have any style information associated with it. The document tree is shown below." Below this, there is an XML error response:

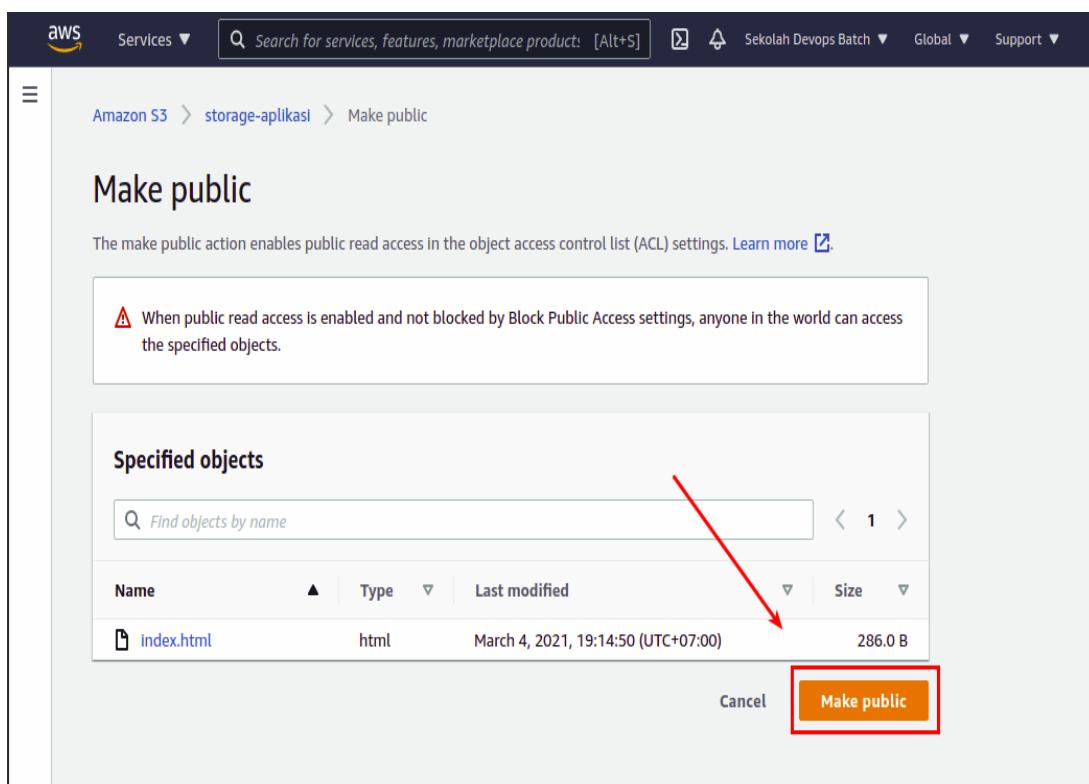
```
<Error>
<Code>AccessDenied</Code>
<Message>Access Denied</Message>
<RequestId>ZQ5A85Z08J850B1</RequestId>
<HostId>669o5WYk0+r9RxtgNhwpoaSnaTetmTkYoCIx5XA620fuKX17TPVe1Gby+4nv7T09avP5Yr2+ePc=</HostId>
</Error>
```

Untuk memberikan akses pada file tersebut kita bisa buat file tersebut menjadi public, akan tetapi hal ini akan membuat file tersebut saja yang menjadi public. Sedangkan file lainnya tidak akan public, sehingga yang lainnya pun harus kita buat public secara manual.

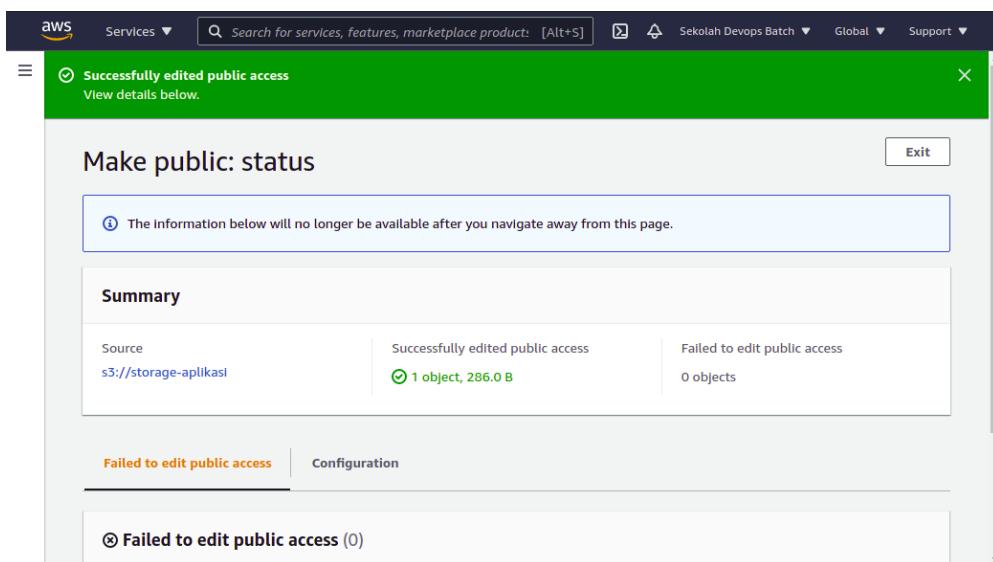
Yang perlu kita lakukan adalah mengeklik/memilih file tersebut, lalu pilih tombol **Object Action** lalu pilih **Make Public**.



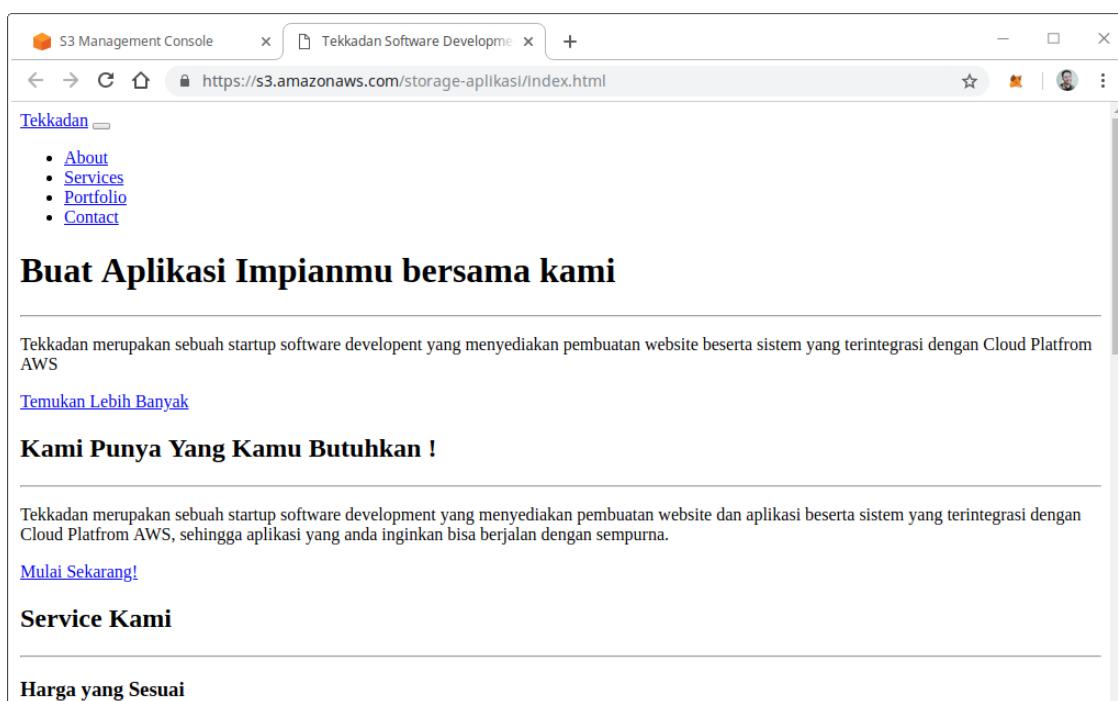
Setelah itu akan muncul sebuah pop-out seperti dibawah ini, pilih tombol **Make public** untuk melanjutkan.



Maka akan muncul notifikasi bahwa file dapat diakses publik sekarang.



Maka setelah kita ujicoba kembali, file tersebut akan bisa kita akses seperti dibawah ini.



The screenshot shows a web browser displaying a website at <https://s3.amazonaws.com/storage-aplikasi/index.html>. The page content includes:

- A navigation menu with links to About, Services, Portfolio, and Contact.
- A main heading: **Buat Aplikasi Impianmu bersama kami**.
- A paragraph: Tekkadan merupakan sebuah startup software developent yang menyediakan pembuatan website beserta sistem yang terintegrasi dengan Cloud Platfrom AWS.
- A link: Temukan Lebih Banyak.
- A section: **Kami Punya Yang Kamu Butuhkan !**
- A paragraph: Tekkadan merupakan sebuah startup software development yang menyediakan pembuatan website dan aplikasi beserta sistem yang terintegrasi dengan Cloud Platfrom AWS, sehingga aplikasi yang anda inginkan bisa berjalan dengan sempurna.
- A link: Mulai Sekarang!
- A section: **Service Kami**.
- A section: **Harga yang Sesuai**.

*Hasil dari konfigurasi make public*

### 6.4.3. Konfigurasi Hosting S3

Apabila pada sebelumnya kita melakukan make public pada satu persatu file pada bucket, sekarang kita akan coba membuat seluruh file menjadi public. Caranya ialah membuat bucket tersebut menjadi sebuah hosting.

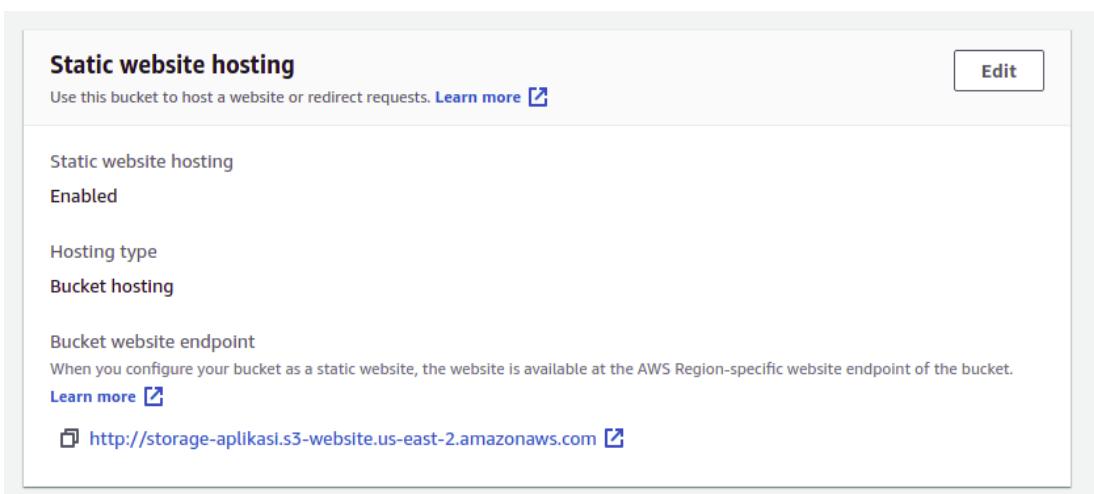
Selain dari pada fungsi menyimpan data, amazon S3 ini bisa kita jadikan sebagai sebuah hosting storage yang cocok untuk menyimpan sebuah web static atau file manager.

Untuk membuat hosting dari S3, kita hanya perlu memilih **Bucket** lalu masuk kedalam tab **Properties**, lalu scroll ke bawah ke menu **Static web hosting** seperti dibawah ini lalu klik **Edit**.

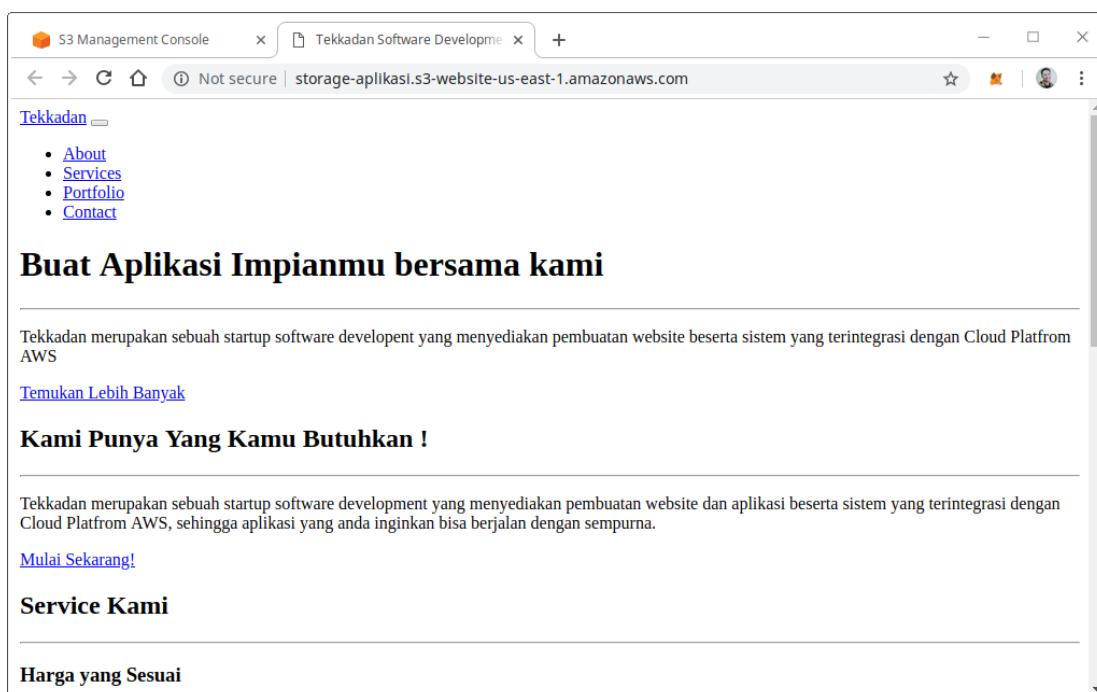
Akan muncul sebuah setting seperti dibawah ini, ubah parameter dari **disable** menjadi **enable**, lalu isikan index dokument dengan file **index.html** yang ada pada bucket kita. Setelah selesai klik **Save**.



Setelah selesai, status properties pada static website hosting tersebut akan berubah menjadi seperti dibawah ini.



Langkah selanjutnya adalah akses alamat **static web hosting** tadi, jika berhasil maka kita akan melihat hasilnya seperti dibawah ini.



The screenshot shows a web browser window displaying a static website. The title bar says 'Tekkadan Software Development' and the URL is 'storage-aplikasi.s3-website-us-east-1.amazonaws.com'. The page content includes a navigation menu with links to About, Services, Portfolio, and Contact. Below the menu, there is a main section with the heading 'Buat Aplikasi Impianmu bersama kami'. The page also contains descriptive text about Tekkadan being a startup software development company and a 'Mulai Sekarang!' button.

*Hasil dari hosting mode amazon S3*

#### 6.4.4. Mounting Amazon S3 kedalam Instance

Pada bagian ini kita akan melakukan konfigurasi pada instance agar bisa melakukan mounting storage S3. Dimana nantinya kita bisa melihat isi data yang ada pada bucket yang kita miliki didalam instance kita, manfaatnya nanti kita bisa memiliki storage yang besar untuk penyimpanan data besar pada

webserver kita seperti video dll yang memang membutuhkan storage yang sangat besar.

Untuk melakukan mounting, ada beberapa tahapan installasi yang harus kita lakukan pada instance, disini saya asumsikan kalian sudah memiliki instance yang siap untuk digunakan.

```
apt-get update
sudo apt-get install automake autotools-dev fuse g++ git libcurl4-gnutls-dev
libfuse-dev libssl-dev libxml2-dev make pkg-config
```

Setelah melakukan installasi aplikasi yang dibutuhkan, sekarang kita clone aplikasi s3fs yang ada di github seperti dibawah ini.

```
git clone https://github.com/s3fs-fuse/s3fs-fuse.git
```

Selanjutnya kita lakukan installasi aplikasi s3fs yang sudah kita clone tadi.

```
cd s3fs-fuse
./autogen.sh
./configure --prefix=/usr --with-openssl
make
sudo make install
```

Selanjutnya kita buat sebuah file password untuk autentikasi kedalam akun IAM yang kita miliki dengan membuat file **passwd-s3fs** seperti berikut.

```
vim /etc/passwd-s3fs
```

Setelah itu masukan access key dan secret key dengan format berikut.

```
accesskey_kamu:secretkey_kamu
```

Selanjutnya simpan file password tersebut dan berikan hak akses.

```
sudo chmod 640 /etc/passwd-s3fs
```

Buat sebuah direktori baru dengan nama S3devopscilsy didalam root, lalu lakukan mouting S3 bucket kita kedalam direktory yang kita buat.

```
mkdir /S3devopscilsy
sudo s3fs storage-aplikasi /root/user -o passwd_file=/home/ubuntu/.passwd-s3fs -o url=https://s3.ap-southeast-1.amazonaws.com -o uid=1001,gid=1001,allow_other
```

**\*\* Note :** yang diberi warna merah adalah nama bucket kita



Jika sudah selesai, kita dapat mengecek keberhasilan mounting yang kita lakukan dengan menggunakan perintah ***df -Th***. Apabila berhasil maka akan muncul seperti dibawah ini.

```
ubuntu@ip-172-31-35-237:~/s3fs-fuse$ df -Th
Filesystem      Type  Size  Used  Avail Use% Mounted on
udev            devtmpfs 481M   0  481M  0% /dev
tmpfs           tmpfs   99M  736K  98M  1% /run
/dev/xvda1       ext4   7.7G  1.9G  5.8G 25% /
tmpfs           tmpfs   492M   0  492M  0% /dev/shm
tmpfs           tmpfs   5.0M   0  5.0M  0% /run/lock
tmpfs           tmpfs   492M   0  492M  0% /sys/fs/cgroup
/dev/loop0       squashfs 88M   88M   0 100% /snap/core/5328
/dev/loop1       squashfs 13M   13M   0 100% /snap/amazon-ssm-agent/495
/dev/loop2       squashfs 91M   91M   0 100% /snap/core/6405
/dev/loop3       squashfs 18M   18M   0 100% /snap/amazon-ssm-agent/1068
+mnfc           +mnfc   a0M   a0M   a0M  0% /run/user/1000
s3fs            fuse.s3fs 256T   0  256T  0% /S3devopscilsy
ubuntu@ip-172-31-35-237:~/s3fs-fuse$
```

Mounting pada tahap ini bersifat manual, apabila kita ingin melakukan mounting secara otomatis, kita harus memasukan beberapa konfigurasi pada bagian crontab.

Sebelum itu kita harus mengecek terlebih dahulu konfigurasi s3fs berada di direktori mana pada server kita dengan menggunakan perintah ***which*** seperti dibawah ini.

```
ubuntu@ip-172-31-35-237:~/s3fs-fuse$ which s3fs
/usr/bin/s3fs
ubuntu@ip-172-31-35-237:~/s3fs-fuse$
```

Setelah kita mengetahui keberadaan aplikasi s3fs, sekarang kita lakukan konfigurasi pada crontab dengan menggunakan perintah ***crontab -e*** lalu masukan script dibawah ini di line paling akhir.

```
@reboot /usr/bin/s3fs storage-aplikasi -o use_cache=/tmp -o allow_other -o
uid=1001 -o mp_umask=002 -o multireq_max=5 /S3devopscilsy
```

**\*\*Note :** yang diberi warna merah adalah lokasi aplikasi s3fs berada.

Setelah selesai, lalu simpan crontab tersebut, ujicoba perubahan yang sudah dilakukan dengan melakukan reboot, apakah storage akan otomatis mounting atau tidak.



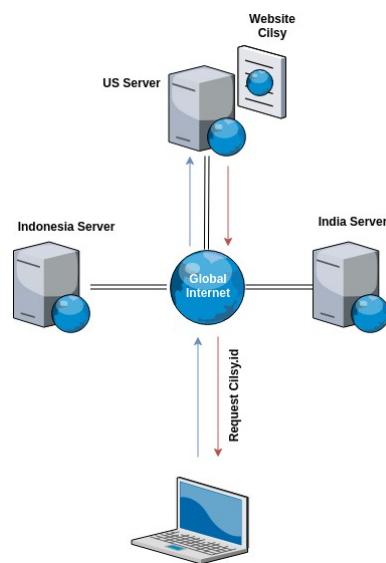
### 6.4.5. Exercise

1. Buat sebuah bucket baru dengan nama kalian, buat bucket tersebut menjadi hosting sehingga bisa diakses.
2. Upload file landing page kedalam bucket tersebut, lalu berikan hak akses public kepada semua konten landing page tersebut.
3. Pastikan landing page dapat diakses menggunakan alamat hosting dari amazon S3.

## 6.5. Pengenalan Amazon CloudFront

### 6.5.1. Apa itu CDN ?

**CDN (Content Delivery Network)** adalah kumpulan dari server global yang terletak di beberapa data center dan tersebar di berbagai negara. Jaringan ini berfungsi untuk mengirimkan konten dari server ke suatu website. CDN meningkatkan kecepatan pengiriman data melalui jaringan server kepada visitor dari lokasi terdekat yang paling memungkinkan.



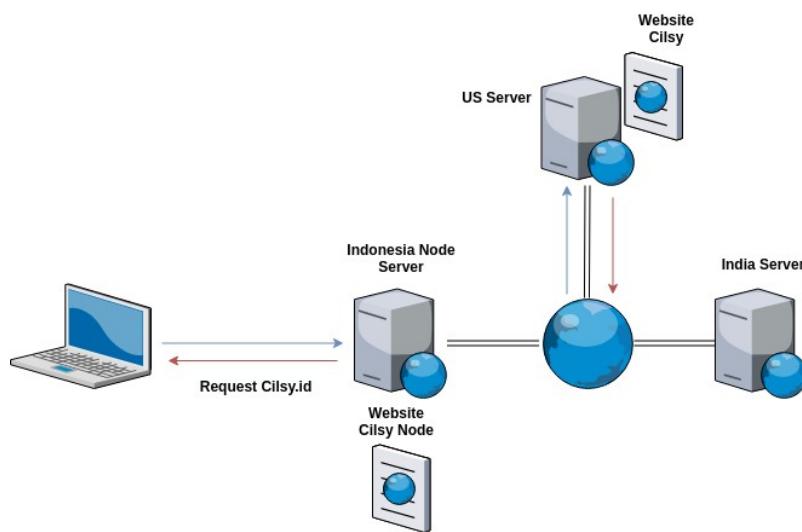
*Ilustrasi tanpa menggunakan CDN*

Katakanlah lokasi visitor berada di indonesia, sedangkan server dari website kita berada di US. Karena jarak antara lokasi visitor dan server, pengiriman



konten memerlukan proses waktu yang lama. Tapi, dengan adanya server CDN yang berada di beberapa tempat seperti indonesia, india, US, proses pengiriman konten dapat menjadi lebih cepat.

Jika visitor kita berasal dari indoneisa, CDN akan mengirimkan file dari lokasi terdekat yang paling memungkinkan. Dalam kasus ini, file bisa jadi dikirimkan dari server yang juga terletak di indonesia.



*Ilustrasi menggunakan CDN*

CDN mempercepat loading website. Fungsinya, memastikan pengiriman konten statis seperti gambar, CSS, JavaScript, video, dan lain-lain dari lokasi server terdekat ke pengguna yang mengakses website kita. Hal tersebut dapat meningkatkan kecepatan respon suatu website ketika diakses.

### 6.5.2. Macam-macam CDN

Karena kecanggihan dari layanan CDN ini, banyak sekali platform yang menediakan fasilitas CDN yang dapat kita gunakan. Mulai dari yang gratis sampai dengan yang berbayar, berikut merupakan beberapa macam CDN yang tersedia sampai saat ini.

- 1.** Cloudflare
- 2.** Amazon CloudFront
- 3.** Incapsula

#### 4. MaxCDN

#### 5. RackSpace

#### 6. Dll

Di AWS sendiri kita memiliki layanan yang memiliki fungsi sebagai Content Delivery Network, yaitu adalah Cloudfront. Kita akan membahas mengenai layanan AWS ini.

### 6.5.3. Apa itu Amazon CloudFront ?

**Amazon CloudFront** adalah layanan **Content Delivery Network** (CDN) yang secara aman mengirimkan data, video, aplikasi, dan API kepada pengguna dengan *low latency* (Jeda waktu yang rendah) dan kecepatan transfer yang tinggi.

**CloudFront** terintegrasi dengan AWS, termasuk lokasi fisik yang terhubung langsung ke infrastruktur global AWS seperti *AWS Shield* untuk mitigasi *DDoS*, *Amazon S3*, *Elastic Load Balancing* atau *Amazon EC2* yang bekerja secara lancar.



Logo AWS CloudFront

Amazon CloudFront mempercepat distribusi konten web statis dan dinamis seperti html, css, js, dan file gambar kepada pengguna. CloudFront memberikan konten melalui jaringan pusat data di seluruh dunia yang disebut ***edge location***.

Ketika seorang pengguna melakukan request konten melalui layanan cloudfont, maka pengguna akan diarahkan ke lokasi yang menyediakan *low*

*latency* sehingga konten tersebut akan dikirimkan dengan kinerja sebaik mungkin.

1. Jika konten sudah berada di *edge location* dengan *low latency*, CloudFront akan segera mengirimkannya.
2. Akan tetapi jika konten tidak berada di *edge location* tersebut, CloudFront akan mengambilnya dari asal yang telah kita tetapkan (misalkan webserver) yang diidentifikasi sebagai sumber untuk versi definitif konten.

## 6.6. Praktek Amazon CloudFront

### 6.6.1. Setup S3 Bucket

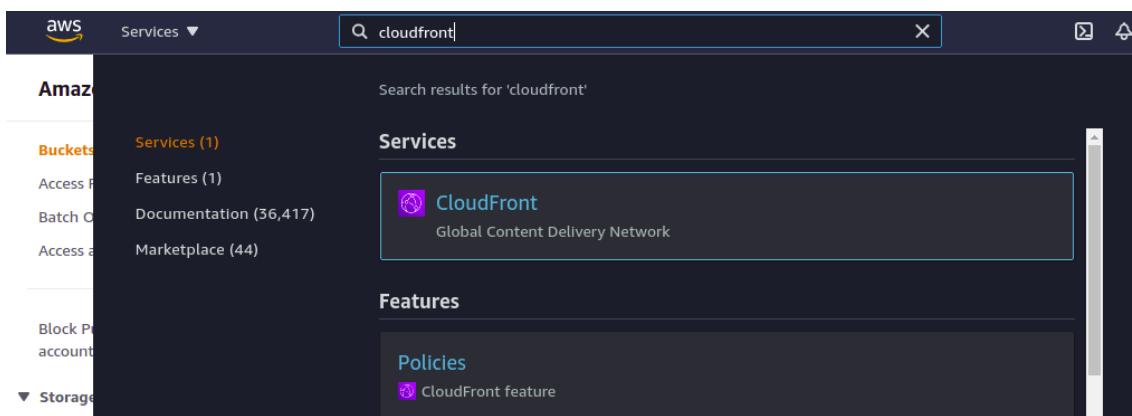
Sebelum melakukan setup pada **cloudfont**, pertama kita harus menyiapkan terlebih dahulu **sebuah bucket di Amazon S3**. Karena cloudfont sendiri menggunakan S3 bucket untuk penggunaannya.

Disini kita anggap kalian sudah mengerti paham bagaimana cara membuat sebuah bucket S3 baru, apabila kalian masih kebingungan, kalian bisa membaca kembali tahap sebelumnya di bab 5 part 3. Berikut adalah requirement untuk S3 bucket yang akan digunakan.

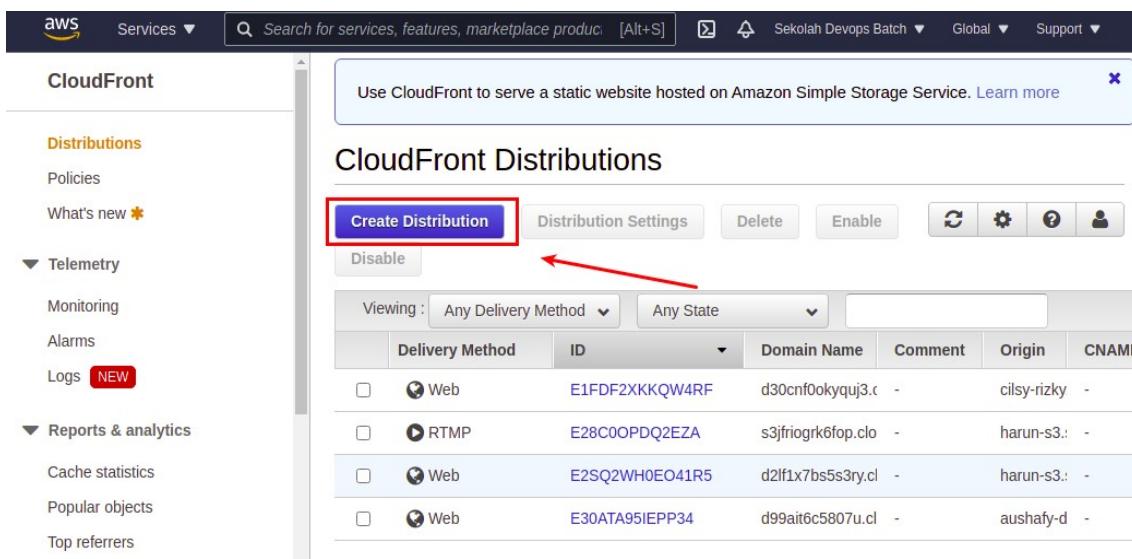
- Sebuah Bucket S3 dengan nama kalian (contoh ***tresna-devops-S3***)
- Buat bucket di region manapun
- Jangan dulu mengisi apapun pada S3 tersebut.

### 6.6.2. Setup Amazon CloudFront dengan S3

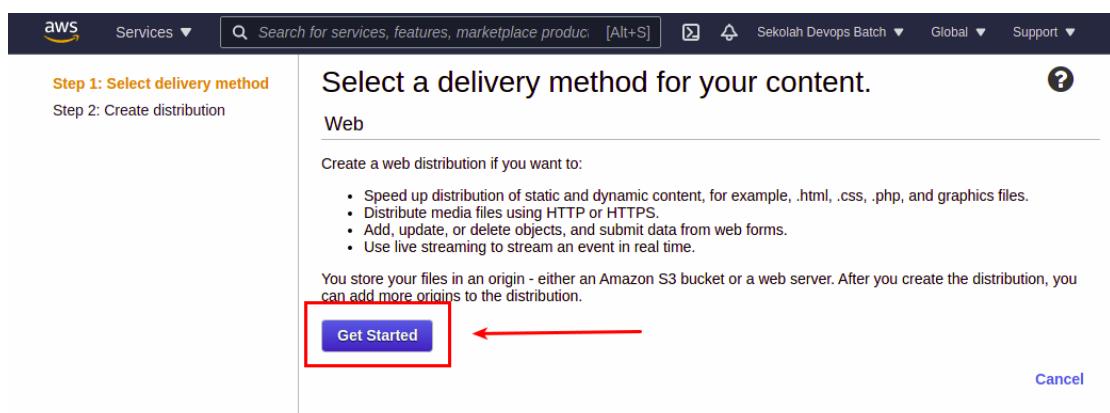
Setelah kita menyiapkan S3 Bucket, selanjutnya kita lakukan setup pada cloudfont. Untuk masuk kedalam dashboard kita hanya perlu memilih menu **Services > Cloudfont** seperti dibawah ini.



Setelah masuk kedalam **dashboard** milik cloudfont, selanjutnya pilih menu **Create Distribution** untuk membuat sebuah CDN baru.

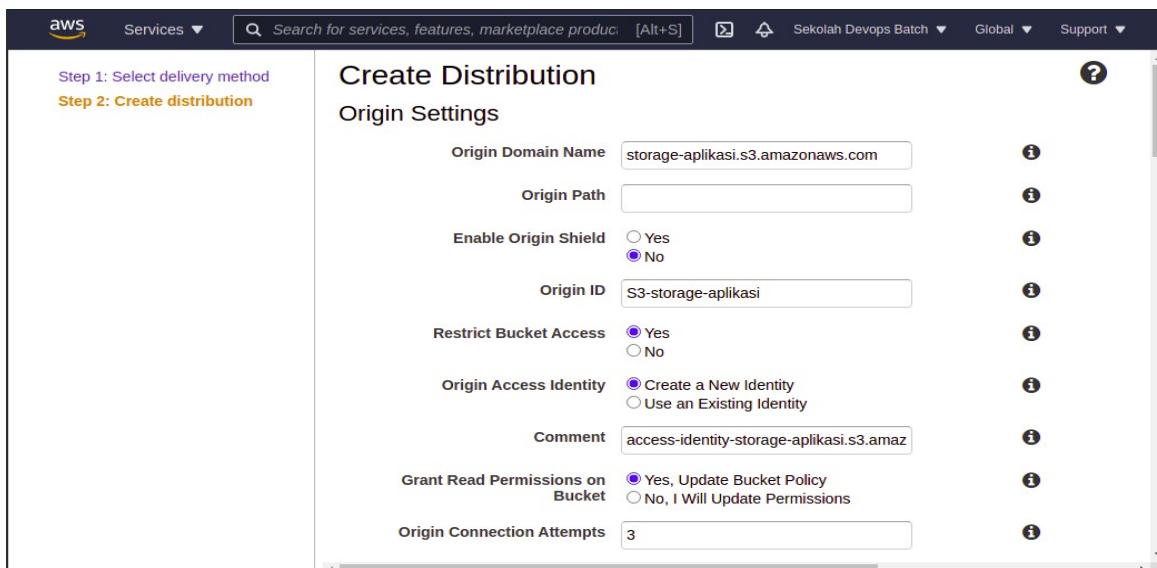


Kita akan diarahkan untuk membuat Distribution dengan mengeklik Get Started.



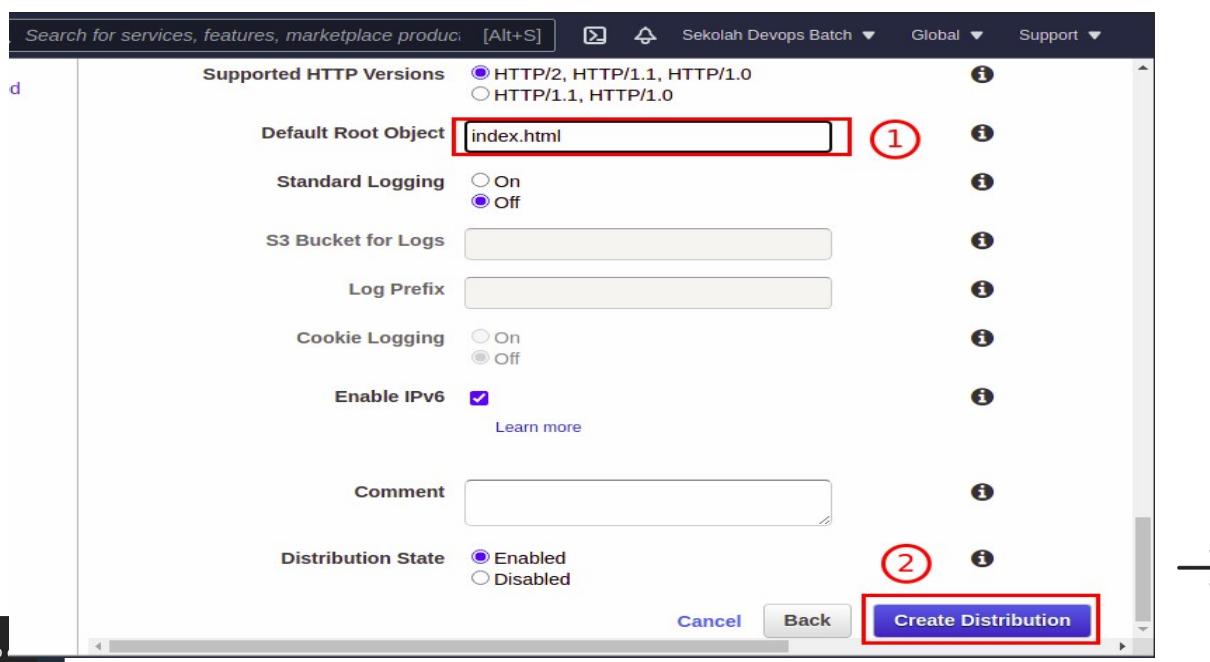
Isikan **Origin Domain Name** dengan **bucket S3** yang kita sudah buat tadi, lalu ubah beberapa option menjadi seperti berikut.

- **Restrict Bucket Access = Yes**
- **Origin Access Identity = Create a New Identity**
- **Grant Read Permissions on Bucket = Yes, Update Bucket Policy**



Setelah itu scroll ke bagian bawah, karena tidak banyak yang akan kita konfigurasi disini. Sampai pada bagian dibawah ini, ubah **Default Root Object** menjadi **index.html**. Sesuaikan nanti dengan konten yang ada di dalam bucket tersebut.

Lalu untuk menyelesaikan konfigurasi, klik **Create Distribution**.



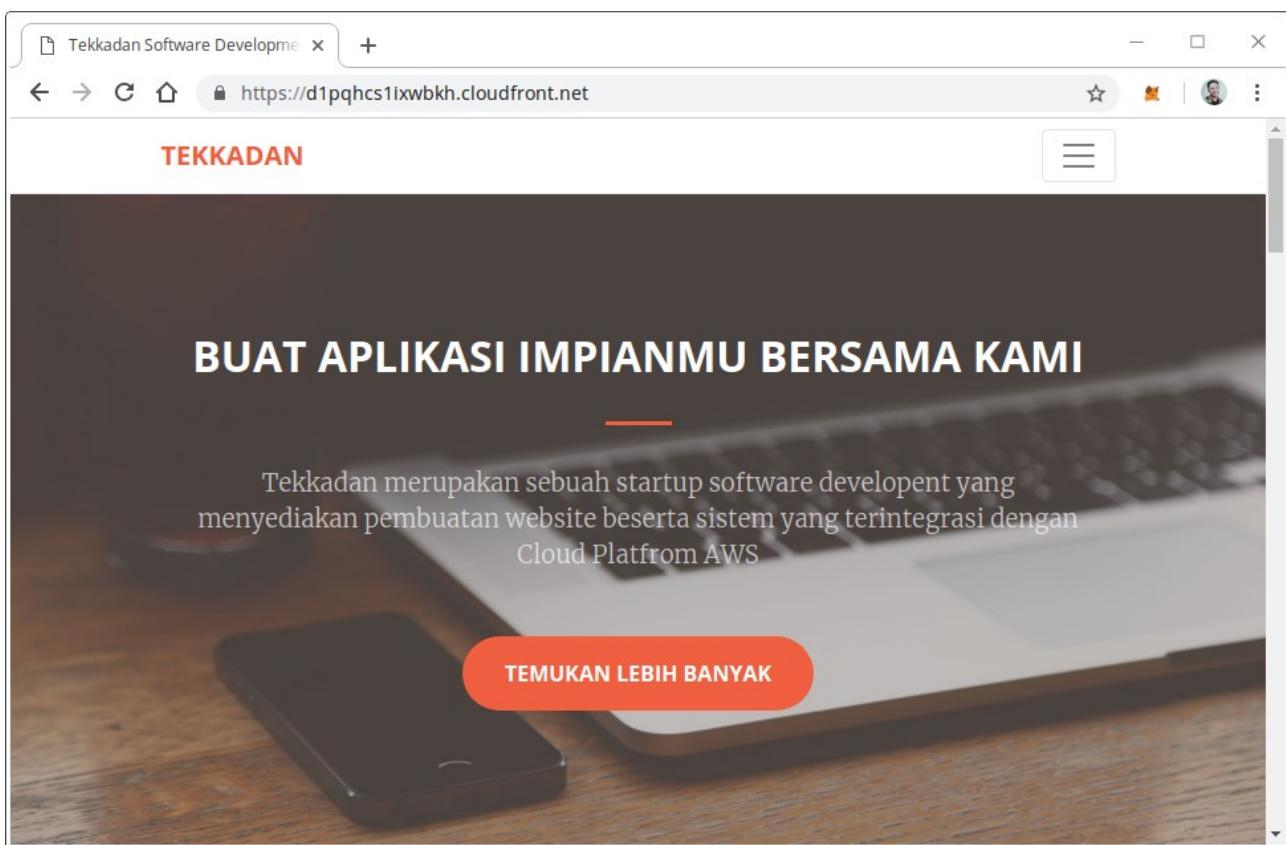
Maka hasilnya akan terlihat seperti dibawah ini, proses pembuatan masih berlangsung. Kita bisa lihat pada bagian **Status** yang masih **in progress**. Kita bisa lihat domain cloudfont kita dengan memilih menu **Distribution Settings**.

Delivery Method	ID	Domain	Com	Ori	CNA	Status	Sta	Last M
Web	E1FDF2XKKQW4RF	d30cnfc	-	cilsj	-	Deployed	En	2021-0:
Web	E1ZSQJUWDS7AGO	d26ynq	-	stor	-	In Prog	En	2021-0:
RTMP	E28C0OPDQ2EZA	s3jfrigl	-	hart	-	Deployed	Dis	2020-1:
Web	E2SQ2WH0EO41R5	d2lf1x7l	-	hart	-	Deployed	En	2020-1:
Web	E30ATA95IEPP34	d99ait6	-	ausl	-	Deployed	Dis	2020-0:

Setelah masuk setting, kita bisa melihat bagian alamat domain pada cloudfont tersebut. Kita bisa coba akses alamatnya, apabila alamat masih belum bisa diakses maka kita hanya perlu menunggu 3-5 menit.

ARN	arn:aws:cloudfront::898130718046:distribution/E1ZSQJUWDS7AGO
Log Prefix	-
Delivery Method	Web
Cookie Logging	Off
Distribution Status	InProgress
Comment	-
Price Class	Use All Edge Locations (Best Performance)
AWS WAF Web ACL	-
State	Enabled
Alternate Domain Names (CNAMEs)	SSL Certificate: Default CloudFront Certificate (* cloudfront.net) Domain Name: d26ynqpc6tpk9.cloudfront.net
Custom SSL Client Support	-
Security Policy	TLSv1
Supported HTTP Versions	HTTP/2, HTTP/1.1, HTTP/1.0
IPv6	Enabled
Default Root Object	index.html
Last Modified	2021-03-04 20:01 UTC+7
Log Bucket	-

Berikut merupakan hasil dari cloudfont yang sudah kita buat, hasilnya akan terlihat seperti dibawah ini.



### 6.6.3. Setup Amazon CloudFront dengan Load Balancer

Selain menggunakan S3 Bucket, CloudFront ini dapat menggunakan **Load Balancer** dan Elastic Beanstalk untuk **Content Delivery Network**. Hanya terdapat sedikit perbedaan konfigurasi di awal untuk dapat menggunakan Load Balancer pada CDN ini.

Langkah pertama yang harus kita lakukan adalah menyiapkan server yang sudah kita load balance, saya asumsikan disini kalian sudah membuat load balancer tersebut. Setelah itu kita buat sebuah cloudfont baru dengan cara yang sama seperti sebelumnya.

Pilih menu **Create Distribution** untuk membuat sebuah CDN baru.

The screenshot shows the AWS CloudFront Distributions page. On the left sidebar, under 'Distributions', there is a 'Create Distribution' button highlighted with a red box. A red arrow points from this button to the 'Origin Domain Name' field in the 'Create Distribution' dialog box on the right.

Isikan **Origin Domain Name** dengan **Load Balancer** yang kita sudah buat tadi, lalu ubah beberapa option menjadi seperti berikut.

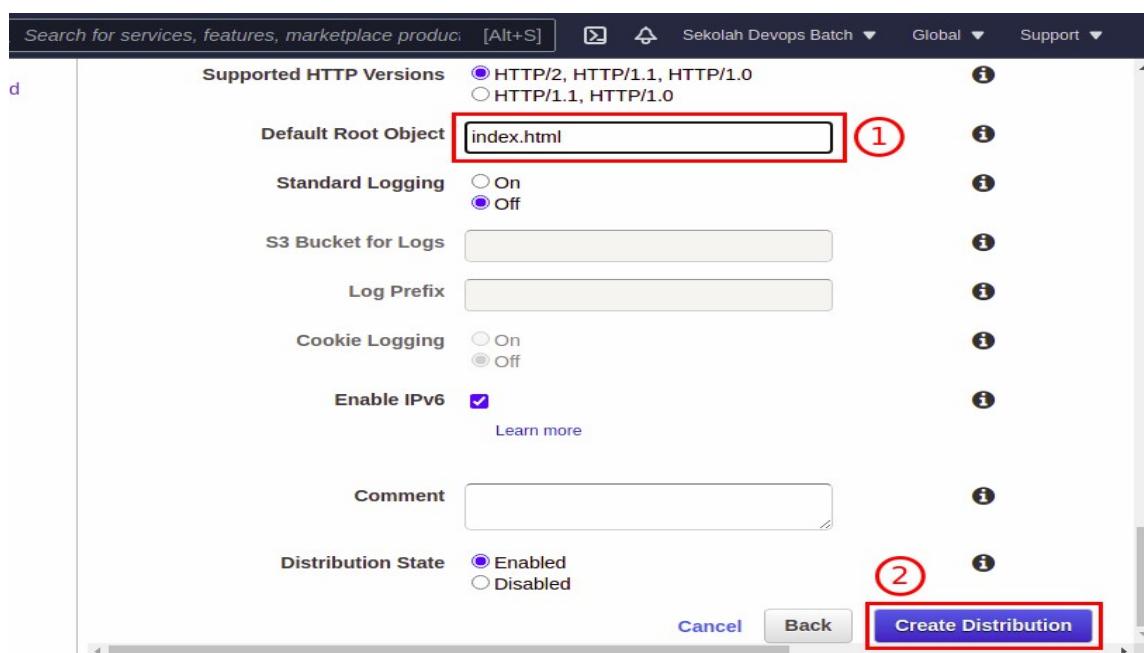
- **Origin Protocol Policy = Match viewer**

The screenshot shows the 'Create Distribution' dialog. The 'Origin Domain Name' field is highlighted with a red box and contains the value 'classic-1953058572.ap-southeast-1.elb.amazonaws.com'. The 'Origin Protocol Policy' section is also highlighted with a red box, showing three options: 'HTTP Only', 'HTTPS Only', and 'Match Viewer'. The 'Match Viewer' option is selected. There are two numbered circles (1 and 2) near these highlighted fields.

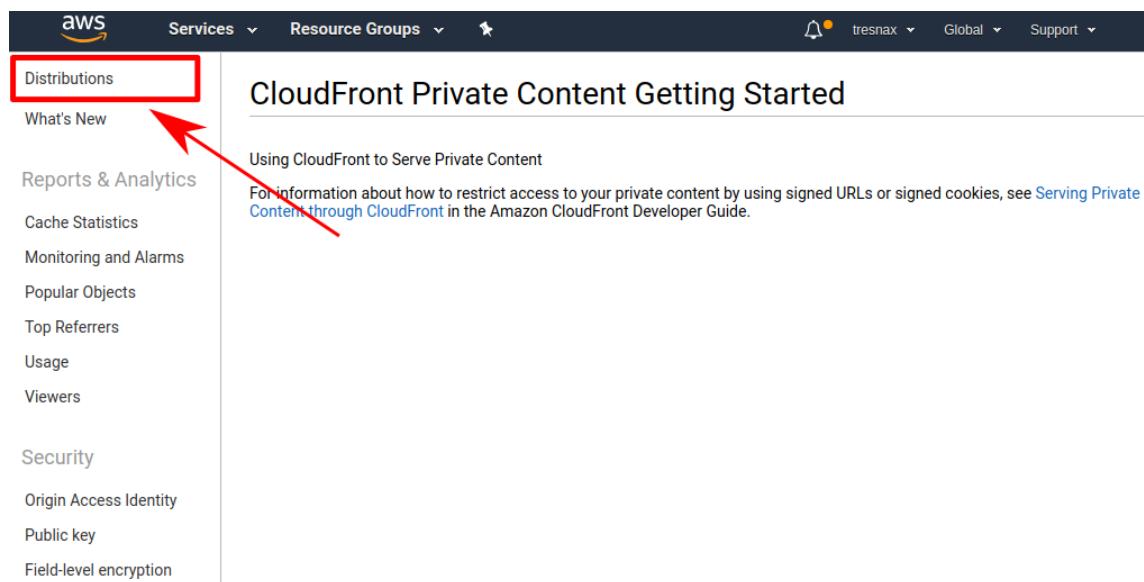
Setelah itu scroll ke bagian bawah, karena tidak banyak yang akan kita konfigurasi disini. Sampai pada bagian dibawah ini, ubah **Default Root Object** menjadi **index.html**. Sesuaikan nanti dengan konten yang ada di dalam bucket tersebut.

Lalu untuk menyelesaikan konfigurasi, klik **Create Distribution**.





Setelahnya akan muncul seperti dibawah ini, klik menu **Distribution** untuk melihat hasil yang sudah kita buat.



Maka hasilnya akan terlihat seperti dibawah ini, proses pembuatan masih berlangsung. Kita bisa lihat pada bagian **Status** yang masih **in progress**. Kita bisa lihat domain cloudfront kita dengan memilih menu **Distribution Settings**.

The screenshot shows the AWS CloudFront Distributions page. On the left sidebar, under 'Logs', there is a red box around the 'Logs' link. In the main content area, a red box highlights the 'Distribution Settings' button. Another red box highlights the checkbox for the second distribution listed in the table.

Delivery Method	ID	Domain	Com	Orig	CNA	Status	Sta	Last M
<input type="checkbox"/> Web	E1FDF2XKKQW4RF	d30cnfc	-	cilsj	-	Deployed	En	2021-0:
<input checked="" type="checkbox"/> Web	E1ZSQJUWDS7AGO	d26ynq	-	stor	-	In Prog	En	2021-0:
<input type="checkbox"/> RTMP	E28C0OPDQ2EZA	s3jfrigj	-	hart	-	Deployed	Dis	2020-1:
<input type="checkbox"/> Web	E2SQ2WH0EO41R5	d2lf1x7l	-	hart	-	Deployed	En	2020-1:
<input type="checkbox"/> Web	E30ATA95IEPP34	d99ait6	-	ausl	-	Deployed	Dis	2020-0:

Setelah masuk setting, kita bisa melihat bagian alamat domain pada cloudfront tersebut. Kita bisa coba akses alamatnya, apabila alamat masih belum bisa diakses maka kita hanya perlu menunggu 3-5 menit.

The screenshot shows the AWS CloudFront Distribution settings page for distribution E1BVIFOKL39G8E. On the left sidebar, under 'Logs', there is a red box around the 'Logs' link. In the main content area, a red box highlights the 'Domain Name' field, which contains 'd3jxh1eke5tpcw.cloudfront.net'.

*Alamat CloudFront yang sudah berhasil dibuat*

## 6.6.4. Exercise

- Buat sebuah Cloudfront baru menggunakan S3 Bucket dan juga Elastic Beanstalk.



## 6.7. Pengenalan Route53

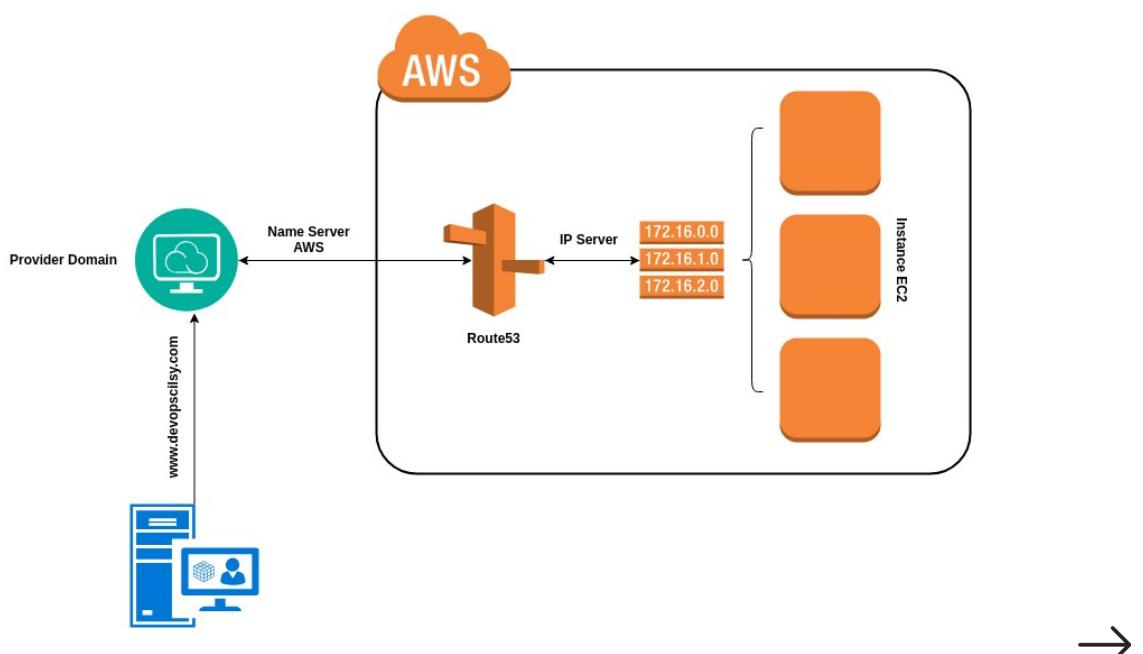
### 6.7.1. Apa itu Route53 ?

**Amazon Route 53** merupakan *domain name system (DNS)*. Pada dasarnya sebuah DNS menerjemahkan alamat IP Public menjadi sebuah nama domain yang dapat kita baca dan ingat dengan sangat mudah, amazon route 53 memiliki fungsi sebagai domain manager dimana layanan ini menghubungkan DNS pada layanan yang ada di AWS mencakup instance, load balancer, S3 bucket, Dan layanan lainnya.



*Amazon Route53 Logo*

Pada layanan ini kita dapat menggunakan domain yang dapat kita beli melalui layanan Route53 ataupun menggunakan domain yang sudah kita beli pada platform lain. Kita hanya perlu mengarahkan Name Server pada Route53 pada domain yang kita beli di platform lain, sehingga domain tersebut dapat kita gunakan.



*Ilustrasi cara kerja Route53*

Dengan begini setiap instance dan load balance yang kita buat, tidak perlu kita akses menggunakan alamat IP Public maupun DNS dynamic dari AWS. Kita hanya perlu mengarahkan layanan tersebut pada domain yang sudah kita daftarkan pada route53.

## 6.8. Praktek Route53

### 6.8.1. Persiapan Domain

Sebelum kita melakukan konfigurasi pada layanan Route53, kita harus sudah memiliki domain terlebih dahulu. Pada bagian ini kita akan coba menggunakan domain yang sudah dibeli dari provider lain dan bukan membeli dari AWS itu sendiri.

Disini kami sudah menyediakan domain untuk kalian satu persatu, akan tetapi jika kalian ingin menggunakan domain sendiri, kami dapat menyarankan untuk kalian membeli di provider-provider langganan kalian.

Mungkin kami akan memberikan beberapa platform yang menyediakan yang menjual domain yang diantaranya adalah berikut :

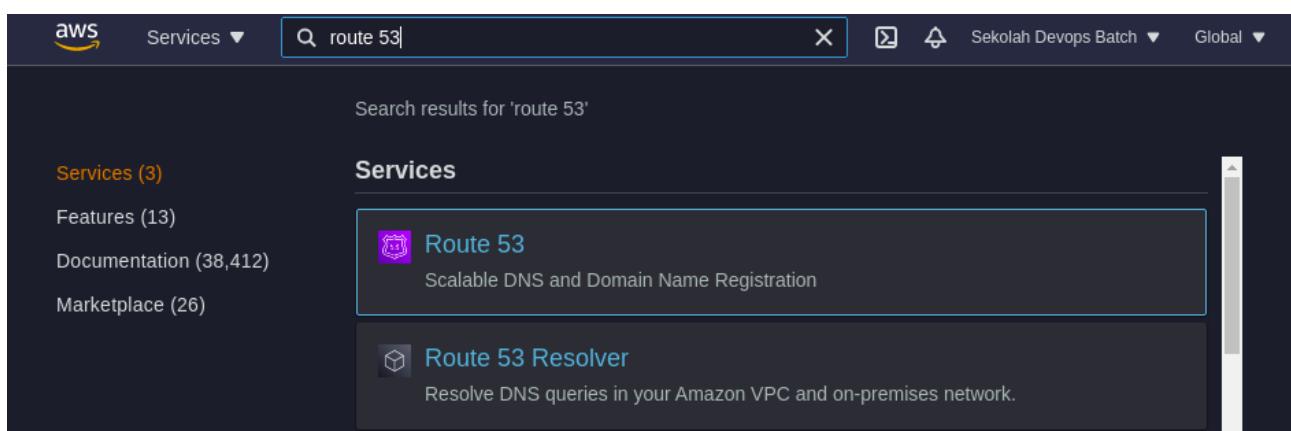
- 1.** IDCloudhost
- 2.** Qwords
- 3.** Dowa Web
- 4.** ID Hostinger
- 5.** Pandi
- 6.** Dll

Disini kami sendiri akan menggunakan provider IDCloudhost untuk domain yang akan kita gunakan. Kami akan memberikan sebuah akun yang dapat kalian gunakan untuk mengakses domain tersebut.

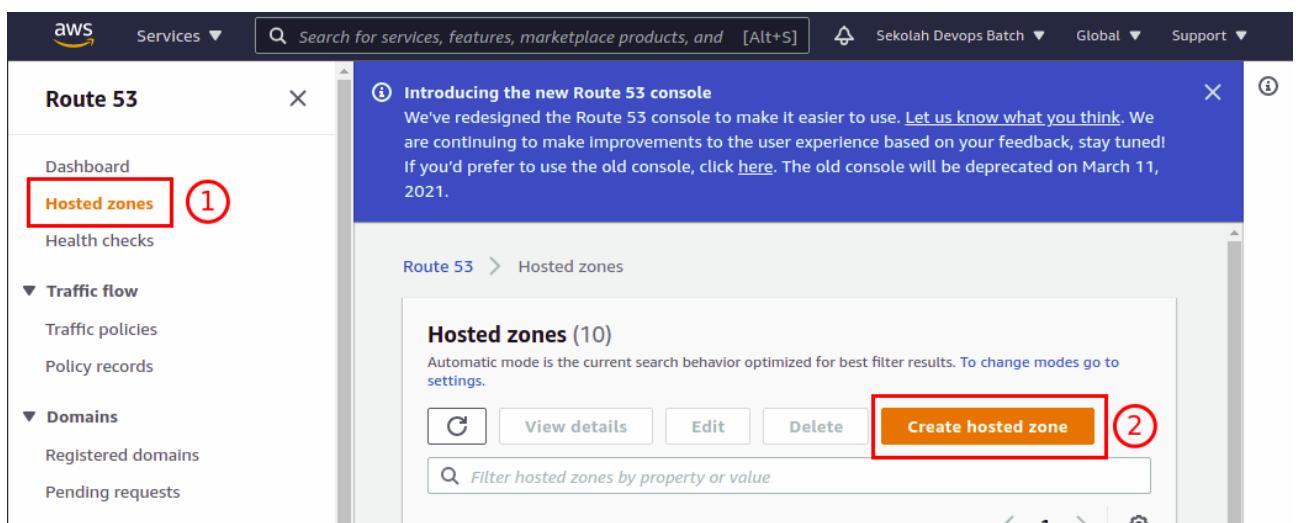
## 6.8.2. Konfigurasi Domain di Route53

Pada bagian ini kita akan coba untuk melakukan konfigurasi pada Route53, disini kita akan memasukan domain yang akan kita gunakan pada layanan ini. Disini kita akan mecontohkan dengan menggunakan domain **sdcilsy-alpha.web.id**.

Kalian dapat menyesuaikan kembali nanti dengan domain yang kalian miliki. Pertama yang harus kita lakukan adalah masuk kedalam **dashboard Route53**. Caranya adalah masuk kedalam **Services** lalu cari **Route53**.



Setelah masuk kedalam dashboard DNS Manager di Route53, selanjutnya kita buat sebuah **Hosted Zone** baru dengan menklik tombol **Create Hosted Zone**.



Setelah itu akan muncul pop-out dari samping, masukan alamat domain yang akan kita daftarkan pada route53. Disini kita akan masukan domain **sdcilsy-alpha.web.id**. Setelah itu klik **Create** untuk menyimpan.

**Create hosted zone** Info

**Hosted zone configuration**  
 A hosted zone is a container that holds information about how you want to route traffic for a domain, such as example.com, and its subdomains.

**Domain name** Info  
 This is the name of the domain that you want to route traffic for.  
 1

Valid characters: a-z, 0-9, ! " # % & ' ) \* + , - / ; < = > ? @ [ \ ] ^ \_ { } , ~

**Description - optional** Info  
 This value lets you distinguish hosted zones that have the same name.

The description can have up to 256 characters. 0/256

**Type** Info  
 The type indicates whether you want to route traffic on the Internet or in an Amazon VPC.

**Public hosted zone**  
 A public hosted zone determines how traffic is routed on the Internet.

**Private hosted zone**  
 A private hosted zone determines how traffic is routed within an Amazon VPC.

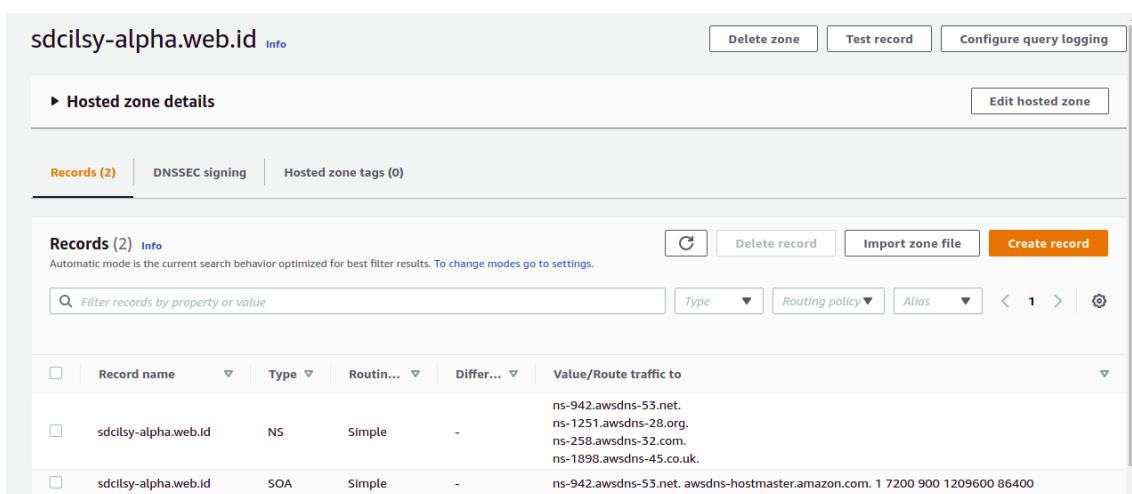
**Tags** Info  
 Apply tags to hosted zones to help organize and identify them.

No tags associated with the resource.

**Add tag** 2  
 You can add up to 50 more tags.

**Create hosted zone** Create hosted zone

Setelah itu kita akan diarahkan kedalam hasil setting dari domain yang kita masukan barusan, disana kita akan melihat Name Server (NS) dan juga SOA dari domain tersebut.

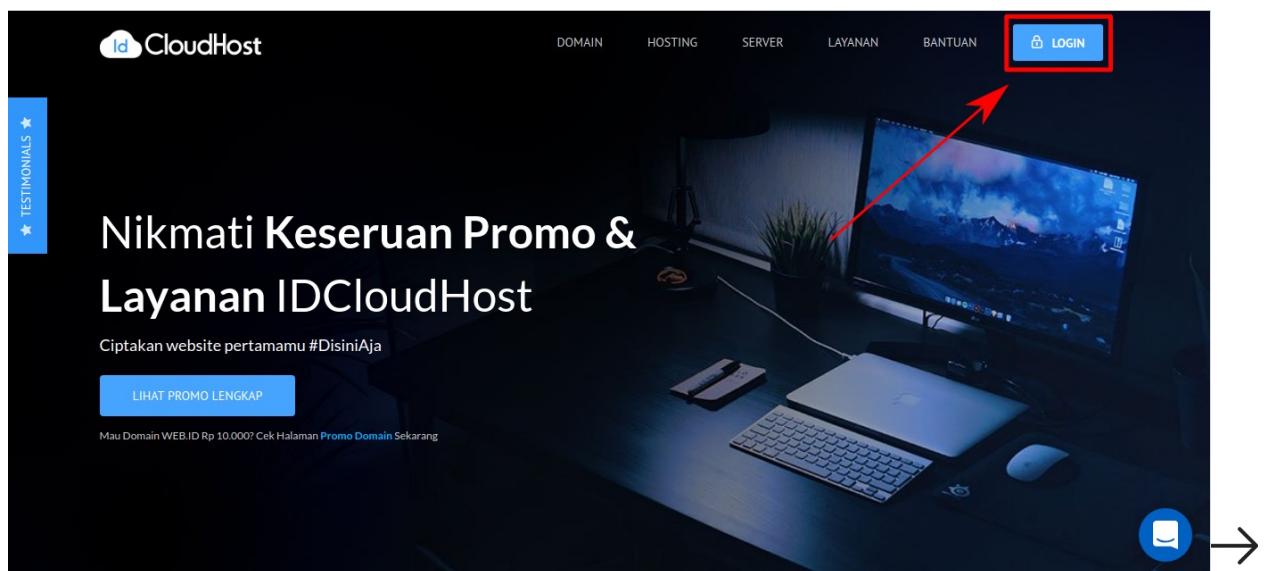


NS ini yang nanti akan kita masukan kedalam domain yang kita miliki, agar nantinya route53 bisa mengarahkan server yang ada di AWS ke domain yang sudah kita miliki.

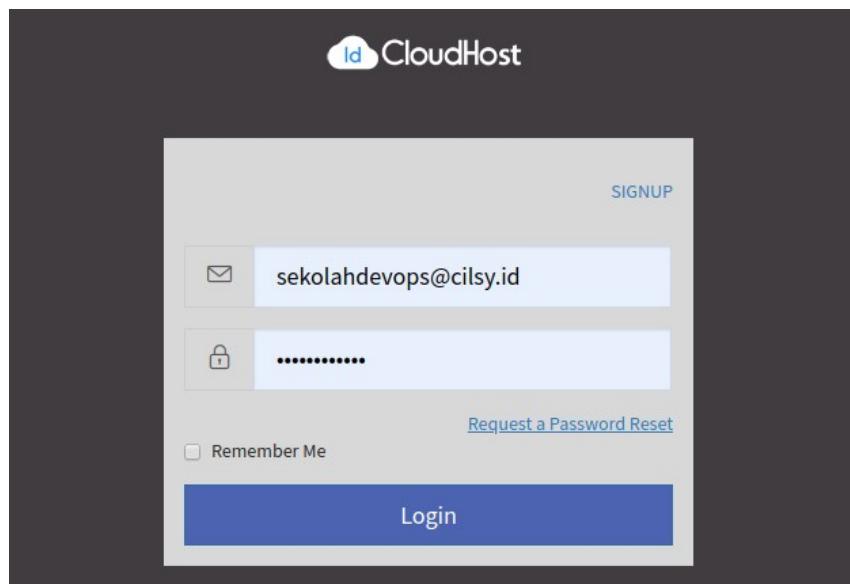
### 6.8.3. Konfigurasi Domain Provider

Setelah kita melakukan konfigurasi pada route53, sekarang kita coba konfigurasi Name Server pada domain yang sudah kita siapkan. Disini kita menggunakan domian dari IDCloudHost untuk providernya.

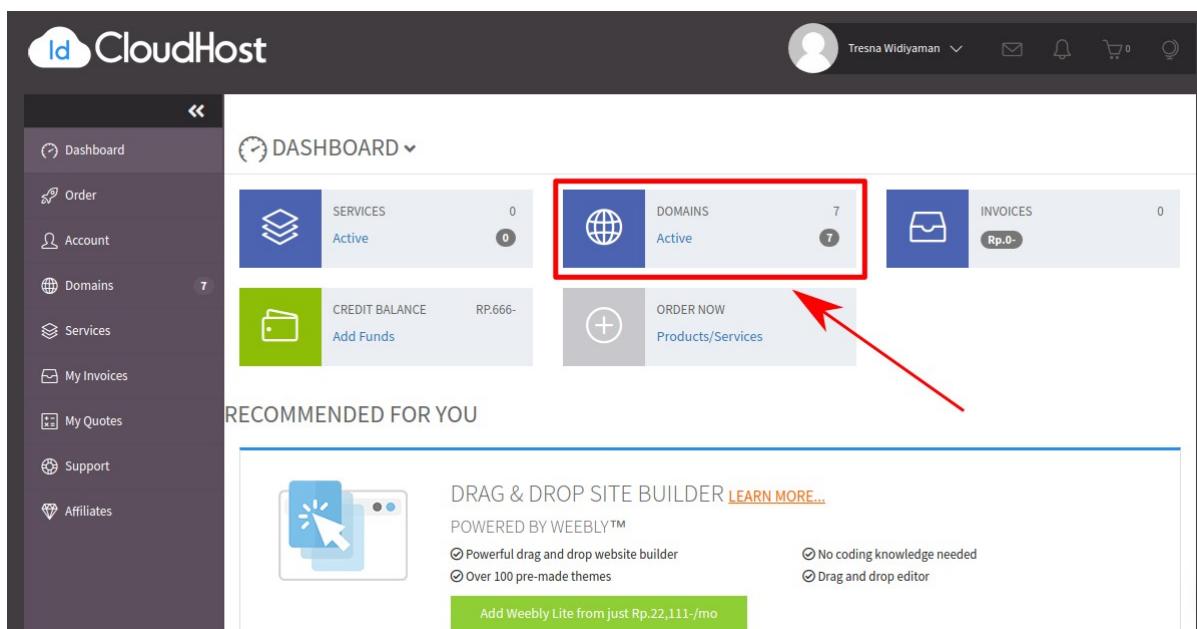
Untuk masuk kedalam dashboard, pertama kita harus mengakses alamat resminya di <https://idcloudhost.com/>. Setelah itu masuk kedalam webnya klik tombol **Login**.



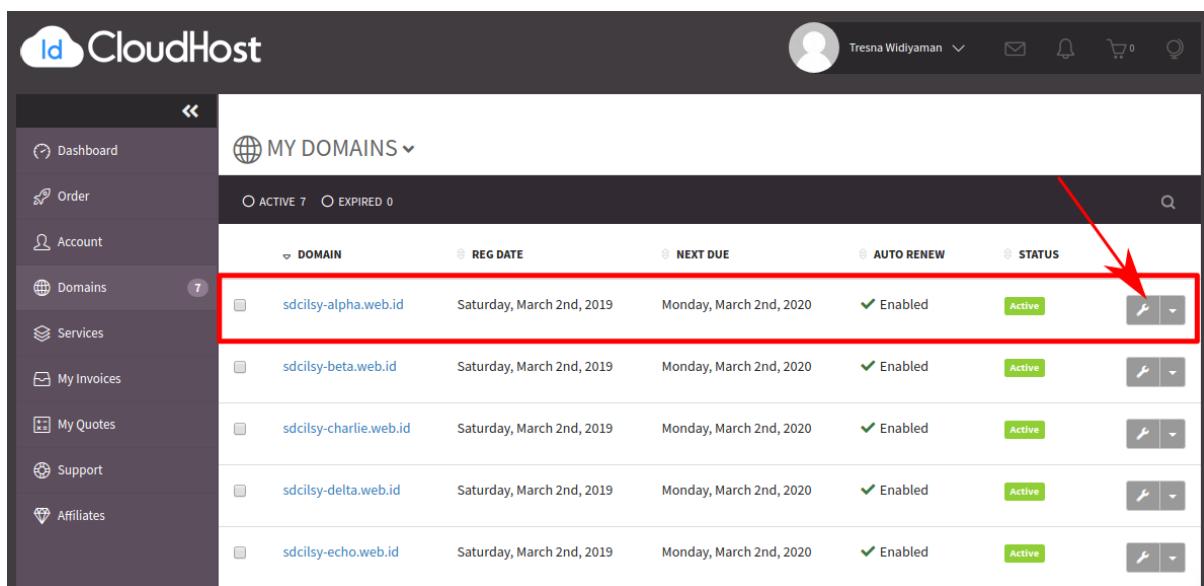
Setelah itu coba login dengan menggunakan user dan password yang sudah diberikan sebelumnya. Lalu klik Login.



Kita akan diarahkan ke dashboard dari IDCloudHost tersebut, pilih menu domain agar kita masuk kedalam list domain yang kita miliki.

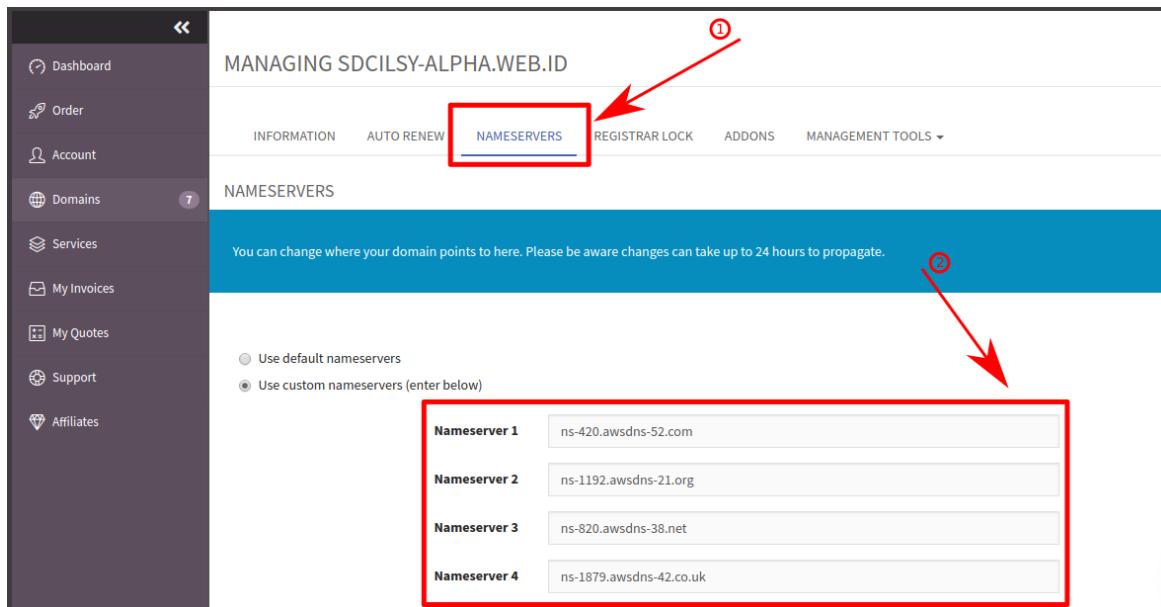


Setelah itu pilih domain yang akan kita set dan sudah kita masukan kedalam route53 tadi, klik tombol setting yang ada disampingnya.



The screenshot shows the CloudHost control panel under the 'Domains' section. It lists seven active domains, each with details like registration date, next due date, auto-renew status, and current status (all marked as 'Enabled' and 'Active'). The first domain, 'sdcilsy-alpha.web.id', is highlighted with a red box, and a red arrow points to its edit icon.

Kita akan masuk kedalam menu setting, pilih tab NameServers, pilih use custom nameserver dan isikan NS yang sebelumnya kita dapatkan di route53 seperti dibawah ini.



The screenshot shows the 'MANAGING SDCILSY-ALPHA.WEB.ID' page. The 'NAMESERVERS' tab is selected and highlighted with a red box. Red arrows point from the text above to the 'Use custom nameservers' radio button and the list of four nameservers (Nameserver 1 to Nameserver 4), which are also highlighted with a red box.

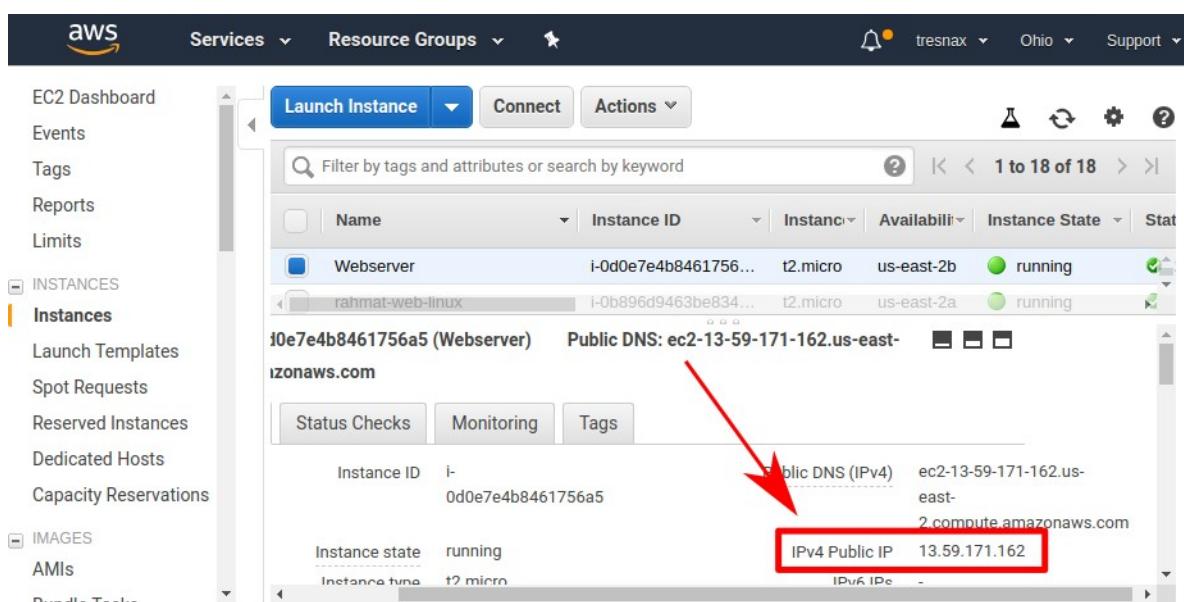
Nameserver	Value
Nameserver 1	ns-420.awsdns-52.com
Nameserver 2	ns-1192.awsdns-21.org
Nameserver 3	ns-820.awsdns-38.net
Nameserver 4	ns-1879.awsdns-42.co.uk

Setelah selesai klik **save** untuk menyimpan perubahan yang telah dilakukan. Selanjutnya kita akan coba hubungkan instance yang sudah kita siapkan untuk web server agar dapat diakses oleh domain tersebut.

### 6.8.4. Integrasi Instance ke Route53

Pada bagian ini kita akan coba untuk mengarahkan instance pada domain yang sudah kita setting tadi. Hal pertama yang harus kita lakukan adalah menyiapkan **webserver** yang sudah berjalan. Kalian dapat membuat sebuah server baru atau juga menggunakan server yang sudah ada.

Disini saya sudah menyimpulkan satu buah webserver yang sudah berjalan, pilih webserver tersebut lalu salin **IP Public** yang dia miliki.



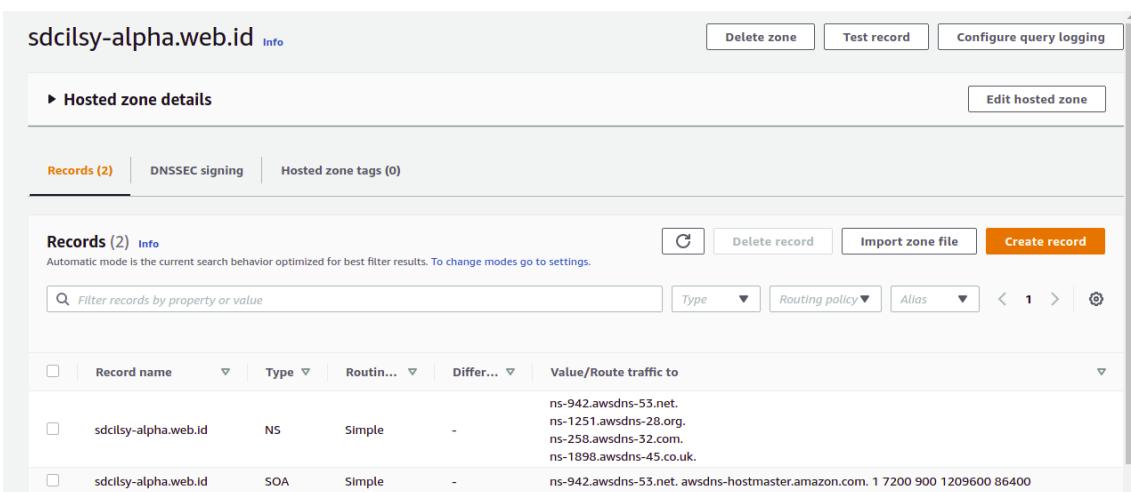
The screenshot shows the AWS EC2 Instances page. On the left sidebar, under the 'INSTANCES' section, 'Instances' is selected. In the main content area, there are two instances listed:

- Webserver**: Instance ID i-0d0e7e4b8461756a5, t2.micro, us-east-2b, running. Its Public DNS is ec2-13-59-171-162.us-east-2.compute.amazonaws.com.
- rahmat-web-linux**: Instance ID i-0b896d9463be834..., t2.micro, us-east-2a, running.

Below the instances, for the 'Webserver' instance, the following details are shown:

Instance ID	i-0d0e7e4b8461756a5	Public DNS (IPv4)	ec2-13-59-171-162.us-east-2.compute.amazonaws.com
Instance state	running	IPv4 Public IP	13.59.171.162
Instance type	t2.micro	IPv6 IP	-

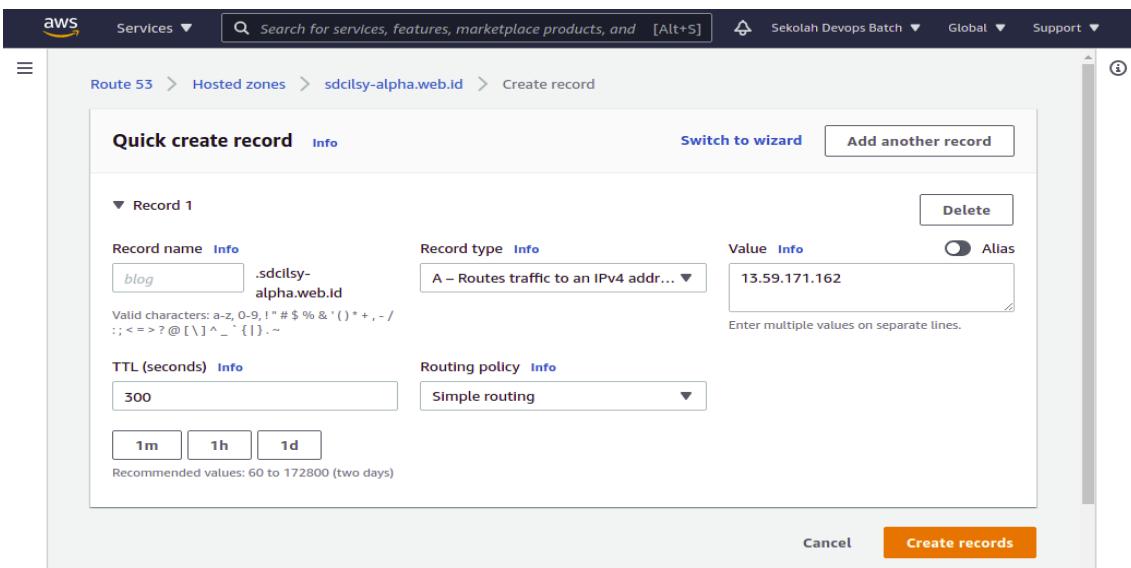
Setelah kita salin IP Public tersebut, selanjutnya kita masuk kembali ke dashboard Route53 yang ada di AWS. Masuk kedalam Zone domain yang sudah kita buat, klik **Create Record**.



The screenshot shows the AWS Route 53 Hosted Zone Details page for the domain `sdcilsy-alpha.web.id`. Under the 'Records' tab, there are two entries:

- NS Record:** Record name `sdcilsy-alpha.web.id`, Type `NS`, Value `ns-942.awsdns-53.net.` This record also points to `ns-1251.awsdns-28.org.`, `ns-258.awsdns-32.com.`, and `ns-1898.awsdns-45.co.uk.`
- SOA Record:** Record name `sdcilsy-alpha.web.id`, Type `SOA`, Value `ns-942.awsdns-53.net. awsdns-hostmaster.amazon.com. 1 7200 900 1209600 86400`

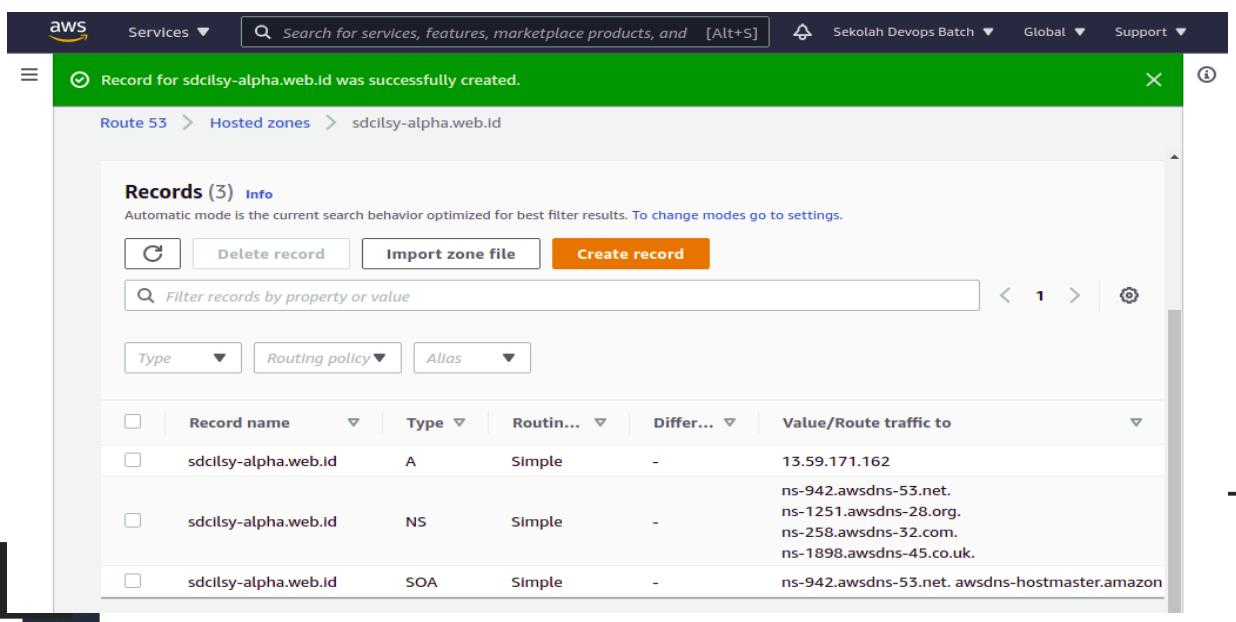
Lalu akan kita akan diarahkan untuk mengisikan alamat **IP Public** tadi dibagian value. Setelah itu klik Tombol **Create Record** untuk menyimpan.



The screenshot shows the 'Quick create record' dialog in the AWS Route 53 console. The form is filled with the following values:

- Record name:** `blog`
- Record type:** `A – Routes traffic to an IPv4 address`
- Value:** `13.59.171.162`
- TTL (seconds):** `300`
- Routing policy:** `Simple routing`

Setelah itu kita bisa melihat hasil yang sudah kita buat seperti dibawah ini.

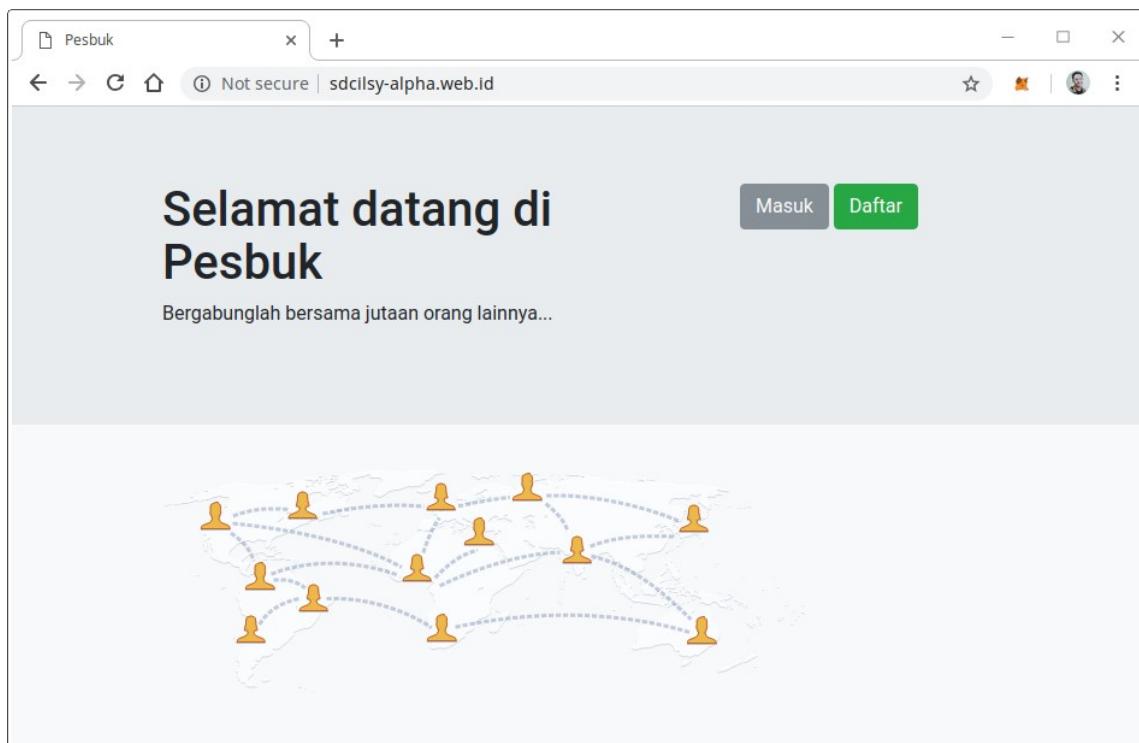


The screenshot shows the AWS Route 53 Hosted Zone Details page for the domain `sdcilsy-alpha.web.id`. Under the 'Records' tab, there are three entries:

- A Record:** Record name `blog`, Type `A`, Value `13.59.171.162`
- NS Record:** Record name `sdcilsy-alpha.web.id`, Type `NS`, Value `ns-942.awsdns-53.net.` This record also points to `ns-1251.awsdns-28.org.`, `ns-258.awsdns-32.com.`, and `ns-1898.awsdns-45.co.uk.`
- SOA Record:** Record name `sdcilsy-alpha.web.id`, Type `SOA`, Value `ns-942.awsdns-53.net. awsdns-hostmaster.amazon.com. 1 7200 900 1209600 86400`

Biasanya proses penerapan domain ini akan memakan waktu maksimal 24 jam, dalam waktu sekitar 1-3 menit biasanya sudah bisa langsung kita akses doamin tersebut. Tapi terkadang kita juga harus menunggu sampai 24 penuh untuk bisa mengakses alamat domainnya.

Berikut merupakan hasil domain yang sudah berhasil mengakses server aws.

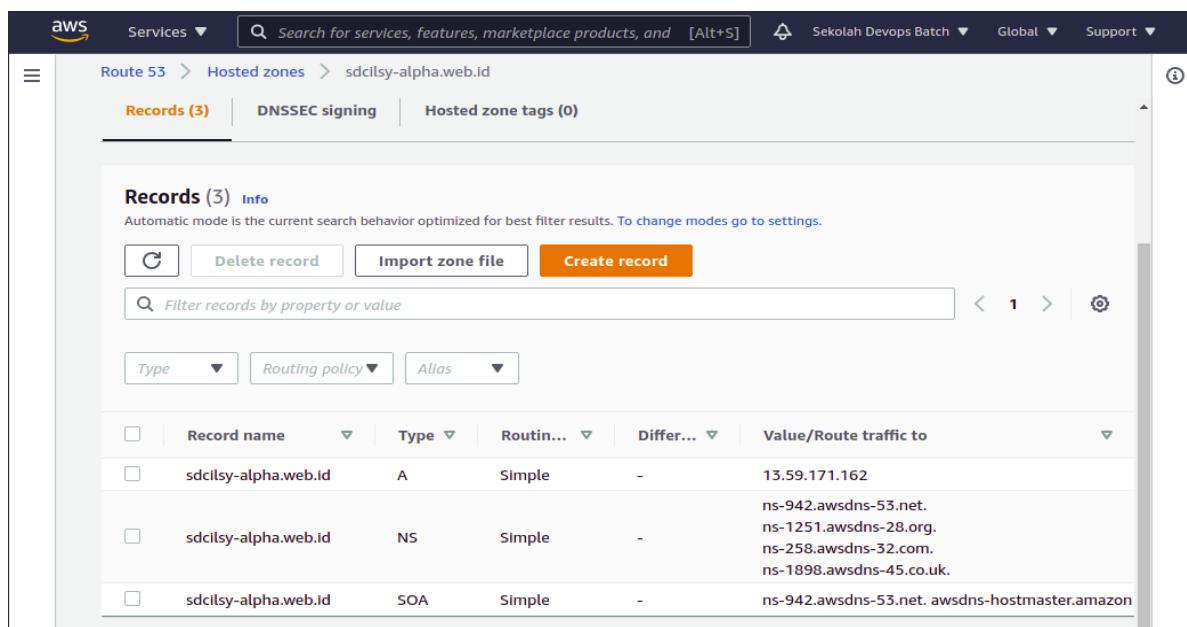


*Hasil domain Route53*

### 6.8.5. Setup Subdomain dan Layanan AWS Lainnya

Setelah kita berhasil membuat domain dapat diakses dan mengarahkan alamat instance, sekarang kita akan coba membuat sebuah subdomain sekaligus mengarahkannya ke salah satu load balance yang ada.

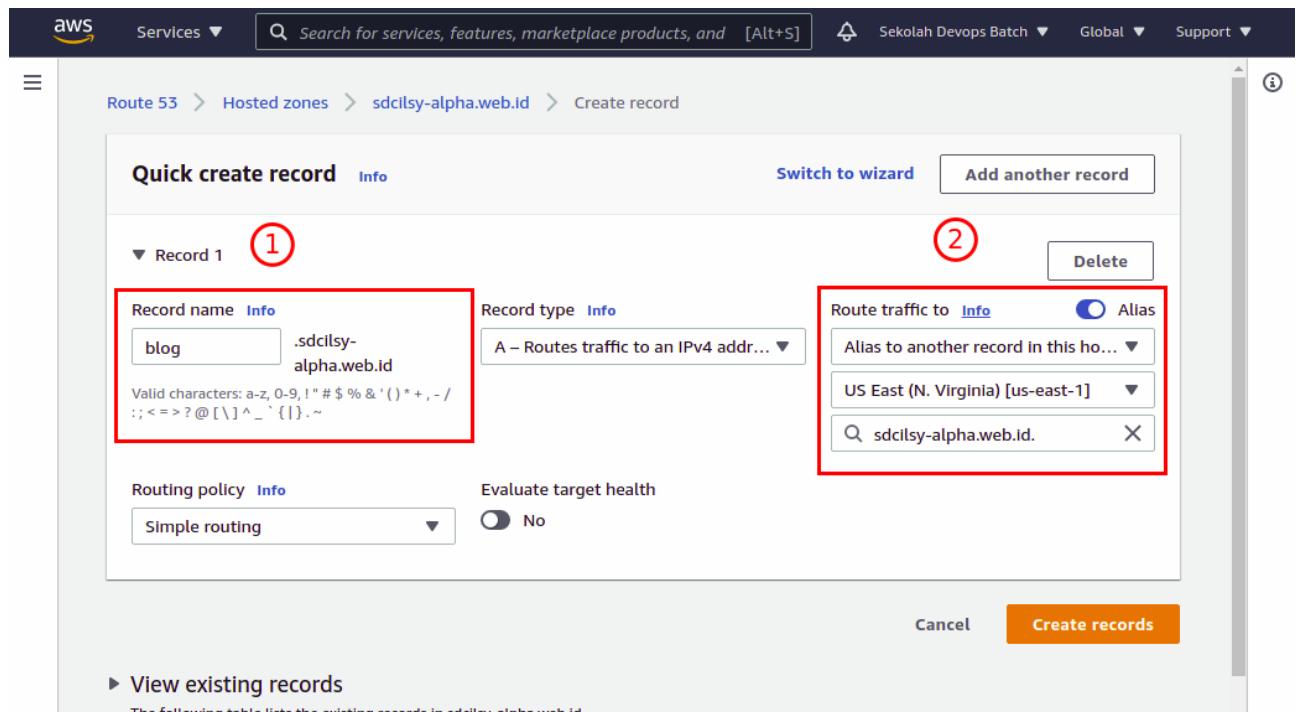
Prosesnya masih sama seperti pada saat akan membuat record pertama kali,



The screenshot shows the AWS Route 53 Hosted Zones interface. In the top navigation bar, 'Route 53' is selected under 'Hosted zones'. Below it, the subdomain 'sdclisy-alpha.web.id' is chosen. The main area displays three records:

Type	Record name	Type	Routing policy	Differ...	Value/Route traffic to
A	sdclisy-alpha.web.id	A	Simple	-	13.59.171.162 ns-942.awsdns-53.net. ns-1251.awsdns-28.org. ns-258.awsdns-32.com. ns-1898.awsdns-45.co.uk.
NS	sdclisy-alpha.web.id	NS	Simple	-	ns-942.awsdns-53.net. awsdns-hostmaster.amazon
SOA	sdclisy-alpha.web.id	SOA	Simple	-	

Pada bagian **Name** diisikan dengan subdomain yang kita inginkan. Pada bagian **Alias** pilih Yes dan pada **Alias Target** pilih Load Balance, Elastic Beanstalk, ataupun Amazon S3 yang akan kita arahkan ke subdomain agar mudah dipanggil nantinya.

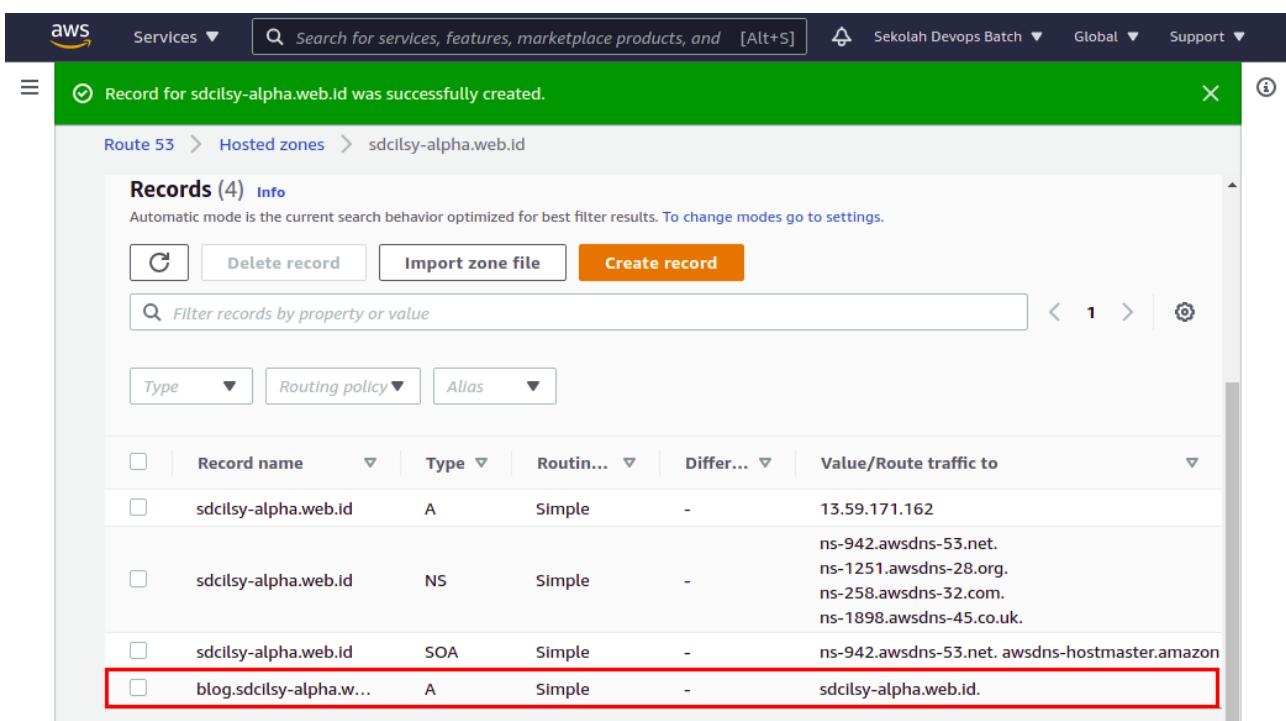


The screenshot shows the 'Create record' dialog for a new record named 'Record 1'. The configuration is as follows:

- Record name:** blog.sdcilisy-alpha.web.id (highlighted by a red box)
- Record type:** A – Routes traffic to an IPv4 address (highlighted by a red box)
- Route traffic to:** Alias (radio button selected) (highlighted by a red box)
  - Alias to another record in this hosted zone: sdclisy-alpha.web.id (highlighted by a red box)
  - Region: US East (N. Virginia) [us-east-1]
- Routing policy:** Simple routing
- Evaluate target health:** No

Setelah kita simpan, hasilnya konfigurasinya akan terlihat seperti dibawah ini.

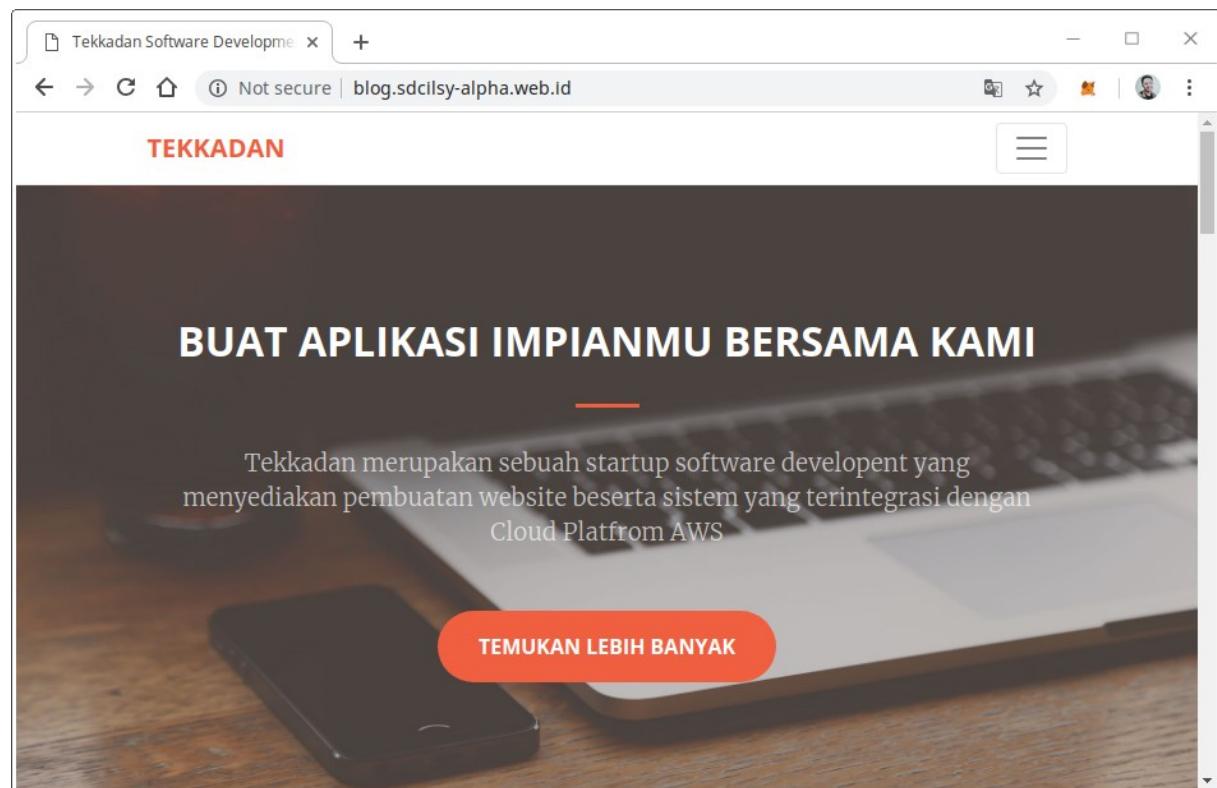




The screenshot shows the AWS Route 53 service page. At the top, there is a green success message: "Record for sdcilsy-alpha.web.id was successfully created." Below this, the navigation path is "Route 53 > Hosted zones > sdcilsy-alpha.web.id". The main area is titled "Records (4) Info" and contains a table of records:

	Record name	Type	Routing policy	Differ...	Value/Route traffic to
<input type="checkbox"/>	sdcilsy-alpha.web.id	A	Simple	-	13.59.171.162 ns-942.awsdns-53.net. ns-1251.awsdns-28.org. ns-258.awsdns-32.com. ns-1898.awsdns-45.co.uk.
<input type="checkbox"/>	sdcilsy-alpha.web.id	NS	Simple	-	ns-942.awsdns-53.net. awsdns-hostmaster.amazon
<input type="checkbox"/>	blog.sdcilsy-alpha.w...	SOA	Simple	-	sdcilsy-alpha.web.id.
<input type="checkbox"/>	blog.sdcilsy-alpha.w...	A	Simple	-	sdcilsy-alpha.web.id.

Jika kita coba akses alamat subdomain tersebut, maka akan otomatis mengarahkan pada instance yang kita miliki seperti dibawah ini.



Dengan begini kita sudah bisa mengarahkan server yang sudah kita siapkan sehingga bisa diakses oleh user melalui domain.

### 6.8.6. Exercise

1. Konfigurasi main domain dengan load balance server yang kalian miliki, sehingga load balance bisa dipanggil melalui main domain !
2. Konfigutasi Beanstalk, Amazon S3 dengan menggunakan subdomain pada amazon Route53 !

## 6.9. Amazon RDS (Relational Database Service)

### 6.9.1. Apa itu Amazon RDS ?

**Amazon Relational Database Service (Amazon RDS)** adalah layanan server basis data (*database*) dimana data dan server akan berada di cloud yang akan menjamin kualitas koneksi, kecepatan, keamanan dan kehandalan. Kita dapat memiliki aplikasi server yang kita mau seperti MySQL, Oracle, PostgreSQL dan SQL Server.



*Amazon RDS Logo*

Dengan menggunakan amazon RDS, server dari database yang kita miliki bisa dimuat secara terpisah dari server tersebut. Kita bisa sesuaikan kapasitas dari server database ini seperti halnya menyesuaikan sebuah instance pada umumnya.

Salah satu kelebihan utama yang dimiliki layanan ini adalah replica database, layanan ini memungkinkan kita untuk melakukan replica atau mirroring database. Sehingga database yang kita miliki akan aman apabila terjadi kesalahan yang fatal. Inilah yang membedakan database RDS dengan database yang biasanya kita gunakan.

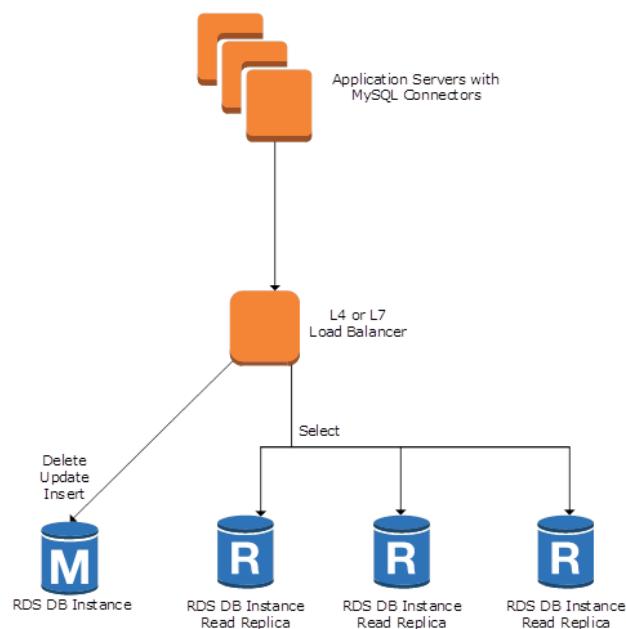
### 6.9.2. Fungsi dan kegunaan Amazon RDS

Selain berfungsi sebagai layanan server database, Amazon RDS sendiri memiliki beberapa fungsi lain dan keunggulannya. Berikut keuntungan menggunakan Amazon RDS:

- Tidak membutuhkan Maintenance atau perawatan. Apalagi bila tidak memiliki database administrator yang bertugas merawat, mengoptimasi, memantau, termasuk urusan update dan perbaikan software. Beda halnya jika kita menggunakan database pada Amazon EC2, kita harus melakukan sendiri proses instalasi, konfigurasi, optimasi, hingga menangani masalah ketika menggunakan database.
- Mudah melakukan pengaturan, mulai dari meningkatkan kapasitas storage, CPU, memori, hingga duplikasi read-replica melalui halaman AWS Management Console.
- Backup otomatis. Amazon RDS akan melakukan backup rutin pada waktu yang telah kita tentukan. Jika ingin menambah waktu backup, ada tambahan biaya lagi. Backup berupa snapshot (kondisi mesin) sehingga bisa dengan mudah di-restore jika terjadi masalah. Beda halnya dengan menggunakan Amazon EC2, kita harus membuat script untuk melakukan backup. Script melakukan dumping ke format SQL kemudian mengompresi dan menyimpan ke Amazon S3 yang dilakukan lewat cron.
- Tersedia halaman monitoring. Untuk mengetahui penggunaan CPU dan jumlah koneksi yang terjadi. Akan tetapi kita tidak bisa mengetahui penggunaan storage.

Selain daripada kelebihan yang dapat kita nikmati, adapula beberapa kekurangan yang menjadi nilai minus dari Amazon RDS. Berikut merupakan kekurangan Amazon RDS :

- Tidak ada akses langsung ke file konfigurasi. Jika pada MySQL kita bisa melakukan konfigurasi spesifik melalui file my.cnf atau my.ini, Amazon RDS menggunakan API yang bisa diakses menggunakan Amazon RDS Command Line Tools.
- Amazon RDS dibatasi oleh jumlah koneksi maksimal. Tidak ada cara lain untuk menaikkan jumlah koneksi maksimal kecuali menaikkan spesifikasi instance.



*Gambaran Amazon RDS*

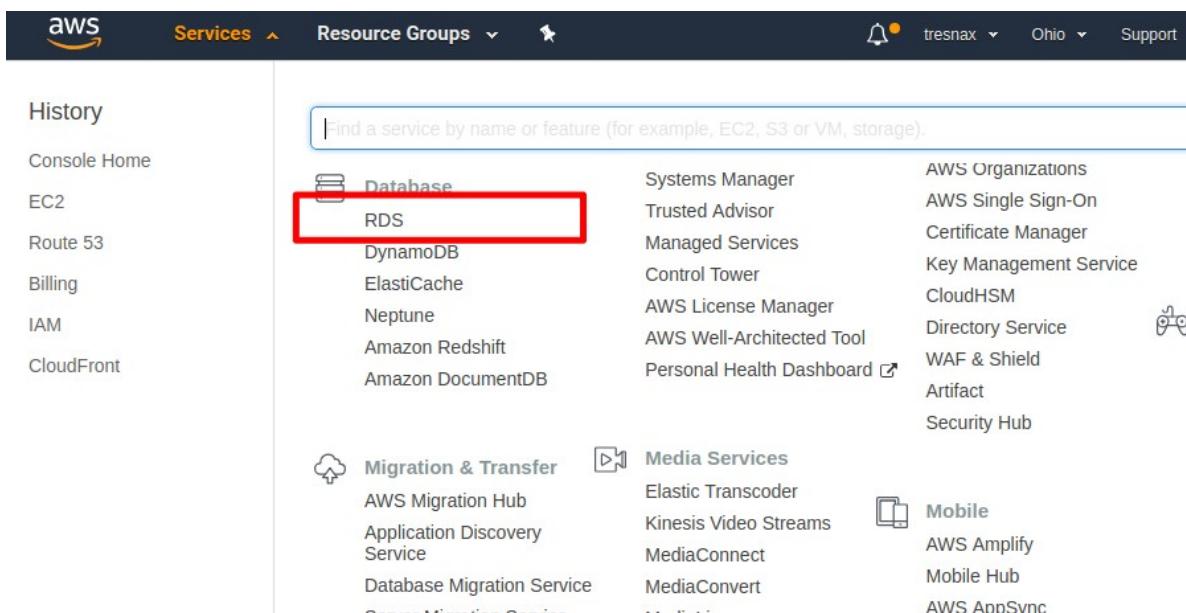
### 6.9.3. Exercise

1. Sebutkan aplikasi server apa saja yang dapat berjalan di Amazon RDS ?
2. Jelaskan secara singkat keunggulan dan kekurangan Amazon RDS !

## 6.10. Setup Amazon RDS

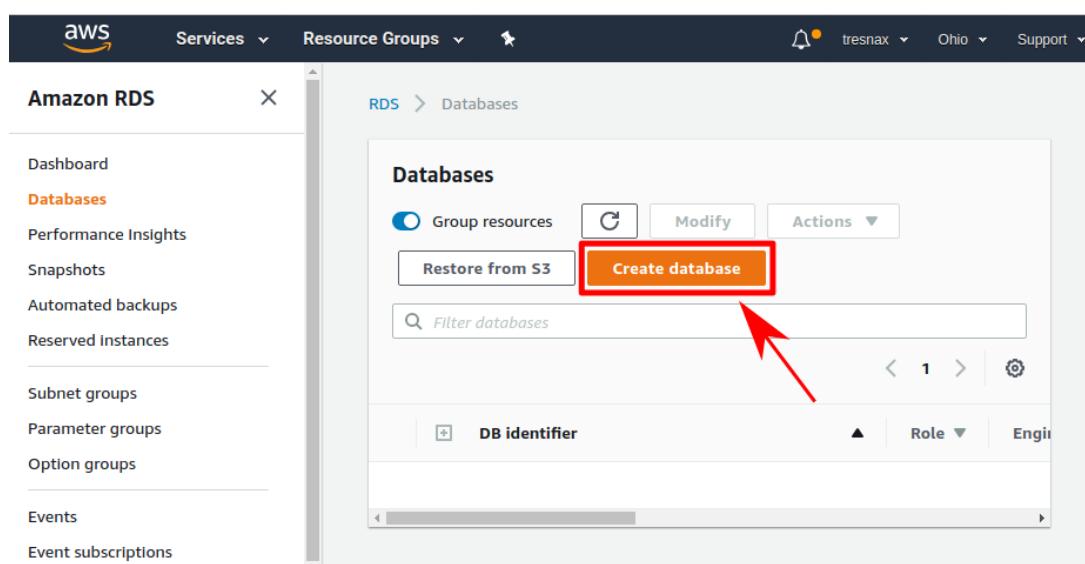
### 6.10.1. Launch DB Instance

Pada bagian ini kita akan melakukan **setup pada Amazon RDS**, Hal pertama yang harus kita lakukan adalah masuk kedalam menu **Service** lalu cari **RDS** atau Search pada kolom text dengan pencarian RDS seperti berikut.



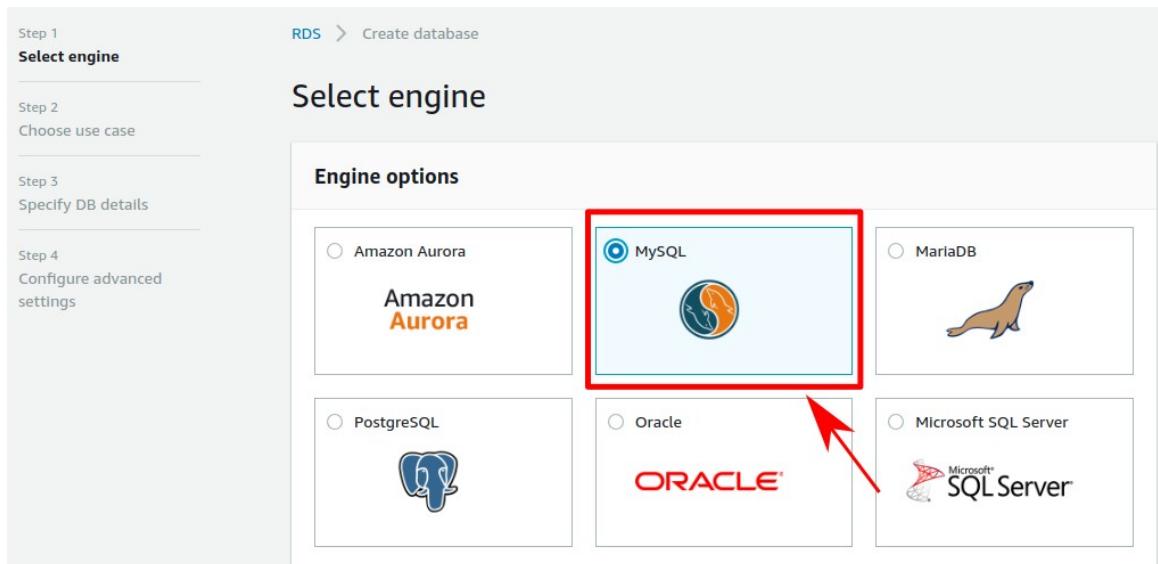
The screenshot shows the AWS Services dashboard. At the top, there's a search bar with placeholder text "Find a service by name or feature (for example, EC2, S3 or VM, storage)". Below the search bar, the "Database" section is expanded, showing various services: RDS (which is highlighted with a red box), DynamoDB, ElastiCache, Neptune, Amazon Redshift, and Amazon DocumentDB. To the right of the database section, there are other service categories like Systems Manager, Trusted Advisor, Managed Services, etc., and a sidebar with links to History, Console Home, EC2, Route 53, Billing, IAM, and CloudFront.

Selanjutnya kita akan diarahkan pada dashboard Amazon RDS. Untuk membuat DB Instance baru kita harus masuk ke menu **Database** lalu klik tombol **Create database** yang ditunjuk seperti gambar dibawah ini.

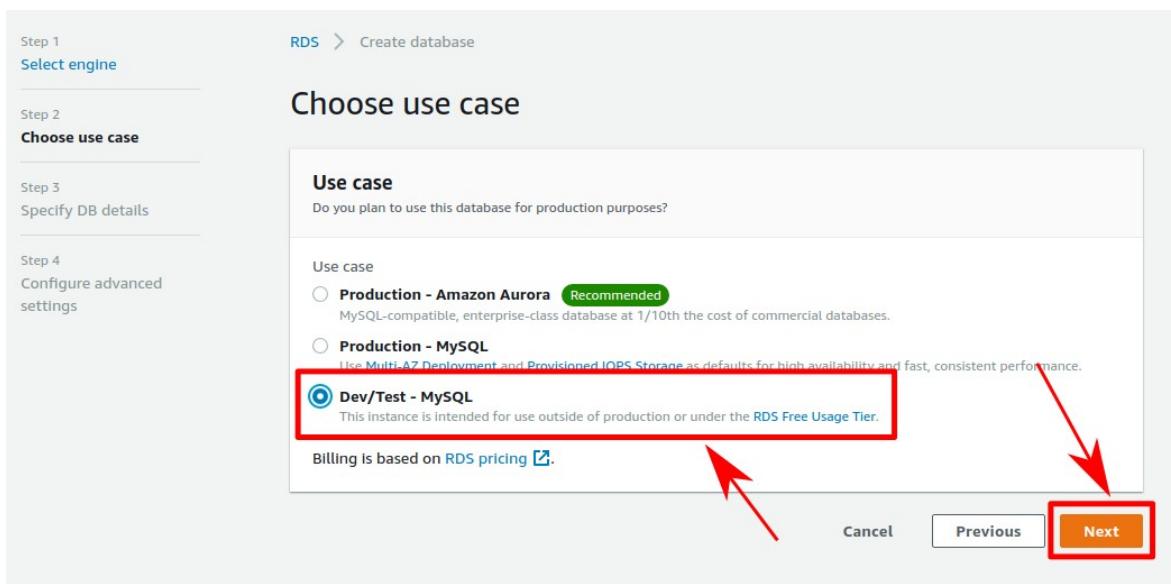


The screenshot shows the "Amazon RDS" service dashboard. On the left, there's a sidebar with links to Dashboard, Databases (which is highlighted in orange), Performance Insights, Snapshots, Automated backups, Reserved instances, Subnet groups, Parameter groups, Option groups, Events, and Event subscriptions. The main area is titled "Databases" and shows a table with columns for DB identifier, Role, and Engine. At the top of the table, there are buttons for "Group resources", "Restore from S3", "Create database" (which is highlighted with a red box and has a red arrow pointing to it), and "Actions".

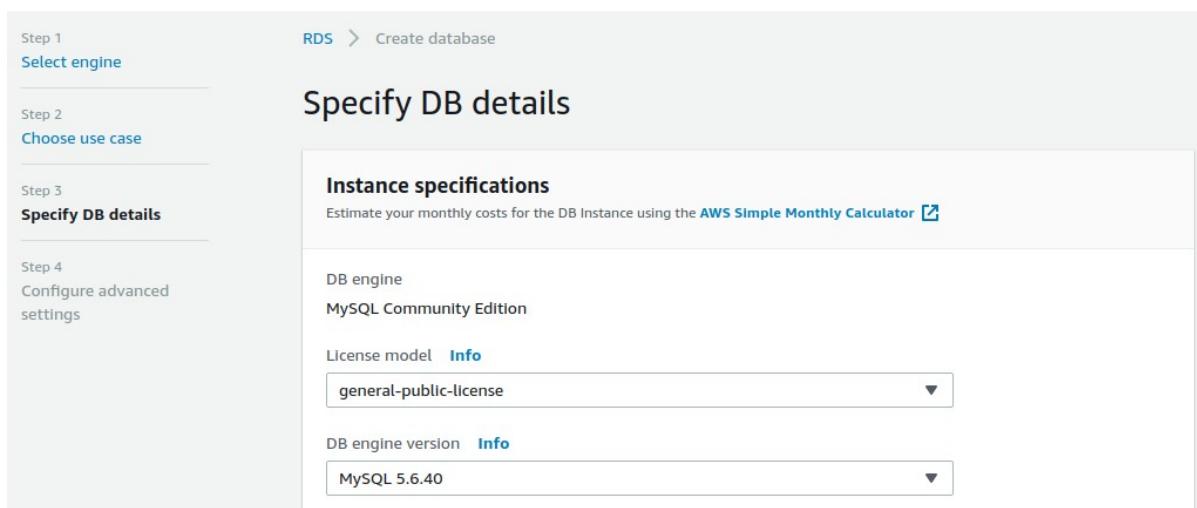
Kemudian pilih **Engine database** yang akan kita gunakan, disini kita akan menggunakan **MySQL**. Selain MySQL, Amazon RDS juga menyediakan database lain seperti Amazon Aurora, MariaDB, PostgreSQL, Oracle dan Microsoft SQL Server. Setelah memilih selanjutnya klik **Next**.



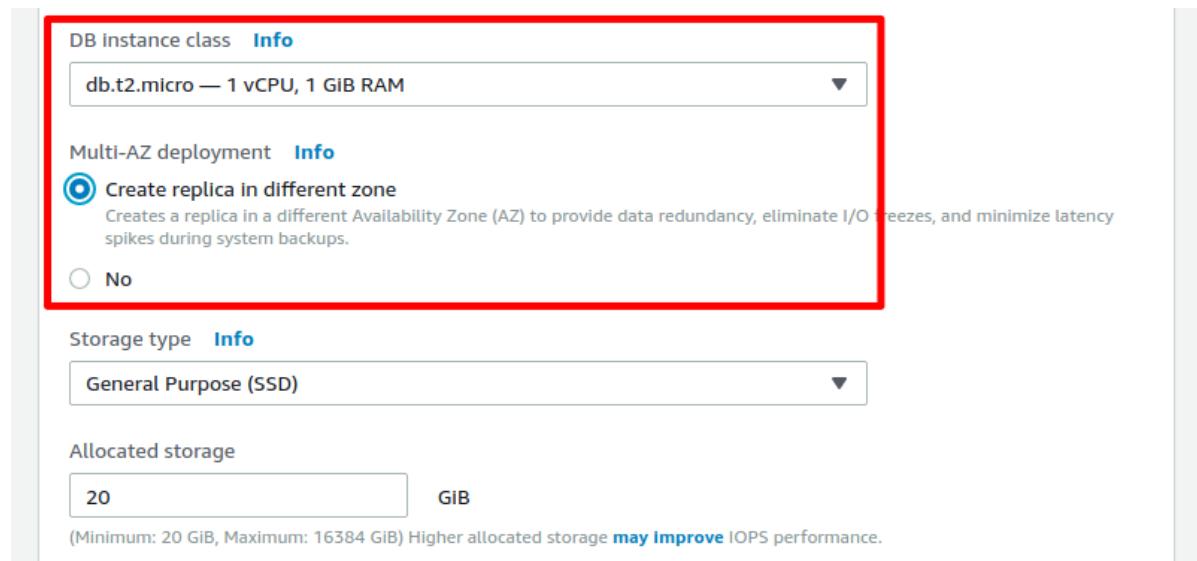
Pada bagian **Use Case** kita pilih **Dev/Test** lalu klik **Next** untuk melanjutkan konfigurasi.



Pada bagian **Specify DB details**, kita akan memilih versi mysql yang akan kita gunakan. Disini kita akan coba gunakan **versi 5.6.40**.



Kita dapat membuat **replikasi** ke region atau zona lain pada Amazon RDS dengan memilih **Create replica in different zone**. Default size Strorage 20GB, dan type storage dengan SSD.



Tahap selanjutnya kita setting **DB Instance Identifier** dengan nama **db-test** dan user **root** lalu master password untuk database yang akan kita buat dengan amazon RDS.

**Settings**

**DB Instance Identifier** [Info](#)  
Specify a name that is unique for all DB instances owned by your AWS account in the current region.

DB instance identifier is case insensitive, but stored as all lower-case, as in "mydbinstance". Must contain from 1 to 63 alphanumeric characters or hyphens (1 to 15 for SQL Server). First character must be a letter. Cannot end with a hyphen or contain two consecutive hyphens.

**Master username** [Info](#)  
Specify an alphanumeric string that defines the login ID for the master user.

Master Username must start with a letter. Must contain 1 to 16 alphanumeric characters.

**Master password** [Info](#)

**Confirm password** [Info](#)

Master Password must be at least eight characters long, as in "mypassword". Can be any printable ASCII character except "/", "", or "@".

**Cancel** **Previous** **Next**



Pada bagian **Network and Security Group**, kita pilih **Default VPC** atau VPC yang sudah pernah kita buat. Pada **Public accessibility** pilih yes agar kita dapat membuka database melalui SQL client yang kita miliki.

**Step 4**  
**Configure advanced settings**

**Virtual Private Cloud (VPC)** [Info](#)  
VPC defines the virtual networking environment for this DB instance.  
 [C](#)

Only VPCs with a corresponding DB subnet group are listed.

**Subnet group** [Info](#)  
DB subnet group that defines which subnets and IP ranges the DB instance can use in the VPC you selected.

**Public accessibility** [Info](#)  
 Yes  
 EC2 instances and devices outside of the VPC hosting the DB instance will connect to the DB instances. You must also select one or more VPC security groups that specify which EC2 instances and devices can connect to the DB instance.  
 No  
 DB instance will not have a public IP address assigned. No EC2 instance or devices outside of the VPC will be able to connect.

**Availability zone** [Info](#)

**VPC security groups**  
Security groups have rules authorizing connections from all the EC2 instances and devices that need to access the DB instance.  
 Create new VPC security group  
 Choose existing VPC security groups  
 [▼](#)

Selanjutnya kita buat nama database dengan nama **db-test** dan port **default** pada database dengan menggunakan port 3306.

**Database options**

Database name [Info](#)  
 db\_test

Note: if no database name is specified then no initial MySQL database will be created on the DB Instance.

Port [Info](#)  
 TCP/IP port the DB instance will use for application connections.

DB parameter group [Info](#)

Option group [Info](#)

IAM DB authentication [Info](#)

Enable IAM DB authentication  
 Manage your database user credentials through AWS IAM users and roles.

Disable

Coba Scroll lagi ke bagian bawah, kita akan mendapatkan pengaturan monitoring. Disini kita pilih **Enable Enhanced Monitoring**, dengan **Granularity 60s**.

**Monitoring**

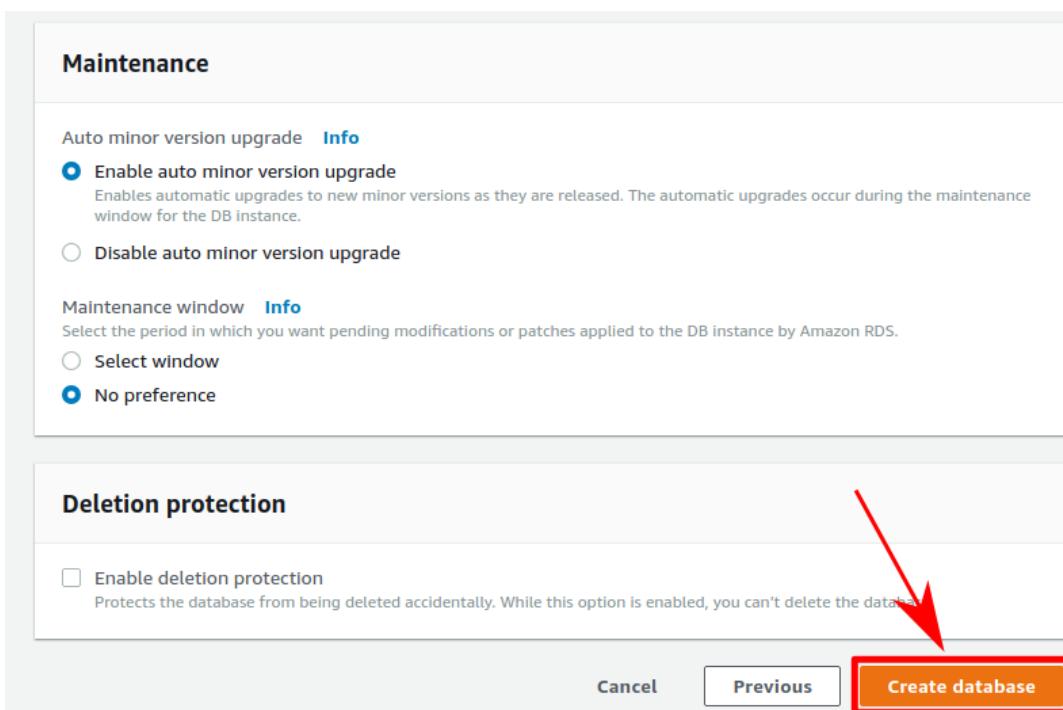
Enhanced monitoring  
 **Enable enhanced monitoring**  
 Enhanced monitoring metrics are useful when you want to see how different processes or threads use the CPU.

Disable enhanced monitoring

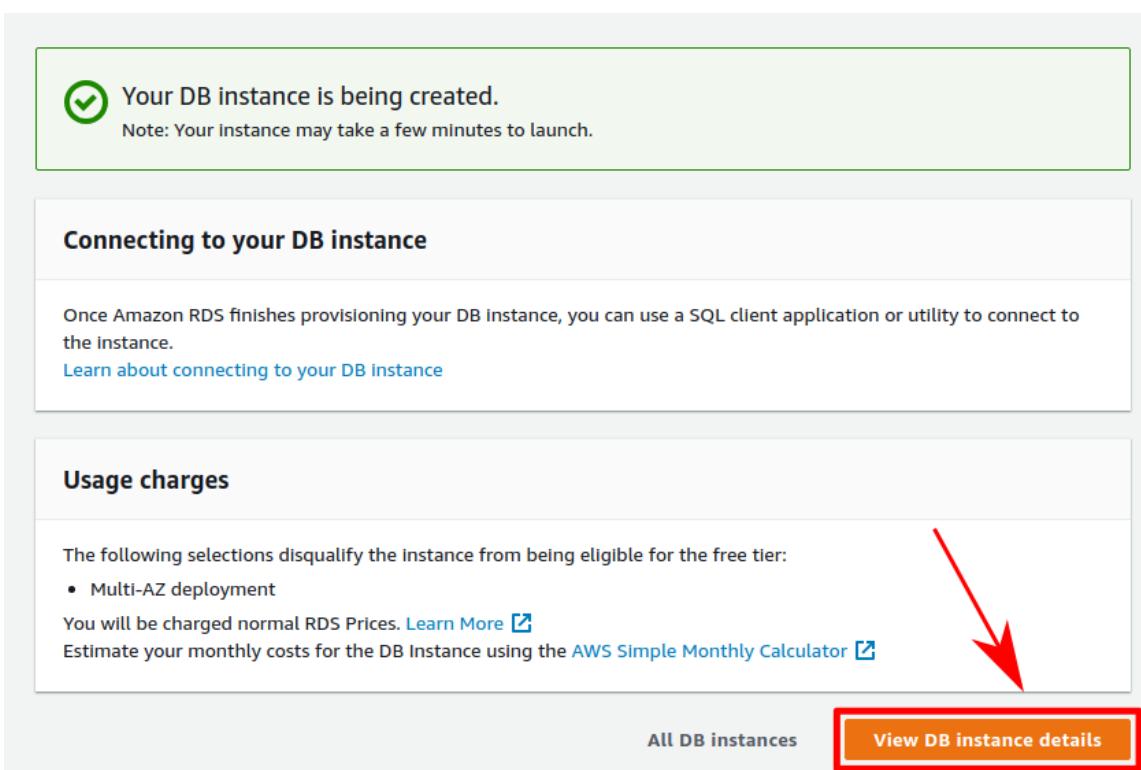
Monitoring Role	Granularity
<input type="text" value="Default"/>	<input type="text" value="60 seconds"/>

I authorize RDS to create the IAM role rds-monitoring-role.

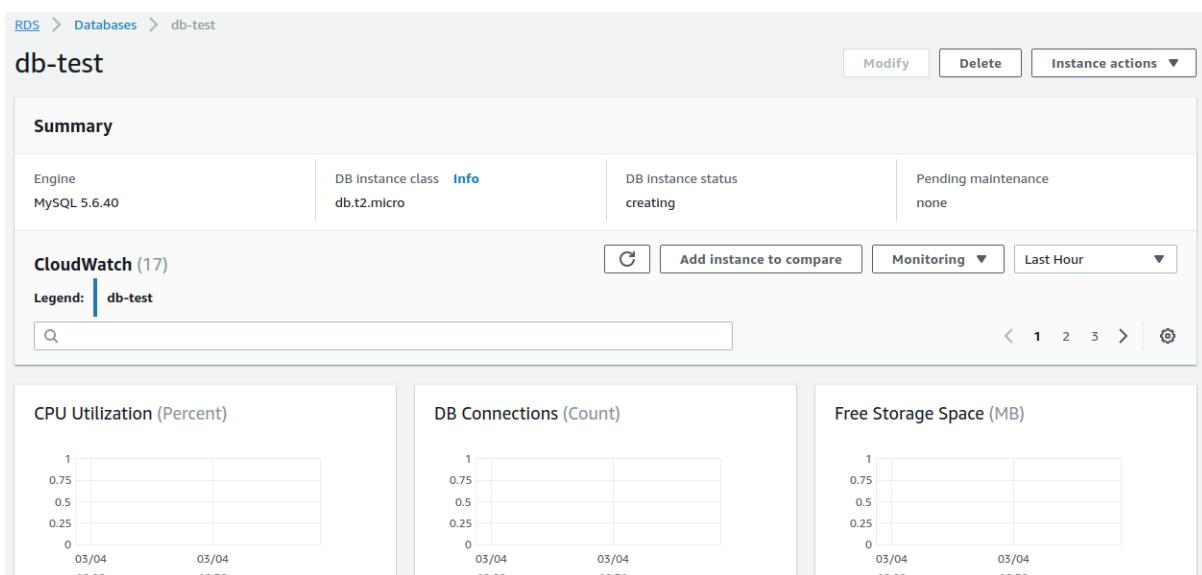
Selanjutnya untuk menyimpan konfiguras yang sudah kita buat, dan mulai menjalankan database, kita klik tombol **Create database**.



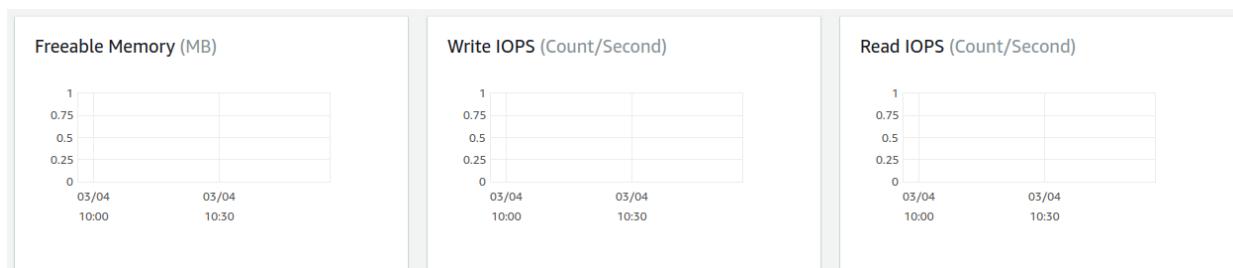
Setelah proses konfigurasi berhasil dilakukan, kita dapat melihat detail database dengan menekan tombol **View DB instance detail**.



Berikut merupakan tampilan dashboard dari database yang sudah kita buat. Kita dapat memonitoring penggunaan CPU yang kita gunakan, DB Connection, dan storage kosong.



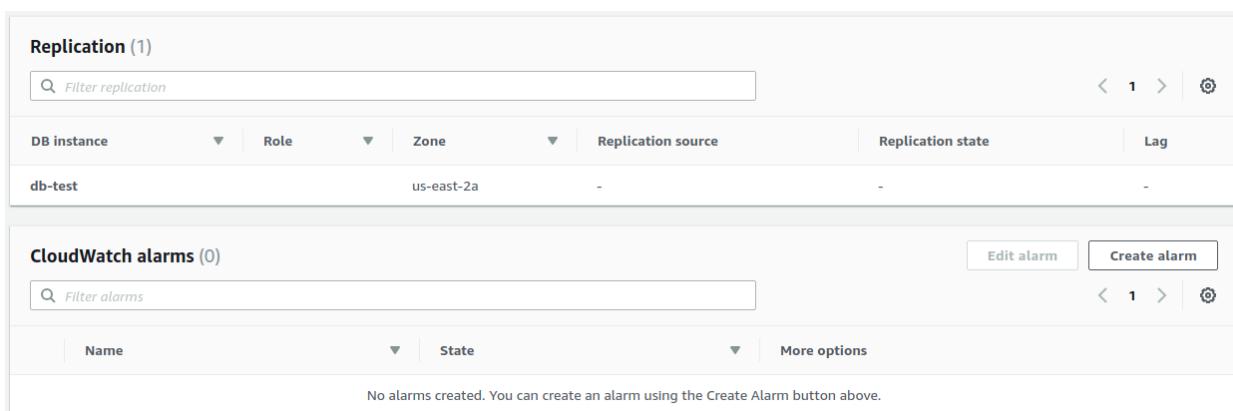
Selain itu kita dapat melihat memori yang tidak di gunakan, melihat proses IOPS write dan IOPS read. IOPS densiri mempengaruhi kecepatan dalam mengakses menulis atau membaca database.



Kita juga dapat melihat detail data mengenai database, seperti versi MySQL, username database, hingga endpoint. Endpoint ini berguna untuk mengakses database MySQL dari SQL Client maupun phpmyadmin yang nanti akan kita setting.

Details			
Configurations	Security and network	Instance and IOPS	Maintenance details
ARN arn:aws:rds:us-east-2:081602682094:db:db-test	Availability zone us-east-2a	Instance Class db.t2.micro	Auto minor version upgrade Yes
Engine MySQL 5.6.40	VPC <a href="#">Default (vpc-d689a6be)</a>	Storage Type General Purpose (SSD)	Maintenance window fri:10:26-fri:10:56 UTC (GMT)
License Model General Public License	Subnet group default	Storage 20 GiB	Pending Modifications Master User Password: ****
DB Name db_test	Subnets <a href="#">subnet-ad65ebd7</a> <a href="#">subnet-eeeb1aa2</a> <a href="#">subnet-5c451534</a>	Availability and durability	Pending maintenance none
Username root	Security groups <a href="#">default (sg-19500973)</a> ( active ) <a href="#">HTTP-SSH (sg-0bc27a463d2aea87d)</a> ( active )	DB instance status creating	Encryption details
Option Group default:mysql-5-6		Multi AZ Yes	Encryption enabled No
		Backup and Restore	

Selain data-data di atas, kita juga dapat melihat **snapshot** (*backup database*) dan juga **CloudWatch** alarm atau notifikasi yang di berikan oleh database MySQL.



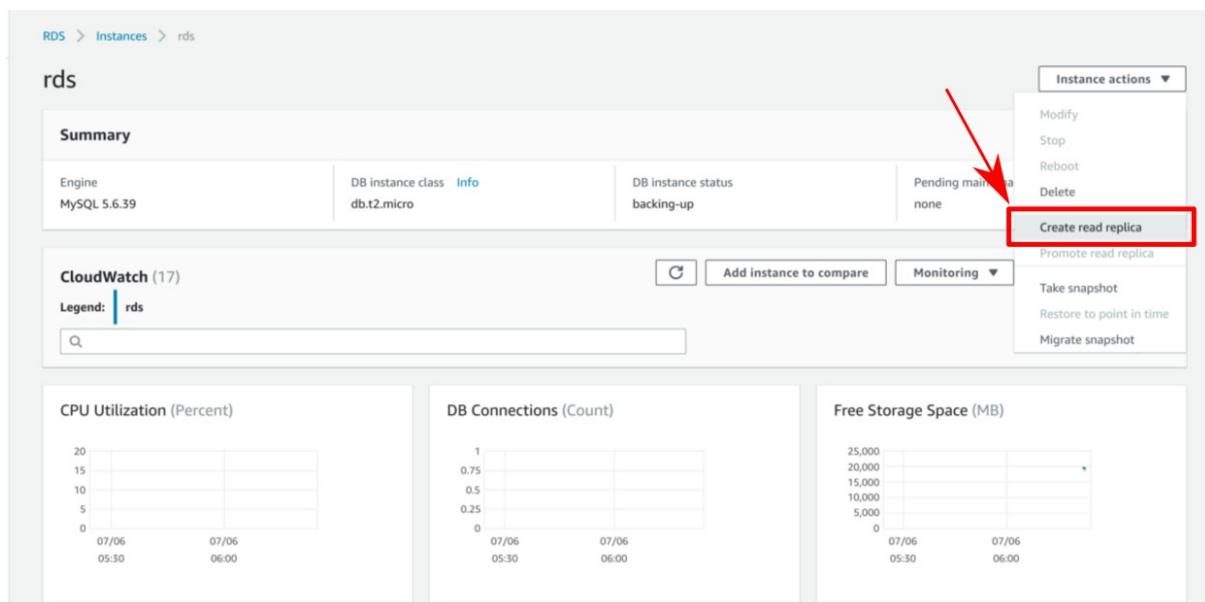
The screenshot shows the AWS RDS 'Replication' section with one entry: db-test in us-east-2a with a replication source of '-' and a replication state of '-'. Below it is the 'CloudWatch alarms' section, which is currently empty with a message: 'No alarms created. You can create an alarm using the Create Alarm button above.'

Dengan begini, kita sudah memiliki sebuah database yang siap kita isi dengan data-data yang kita miliki, kita bisa mengkombinasikan database ini dengan aplikasi yang kita miliki didalam Server AWS tersebut.

### 6.10.2. Replikasi Amazon RDS

Apabila pada bagian pembuatan database Amazon RDS tadi kita tidak mengaktifkan **replikasi**, maka kita bisa melakukan replikasi secara manual yang akan kita bahas dibagian ini. Tujuan dari replikasi ini adalah untuk menduplikat database yang sudah kita buat dengan menggunakan sistem mirroring sehingga isi dari duplikasi database yang baru akan sama dengan database pertama yang sudah kita buat sebelumnya.

Pertama kita harus memilih **Database RDS** yang akan kita replikasi. Klik pada tombol **Instance Action**, kemudian klik **Create Read Replica** seperti berikut.



Kemudian pada bagian konfigurasi **Network & Security**. Pilih **Destination region** (sebaiknya pilih region yang berbeda) kemudian atur **Publicly Accessible**, jika kita ingin membuka akses untuk jaringan public maka pilih **yes**. Jika hanya untuk jaringan private maka pilih **no**.

**Create read replica DB instance**

You are creating a replica DB instance from a source DB instance. This new DB instance will have the source DB instance's DB security groups and DB parameter groups.

**Network & Security**

Destination region  
The region in which the replica will be launched  
Asia Pacific (Tokyo)

Destination DB subnet group  
None

Availability zone  
The EC2 Availability Zone that the database instance will be created in.  
No preference

Publicly accessible  
 Yes  
 EC2 instances and devices outside of the VPC hosting the DB instance will connect to the DB instances. You must also select one or more VPC security groups that specify which EC2 instances and devices can connect to the DB instance.  
 No  
 DB instance will not have a public IP address assigned. No EC2 instance or devices outside of the VPC will be able to connect.

**\*\*Note :** Pilihan *No* pada *Publicly Accessible* digunakan untuk meningkatkan security dan keamanan pada database kita.

Kemudian kita masuk ke **Instance specifications**, sesuaikan **DB Instance Class** sama seperti dengan type EC2. Secara default typenya adalah db.t2.micro (1 vCPU, 1 GiB RAM).

**Instance specifications**

DB instance class  
Contains the compute and memory capacity of the DB instance.  
db.t2.micro — 1 vCPU, 1 GiB RAM

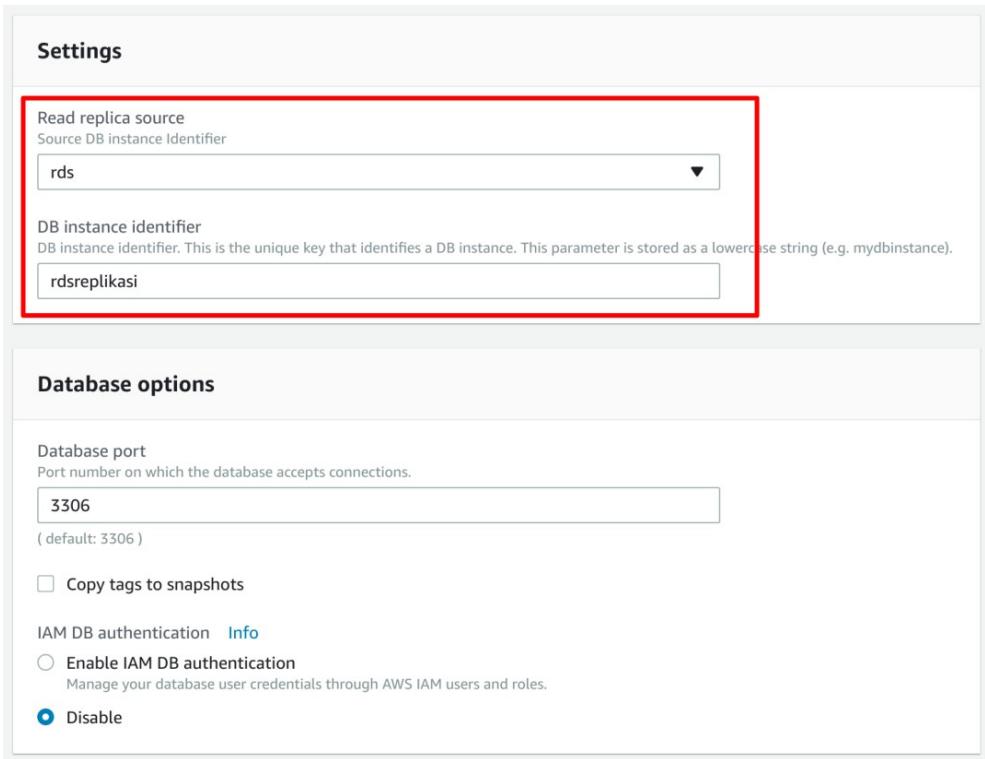
Multi-AZ deployment  
Specifies if the DB instance should have a standby deployed in another availability zone.  
 Yes  
 No

Storage type [Info](#)  
General Purpose (SSD)

Provisioning less than 100 GiB of General Purpose (SSD) storage for high throughput workloads could result in higher latencies upon exhaustion of the initial General Purpose (SSD) IO credit balance. [Click here](#) for more details.

Kemudian pada menu **Settings** kita akan memilih **source replika database** yang ingin kita buat, kemudian kita buat nama replikasi (**DB instance**

**identifier)** dengan nama **rdsreplikasi**. Pada Database Options masukan port dengan **3306**, dan pilih **disable** untuk IAM DB authentification.



**Settings**

Read replica source  
Source DB instance Identifier

DB instance identifier  
DB instance identifier. This is the unique key that identifies a DB instance. This parameter is stored as a lowercase string (e.g. mydbinstance).

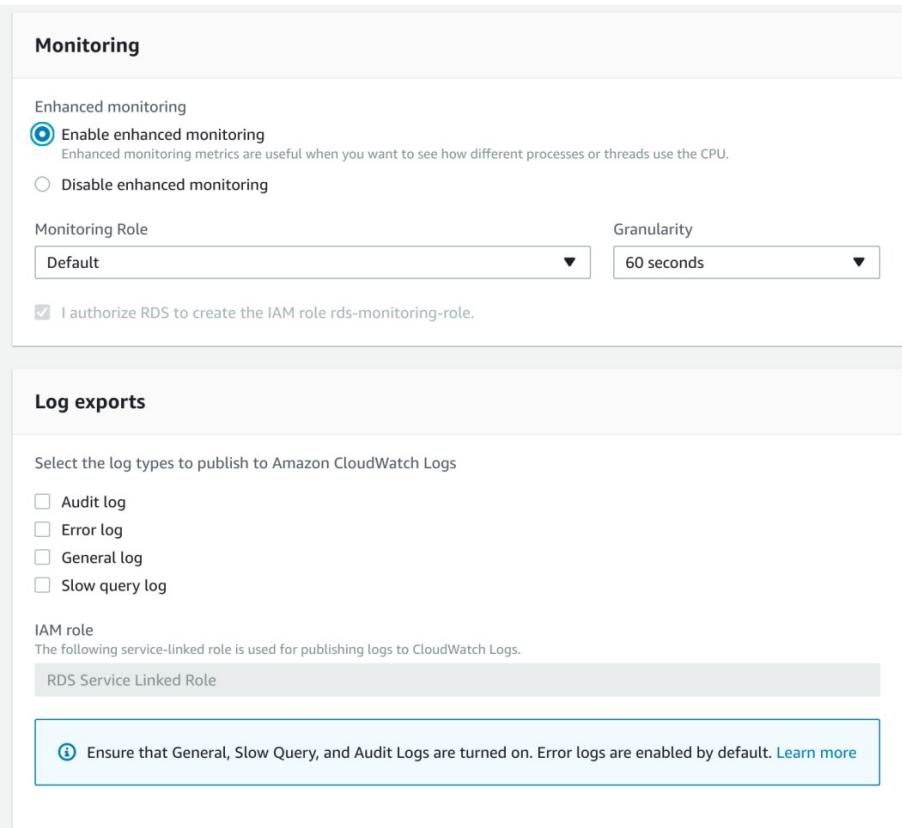
**Database options**

Database port  
Port number on which the database accepts connections.  
  
( default: 3306 )

Copy tags to snapshots

IAM DB authentication [Info](#)  
 Enable IAM DB authentication  
Manage your database user credentials through AWS IAM users and roles.  
 Disable

Kemudian nyalakan pengaturan untuk monitorin dengan klik **enable enhanced monitoring** seperti gambar berikut.



**Monitoring**

Enhanced monitoring  
 Enable enhanced monitoring  
Enhanced monitoring metrics are useful when you want to see how different processes or threads use the CPU.  
 Disable enhanced monitoring

Monitoring Role

Granularity

I authorize RDS to create the IAM role rds-monitoring-role.

**Log exports**

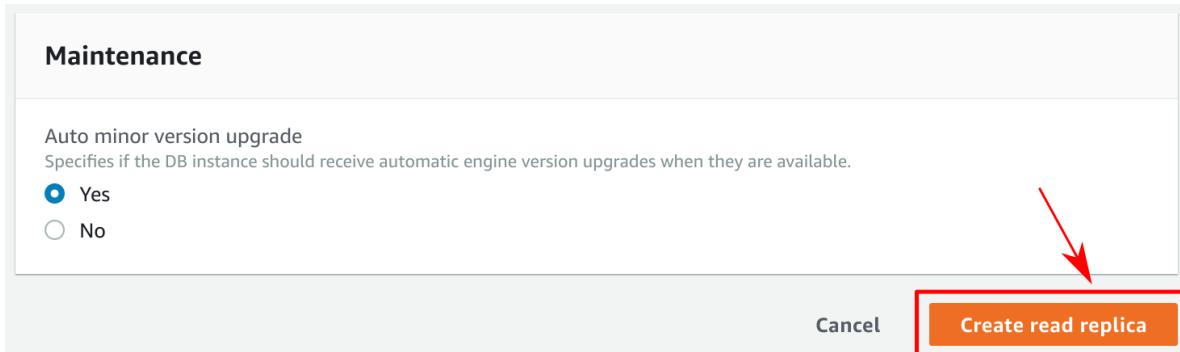
Select the log types to publish to Amazon CloudWatch Logs

Audit log  
 Error log  
 General log  
 Slow query log

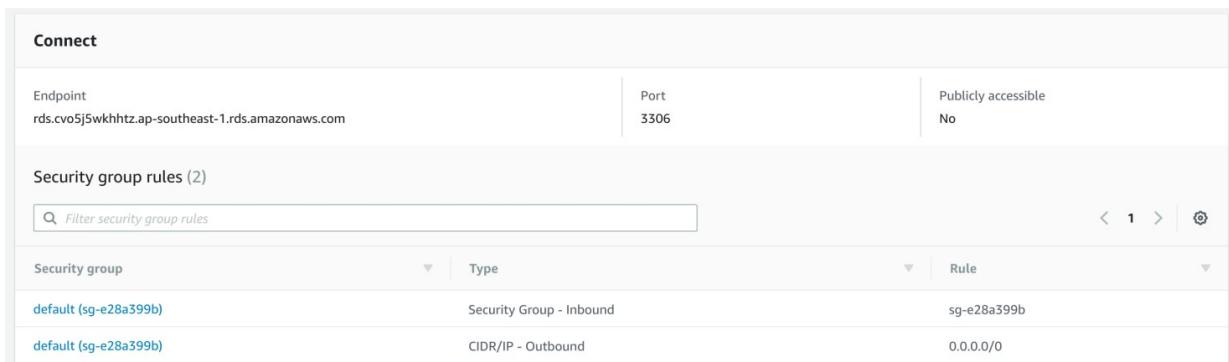
IAM role  
The following service-linked role is used for publishing logs to CloudWatch Logs.

**Ensure that General, Slow Query, and Audit Logs are turned on. Error logs are enabled by default. [Learn more](#)**

Selanjutnya di bawah pengaturan monitoring akan muncul **Maintenance** dan pilih **Yes** untuk auto maintenance. Setelah semuanya selesai maka kita dapat meng klik **Create read replica**. Tunggu proses pembuatan replikasi sampai dengan selesai.



Setelah selesai kita akan masuk kedalam halaman **dashboard**, kita dapat melihat informasi untuk koneksi ke amazon RDS , melalui aplikasi yang kita miliki.



Kita juga dapat melihat status Replication pada halaman dashboard

Replication (2)						
Filter replication < 1 > ⚙️						
DB instance	Role	Zone	Replication source	Replication state	Lag	
rds	master	ap-southeast-1c	-	-	-	
rdsreplikasi (Tokyo)	replica	ap-northeast-1	rds	-	-	

Dengan begini kita sudah menyelesaikan untuk membuat sebuah replikasi database dari amazon RDS. Apabila kita lupa membuat replikasi, kita bisa membuatnya secara manual seperti ini. Dan database kita akan aman karena datanya sudah di replikasi.

### 6.10.3. Exercise

1. Buat sebuah database menggunakan Amazon RDS !
2. Setup replikasi untuk database yang sudah kalian buat !

## 6.11. Setup Web Aplikasi dengan RDS

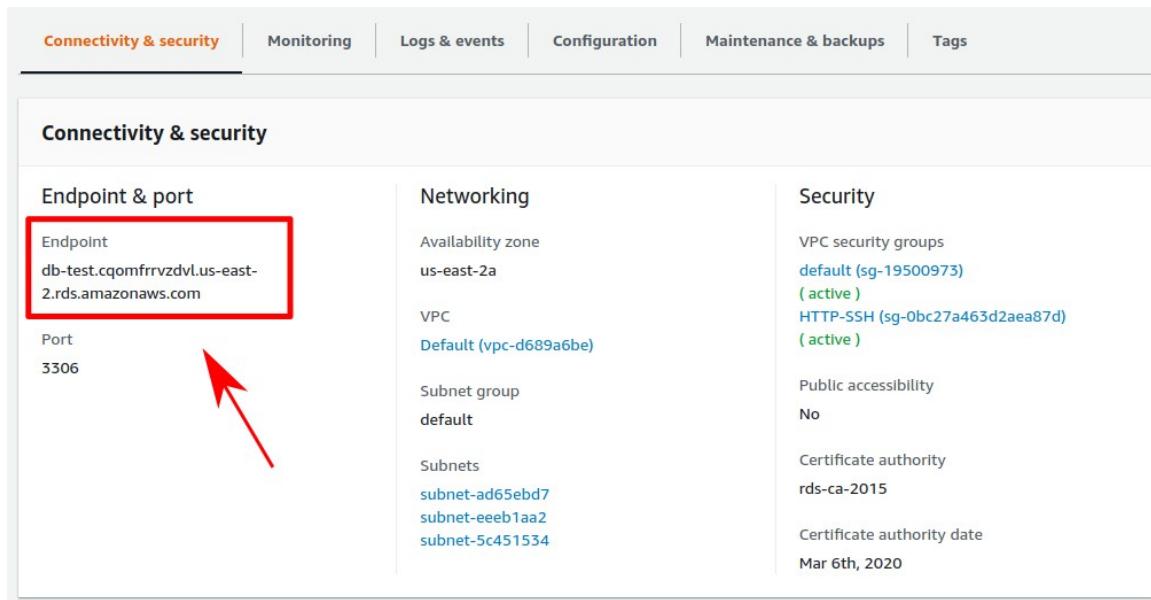
### 6.11.1. Migration Database

Agar aplikasi yang kita miliki dapat menggunakan database dari RDS, pertama kita harus melakukan migrasi database terlebih dahulu agar semua data yang kita miliki bisa langsung digunakan pada database RDS tersebut.

Disini saya asumsikan kita sudah memiliki sebuah server yang sudah ada aplikasi didalamnya, misalkan aplikasi pesbuk yang bisa kita gunakan. Apabila kita sudah memiliki banyak data pada databasenya, kita bisa melakukan export terlebih dahulu dengan menggunakan perintah berikut.

```
sudo mysql -u devopscilsy -p dbsosmed > dbsosmed.sql
```

Tunggu sampai proses export selesai dilakukan, selanjutnya kita masuk kembali ke Amazon RDS pada database yang kita buat. Cek alamat endpoint yang ada pada database tersebut.



The screenshot shows the 'Connectivity & security' tab selected in the AWS RDS console. The 'Endpoint & port' section displays the endpoint address: db-test.cqomfrvzdv.us-east-2.rds.amazonaws.com and port 3306. A red box highlights this information, and a red arrow points from the bottom of the page towards it. The 'Networking' and 'Security' sections provide additional details about the VPC configuration, including availability zones, subnets, and security groups.

Endpoint & port	Networking	Security
Endpoint db-test.cqomfrvzdv.us-east-2.rds.amazonaws.com	Availability zone us-east-2a	VPC security groups <a href="#">default (sg-19500973) ( active )</a> <a href="#">HTTP-SSH (sg-0bc27a463d2aea87d) ( active )</a>
Port 3306	VPC <a href="#">Default (vpc-d689a6be)</a>	Public accessibility No
	Subnet group default	Certificate authority <a href="#">rds-ca-2015</a>
	Subnets <a href="#">subnet-ad65ebd7</a> <a href="#">subnet-e6eb1aa2</a> <a href="#">subnet-5c451534</a>	Certificate authority date Mar 6th, 2020



Alamat ini adalah alamat yang akan kita gunakan untuk mengakses database Amazon RDS tersebut, setelah kita mengetahui alamatnya sekarang kita coba melakukan akses pada database yang sudah kita buat di Amazon RDS dengan menggunakan perintah berikut.

```
sudo mysql -h db-test.cqomfrrvzdvl.us-east-2.rds.amazonaws.com -u root -p
```

**\*\*Note :** yang diberi warna merah adalah endpoint dari database Amazon RDS.

Masukan password sesuai dengan yang sudah kita buat tadi, kita juga bisa cek database yang ada didalamnya menggunakan perintah **show databases;** seperti biasa

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
mysql> show databases;  
+-----+  
| Database |  
+-----+  
| information_schema |  
| db_test |  
| innodb |  
| mysql |  
| performance_schema |  
| sys |  
+-----+  
6 rows in set (0.00 sec)
```

Setelah itu coba keluar dari console database tersebut dengan perintah \q lalu kita coba import database yang kita miliki dengan menggunakan perintah berikut.

```
sudo mysql -h db-test.cqomfrrvzdvl.us-east-2.rds.amazonaws.com -u root -p  
db_test < dbsosmed.sql
```

**\*\* Note :** yang berwarna merah adalah database kita di amazon RDS

Dengan ini proses import database dari local ke database Amazon RDS sudah selesai, sekarang kita masuk kedalam tahap konfigurasi dan perubahan pada konten web aplikasi yang kita miliki.

## 6.11.2. Konfigurasi Web Aplikasi

Setelah kita melakukan import data dari local ke server database RDS, sekarang kita akan melakukan konfigurasi pada webapps yang kita miliki. Kita



akan coba mengubah alamat tujuan, user dan password dari koneksi database webapps ke database RDS.

Untuk itu kita perlu melakukan konfigurasi pada file **config.php** pada web aplikasi **pesbuk**. Perlu diperhatikan bahwa "**file yang kita konfig mungkin tidak akan sama dengan dilapangan, karena file konfig ini nanti menyesuaikan dengan file yang dibuat oleh programmer**".

Coba masuk kedalam direktory /var/www/html lalu masukan perintah

```
sudo nano config.php
```

Didalam konfigurasi tersebut akan terdapat script seperti dibawah ini.

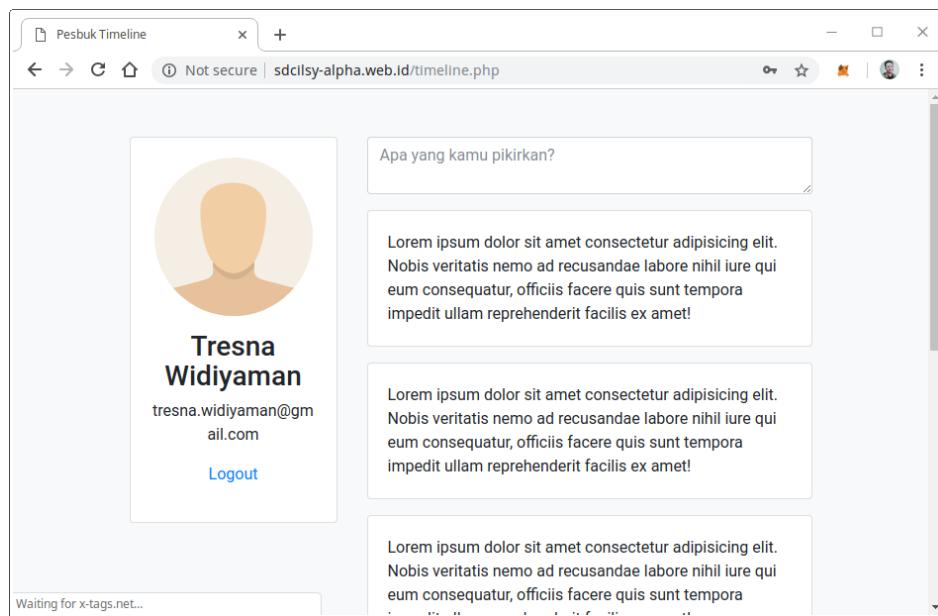
```
$db_host = "localhost";  
$db_user = "devopscilisy";  
$db_pass = "1234567890";  
$db_name = "dbsosmed";
```

Coba sesuaikan konfigurasi dengan database RDS yang sudah dibuat tadi seperti dibawah ini.

```
$db_host = "db-test.cqomfrrvzdvl.us-east-2.rds.amazonaws.com";  
$db_user = "root";  
$db_pass = "1234567890";  
$db_name = "db_test";
```

Setelah itu simpan konfigurasi pada file tersebut dan coba akses kembali web aplikasi yang kalian miliki. Buat sebuah user baru dan login dengan user baru tersebut.

Apabila berhasil maka proses migrasi database bisa dikatakan berhasil dan selesai.



*Hasil dari import database*

### 6.11.3. Exercise

1. Ubah database pada web aplikasi yang kalian miliki menjadi database dari amazon RDS.

## 6.12. Summary

1. Pada bab ini kita sudah membahas mengenai konsep CDN dan Amazon Cloudfront, dimana nantinya konten yang kita miliki akan disebar di berbagai edge location sehingga bisa diakses dengan lebih mudah.
2. Untuk menggunakan cloudfont kita bisa menggunakan S3 Bucket, Load Balancer, dan Elastic Beanstalk untuk sumber yang akan di distribusikan.
3. Kita sudah membahas mengenai Amazon Route53 yang memiliki fungsi sebagai domain manager yang mengarahkan domain dengan instance di AWS.
4. Untuk bisa mengakses domain, kita harus mendaftarkan domain di Route53, setelah itu menerapkan NS yang ada di Route53 ke Name Server domain kita yang kita beli di Provider domain.

5. Kita cukup memasukan alamat ip public server instace agar bisa diakses dengan domain di route53, selain itu kita bisa masukan dns dari amazon s3, load balancer, dan elastic beanstalk agar bisa diakses melalui domain di route53.
6. Kita sudah membahas mengenai Amazon RDS dan keunggulan yang dimiliki oleh layanan tersebut, harusnya kita sudah paham cara penggunaan layanan tersebut.
7. Kita bisa membuat database dengan mudah di Amazon RDS, dengan menyesuaikan server seperti halnya membuat sebuah instance baru. Kita dapat melakukan replika atau mirroring pada database yang sudah kita buat.
8. Database yang sudah kita buat bisa langsung kita terapkan pada web aplikasi yang kita miliki, sehingga database kita akhirnya terpisah dengan server web aplikasi yang selama ini bersatu dengan web database.