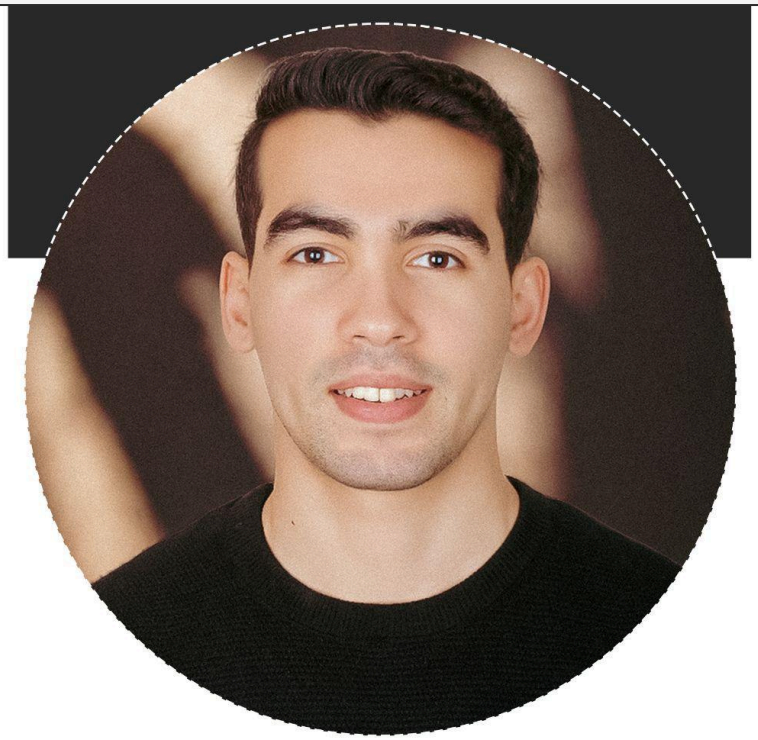


WHOAMI

HI, I'M MOHAMED EID

Passionate DevOps enthusiast with expertise in Linux system administration and automation. Proficient in Bash, Git, Ansible, Docker, AWS, and Terraform, I focus on streamlining processes and enhancing system reliability in cloud environments. Always eager to learn and share knowledge with others.

/mohamedeid404/

I'm Here

Table Of Content

I'm Here

IT & Cloud Fundamentals

💡 All From Eisa_Abo_Sherif & Stephane_Maarek

➥IT

- **IT Fundamental:**

- اي اجهزة احنا بنستخدمها اسمها Computing Device

- هتلاقي فى الاجهزة دي حاجات مشتركة كتير وهيا:

1. CPU (Central Processing Unit):

- وحدة المعالجة المركزية .. دا المخ اللى بي process كل حاجة
- بي fetch , execute , decode process كل ال instructions اللى جياله من ال programs
- الاداء بتاعه بيتقاس بعدد ال cores وال clock speed علاقة طردية
- جوا ال CPU حاجة اسمها Registers ودي عباره عن small fast storage بيستخدمها بديلة عن ال RAM لانها اسرع منها .. بيستخدمها ف حالة انو عاوز ي quick access على حاجة.
- فى ال CPU الجديدة هتلاقي small amount of high-speed memory تستخدم فال cashing علشان موضوع ال processing مياخدش وقت اطول

- فى ال CPU الجديدة بيكون فيها Multi-core Processing ياعني تقدر ت perform multiple operations فى نفس الوقت
- سرعته بتتقاس بال GHz

---

2. GPU (Graphics Processing Unit):

- عباره عن processor مخصص انو ي process ال graphic renders واي processing task خاصة بالجرافيك زي ال ,videos , animation , images 3D environment

---

3. RAM (Random Access Memory):

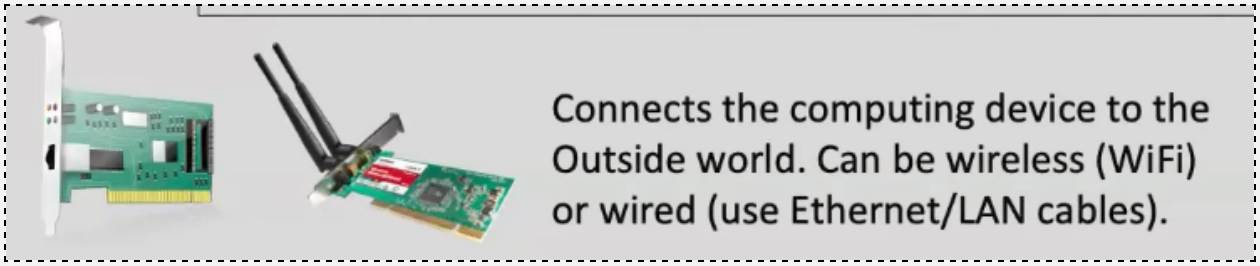
- اسرع من ال HDD , SSD
- عبارة عن temporary storage لل data اللى بيستخدمها ال CPU وي proccess it .. بتخلي ال CPU ي Access الداتا وي process it بطريقة سريعه من خلالها زي انك ت run app او ت open file
- كل ما تكون ال RAM اكبر كل ما تكون قادر انك تعمل Multi tasking ياعني تفتح برامج وتعمل حاجات كتير فى نفس الوقت بدون ما الجهاز يهنج
- بتستخدم بردو فى ال buffer , cashing
- لو ال RAM مش كافية ممكن انك تاخد من ال HDD او SDD جزأ portion ك virtual memory
- سرعتها بتتقاس بالGiga byte

---

4. HDD & SSD

- بيخذنو الداتا العادية وال SSD اسرع من ال SSD
- سرعتها بتتقاس بـ Gigabyte او tera

5. Network Cards / Interface



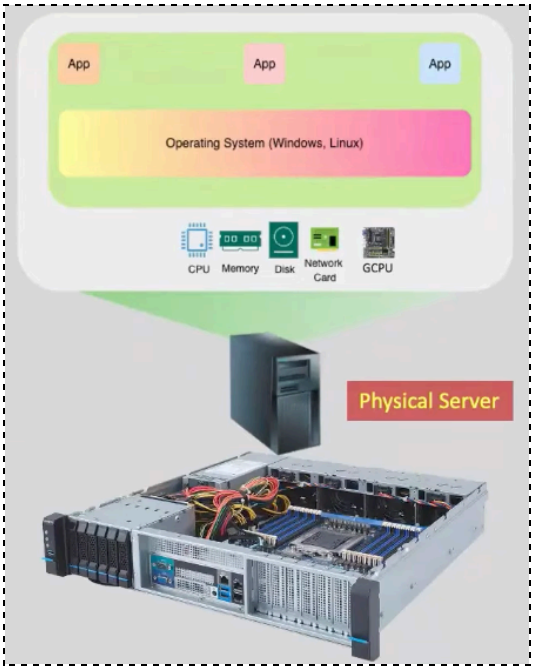
- عن طريقها بنوصل لل internet سواء wireless زي اللاب كدا او لو كمبيوتر ممكن بكابل او ممكن ب وصلة تخليه يلقط وايفاي.
- سرعتها بتتقاس بال**Gbps** or **Mbps** .. ميجا او جيجا بايت / ثانية

💡 اتكلمنا عن كل اللى فات دة لان دول نفس مكونات ال Servers اللى بتبقى موجودة فى ال Data Centers

---

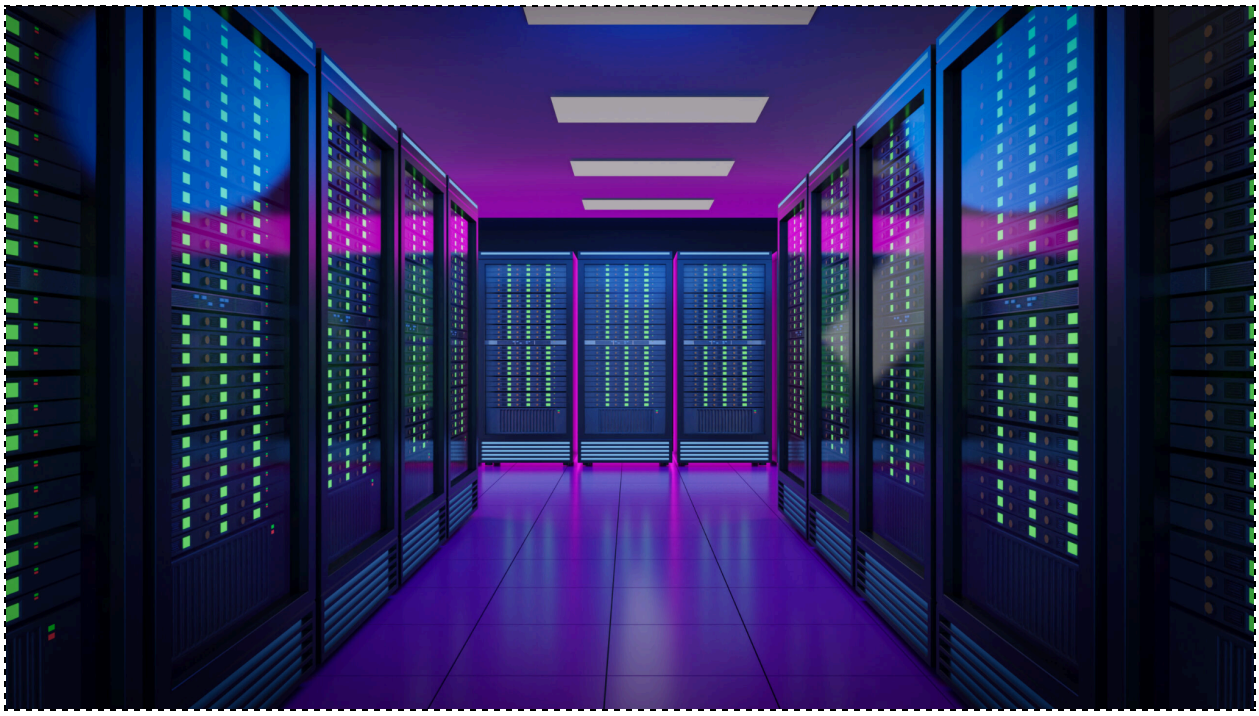
➥Cloud

- **Physical Server (Powerful Computers):**



- بيكون عبارة عن نفس المكونات بتاعة الكمبيوتر العادي ولكن اكبر بمراحل ومحطوطين في Racks زي اللى فى الصورة دي وال Racks دي فى اماكن اسمها Data Centers.
- بين ال server hardware وال Applications بيكون موجود ال Operating System اللى بيوصل ال requests لل hardware علشان يتعملها process ويرجع تاني ال responds لل Applications.

• Data Centers:

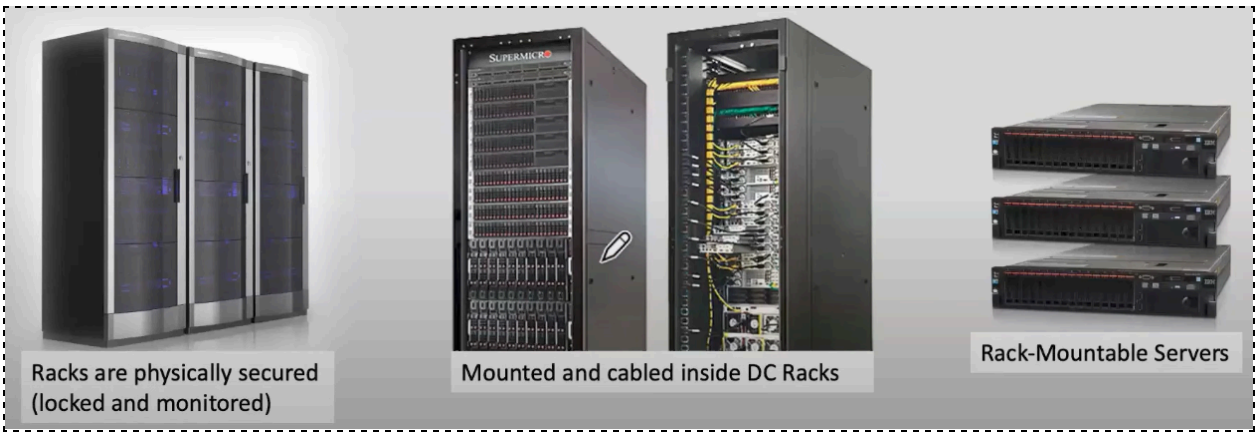


- دي الاماكن المجهزة اللى بيكون فيها ال servers بتاعتنا .. حماية وكاميرات وانظمة مراقبة وتكيفات وكل حاجة تحافظ على ان ال servers دي تفضل شغالة ومحدش يخترقها.
- هتلاقي جوا يمكن شوارع ويمكن levels فوق بعض طوابق .. ف بيبقي عندك كميات رهيبة .. بتتقسم فى كابينات كدا اللى هيا بالطول دي اسمها racks بيكون جواها ال servers وعلي كل حاجة stickers بحيث نعرف ال server دة بتاع مين وتابع لاي

تيم ولا اي business unit وممكن يكون مكتوب عليه كمان ال IP address ..
الارضيه حتي دي بتبقي تحتها كابلات.

- كل ال servers دي بتبقي معزولة عن بعضها .. يقدرو يتكلمو من خلال ال network لو انت سامحلهم او محتاج دة.
- ممكن تاخد server او rack او room كاملة .. كل حاجة متاحة على حسب احتياجاتك

- **المكونات بالتفصيل :**



- **Rack-Mountable Servers** : دي ال Physical servers وال Rack-Mountable دة نوع معين .. ال servers دي بناخدها نحطها فى ال Racks ال Racks دي من الخلف بيكون فيها ال network interface وال storage وال power وكدا .. ال servers دي بقي اللى بيكون عليها ممكن تطبيقات عادية زي اوبر او مثلا data bases .. ال Racks دي كل واحدة فيهم بتكون secured ومراقبة والمبني كمان متراقب وكله تحت الكنترول 😁

- **أنواع الـ Data Centers :**

- **On-Premises** : بيكون مملوك للهيئة او الشركة او الحكومة وهو اللى بيتحمل كل التكاليف من الصفر .. , OS , Virtualization , hardware , storage , network , building Data , App
- **On-Cloud** : عالكلاود زي AWS , Azure ومن مميزاته ..

• Cloud Types

1. Public & Private Cloud

- **Public** : زي AWS , Azure متاح للعامة
- **Private** : ال Data centers باللى فيها بتاعتنا ومحدش بيوصلها غيرنا ودة مهم فى ال critical use case زي البنوك او كدا

2. Hybrid Cloud

- لو هتستخدم ال Private وال Public مع بعض ف دة اسمو **Hybrid Cloud** سواء ال connection بينهم by the internet او private connectivity .. هيئة حكومية مثلا وعندها بيانات عملاء ف دي هتخليها private وبقيت البيانات دي عادي تكون عال public ف بتخليها عال public ويبقي ده **Hybrid**

💡 التحدي بيكون فى اننا نـ Authenticate ال users مابين ال private , Public Cloud .. يعني لو فيه user بيـ Access حاجة عال Public ومحتاج يـ Access حاجة عال Private ف ازاي نخليه يقدر يعمل دة.

3. Multi Cloud

- اني اقسم ال work load بتاعي علي كذا cloud provider
- لو عندنا تطبيقات خاصة ب ال E-Commerce ف افضل استخدم AWS
- لو عندنا تطبيقات خاصة ب Microsoft فـ افضل Azure
- لو بنك ف افضل IBM
- لو AI/Machine Learning ف Google افضل
- لو عندي جزئية خاصة بال Virtualization ف هتلاقي جزأ خاص بال VMware على كل ال Cloud Providers

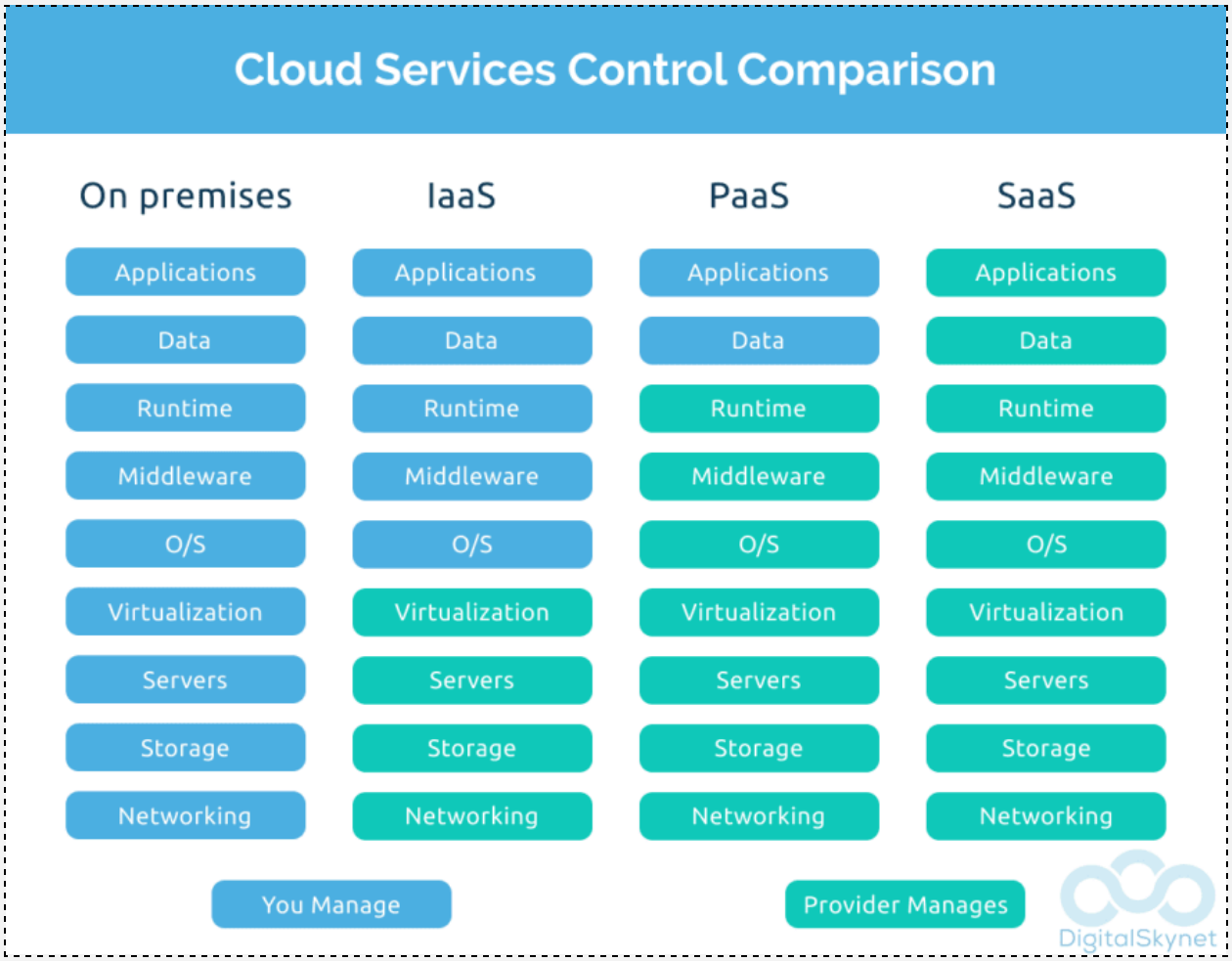
💡 ممكن تستخدم Private & Public & Multi Cloud مع بعض .. شوية data على AWS وشوية على Azure وشوية على ال Private data center .. كدا دة اسمو Hybrid & Multi Cloud

---

- **Shift from CAPEX to OPEX model :**

- بوجود الـ Cloud اتحولنا من CAPEX إلي OPEX .. ياعني بدل ماانا هدفع واكلف واعمل Data Center واصرف عليه من الصفر .. لأ انا بس هدفع تكاليف التشغيل فقط ودة اللى وفرتو ال cloud providers زي AWS , Azure واي حاجة تانية بتكون من مسؤولية ال Cloud Provider
- **capital expenditure (النفقات الرأسماليه : CAPEX)** : ودة راس المال اللى بحتاجة علشان ابدا بيه المشروع او ابني بيه ال data center
- **Operational expenditure (النفقات التشغيليه : OPEX)** :ودي هيا المصاريف التشغيلية بعد ما بنيت ال data centers
- **Consumption Based Model :** دة ياعني هنشتغل بمبدأ Pay As You Go ادفع على قد استخدامك.

- **Cloud Services:**



- **On-Premises :**

- انت اللى شايل كل حاجة من البداية للنهاية.

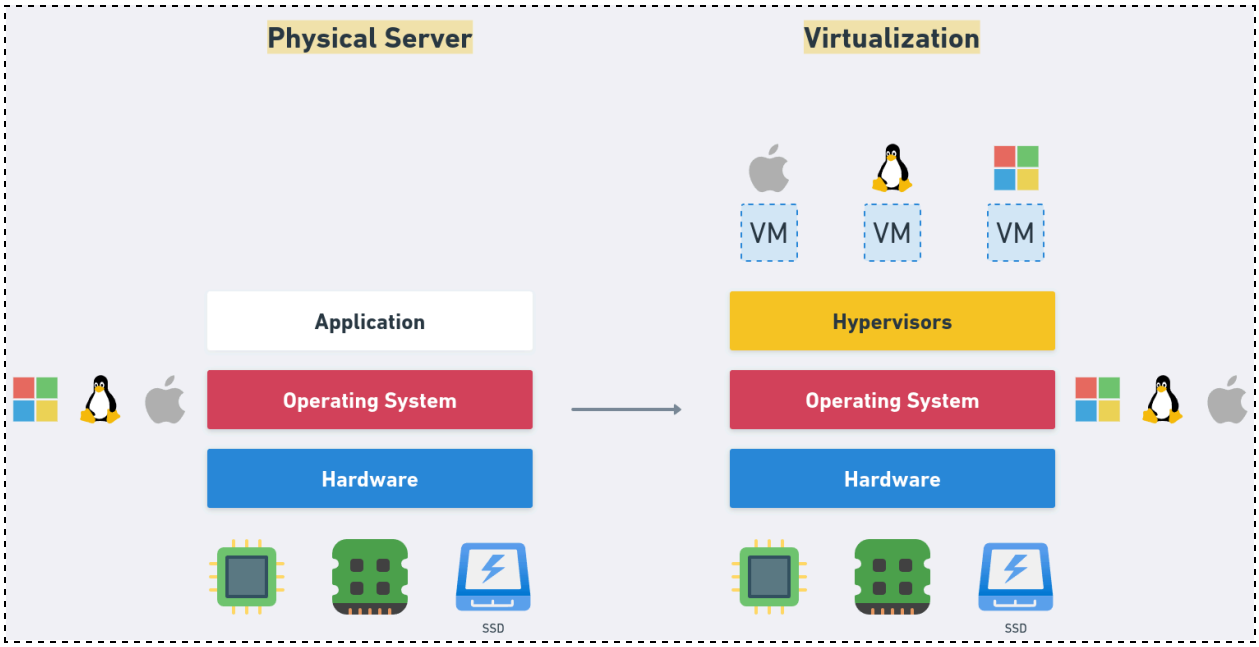
- **Infrastructure as a service (IaaS)**

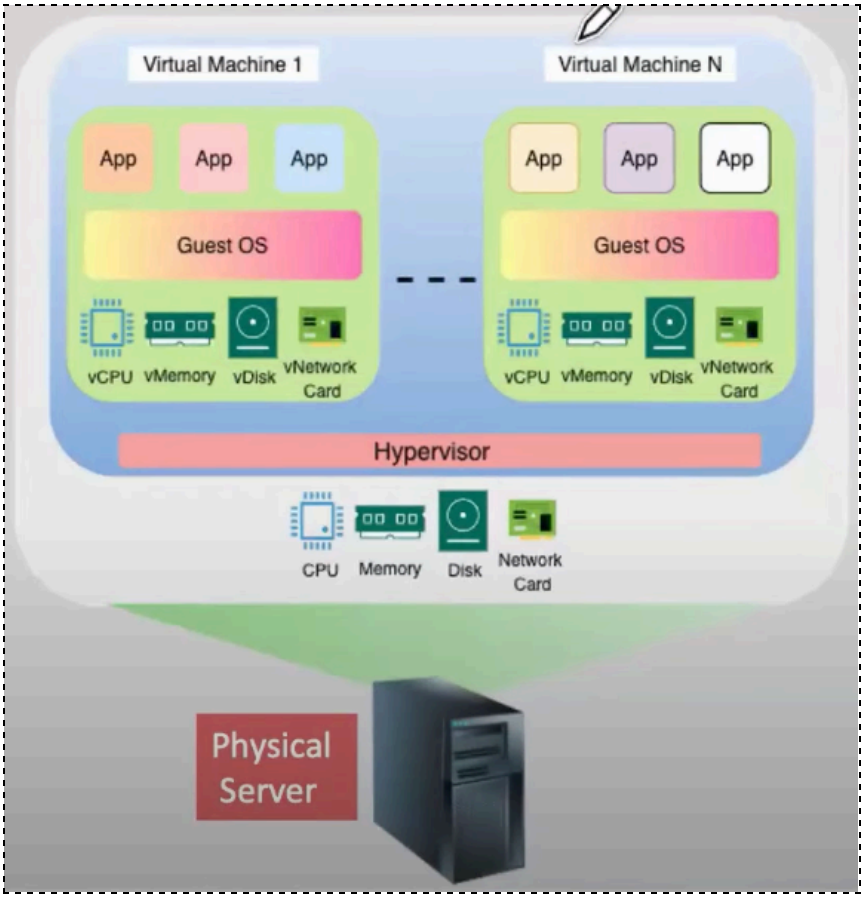
- بتاخد فقط ال Virtual Machines زي ال EC2 instance فى AWS .. والباقي من عند ال Providers
- **Run Time** : هتشتغل python ولا java ولا بتستخدم انهي interpreter
- **Middleware** : ال software اللى بين ال Applications وال OS زي Message brokers like RabbitMQ, Apache Kafka.
- AWS هي الرائدة فى الجزء دة

- **Platform as a service (PaaS)**
  - ف ال model دة ال customer بيأجر platform جاهزة بي run عليها ال app بتاعه.
- **Software as a service (SaaS)**
  - ف ال model دة ال customer بيأجر software جاهز managed by , hosted ال cloud provider

---

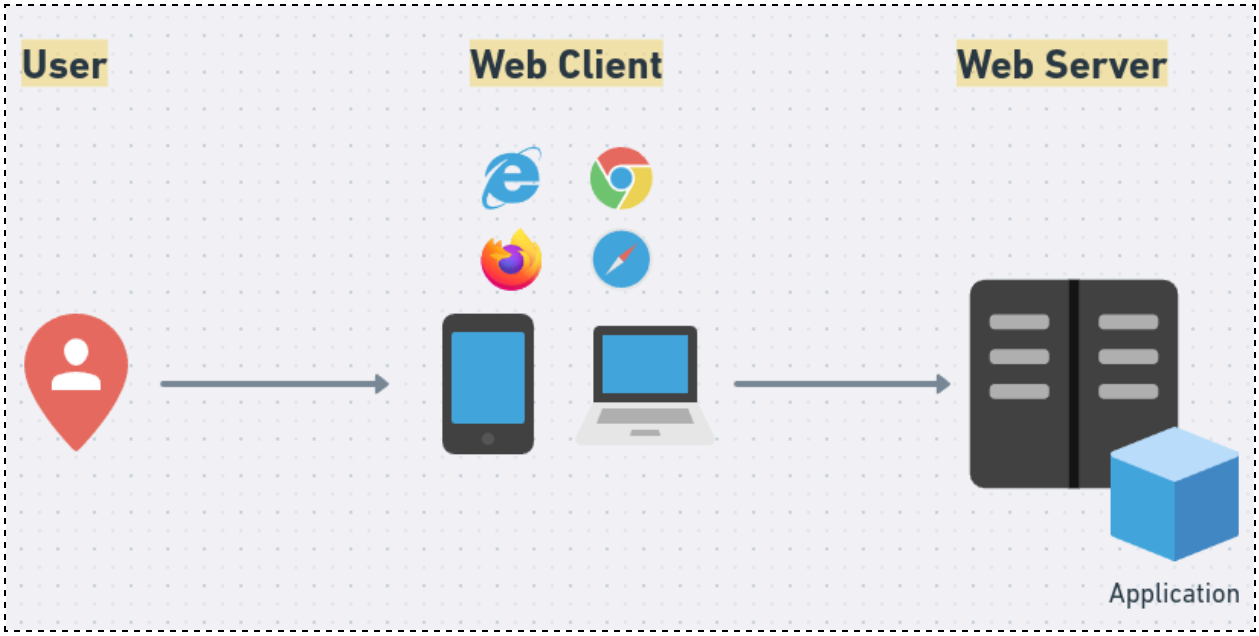
- **Virtualizations:**





- اشتريت server .. كمبيوتر قوي جدا ومستشغلتش امكانياته كلها يبقي كدا غلط
- ال **Hypervisor** عبارة عن Software .. من خلال ال Hypervisor اللى هو المحاكي او ال Emulator بتتشارك ال Virtual Machines .. VM فى ال Hardware اللى تحت ومن امثلته VMware , Virtual box , HyperV
- وانت بتكريت ال VM اللى فوق دي بتحددلها ال Storage وال Ram وكل حاجة من ال hardware اللى هتحتاجها لانك بتكون عارف ال VM دي هينزل عليها تطبيقات اية بالظبط وعارف هيا عاوزة متطلبات اية
- كل ال VM دي بتبقي معزولة عن بعضها .. يقدرو يتكلمو من خلال ال network لو انت سامحلهم ب دة.

- **Web Application:**



- **Web Clients** : هي الاجهزة اللى ال user بيخش من خلالها عال web
- **Web Server** : هو ال server اللى بيستضيف ال Application بتاعنا .. ياعني لو ال server دة معانا ال ip بتاعه ودخلنا عليه هيدخلنا علطول عال Application
- اشهر تطبيقات web server ف ويندوز iis وف لينكس apache , nginx
- علشان تبعت ملفاتك لل server البعيد دة بتستخدم اي software من ال FTP زي filezilla

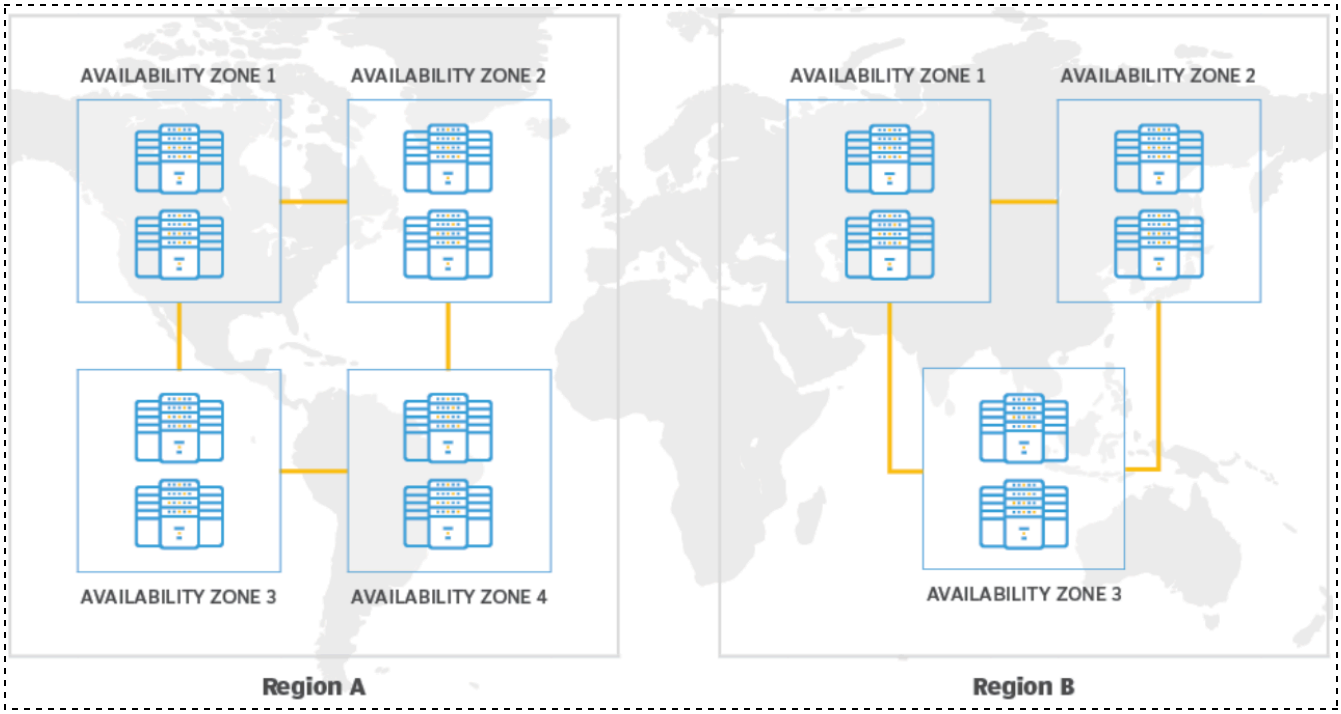
- **After Access URL:**

- الurl بيتحول ل ip عن طريق ال DNS
- الip دة بيروح للسيرفر اللى هو ف الحالة دي ويب سيرفر ومن الip بيعرف انت عاوز انهي صفحة ويب او موقع ياعني ويا بيكون فيه السورس كود بتاعها او بي generate دة ويعرضهالك.

- **DNS & Hosts File:**
  - ملف الهوستس دة بيكون فيه الip بتاع المواقع بمعني انا لو عندي موقع example.com ف بيكون ف ملف الهوستس دة لازم لية ip معين بحيث مثلا لو بنجت عليه هيرد عليا بالip دة او لو كتبتو ك url ف بيتحول عن طريق الdns لنفس الip الموجود ف ملف الهوستس.

CH-03 - Getting started with AWS

➥Region:



- منطقة جغرافية بيكون جواها Availability Zones اللى بيكون جواها cluster of data centers اللى ب deploy عليها ال applications بتاعتي
- بتبقي منتشرة فى العالم وكل region منفصلة عن الثانية وكل شوية بيظهر regions جديدة بيحصل توسع يعني.
- كل region بيبقي ليها كود Sydney مثلا الكود بتاعها ap-southeast-2

- **بختار ال Region دي بناءا على الآتي:**

- **Latency**
  - لو عندي مثلا website و ال end user بتاعي اللى هيستخدم ال website من Europe مينفعش اروح استخدم region ف Asia مثلا هيكون فيه Latency عليه وهو بي Access ال website
  - لو ال end user دة هي Access - latency sensitive applications or services زي ال video gaming or real time videos مثلا مينفعش يكون فيه تاخير عليه
- **Services**
  - مش كل ال regions فيها نفس ال services ممكن تلاقي service موجودة ف region وف region تانية مش موجودة
- **Cost**
  - مش كل ال services ليها نفس التكلفة ف كل ال regions .. تكلفة ال services بتختلف من region للتانية
- **Compliance or Data Sovereignty (سيادة)**
  - ممكن دولة يكون ليها قوانين معينة تشترط بقاء ال data داخل حدود الدولة دي مينفعش تخرج براها.

➥Availability Zones (AZ):

- عباره عن data centers او اكتر ومنفصلين عن بعضهم physically .. ومنفصلين وبعاد عن بعض الى حد ما يكفي ان لو حصل كارثة زي مثلا زلزال او قطع تيار كهربي او كدا ال Availability Zones بتاعتي متتاثرش كلها .. اللى تقع التانية تشيل مكانها وهكذا
- كل AZ ليها ال power,cooling and networking منفصلة.
- كل AZ تحتوي على عدد من ال Data Centers مش معروف
- بيكونو متوصلين ببعض بـ high bandwidth, ultra-low latency networking
- من 3 اقل حاجة ل 6 اقصي حاجة فى كل region

- **ف دة بيضمنلك حاجتين من مميزات ال cloud :**

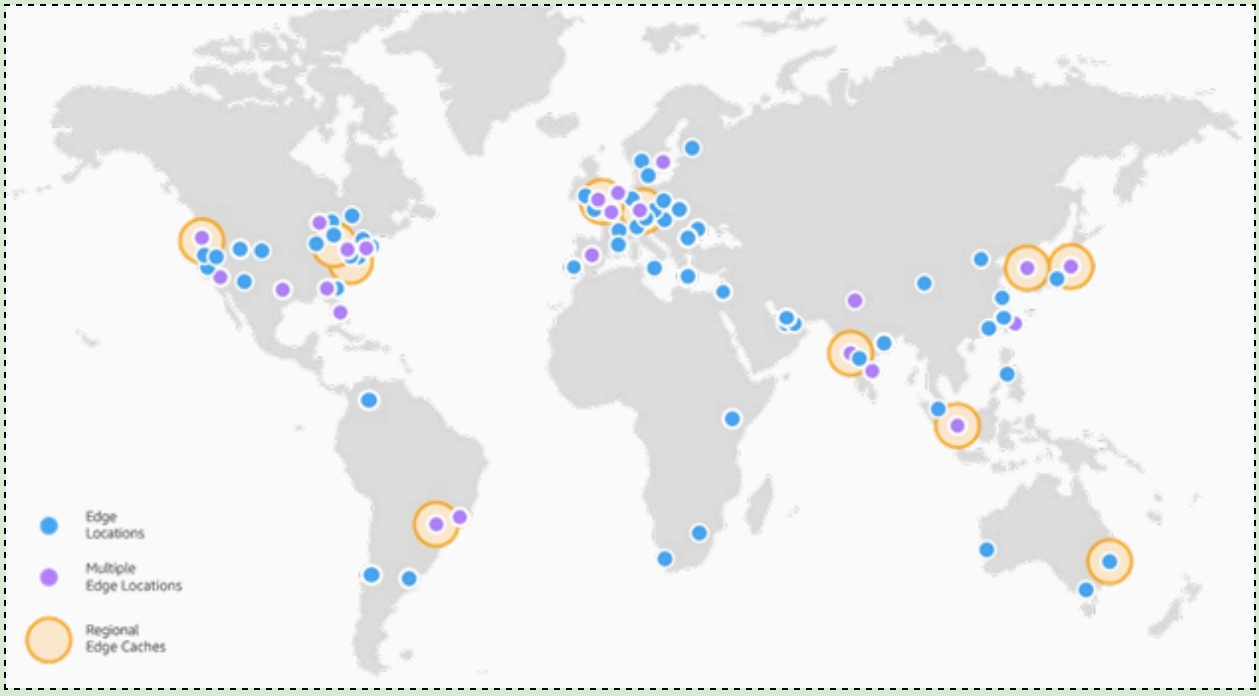
- **Disaster recovery:**

- ان لو حصل failure ف الservices بتاعتي بقدر ا recover بشكل سريع و automatic عن طريق ان بيكون عندي ال services دي فى data center تاني ف AZ تانية

- **Fault tolerance:**

- ودي عباره عن ان لو عندي services ف مكان وredundant زيها ف مكان تاني لو حصل مشكلة ف المكان الاولاني ال services اللى ف المكان التاني بتاخد مكانها من غير ما ال end user يحس

➥AWS Points of Presence (Edge Locations)

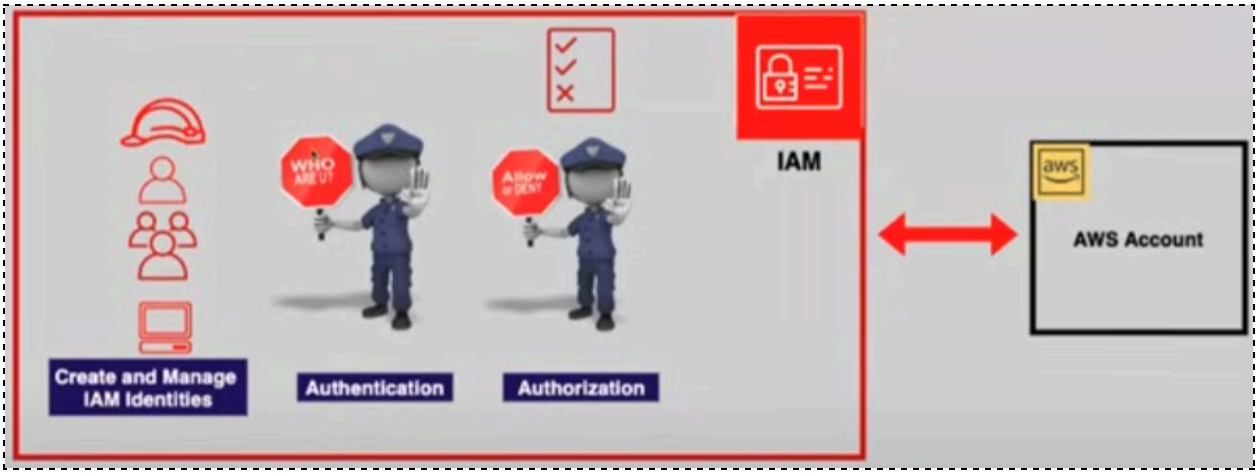


- Amazon has 400+ Points of Presence (400+ Edge Locations & 10+ Regional Caches) in 90+ cities across 40+ countries.
- Content is delivered to end users with lower latency.

CH-04 - IAM & AWS CLI

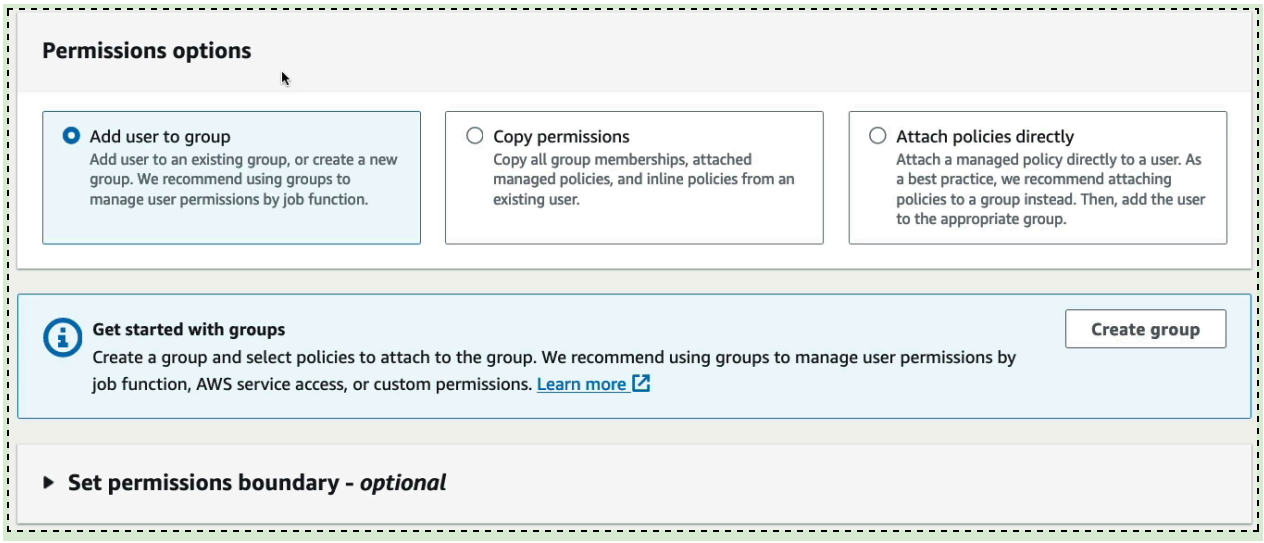
➥IAM Introduction (Identity and Access Management): Users, Groups, Policies

- عباره عن Free Service موجودة فى AWS ونوعها Global Service
- ال Global Services بتبقي services متاحة فى كل ال regions .. وفيه Services تانيه بتكون متاحة فى regions معينه بس عندك مثلا EC2 instance لازم تحدد معاها ال Region علشان لو عملت Launch ل EC2 instance فى Region معينه مش هتلاقيها ف ال Region التانية علي عكس ال Global Services
- بتمكنك من انك تكريت users & groups وتديهم permissions وكدا
- انت By Default بتخش بال Root User ولكن دة مش مفضل انك تستخدمه فى العادي بتستخدم Admin Or Normal User
- علشان تعرف ال Service دي متاحة عندك فى ال region ولا لا من هنا



➥IAM Users, Groups Hands ON.

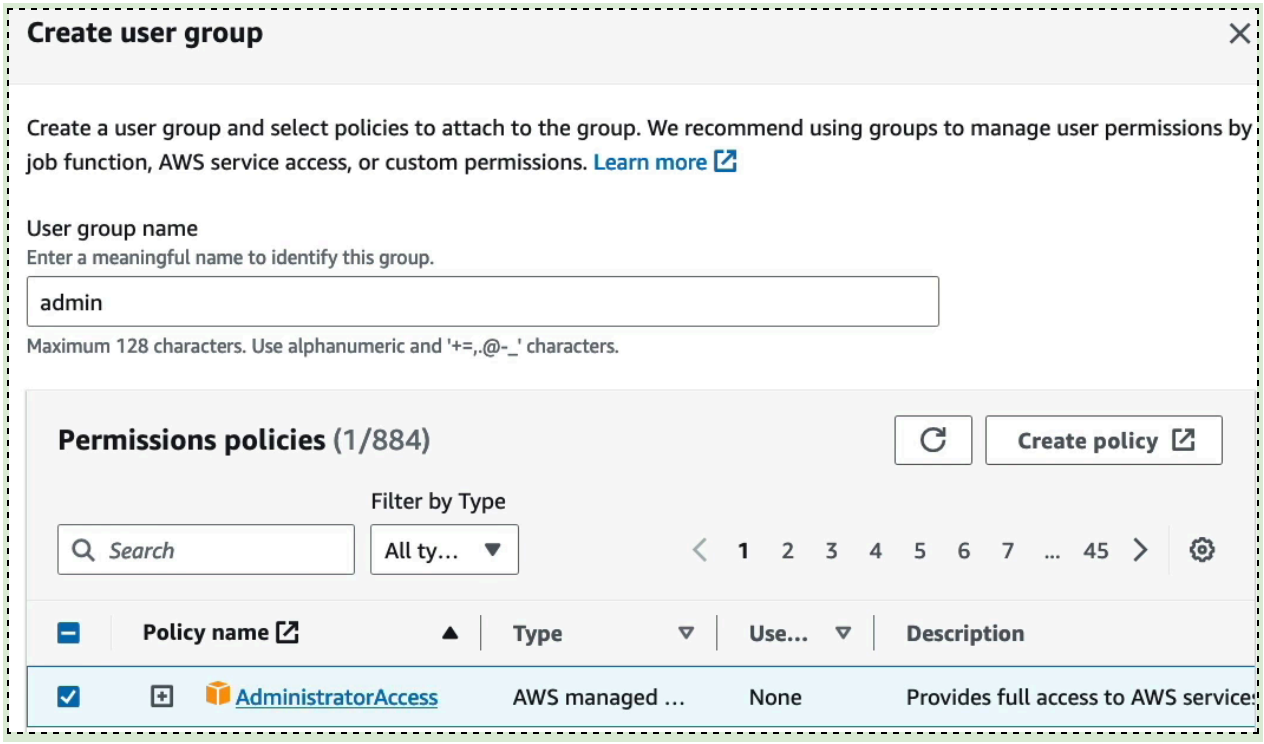
```
Search >> IAM >> User Name (Stephan), Password >>
Permissions
```



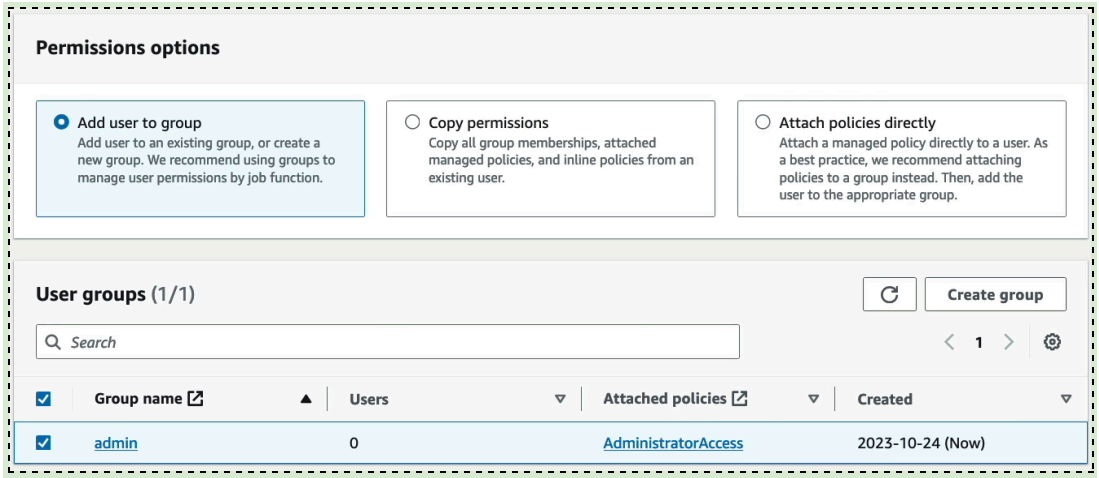
" تقدر تدخل ال user جروب انت عامله وال permissions فيه جاهزة او ت create

group أو ت Copy ال Permissions او ت Attach Policy"

"بما ان هنا مفيش Groups قديمة ف احنا هنكريت جروب جديد"



"في الحالة بتاعتنا عملنا user وعاوزينو Admin ف عملنا create group وال policy بتاعته خليناها Administrator Access بحيث يكون واخد كل ال Access عال AWS Services"

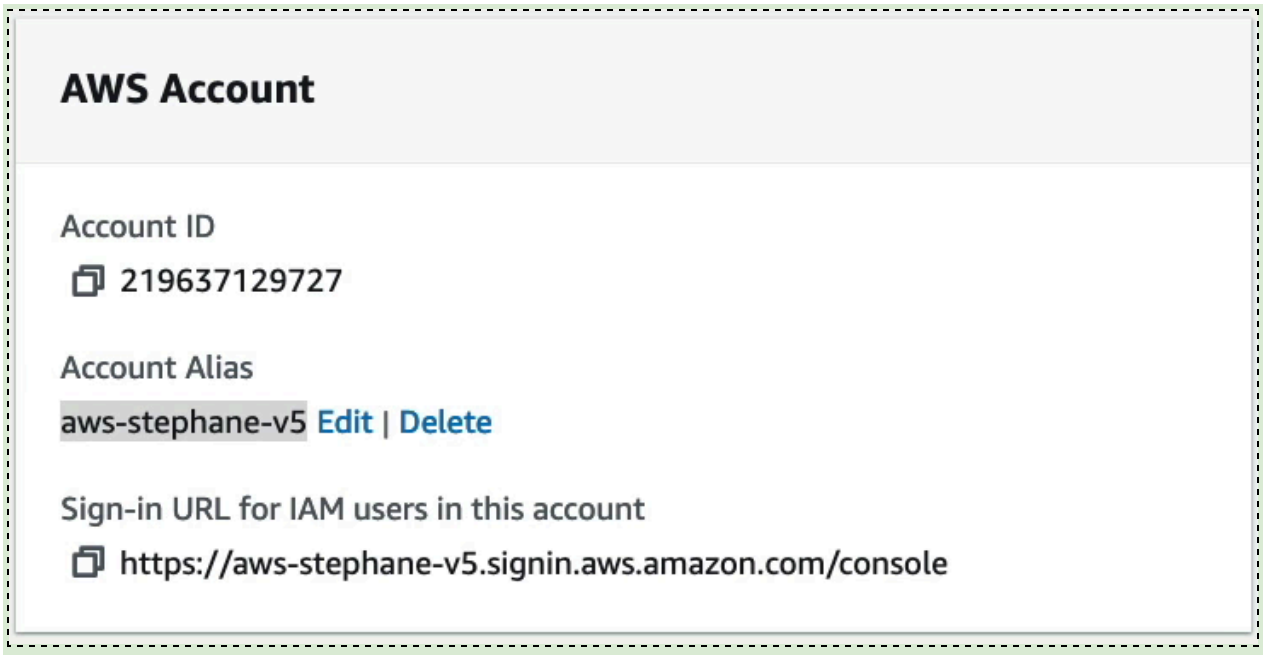


"كدا بقا الجروب جاهز .. ف ظهرلنا فى ال groups ف عملنا check عالجروب اللى لسة عاملينه علشان نضيف ال user فيه."

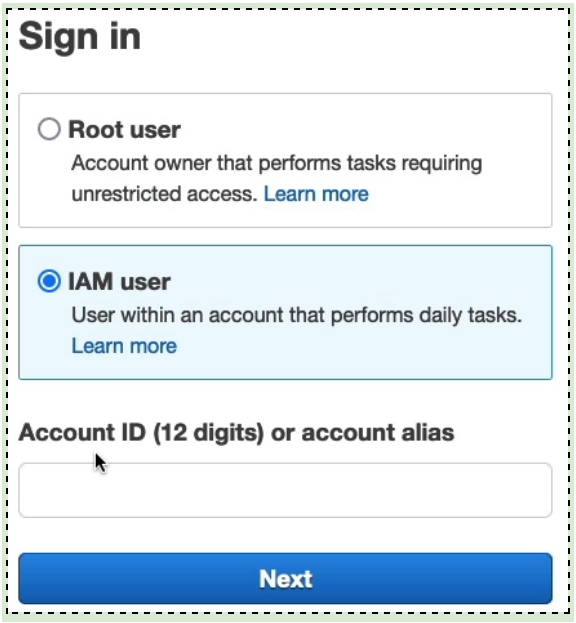
" تقدر وانت بتعمل ال user تضيف Tags ودي بتسهل عليه ت Access ال users مثلا تقسمهم : department, project, environment, or cost center."

```
Search >> Users , User Groups
```

"هتلاقي عندك بقا ال user وال permissions اللى واخدها اللى هيا بتاعة الجروب وهتلاقي ال groups اللى عندك وكل ال Configurations اللى انت عملتها وانت بتcreate ال user , group"

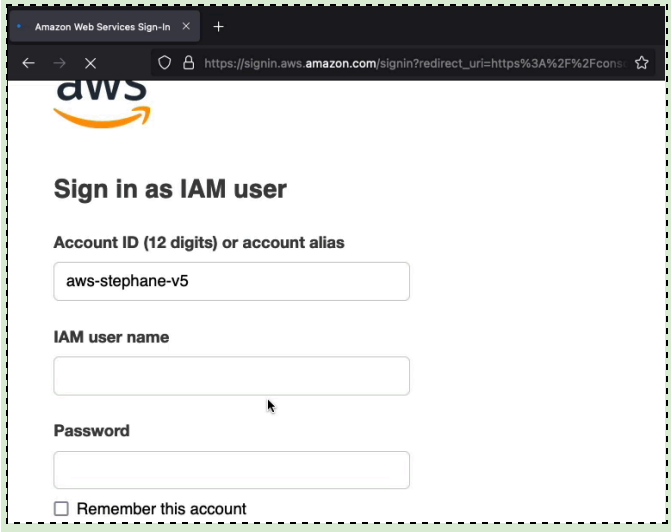


"ال user اللى انت عملته بيكون ليه Account id و Account Alias من خلالهم بتخش بيهم عالاكونت بجانب ال password"

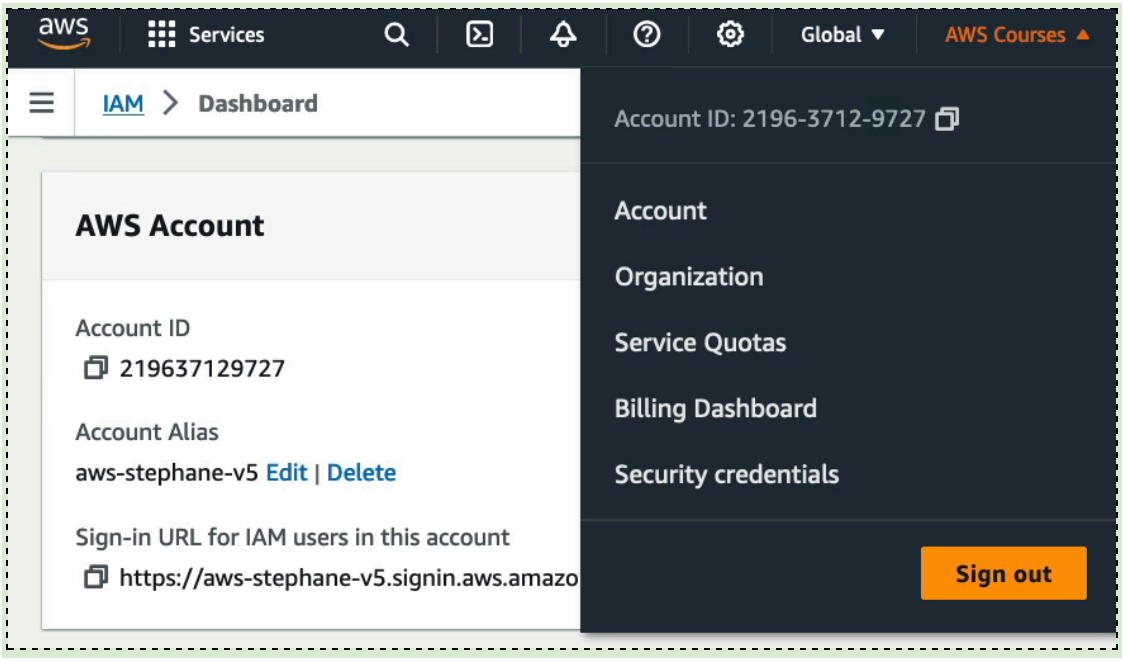


"بالنسبة لل url دة من خلالة بيدخلك مباشر على ال user دة بتكون محتاج ال password

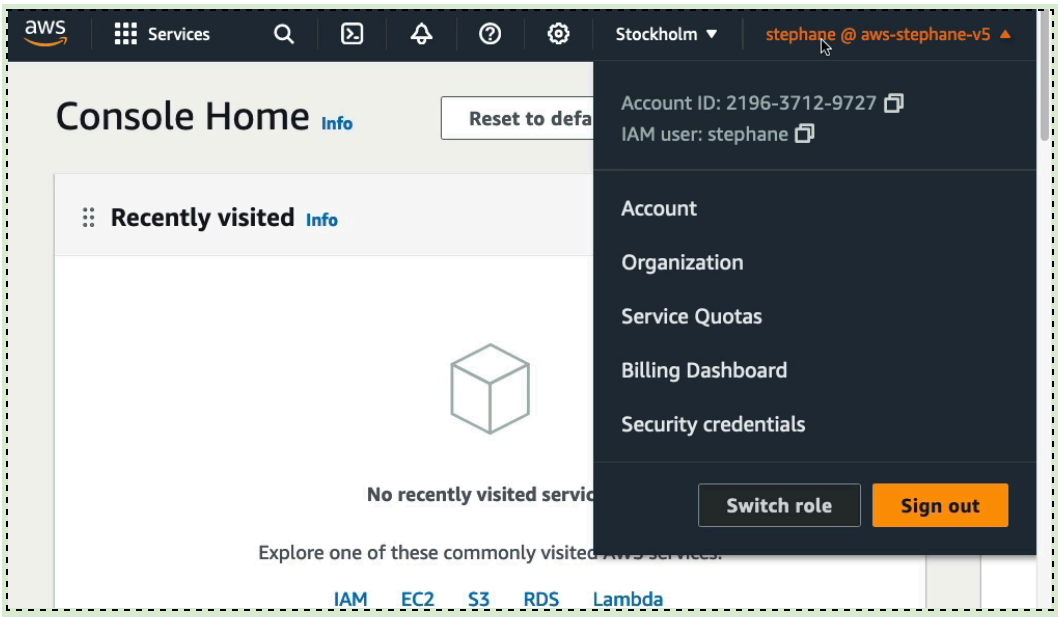
بجانب ال alias او ال id"



"دة بيكون شكل ال root اللى انت دخلت بيه فى البداية خالص .. Account ID فقط"

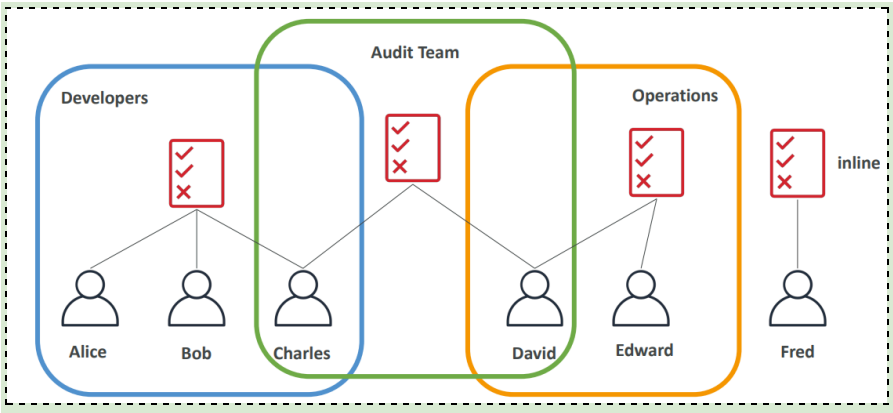


"دة بيكون شكل ال IAM User"



" ان حصلت مشكلة فى ال IAM Users Accounts هتهندلها من خلال ال Root الاصلي ..
إنما لو حصلت مشكلة فى ال Root Account مشكلة كبيرة وهتكلم ال Support."
" من خلال ال Root user تقدر ت delete stephane user وميكونش ليه access علي اي حاجة"

➥IAM Policies:



- عباره عن JSON Files بتحدد ال Permissions لل User & Group
- ال Policy اللى واخدها الجروب بتطبق علي كل user فيه.

• Policy Structure:

- بيكون عبارة عن JSON file .. من خلالة بت define ال permissions وبتـ specify ال actions هل allowed or denied ل resources معينه .. ال Structure دة بيتكون من :

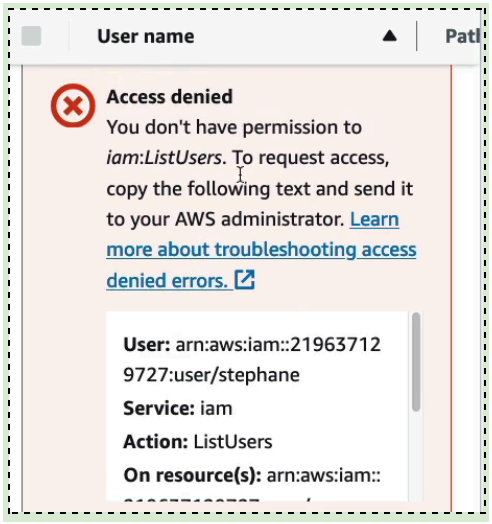
```
{
    "Version": "2012-10-17",
    "Id": "S3-Account-Permissions",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "AWS": ["arn:aws:iam::123456789012:root"]
            },
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": ["arn:aws:s3:::mybucket/*"]
        }
    ]
}
```

- **Version :**
  - دة version ال policy language ودة حاليا 17-10-2012
- **ID (Optional):**
  - عبارة عن identifier لل policy
- **Statement:**
  - بتحتوي علي Statement او اكثر ال Statement عبارة عن rule وبتحتوي علي كذا عنصر:
    - **SID (Optional):**
      - عبارة عن identifier لل Statement
    - **Effect:**
      - هل ال Statement دي Allow or Deny
    - **Action:**
      - ال account or user or which role اللى ال policy دي هتطبق عليه.

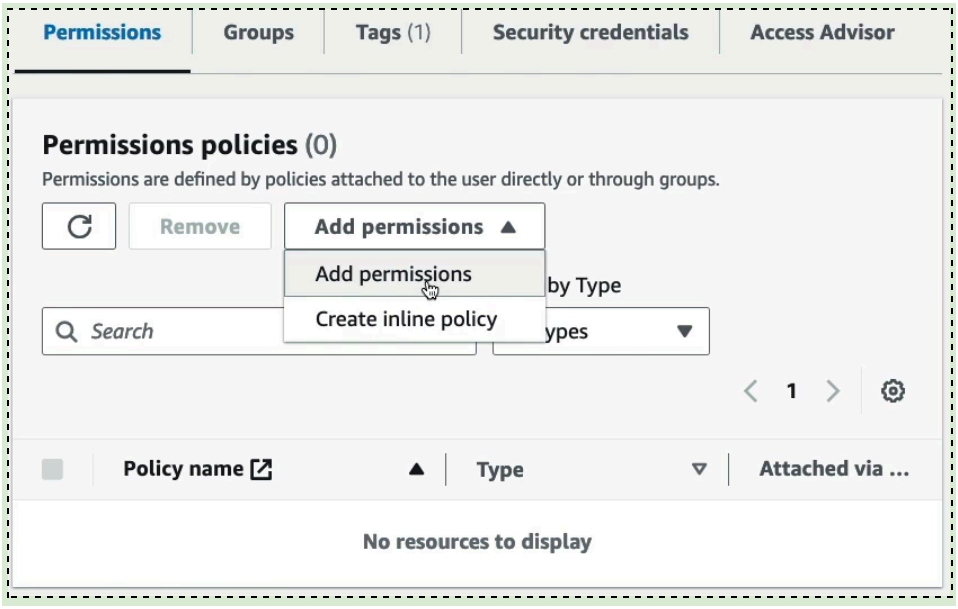
- **Resources:**
  - ال Resources اللى ال Actions هتطبق عليها.
- **Condition(Optional):**
  - شروط تفعيل ال policies دي.

➥IAM Policies Hands On:

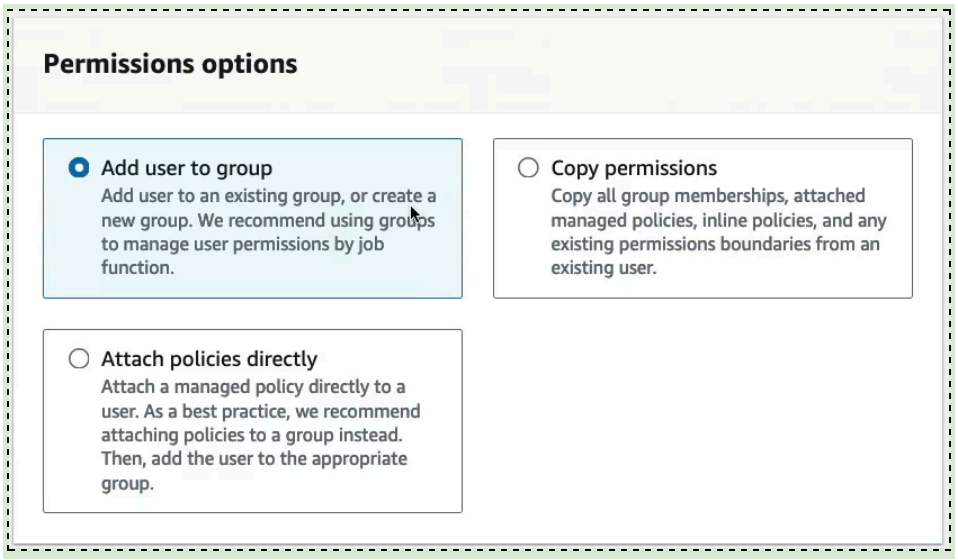
"لو شيلت stephane من ال group كدا هيفقد ال admin permissions اللى كان واخدها لانها تبع الجروب.. ف مش هيقدر مثلا انو ي list ال users"




```
Users >> Stephane >> Permissions
```

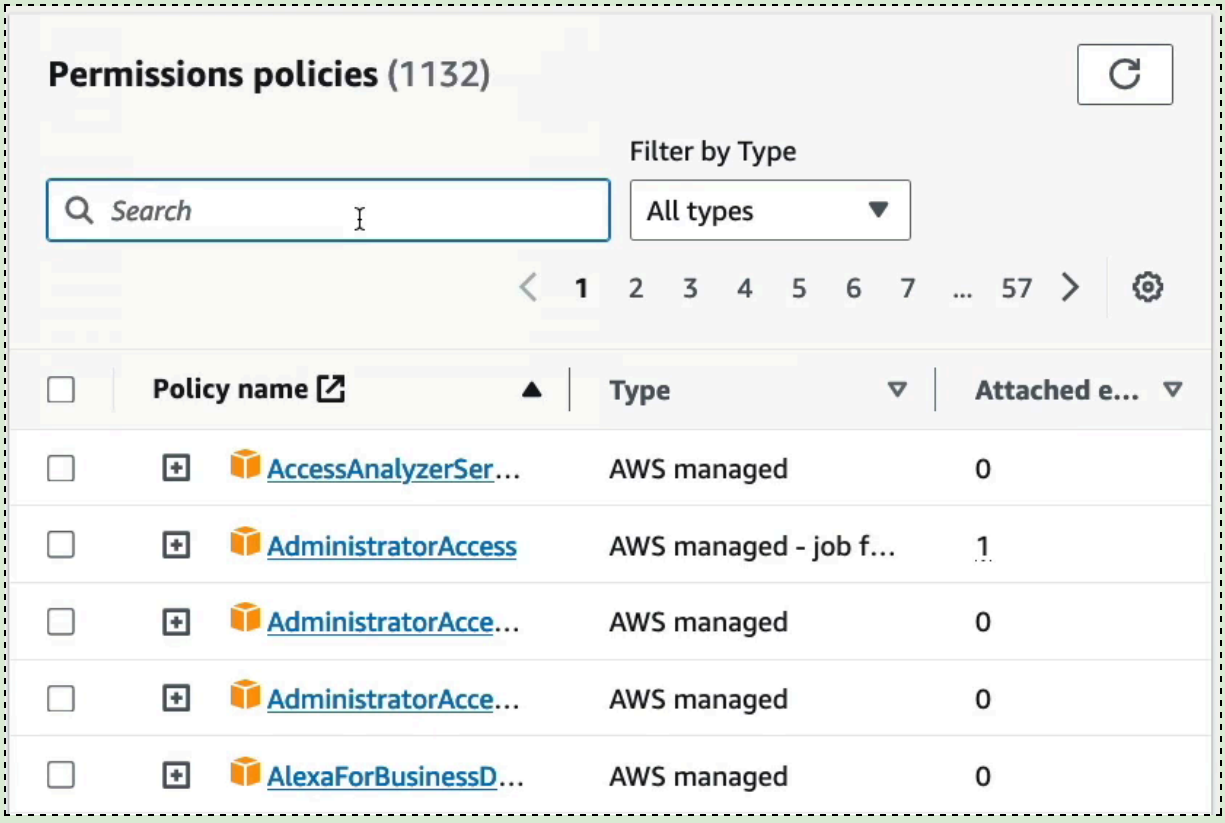


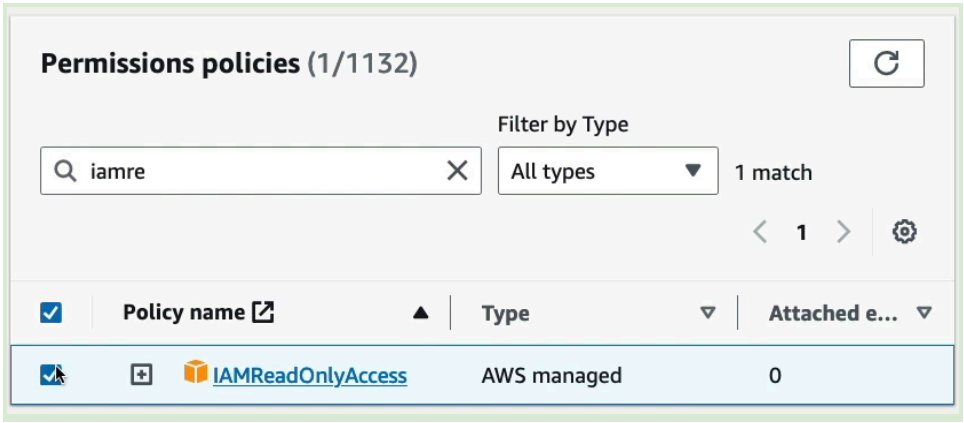
"تقدر من تاني تحط ال permissions اللى انت عاوزها لل user stephane"



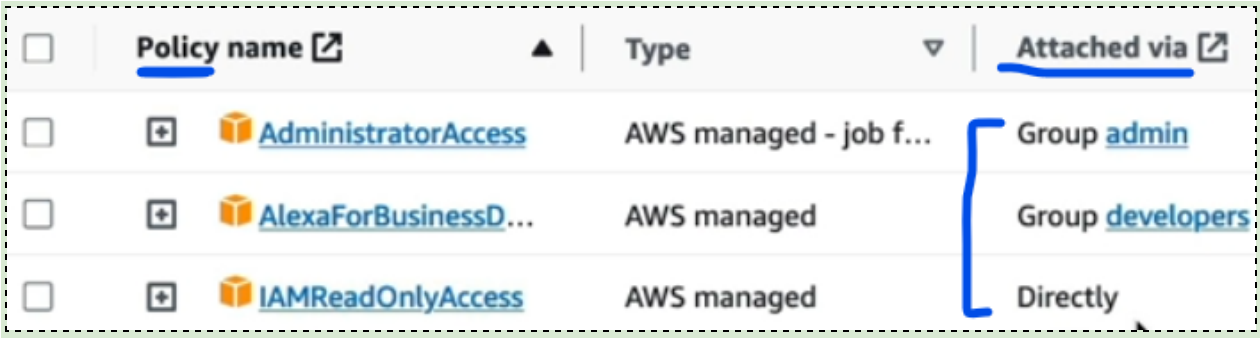
"قبل كدا دخلناه جروب وخد ال permissions بتاعته اللى كانت Administration

permissions ودلوقتي هن Attach policies .. وليكن read only policy"



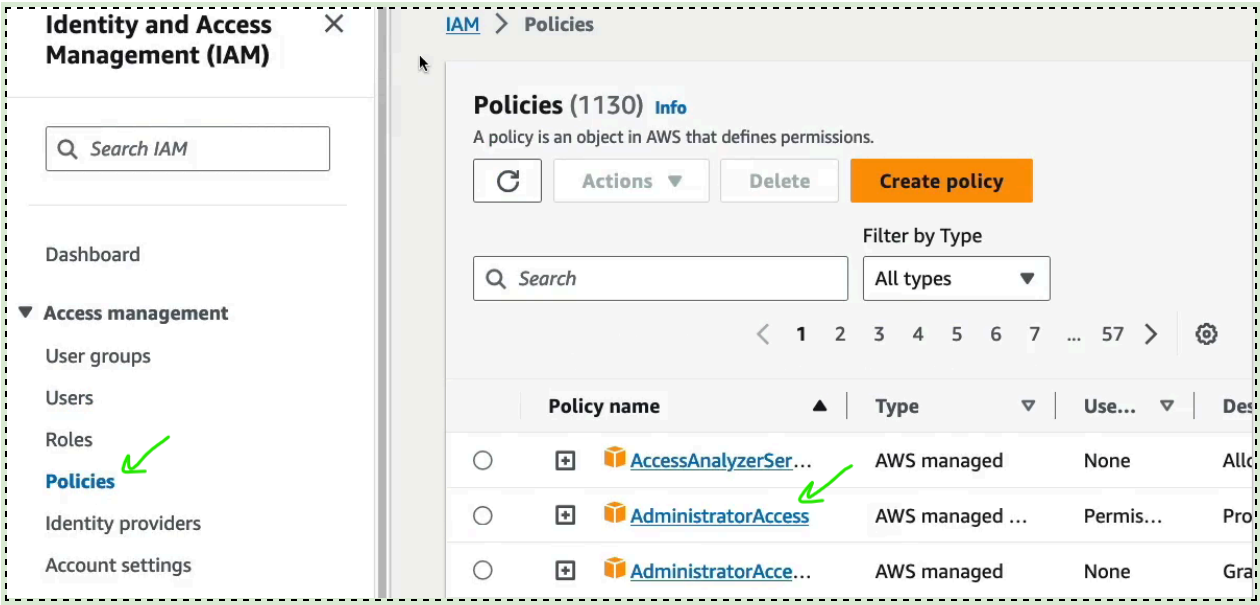


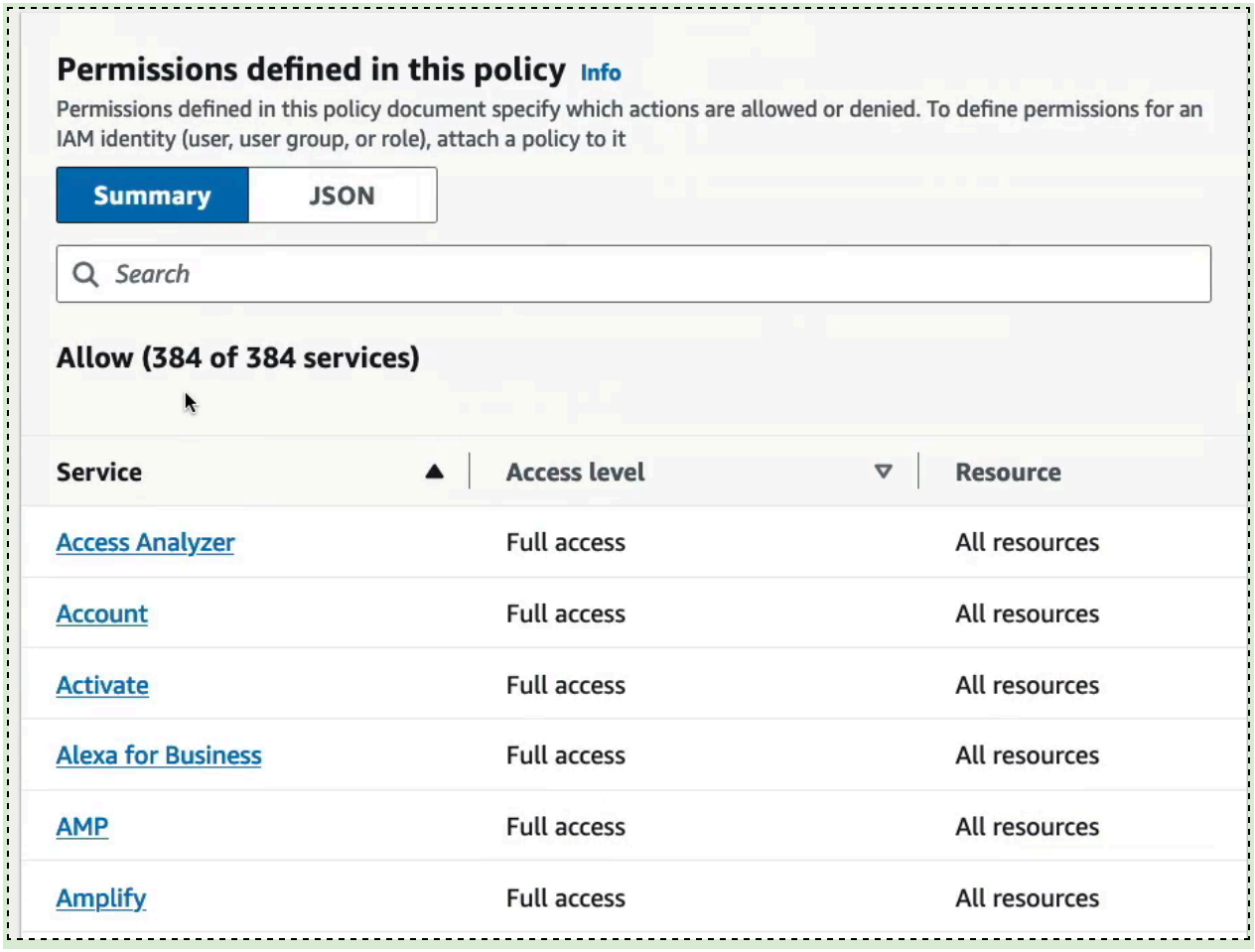
"بعد ما staphane خد ال readonly permission كدا يقدر ي list ال users مثلا.. لو جيت ب stephane اللى واخد readonly permission تحاول ت create group مش هينفع."



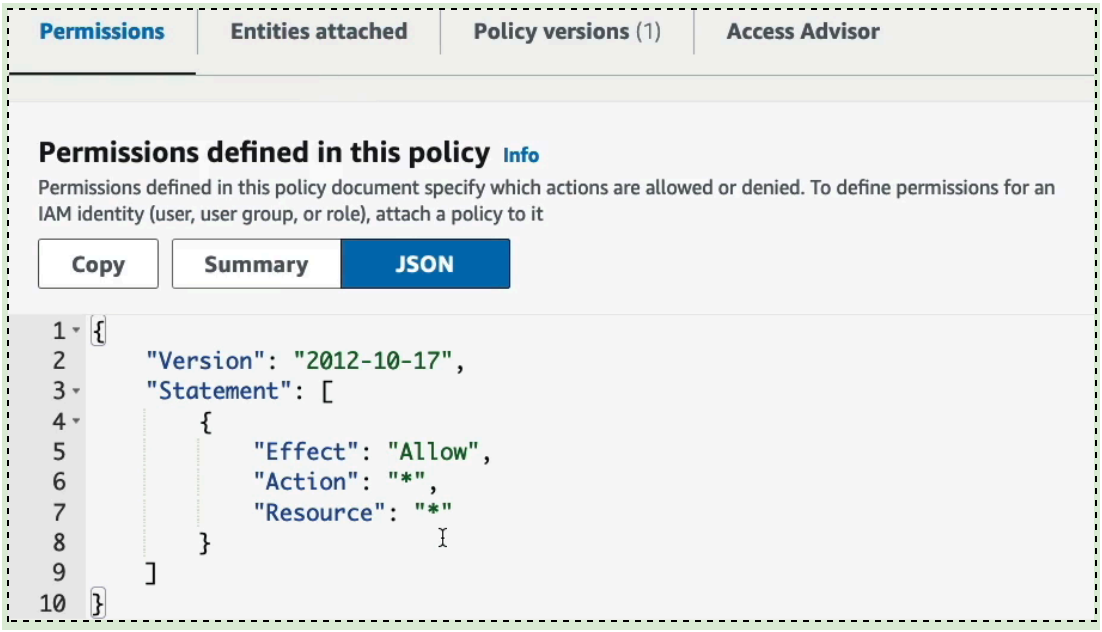
"بيكون واضحلك انت واخد permissions اية ومن انهي groups بالظبط ولو directly ف دة اللى root مديهولك ال root مدي ل stephane وهو بيكريتة readonly permission."

```
IAM >> Policies >> AdministratorAccess
```



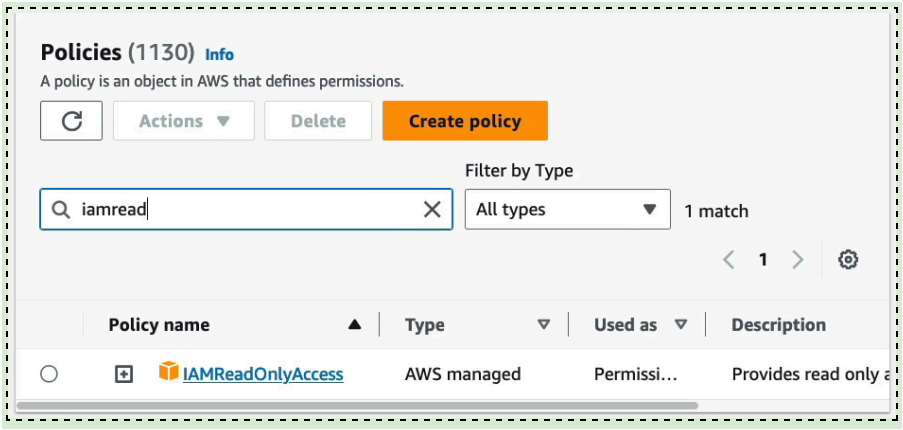


"لما تخش علي ال policies وتخش علي واحدة منهم وليكن AdministratorAccess هتلاقي كل ال services اللى ال policy دي تقدر تتحكم فيهم وكمان اية نوع ال access اللى واخداه هنا مثلا Full Access"

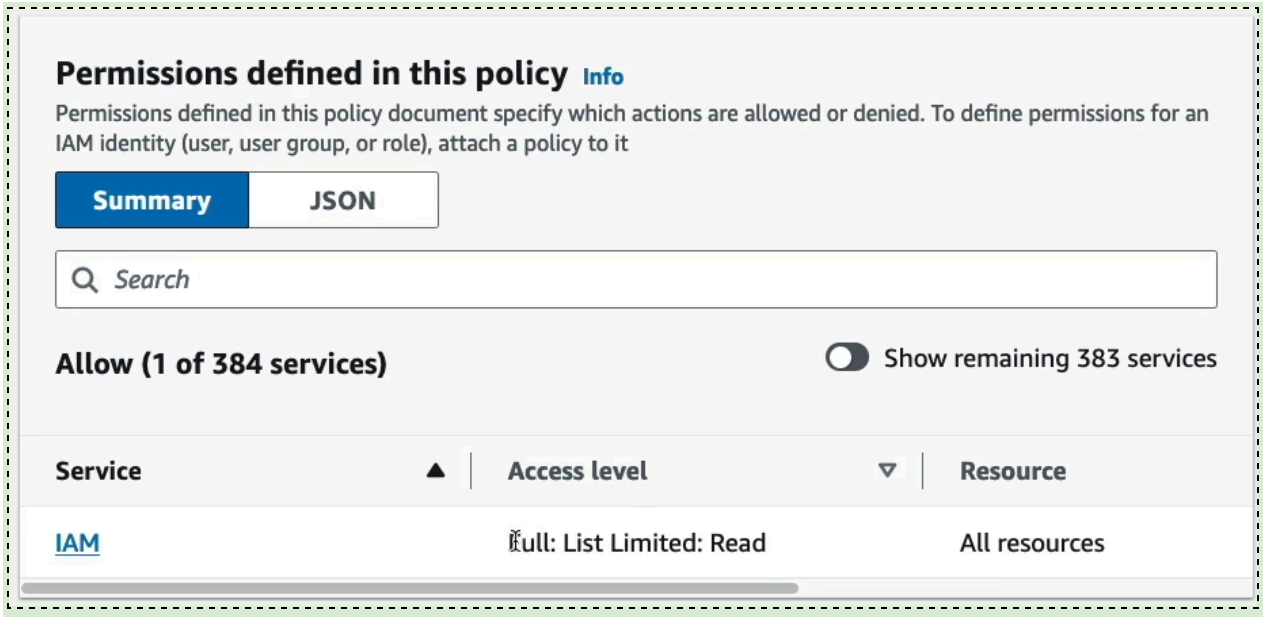


"لو دخلت على JSON هتلاقي محتوي ال JSON File خاص بال Policy ياعني لو Policy تانية زي ما هنشوف تحت هتلاقي محتوي تاني خصوصا فال statements لان كل policy ب rules مختلفة"

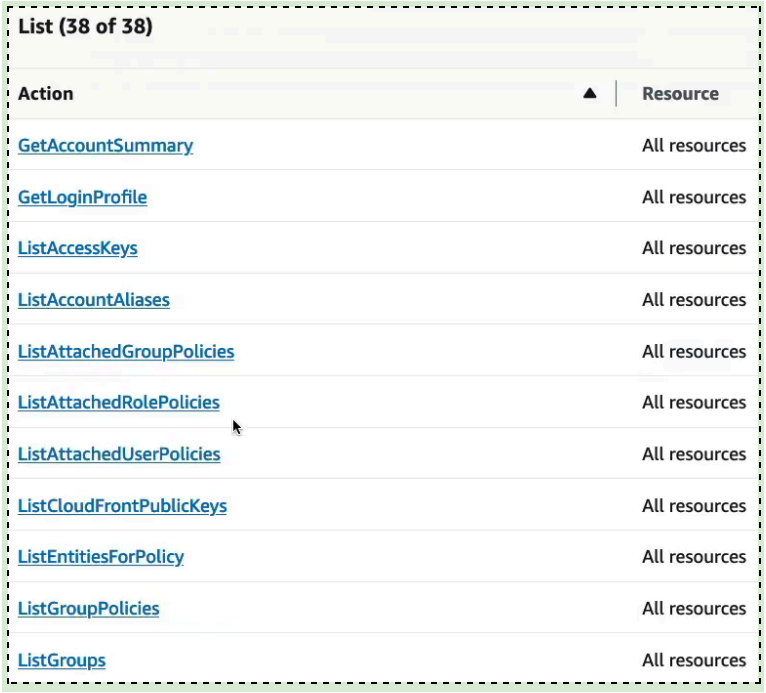
"هنا معني ال JSON File دة .. بن Allow any action على كل ال resources .. ودة فعلا ال access اللى بياخدو ال AdministratorAccess."



"لو دخلت مثلا على policy تانية زي IAMReadOnly هتلاقي واخدة access علي service واحدة "

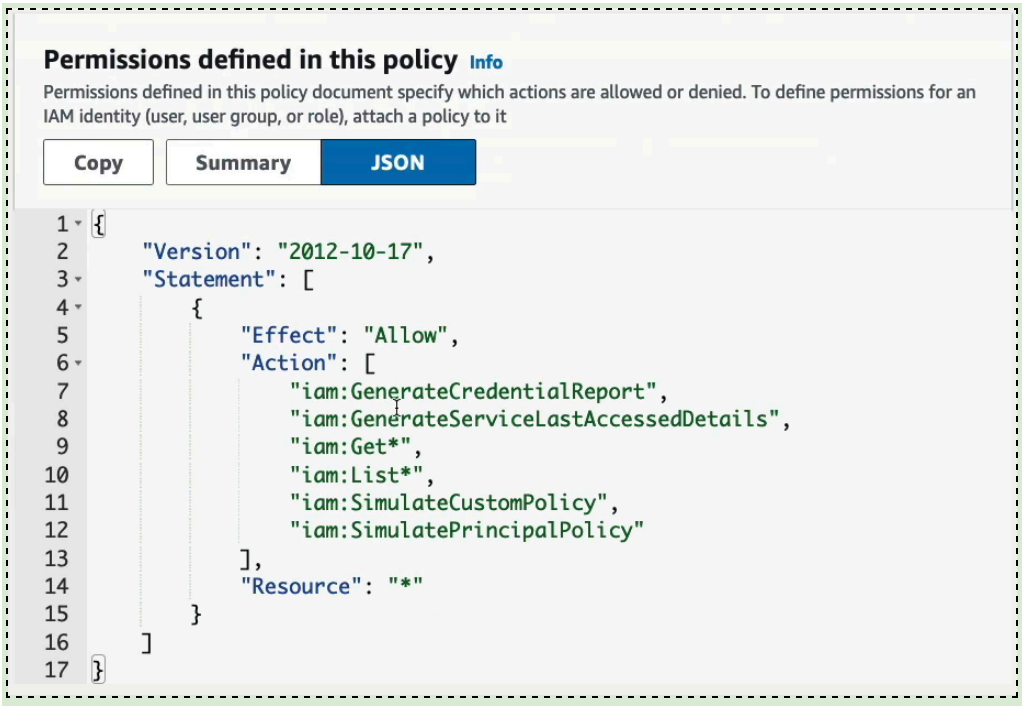


"ولو دخلت عال service دي هتوضحلك اية بالظبط اللى اقدر أ read only عليه.. اية هيا ال API Calls اللى مسموح بيها"



List (38 of 38)

Action	Resource
GetAccountSummary	All resources
GetLoginProfile	All resources
ListAccessKeys	All resources
ListAccountAliases	All resources
ListAttachedGroupPolicies	All resources
ListAttachedRolePolicies	All resources
ListAttachedUserPolicies	All resources
ListCloudFrontPublicKeys	All resources
ListEntitiesForPolicy	All resources
ListGroupPolicies	All resources
ListGroups	All resources



Permissions defined in this policy Info

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Copy | Summary | JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:GenerateCredentialReport",
                "iam:GenerateServiceLastAccessedDetails",
                "iam:Get*",
                "iam:List*",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

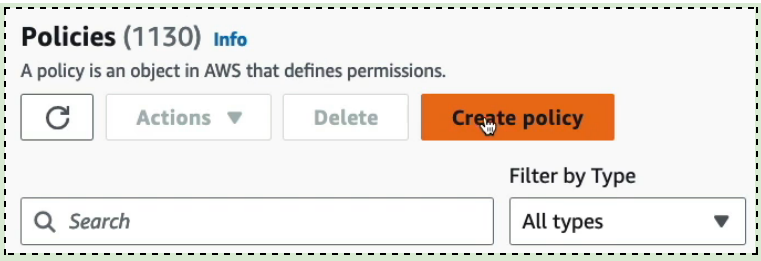
"هنا فال read only policy محتوي ال JSON File اختلف عن اللى policy اللى فاتت وهكذا."

"Get"* دي wild card .. مثلا Get users , Get Groups وهكذا نفس الكلام على "list

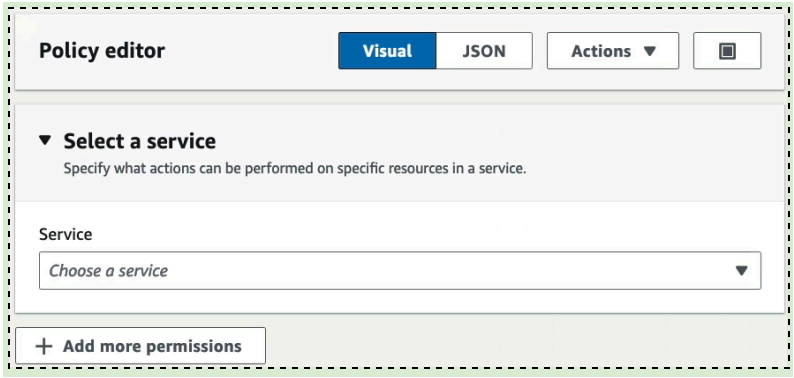
💡 يبقي الخلاصة : دة الترتيب

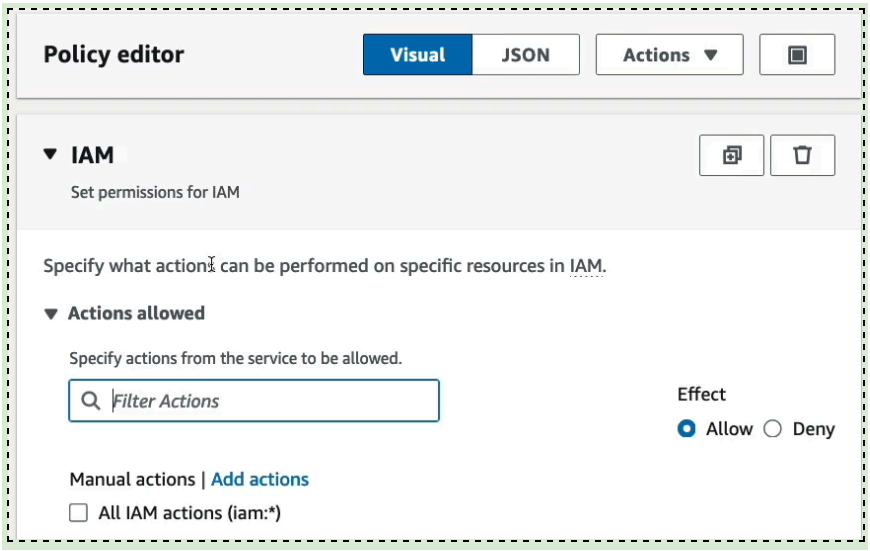
```
IAM >> Policies >> One Of The Tolices >> services can
access by the policy , json file content
```

```
IAM >> Policies >> Create Policy
```



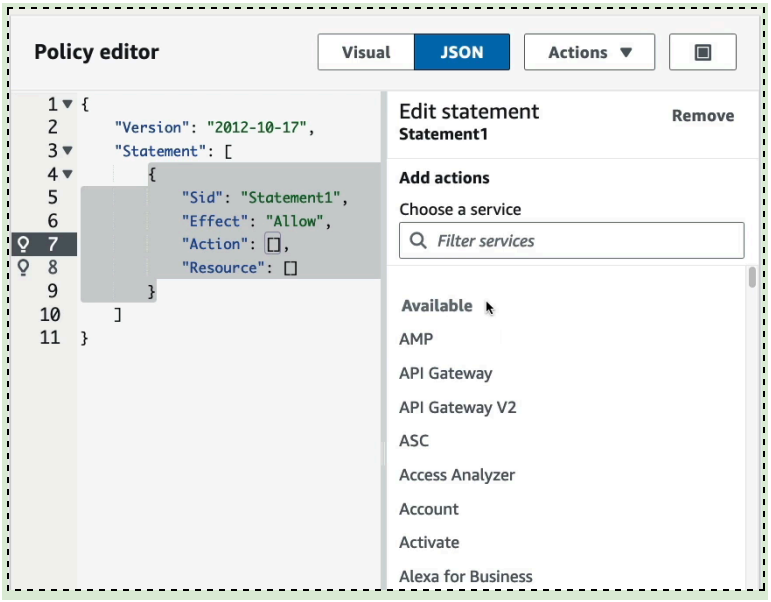
"ممكن انت ت Create ال policy الخاصة بيك .. هتلاقي عندك visual editor or json editor"





"وبتختار بقا ال service ثم ال actions وهكذا

وبالنسبة لل JSON .. نفس الكلام"



"وبعدها بقا تحط لل policy name & description وهكذا"

➥IAM – Password Policy

- **Passwords:**

```
IAM >> Account Setting >> Edit Password Policy
```

- تقدر تحدد لل users ال passwords بتاعتهم لازم يكون شكلها عامل ازاي
- تقدر تحدد المدة اللى لازم يغيرو فيها ال passwords دي
- تقدر تمنع انهم يعيدو استخدام passwords قديمة

- **Multi Factor Authentication - MFA**

- عباره عن password محفوظ جوا device معاك

- تقدر تطبق ال Multi Factor Authentication عن طريق تطبيقات زي Google Authenticator او Authy .. او من خلال Physical Devices

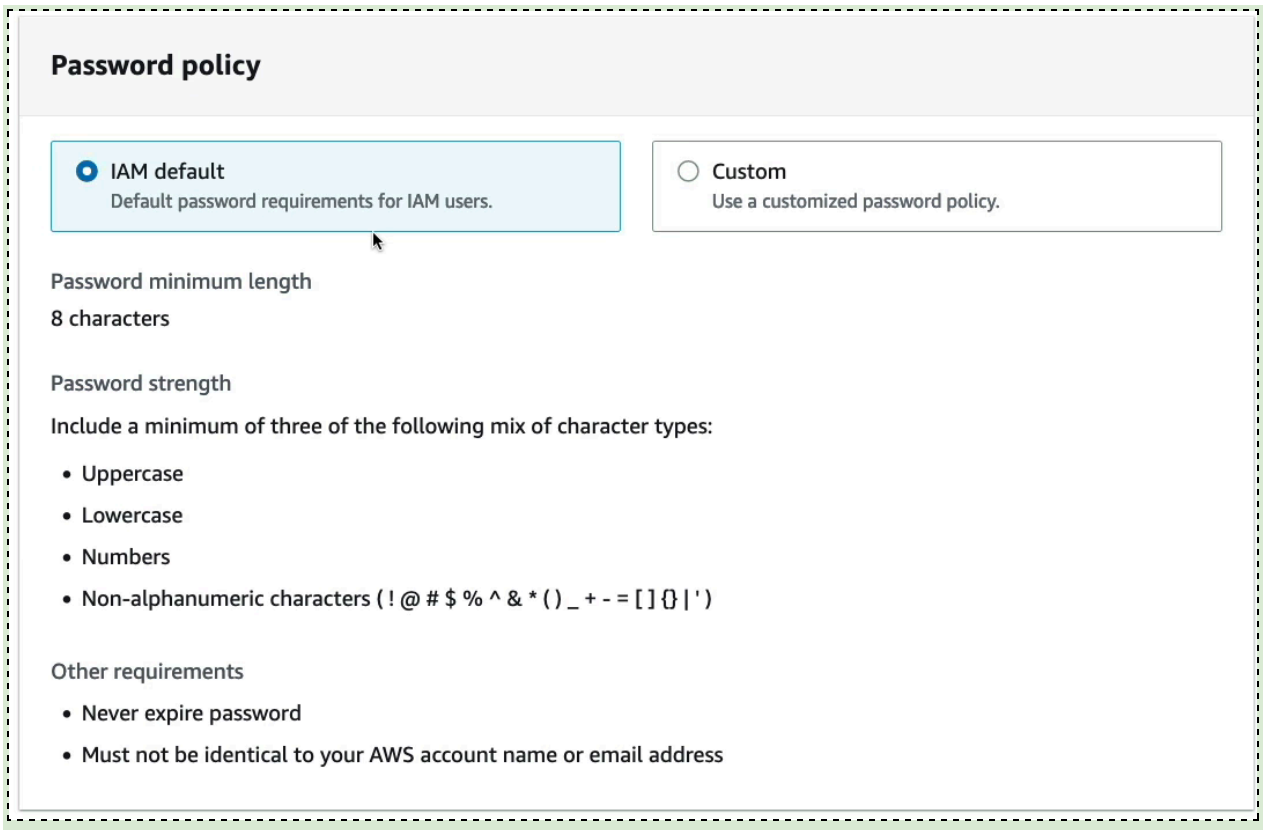
➥IAM MFA Hands ON

- **Passwords Policy:**

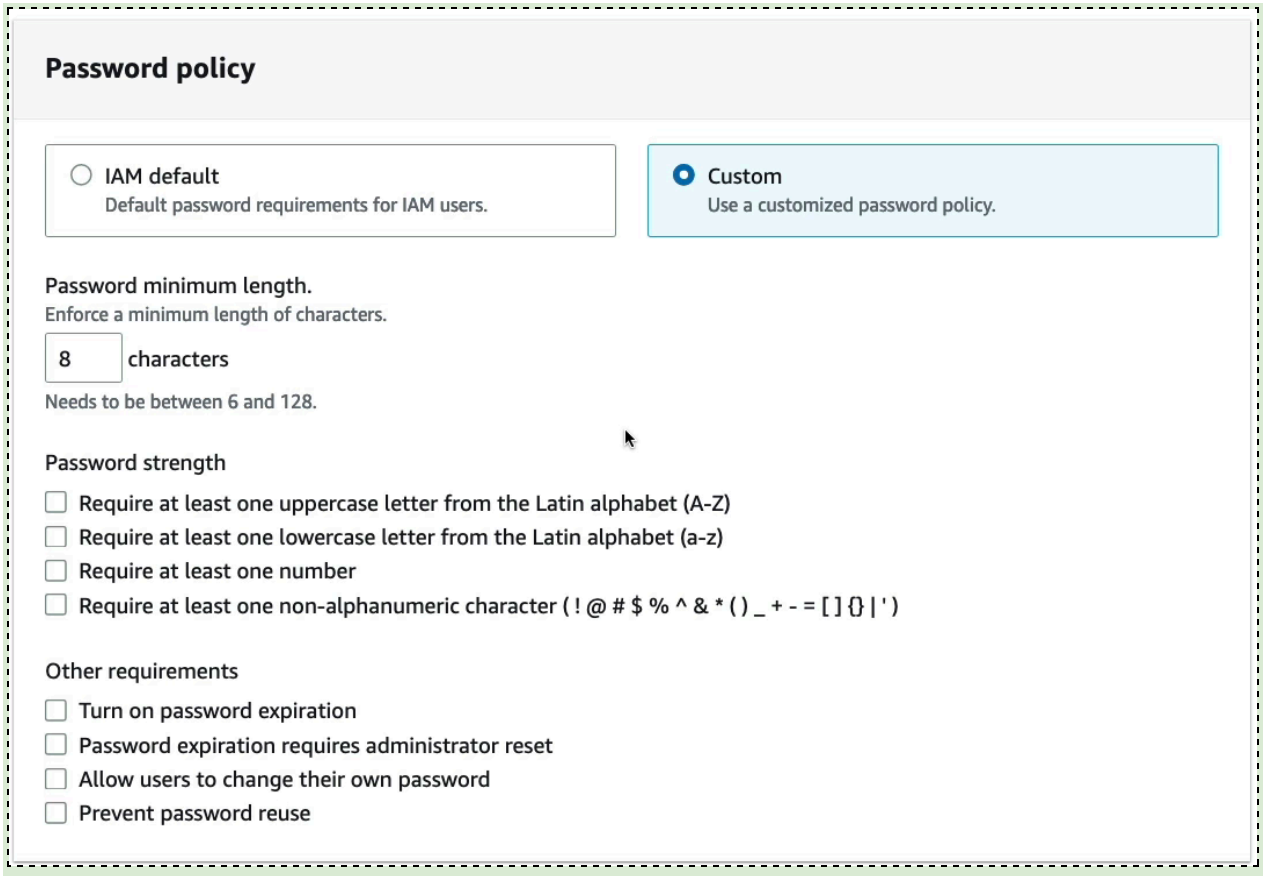
```
IAM>> Access Management >> Account Setting >> Password
```

```
Policy >> Edit
```

تقدر تستخدم ال IAM Default بالشكل دة

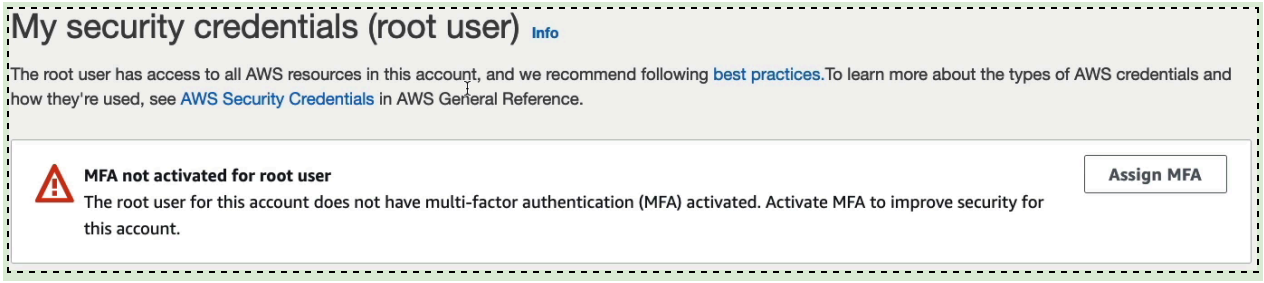


أو ت Custom انت ال Password Policy الخاصة بيك بحيث تستخدمها بعدين.



- **Assign MFA:**

```
Account Name >> Security Credentials
```



من خلال Assign MFA بت set ال Device name , Device Type

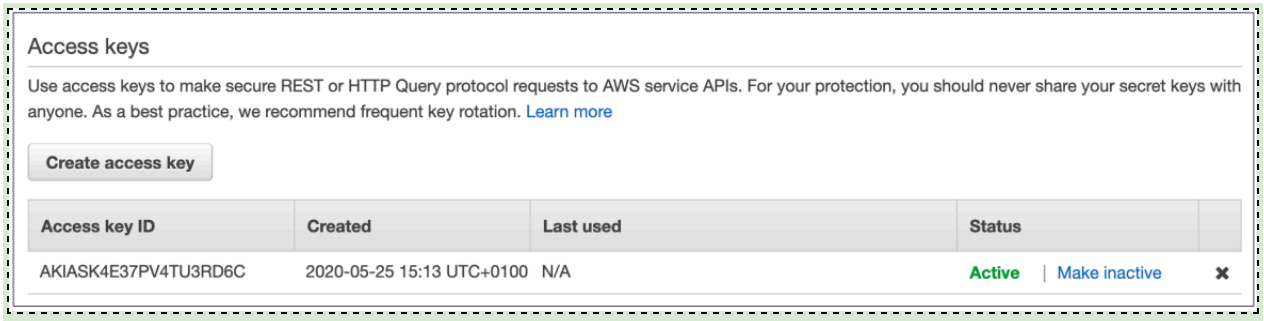
ودي ال Authenticating Apps لو مش معاك device تقدر تستخدمها

Android	Twilio Authy Authenticator, Duo Mobile, LastPass Authenticator, Microsoft Authenticator, Google Authenticator, Symantec VIP
iOS	Twilio Authy Authenticator, Duo Mobile, LastPass Authenticator, Microsoft Authenticator, Google Authenticator, Symantec VIP

لو قفلت ال root acc وجيت تفتح تاني هيطلب منك MFA Code هتجيبه من خلال التطبيق او الجهاز ايا كان ياعني .. لو التطبيق او الجهاز ضاعو هتبقي مشكلة كبيرة.

➥AWS Access Keys, CLI and SDK:

- اتعاملنا مع AWS من خلال ال Management Console من خلال المتصفح ياعني.. وعندك طريقتين كمان هما ال CLI وال SDK والاتنين علشان نشتغل منهم لازملنا Access Keys



💡 ال Access Key بتاعك انت بس ميخرجش بره.

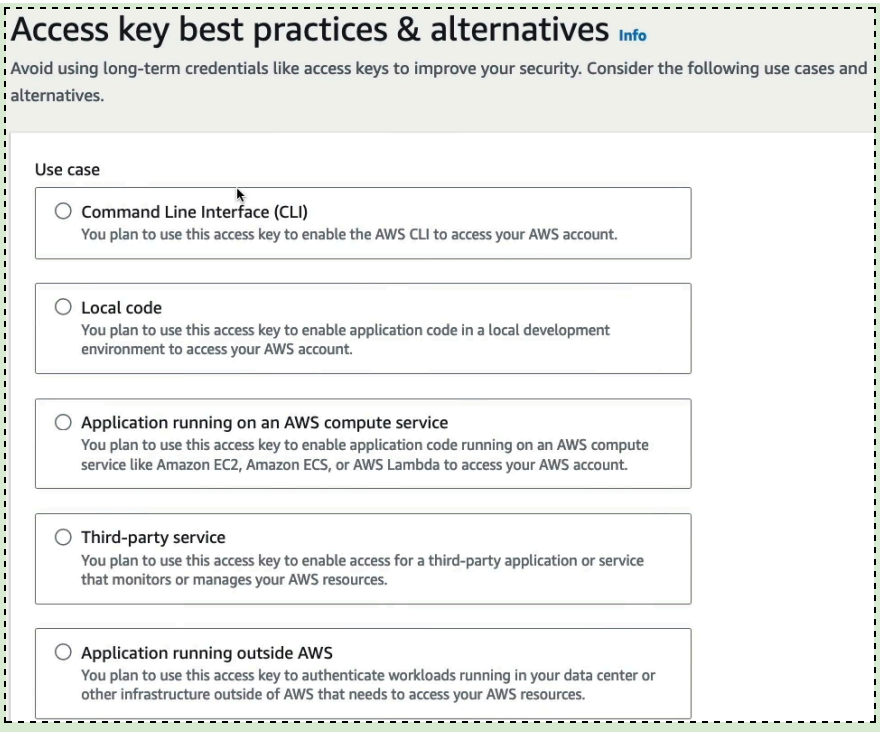
• SDK (Software Development Kit):

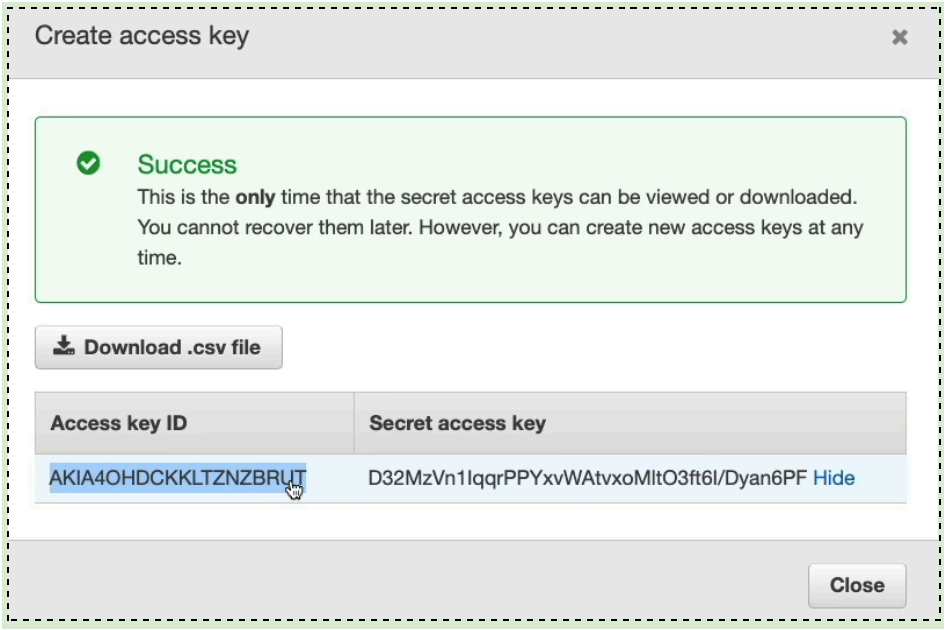
- عبارة عن مجموعة من ال libraries & tools اللى developers يقدرو يستخدموهم وهما بيكتبو كود لل Applications بتاعتهم بحيث يقدرو يخلو ال Applications تـ interact مع AWS Services
- **DevOps and CI/CD**: SDKs are used in continuous integration and continuous deployment (CI/CD) pipelines to automate the deployment and management of applications on AWS

💡 مثال على استخدام SDK هو ال CLI اللى هنستخدمه فى اننا ن Access AWS كمان شوية .. دة مبني على AWS SDK For Python واسمو boto3

➥AWS CLI Hands On:

Users >> Stephane >> Security Credentials >> Access Keys >> Create Access Key





بعد ما وصلت لل Access Key وليكن بعد ما اخترت CLI بتروح بقا على ال terminal

علشان تربط الاتنين ببعض

```
~ aws configure
AWS Access Key ID [None]: AKIA4OHDCKKLTZNZBRUT
AWS Secret Access Key [None]: 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
Default region name [None]: eu-west-1
Default output format [None]:
~ aws iam list-users
```

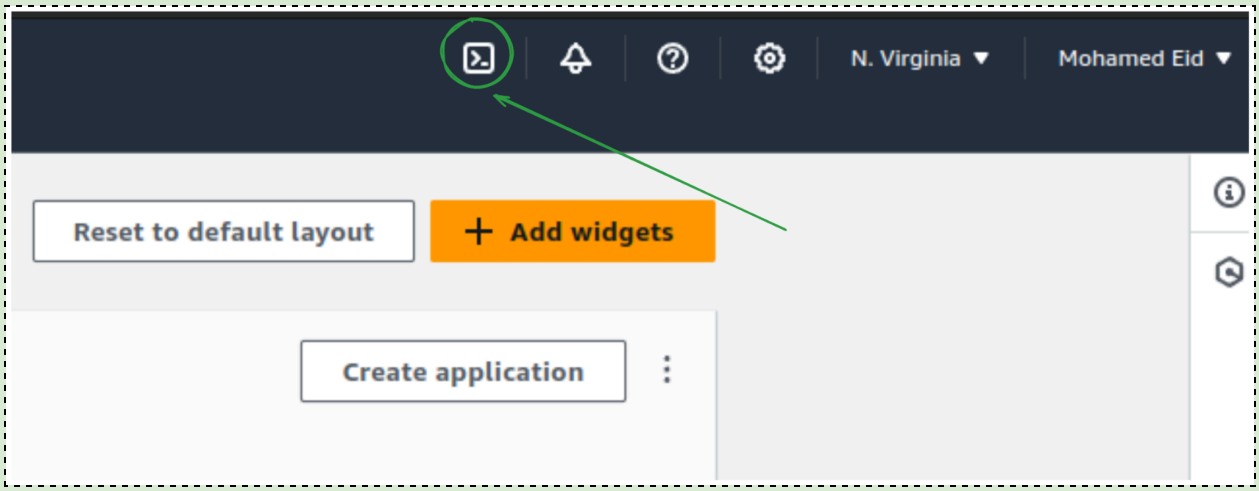
```
{
    "Users": [
        {
            "Path": "/",
            "UserName": "stephane",
            "UserId": "AIDA4OHDCKKLSE4HHDCOE",
            "Arn": "arn:aws:iam::855174697623:user/stephane",
            "CreateDate": "2020-05-27T16:28:16+00:00",
            "PasswordLastUsed": "2020-05-27T16:33:03+00:00"
        }
    ]
}
(END)
```

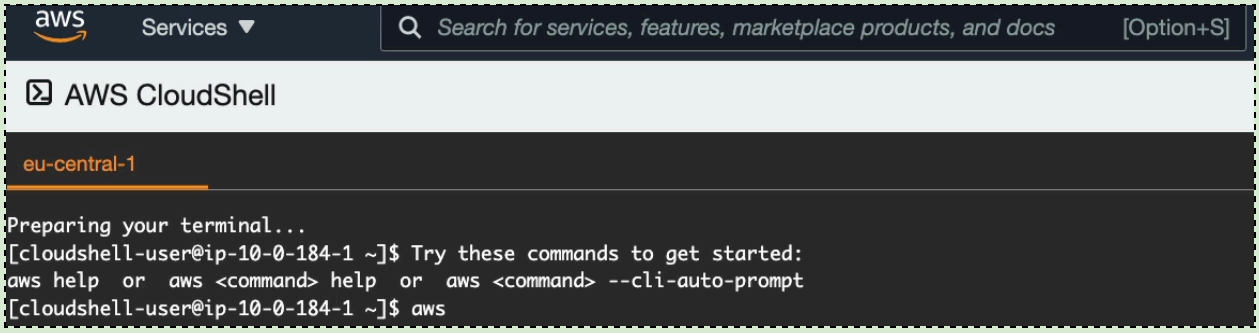
بت Configure .. وبالنسبة لل Region بتختار الاقرب ليك.

من خلال ال commands تقدر تعمل بقا كل اللى كنت بتعملو على aws management console قبل كدا

💡 لو شيلت ال user اللى هو stephane من جروب ال admins ياعني مبقاش واخد اي permissions وجيت فى ال terminal نفذت نفس ال command اللى هو list the users هتلاقي انو مدكش نتيجة ودة حصل مباشرة.

➥AWS CloudShell:





- Terminal عادي خالص علي AWS وبيكون متاح فى بعض ال regions تقدر تشوفهم من هنا
- بيكون فيه repository عادي ياعني لو انشأت file هتلاقيه موجود متخزن وتقدر تعمله download وتعمل upload وهكذا اكنك فاتح terminal عادي.
- ال terminal اللى على جهازي افضل اكيد.

➥IAM Roles for AWS Services:

- Role ياعني دور ك ترجمه حرفيه .. ياعني هديك permission تقوم ب دور معين وغالبا بيكون الموضوع دة لمدة قصيره ..
- لو عملت access على EC2 instance وجيت ت list ال user بالشكل دة aws iam list-users مش هتلاقيها واخده ال permission دة ف بنروح ن attach role ليها تكون ليها policy allow read only وبالتالي تقدر ت list users .. **ودي امثله اكتر :**

- **Cross-Account Access**

- شركة A عاوزه تدي Permission للأكونت بتاع شركة B إنو يقدر يستخدم الداتا اللى على ال S3 Bucket بتاع شركة A ف دة بيتم عن طريق ان شركة A تكريت role وتديها لاكونت شركة B .. ال role بتبقي عباره عن policies ف بتحدد فى الpolicy انهي permission مسموح بيها.

- **AWS Lambda Functions**

- Lambda function ودي service عاوزين نديها role دور ياعني تمكنها من انها يكون ليها ال permissions اللى تخليها قادره ت access DynamoDB tables and S3 buckets علشان ت process الداتا مثلا
- يبقي كدا عندنا service اديناها permission مكنها انها ت perform operations على some resources

- **EC2 Instances and S3 Access**

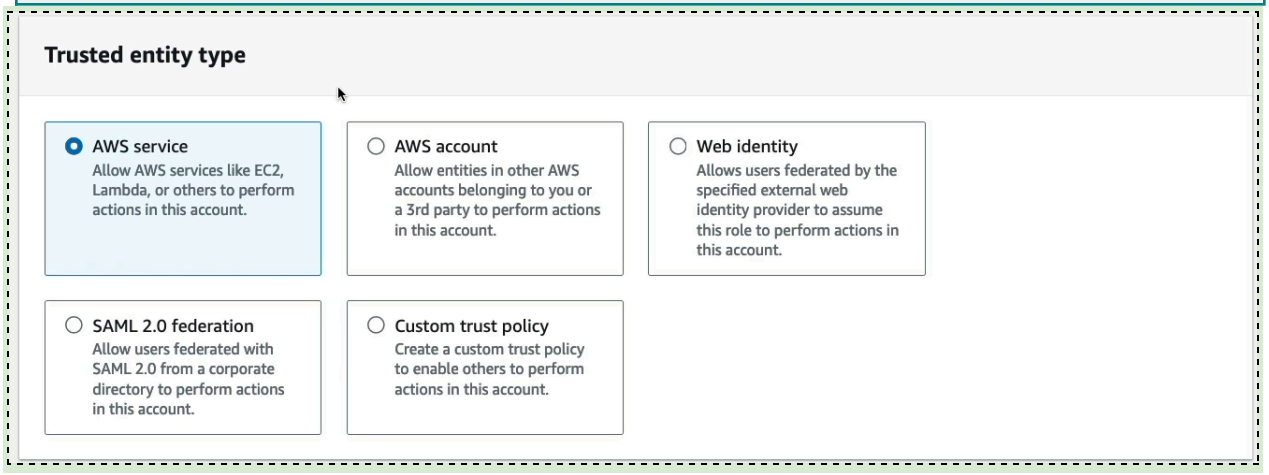
- عاوزين نخلي ال service اللى اسمها EC2 Instance قادرة انها ت retrieve configurations files & upload logs ل S3 Bucket ف بنديها role علشان تقدر تعمل دة

• Most Common Roles:

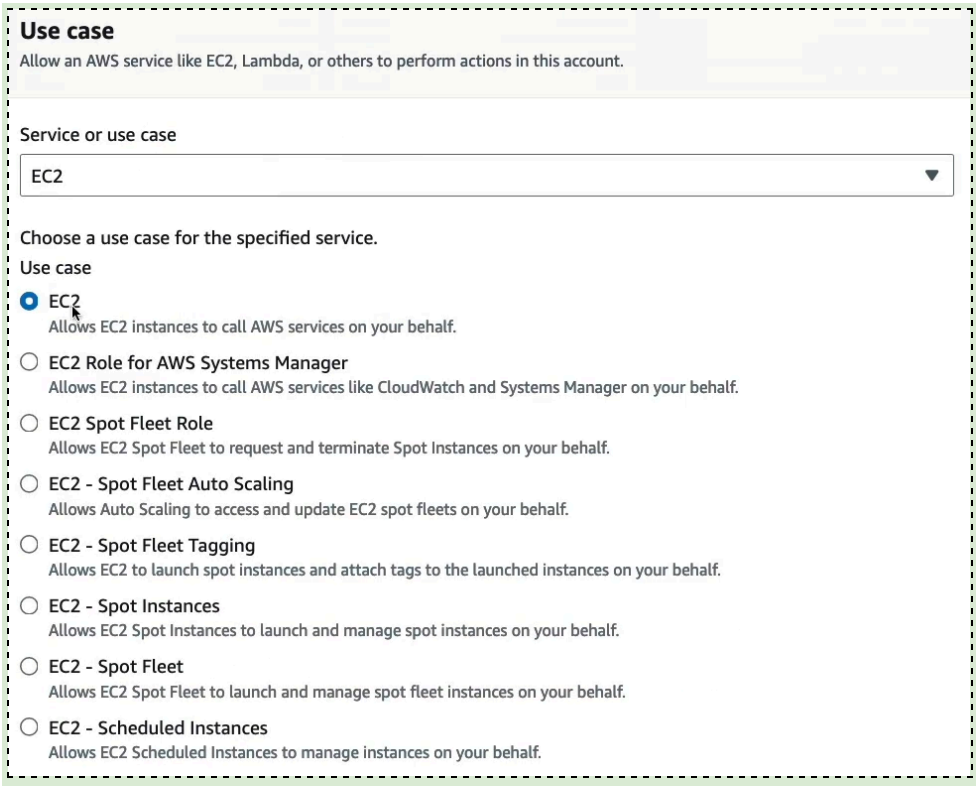
- EC2 Instance Roles
- Lambda Function Roles
- Roles for CloudFormation

➥IAM Roles Hands ON:

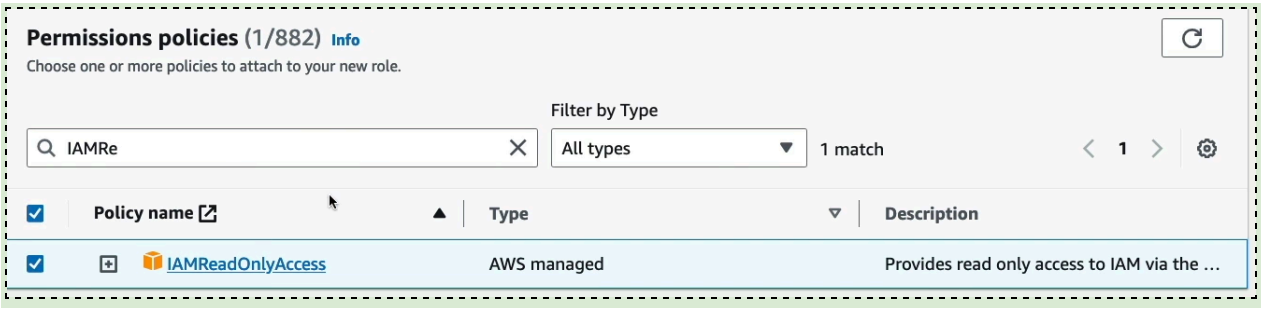
Access Management >> Roles >> Create Role



زي مانت شايف فيه كذا نوع ولكن هنشتغل حاليا على AWS Service

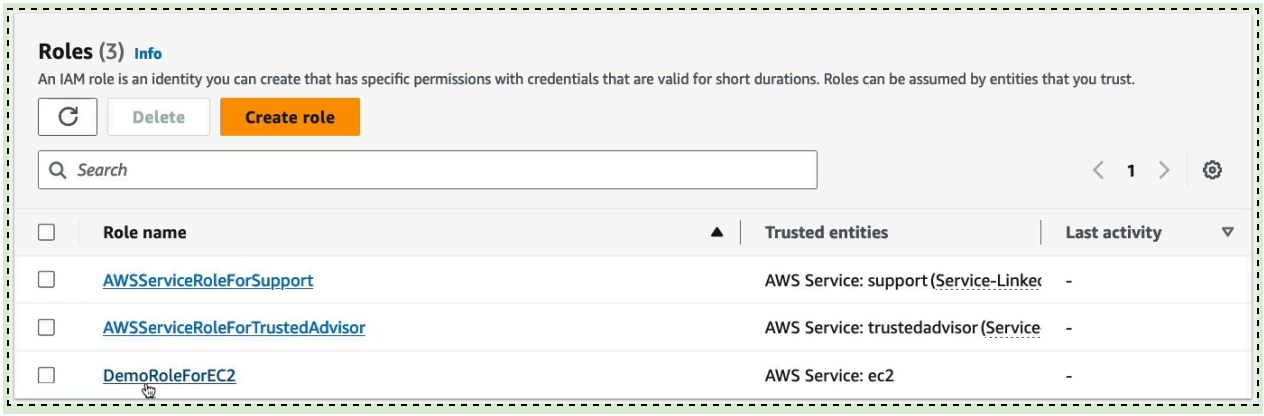


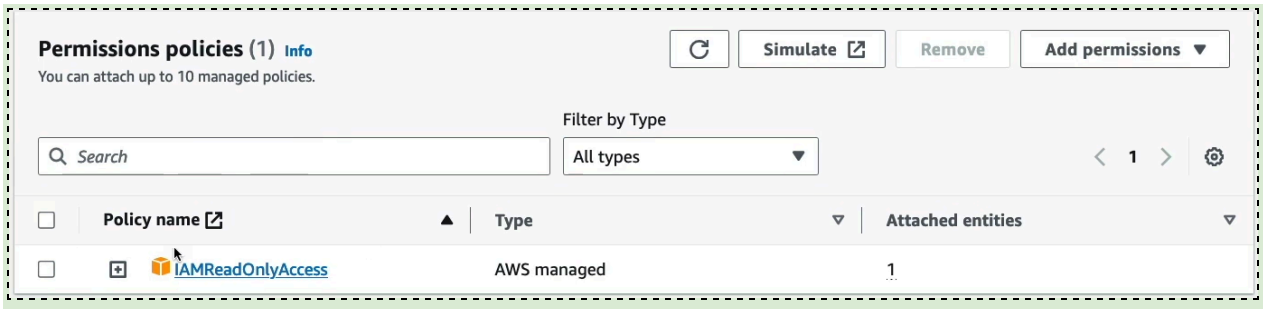
بعدها بتختار ال Service اللى عاوز تديها ال role



بعدها بت attach ال policy وليكن هندي ال EC2 IAMReadOnly .. كدا قادرة ت read اي حاجة فى IAM

بعدها بتحط ال role name , description وبكدا انت كريتت ال role





ولو ضغطت عليها هتلاقي ال permission اللى واخداه اللى هو IAMReamOnly

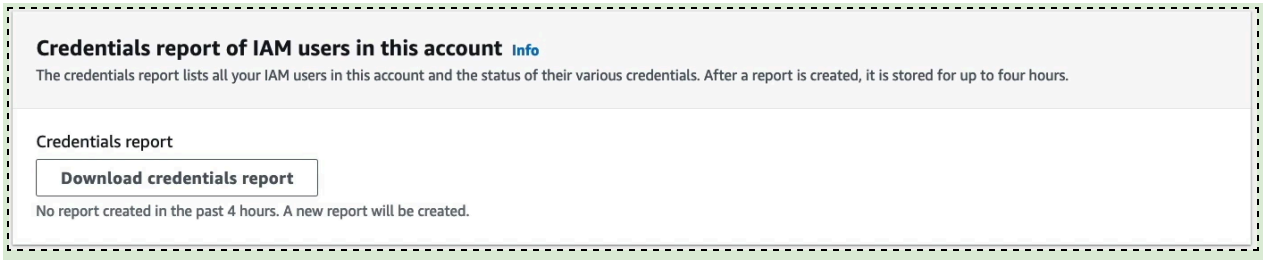
---

➥IAM Security Tools, Hands On:

- **IAM Credentials Report (account-level)**

عباره عن report بتعمله download من خلال الroot بيكون فيه معلومات عن كل ال users

Access Reports >> Credential Report



- **IAM Credentials Report Content**

الاسم

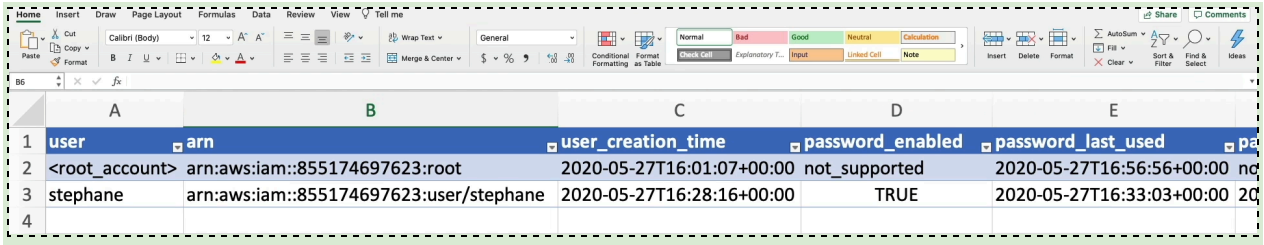
نوع الuser

امتي اتكريت ال user

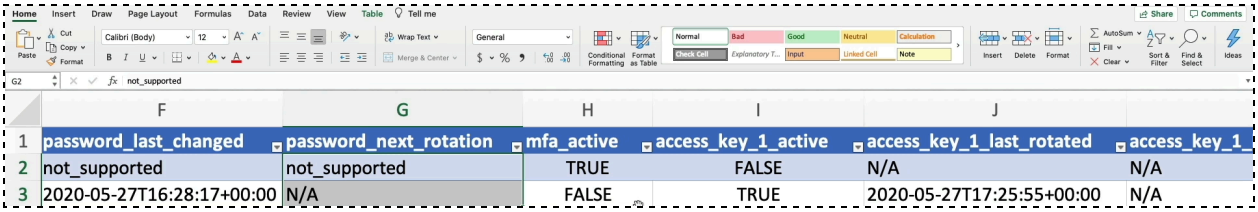
امتي اخر مرة ال password اتغير

هل فيه MFA ولا لا

Password next rotation : امتي الpassword هي expired وامتي المفروض هيتغير



	A	B	C	D	E
1	user	arn	user_creation_time	password_enabled	password_last_used
2	<root_account>	arn:aws:iam::855174697623:root	2020-05-27T16:01:07+00:00	not_supported	2020-05-27T16:56:56+00:00
3	stephane	arn:aws:iam::855174697623:user/stephane	2020-05-27T16:28:16+00:00	TRUE	2020-05-27T16:33:03+00:00
4					



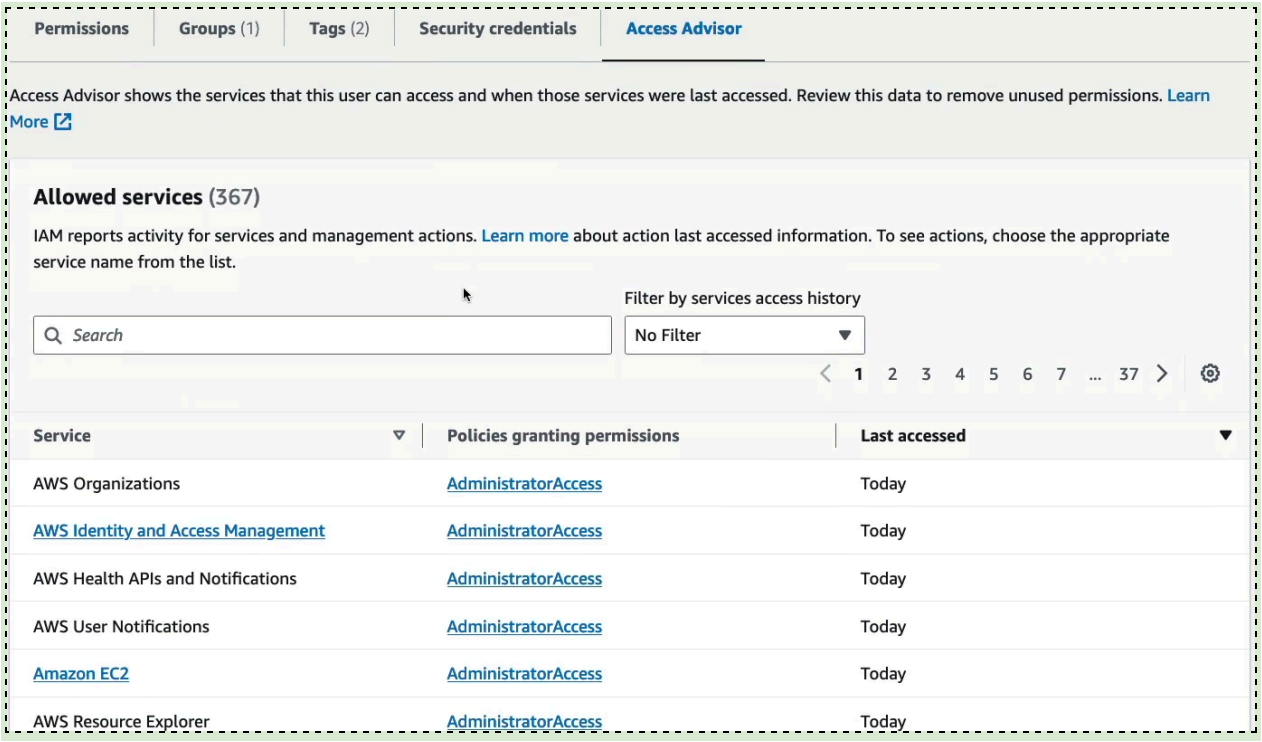
	F	G	H	I	J	
1	password_last_changed	password_next_rotation	mfa_active	access_key_1_active	access_key_1_last_rotated	access_key_1
2	not_supported	not_supported	TRUE	FALSE	N/A	N/A
3	2020-05-27T16:28:17+00:00	N/A	FALSE	TRUE	2020-05-27T17:25:55+00:00	N/A

- **IAM Access Advisor (user-level)**

بيوضحلك اية ال services اللى ال user دة استخدمها وامتي استخدمها وكمان لما استخدمها

كان واخد عليها permission اية .. زي تحت كدا واخد admin access

Access Management >> Users >> Access Advisor



➥IAM Summary:

- **Users**: mapped to a physical user, has a password for AWS Console
- **Groups**: contains users only
- **Policies**: JSON document that outlines permissions for users or groups
- **Roles**: for EC2 instances or AWS services
- **Security**: MFA + Password Policy
- **AWS CLI**: manage your AWS services using the command-line

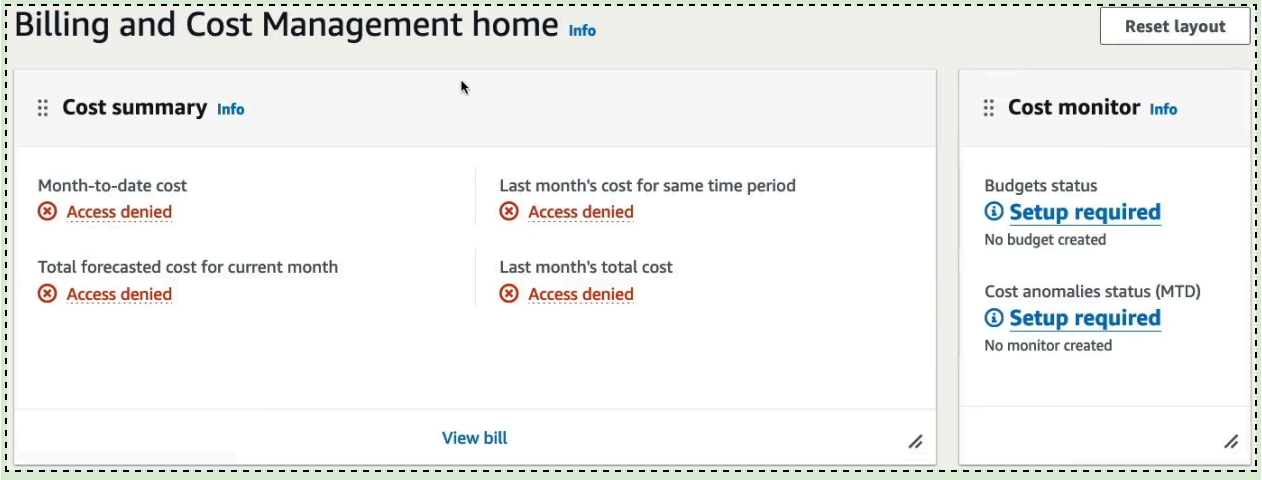
I'm Here

- **AWS SDK**: manage your AWS services using a programming language
- **Access Key**s: access AWS using the CLI or SDK
- **Audit**: IAM Credential Reports & IAM Access Advisor

CH-05 - EC2 Fundamentals

➥AWS Budget Setup:

Your Name >> Billing And Cost Management

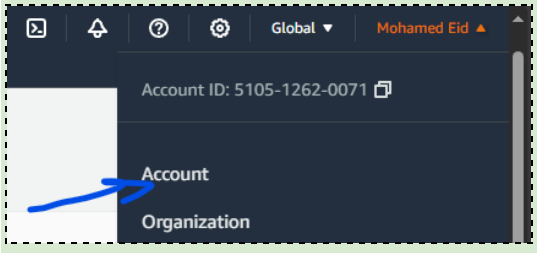


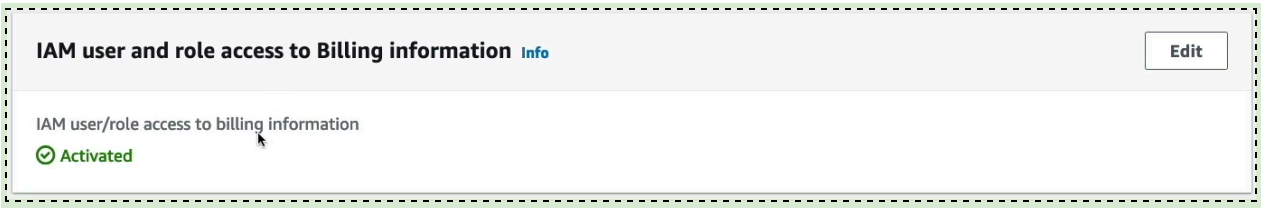
- هتلاقي انك معندكش Access عال Billing & Cost لانك فاتح من user عادي رغم انك واخد Administrator Access .. ف علشان نحل المشكلة دي لازم نفعل ال access دة من ال root user

---

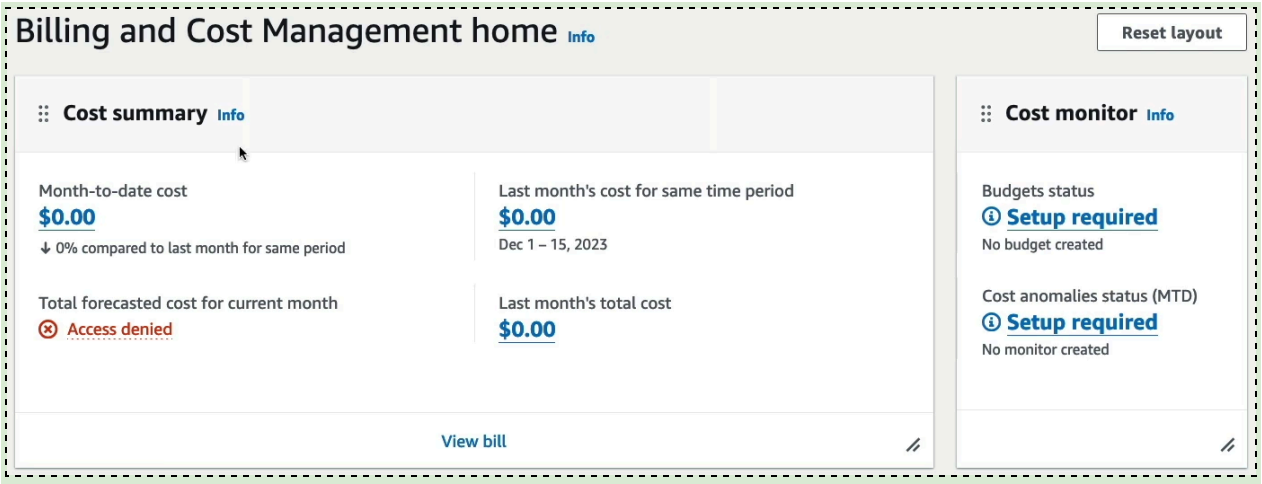
I'm Here

Open the root user >> Account

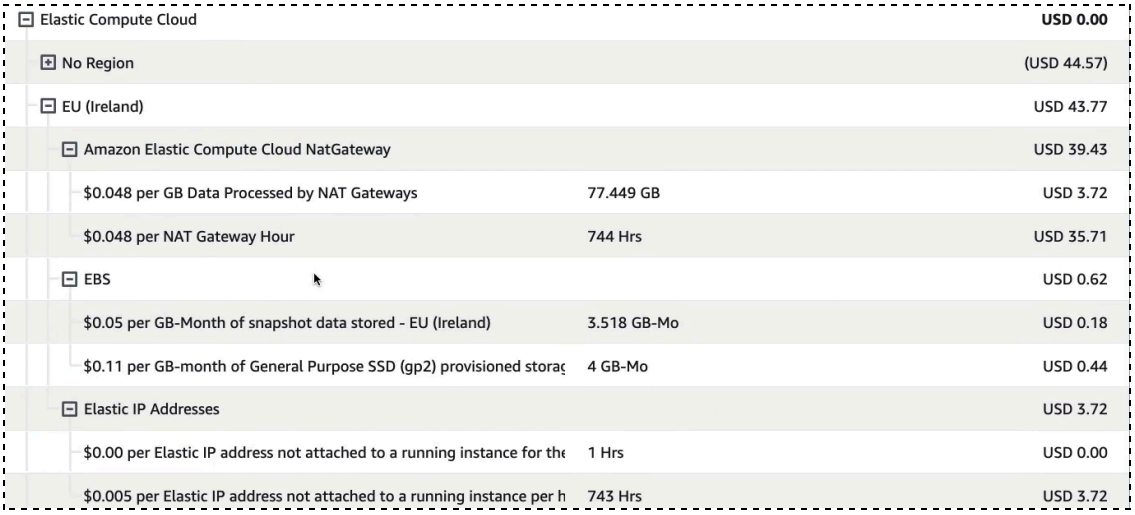




- وبعد activate ان ال user يقدر ي access ال billing & cost تقدر عادي تتحكم فيهم من خلال ال regular user



Billing And Cost Management >> Billing and Payments >> Bills >> Charges By services



Elastic Compute Cloud		**USD 0.00**
No Region		(USD 44.57)
EU (Ireland)		USD 43.77
Amazon Elastic Compute Cloud NatGateway		USD 39.43
$0.048 per GB Data Processed by NAT Gateways	77.449 GB	USD 3.72
$0.048 per NAT Gateway Hour	744 Hrs	USD 35.71
EBS		USD 0.62
$0.05 per GB-Month of snapshot data stored - EU (Ireland)	3.518 GB-Mo	USD 0.18
$0.11 per GB-month of General Purpose SSD (gp2) provisioned storag	4 GB-Mo	USD 0.44
Elastic IP Addresses		USD 3.72
$0.00 per Elastic IP address not attached to a running instance for the	1 Hrs	USD 0.00
$0.005 per Elastic IP address not attached to a running instance per h	743 Hrs	USD 3.72

- من خلالها تقدر تشوف كل ال Active Services وكل Service انت اشتغلت عليها كلفتك فلوس قد اية

Billing And Cost Management >> Cost Analysis >> Free tier

AWS Free Tier (43) Info

Find service name

1 2 3 4 5

Service	AWS Free Tier usage limit	Current usage	Forecasted usage	MTD actual usa
AWS Lambda	400000.0 seconds are always free per month as part of AWS Free Usage Tier (Global-Lambda-GB-Second)	400,000 seconds	826,667 seconds	
Amazon Simple Queue Service	1000000.0 Requests are always free per month as part of AWS Free Usage Tier (Global-Requests)	1,000,000 Requests	2,066,667 Requests	
AWS X-Ray	100000.0 Traces are always free per month as part of AWS Free Usage Tier (Global-XRay-TracesStored)	100,000 Traces	206,667 Traces	
AmazonCloudWatch	10.0 Alarms are always free per month as part of AWS Free Usage Tier (Global-CW:AlarmMonitorUsage)	10 Alarms	21 Alarms	
AWS Lambda	1000000.0 Request are always free per month as part of AWS Free Usage Tier (Global-Request)	646,310 Request	1,335,707 Request	

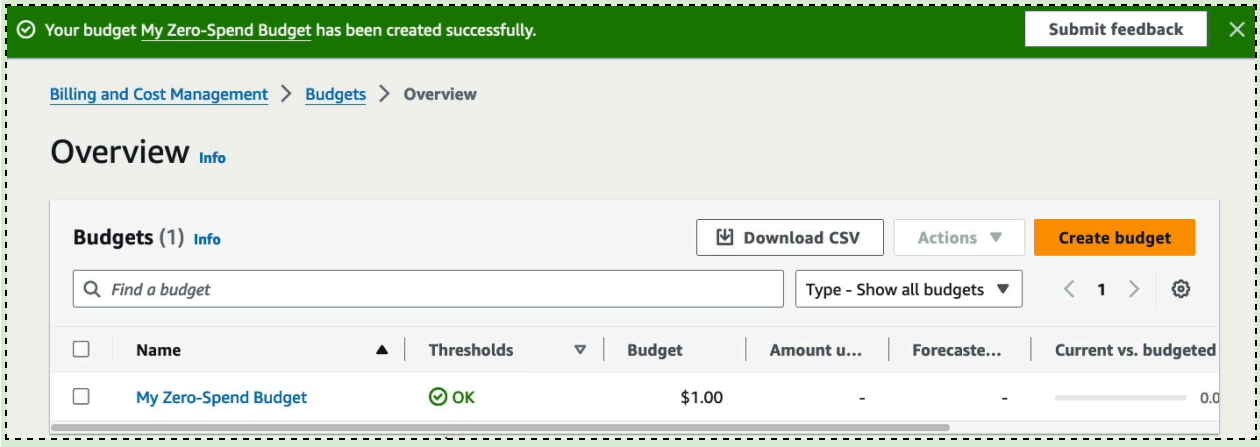
- من خلالها تقدر تتابع استخدامك الحالي والمتوقع لو فاتح free tier acc

Billing And Cost Management >> Budget And Planning >> Budgets

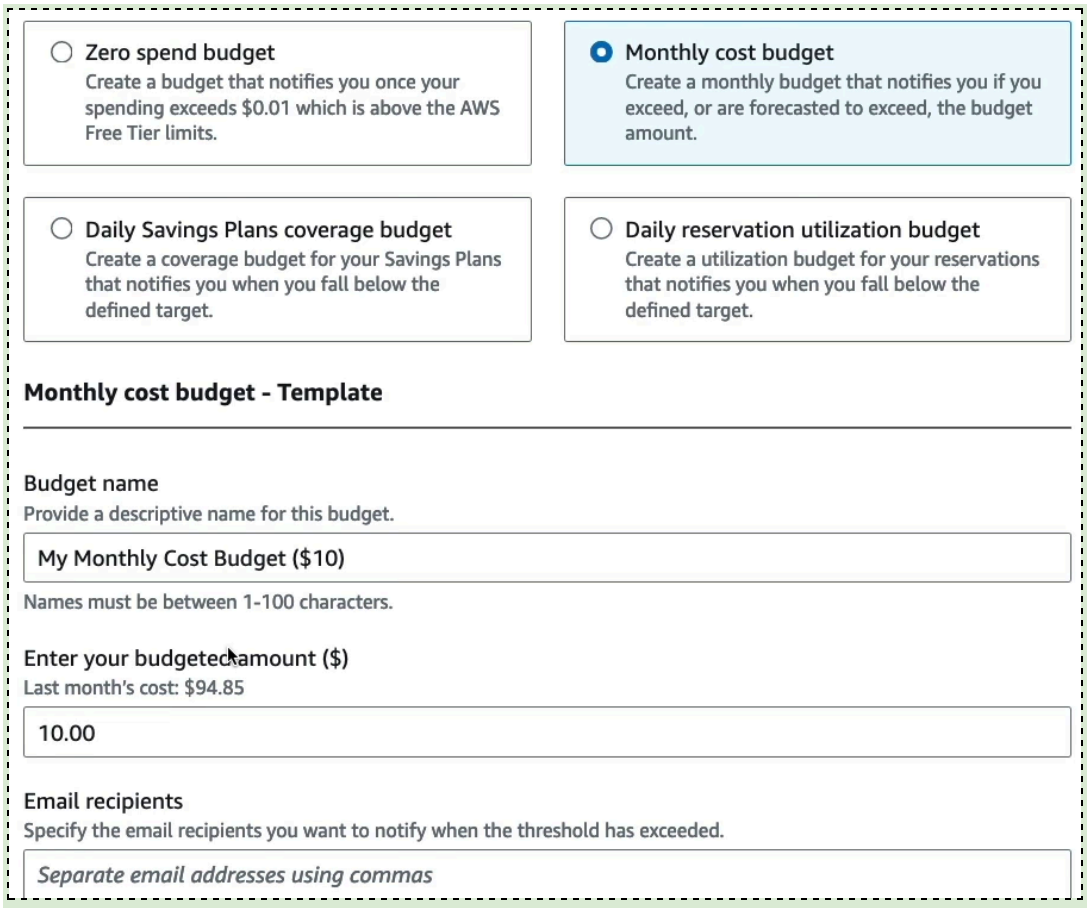
Templates - *new*

Choose a template that best matches your use case.

- **Zero spend budget** — Create a budget that notifies you once your spending exceeds $0.01 which is above the AWS Free Tier limits.
- **Monthly cost budget** — Create a monthly budget that notifies you if you exceed, or are forecasted to exceed, the budget amount.
- **Daily Savings Plans coverage budget** — Create a coverage budget for your Savings Plans that notifies you when you fall below the defined target.
- **Daily reservation utilization budget** — Create a utilization budget for your reservations that notifies you when you fall below the defined target.



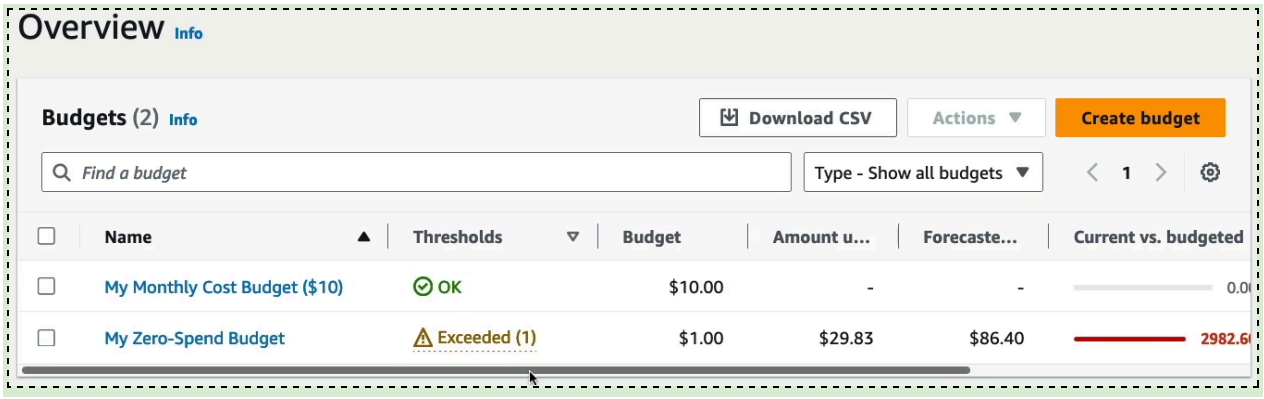
- من خلالها تقدر تكريت Badget setup ي notificate you لو تعديت $0.01 ك free tier account .. وبتكتب ال email بتاعك اللى هيبعتلك عليه ال alert دة.



- بالنسبة لل monthly بتحددله انت عاوز تدفع كام هنا $10 خلال الشهر.

You will be notified when 1) your **actual spend** reaches 85% 2) your **actual spend** reaches 100% 3) if your **forecasted spend** is expected to reach 100%.

💡 هيبعتلك mail لو صرفت 85% من ال 10$ ولو وصلت 100% او لو المتوقع توصل 100%



- لما بتعدي ال budget بتتحول للشكل دة

➥EC2 Basics:

- **EC2** = Elastic Compute Cloud = Infrastructure as a Service

- **عباره عن service من خلالها تقدر :**
  - تأجر virtual machines << **EC2**
  - تخزن Data علي Virtual Drives << **EBS**
  - توزع ال load عال machines من خلال **ELB**
  - ت Scale ال Services من خلال auto-scaling group << **ASG**

- **EC2 sizing & configuration options**

- Operating System (OS): Linux, Windows or Mac OS
- How much compute power & cores (CPU)
- How much random-access memory (RAM)
- How much storage space
- Network-attached (EBS & EFS)
- hardware (EC2 Instance Store)
- Network card: speed of the card, Public IP address
- Firewall rules: security group
- Bootstrap script (configure at first launch): EC2 User Data

---

- **User Data** : عباره عن mechanism فى AWS من خلالة ب pass configurations or commands لل EC2 Instance وهيا بتـ launch فى الغالب بتكون فى شكل shell script
- **Bootstrapping**: عبارة عن process معناها انك ت Run Commands لما ال machine ت Start .. ال Script دة بي run مرة واحدة بس لما ال machine ت start

---

💡 يبقي ال User data بحط فيه ال script بتاعي اللى بيتنفذ من خلال ال bootstrapping لما ال machine ت start

User data **Info**

```
#!/bin/bash
# Use this for your user data (script from top to bottom)
# install httpd (Linux 2 version)
yum update -y
yum install -y httpd
systemctl start httpd
systemctl enable httpd
echo "<h1>Hello World from $(hostname -f)</h1>" > /var/www/html/index.html
```

- **ال Script دة بيكون عباره عن :**

- Install updates , softwares
- Download something from the internet or any thing you can think of

---

💡كل ما كان ال Script اللى هيتنفذ اثناء ال boot time اكبر كل ما ال instance هتاخد وقت علي ما تـ launch

💡ال Script دة بيتنفذ من خلال ال root user ياعني فى ال script هتشتغل as sudo

---

- عندك انواع كتير من ال EC2 Instances دول امثله ليهم.
- ال t2.micro اللى هنشتغل عليه وفى ال free tier بيتيحلك 750 hour / month

---

➥Create An EC2 Instance with EC2 User Data to have a website Hands On:

هيكريت Instance زي ما عملنا فى ال cloud practitioner و install web server عليها من خلال ال user data

💡لو عملت Stop لل Instance وجيت شغلتها تاني هتلاقي ال IPV4 اتغير ف خلي بالك.

➥EC2 Instances Types:

- هتلاقي كل الانواع وكل اللى محتاج تعرفه عنهم هنا

PAGE CONTENT

- General Purpose
- Compute Optimized
- Memory Optimized
- Accelerated Computing
- Storage Optimized
- HPC Optimized
- Instance Features
- Measuring Instance Performance

M7g | M7i | M7i-flex | M7a | Mac | M6g | M6i | M6in | M6a | M5 | M5n | M5zn | M5a | M4 | T4g | T3 | T3a | T2

Amazon EC2 M5 instances are the latest generation of General Purpose Instances powered by Intel Xeon® Platinum 8175M or 8259CL processors. These instances provide a balance of compute, memory, and network resources, and is a good choice for many applications.

Features:

- Up to 3.1 GHz Intel Xeon Scalable processor (Skylake 8175M or Cascade Lake 8259CL) with new Intel Advanced Vector Extension (AVX-512) instruction set
- New larger instance size, m5.24xlarge, offering 96 vCPUs and 384 GiB of memory
- Up to 25 Gbps network bandwidth using Enhanced Networking
- Requires HVM AMIs that include drivers for ENA and NVMe
- Powered by the AWS Nitro System, a combination of dedicated hardware and lightweight hypervisor
- Instance storage offered via EBS or NVMe SSDs that are physically attached to the host server
- With M5d instances, local NVMe-based SSDs are physically connected to the host server and provide block-level storage that is coupled to the lifetime of the M5 instance
- New 8xlarge and 16xlarge sizes now available.

Instance Size	vCPU	Memory (GiB)	Instance Storage (GB)	Network Bandwidth (Gbps)***	EBS Bandwidth (Mbps)
m5.large	2	8	EBS-Only	Up to 10	Up to 4,750
m5.xlarge	4	16	EBS-Only	Up to 10	Up to 4,750

- **m5.2xlarge**

- **m** : تشير إلي ال Class

- **5** : تيشر إلى ال generation .. هتلاقي AWS بتطور ال hardware ومع كل تطوير بتقدم generations جديدة ف هتلاقي m5 , m6 m7
- **2xlarge** : بتيشر إلى ال size هتلاقي عندك 12x , 8x , 4x وهكذا

💡 Size اكبر ياعني CPU , Memory اكبر وهكذا.

- **هتلاقي ان AWS مقسمالك الأنواع دي على حسب غرضك:**

- **General Purpose :** مناسب لل work loads زي ال load balancers وال code repository وفيه balance بين ال CPU , Memory , Networking

💡 النوع اللى هتستخدمه اللى هو T2 من ال General purpose

- **Compute Optimized** : بستخدمها لل Applications اللى بتحتاج High performance processing زي ال media transcoding
- **Memory Optimized** : بستخدمها لل Applications اللى بتحتاج High performance ف ال Memory
- **Storage Optimized** : بستخدمها لل Applications اللى بتحتاج High performance ف ال Storage
- **Accelerated Computing** : بستخدمها لو محتاج اعمل machine learning او complex calculations او graphic workloads

---

- **مقارنة بسيطة:**

Instance	vCPU	Mem (GiB)	Storage	Network Performance	EBS Bandwidth (Mbps)
t2.micro	1	1	EBS-Only	Low to Moderate	
t2.xlarge	4	16	EBS-Only	Moderate	
c5d.4xlarge	16	32	1 x 400 NVMe SSD	Up to 10 Gbps	4,750
r5.16xlarge	64	512	EBS Only	20 Gbps	13,600
m5.8xlarge	32	128	EBS Only	10 Gbps	6,800

t2.micro is part of the AWS free tier (up to 750 hours per month)

- ودة Website بيقارن بين كل الانواع والاسعار

➥Security Group & Classic Ports Overview

- أكنو Virtual Firewall بيكون على ال EC2 instances بتاعتك وبي control ال inbound and outbound traffic من خلال rules انت بتحددها بتخليك تتحكم ف ال protocols,ports and source/destination ip وهكذا
- هتلاقي ان **by default** كل ال **outbound** traffic are **allowed** ياعني تقدر تكلم اي حد بره
- هتلاقي ان **by default** كل ال **inbound** traffic are **denied** وف الحالة دي بتحدد ازاي هتتعامل مع ال inbound roles من خلال ال rules اللى بتحطها وليكن مثلا:

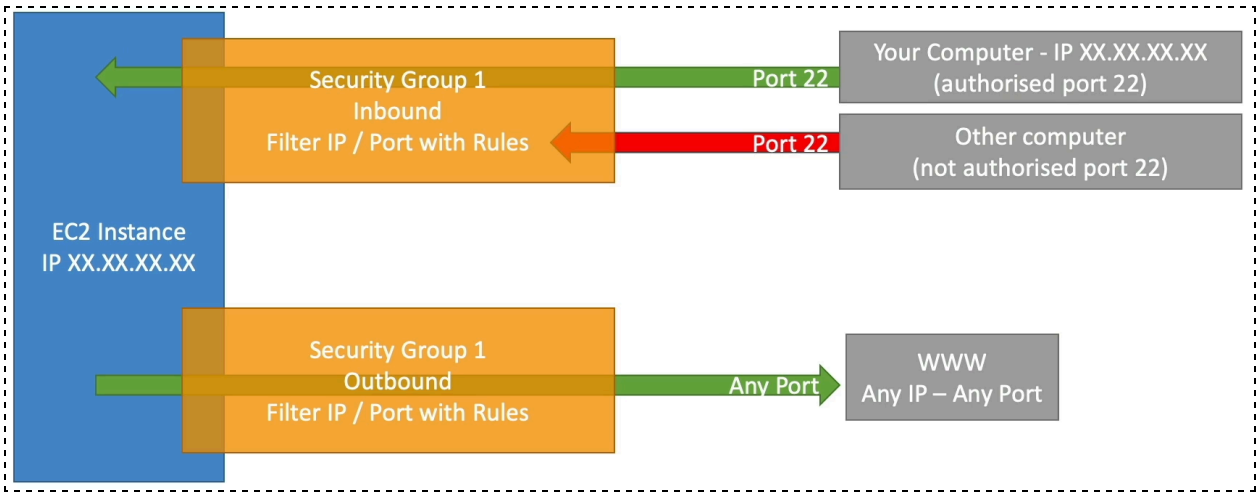
- **Example:**

- حاطط rules بتسمح ب http traffic اللى جاي علي port 80
- حاطط rules بتسمح ب https traffic اللى جاي علي port 443

- حاطط rules بتسمح ب ssh traffic اللى جاي علي port 22 من ip معين فقط وفى الغالب بيكون root user ip
- ممكن ت define اكتر من Security Group ل instance 1
- ال Security Group بتشتغل على ال EC2 instance level مش على ال subnet level , اللي بيشتغل على ال subnet هو ال network access control list

Control the inbound & outbound traffic by:

- IPV4 - IPV6
- Ports
- Inbound & Outbound Network

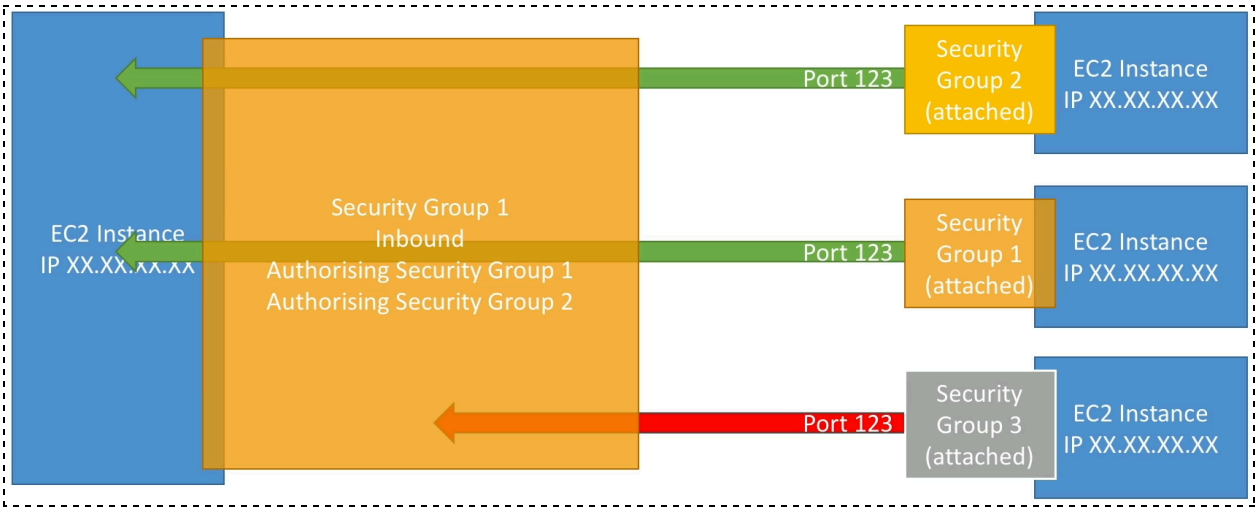


💡يفضل فصل ال SSH فى Security group لوحدها.

💡لو بتـ Access ال App بتاعك وفيه time out او wait كتير ف 100% السبب ال Security group

ولو Connection Refused Error فغالبا دي مشكلة Application or launching

• Important Example:



- فى ال Security Group اللى عالشمال انا سامح فيه بال inbound traffic من ال Security Group اللى علي اليمين اللى هما 1.2 بالتالي اي instances جواهم هيبقي مسموح بال traffic اللى جاي منها على ال instance اللى عالشمال

💡 انك تسمح بال inbound traffic اللى جاي من ال Security Group يغني عن انك تسمح بكل instance جواه مش هتفضل تكتب ip كل instance انت بتشوف مين ال security group اللى لاممهم وتسمح بيه وخلاص .. ودة مناسب فال auto scaling علشان لو فيه instances بتزيد او بتنقص بردو اكيد مش هتفضل تعدل ال inbound rules عال Security Group اللى عالشمال

💡 انك ت Authorized Security Group عند Security Group تاني بيتم من خلال ال ID بتاعه.

• Classic Ports:

- **22** = **SSH** (Secure Shell) - log into a Linux instance

- **3389** = **RDP** (Remote Desktop Protocol) – log into a Windows instance
- 21 = FTP (File Transfer Protocol) – upload files into a file share
- 22 = SFTP (Secure File Transfer Protocol) – upload files using SSH
- 80 = HTTP – access unsecured websites
- 443 = HTTPS – access secured websites

➥Security Groups Hands On:

💡فى حالة ال Security Group لو فتحت ال inbound traffic اللى جاي علي port 80

تلقائي ال outbound من port 80 هيكون allowed

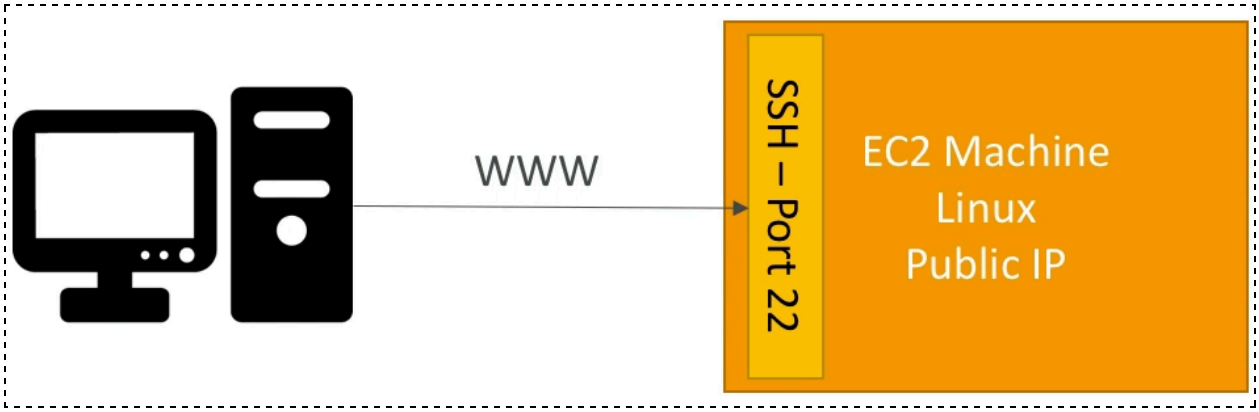
طبعا دة علي عكس ال iptables مثلا لازم تحط للاتنين rules.

➥SSH Overview:

	SSH	Putty	EC2 Instance Connect
Mac	✓		✓
Linux	✓		✓
Windows < 10		✓	✓
Windows >= 10	✓	✓	✓

- اقل من windows 10 مينفعش معاها ssh
- **Putty** ل windows فى العموم
- هتتشرح قدام EC2 Instance Connect

➥How To SSH Using Linux Or Mac



- بتفتح ال Port 22 فى ال security group اللى بيحوي ال EC2 instance بتاعتك وبت ssh عال public ipv4 بتاع ال instance .. ولكن انت اكيد مش بتكون فاتح ال ssh لاي حد ي access ال instance ف لازم يا تكون محدد ال ip اللى هي access او عن طريق key

```
Ssh -i key.pem ec2-user@ipv4
```

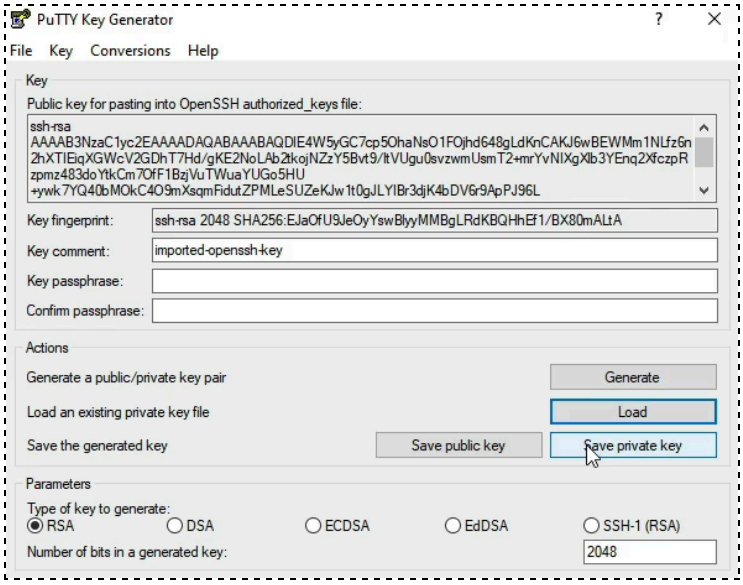
- **Ec2-user** دة ال user اللى معمول عال instance واللى لما تخش هتخش بيه
- **Ipv4** دة بتاع ال instance

```
~/aws-course ssh -i EC2Tutorial.pem ec2-user@3.250.26.200
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0644 for 'EC2Tutorial.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0644 for 'EC2Tutorial.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "EC2Tutorial.pem": bad permissions
Received disconnect from 3.250.26.200 port 22:2: Too many authentication failures
Disconnected from 3.250.26.200 port 22
✘ ~/aws-course chmod 0400
```

- ال error دة الحل بتاعه ف انك تدي الملف read permission لل owner فقط.

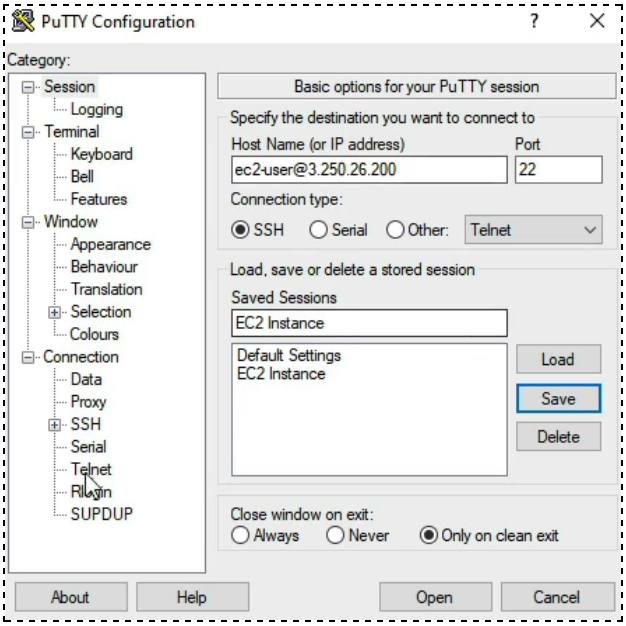
➥How To SSH Using Windows7.8:

1) هتحمل من جوجل putty
2) بعد ما تسطبه هتلاقي عندك ليه جزأين puttygen خاص بال key وال putty ذات نفسه

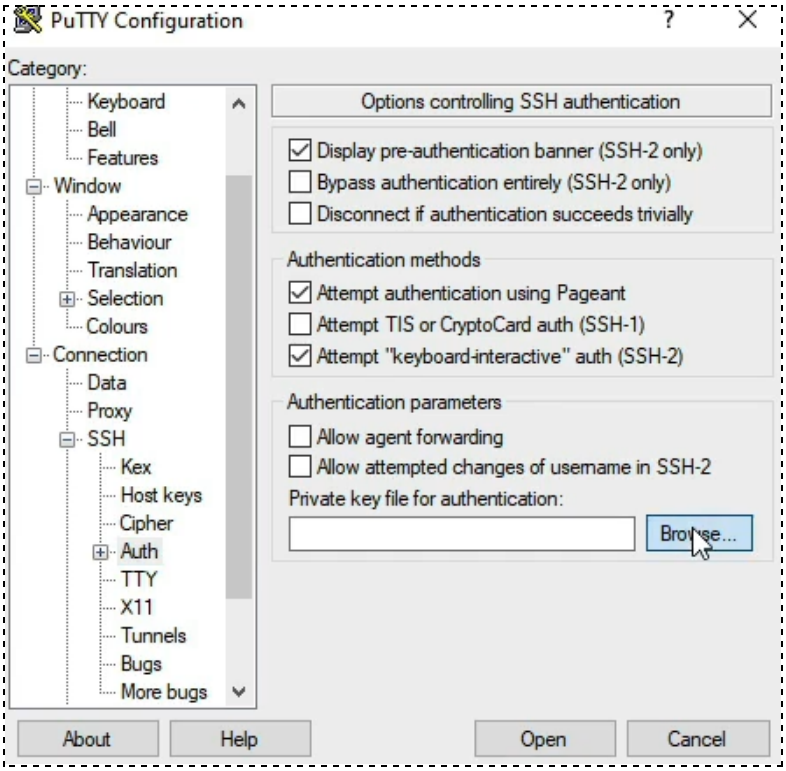


- هت load ال key.pem وتحوله ل key.ppk او لو حملته من الاول بصيغة ppk ف تمام ياعني .. وت save private key

3) هتملي الداتا دي :



4) Ssh ثم auth وبعدين هت load ال private key اللى هو .ppk



5) بت save اللى عملته دة فى الصفحة الاولي بحيث متعملش الخطوات دي كل ما تحتاج تخش عال instance

---

I'm Here

➥How To SSH Using Windows10:

- ممكن من خلال putty عادي

```
Ssh -i key.pem ec2-user@ipv4
```

- لو جالك error هيبقي نفس ال error اللى زي اللى فوق علشان ال owner ال file

وهتغيرو من خلال :

```
Right click on the file >> properties >> security >> Advanced >>
Change >> choose your self.
```

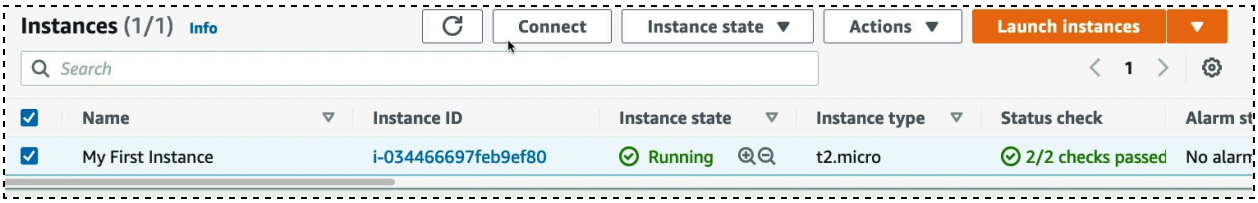
```
PS C:\Users\stephanemaarek\Desktop> ssh -i .\EC2Tutorial.pem ec2-user@3.250.26.200
The authenticity of host '3.250.26.200 (3.250.26.200)' can't be established.
ECDSA key fingerprint is SHA256:INQMVOPTzV+fRKl+EeoRtTp+pWI4koar4F8a6QTmPgE.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.250.26.200' (ECDSA) to the list of known hosts.
Last login: Mon May 16 17:28:25 2022 from bl14-117-32.dsl.telepac.pt

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-33-135 ~]$
```

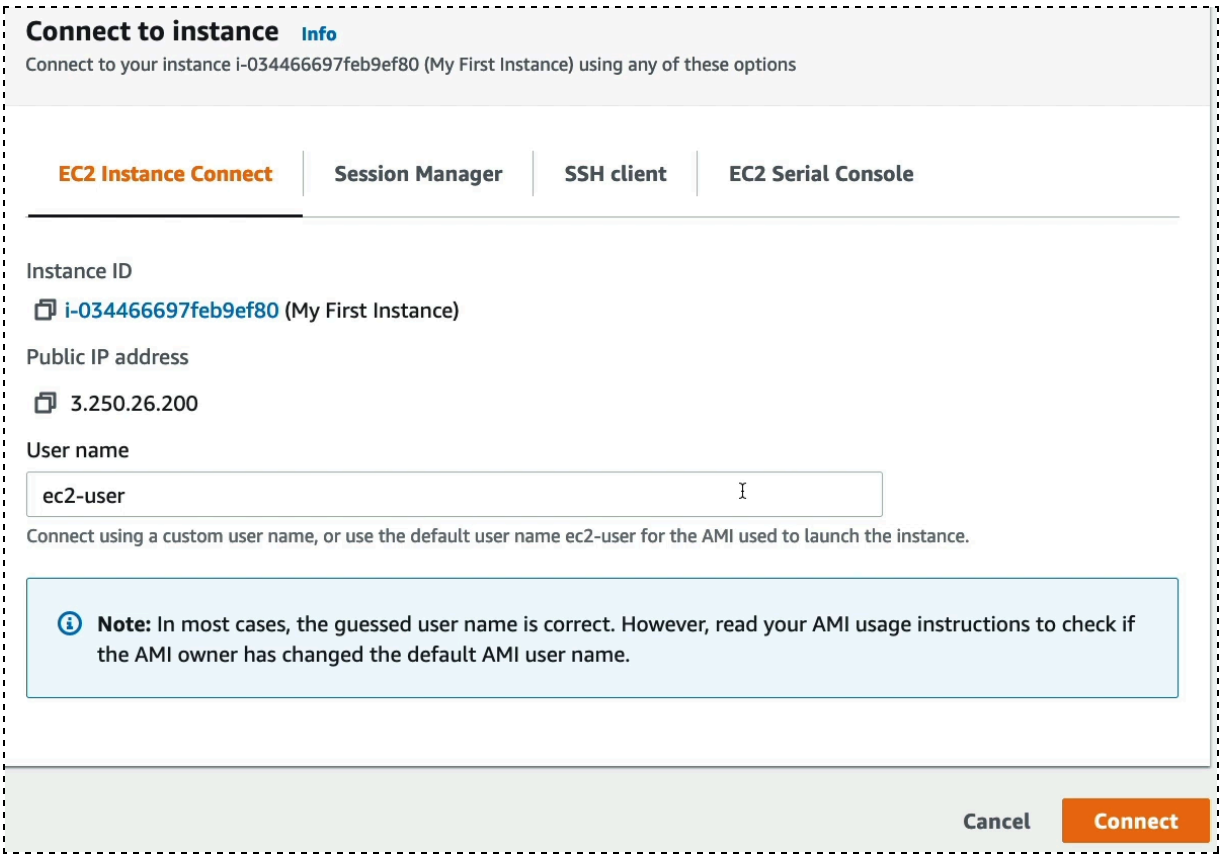
➥SSH Troubleshooting:

- ملف كويس اقرأه

➥EC2 Instance Connect (Easy way)

- هيتم الموضوع من خلال ال browser بجرد ما بتضغط connect ال session بتتفتح عال browser





💡 ممكن حتي لو فتحت ال ssh عال instance من anywhere/ipv4 متعرفش ت connect فى الحاله دي افتح anywhere/ipv6

💡 Any timeout (not just for SSH) is related to security groups or a Firewall

💡 لو عملت Stop لل Instance وجيت شغلتها تاني ممكن تلاقي ال IPV4 اتغير ف خلي بالك..ياعني مش هتعرف تخش لو محدد انك تssh بال ipv4 اللى اتغير

➥EC2 Instance Roles Demo:

- اللى عمله انو access ال instance وحاول ي list ال user ولكن ال instance مكنتش واخده ال role دي .. ف عملنا attach ل readonly role لل instance علشان تقدر ت list ال users

💡 لما تيجي ت attach ال role ممكن تاخد وقت على ما تسمع عال instance ف متستعجلش.

💡 ال instance لما كريتتها ب linux جت بال CLI installed جاهز

```
[ec2-user@ip-172-31-17-250 ~]$ aws --version
aws-cli/1.18.147 Python/2.7.18 Linux/4.14.225-169.362.amzn2.x86_64 botocore/1.18.6
[ec2-user@ip-172-31-17-250 ~]$ aws iam list-users
Unable to locate credentials. You can configure credentials by running "aws configure".
[ec2-user@ip-172-31-17-250 ~]$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]:
Default output format [None]: ^C
[ec2-user@ip-172-31-17-250 ~]$
```

- متعملش configure لل credential بالشكل دة اي حد لو دخل عال instances يقدر ي retrieve ال access key ويخرب الدنيا

➥EC2 Instance Purchasing Options:

- **On-Demand Instances :**
  - ودي pas as you go ف Windows , Linux بندفع بالثانية واي OS تاني بالساعه .. وده اعلى سعر ف ال Options ياعني لو محتاج مثلا سيرفر اسبوع شهر شهرين تلاتة.
  - مفيش فيها upfront payment دفع مقدم ياعني.

💡 ال **usage** : فى ال short-term usage وللل Apps اللى مش توقع ال load بتاعها.

- **Reserved Instances & Convertible Reserved Instances (1&3 year):**
- **Reserved Instances**
  - وهنا AWS بتقلل ال Cost مقابل انك تلتزم ب Configuration ثابته لل Instances بتاعتك زي ال region , type.
  - فيها upfront payment دفع مقدم .. ممكن متدفعش وممكن تدفع جزأ ممكن تدفع كل الحساب وطبعا كل ما دفعت اكتر كل ما هتاخد خصم اكتر.

- بيبقي فيها تقريبا discount %70 عن ال on demand وتقدر تبيع ال instances دى فى ال market place لو مبقتش محتاجها
- سنة او 3 سنين مش سنة الي 3 سنين

💡 ال **usage** : فى ال steady state usage applications زي ال database

- **Convertible Reserved Instances**

- نوع من انواع ال Reserved Instances ميزته انك تقدر تغير ال EC2 instance type, instance family, OS, scope and tenancy
- بيبقي عليه discount بيوصل ل 66%

- **Saving Plans (1&3 year):**

- بيبقي فيها تقريبا discount %70 زي ال reserved
- بتلتزم فيها بالاستخدام مثلا (10 دولار/ساعة لمدة 1 أو 3 سنوات) ولو استخدامك زاد عن اللى انت محددة هتدفع بال on-demand
- Locked لل instance family & region ياعني لو حددت M5 على us-east-1 مش هتعرف تغيره

- **Spot Instances:**

- ودي سعرها بيكون مخفض لنسبة قد تصل ل 90% من سعر ال On-Demand Instances ودي يعتبر اقل سعر ف ال options المتاحة. ف ال option دة انت بتستخدم ال unused EC2 Instances بسعر مخفض لكن ممكن تخسر ال instances بتاعك ف اي وقت لو سعر ال spot instance تخطي ال max price بتاعك انت ك customer ودة طبعا مش مناسب لل critical services.

💡 ال **usage** : مش مناسب لل critical jobs ومناسب للآتي :

Batch jobs , Data analysis , Image processing , Any distributed , workloads , Workloads with a flexible start and end time

---

- **Dedicated Hosts & Dedicated Instances:**
  - الاتنين متقاربين ال **hosts** بتاخد physical server وبيكون ليك full control عال hardware اما ال **instances** بيكون ليك control اقل ف بتختارها لما متكونش محتاج detailed hardware control
  - علشان اللغبطة من الاسم Dedicated Host ياعني السيرفر ملكك .. Dedicated Instances ياعني ال instances ملكك على hardware ملكك
  - **الدفع بيكون حاجة من الاتنين :**
    - **On-demand**
      - pay per second for active Dedicated Host
    - **Reserved**
      - 1 or 3 years (No Upfront, Partial Upfront, All Upfront)

Comparison Summary

Feature	Dedicated Hosts	Dedicated Instances
Control	Full control over instance placement on specific physical servers	AWS manages instance placement
Hardware Visibility	Full visibility into physical hardware (sockets, cores)	No visibility into physical hardware
Licensing	Ideal for BYOL scenarios requiring specific hardware licensing	Less suitable for complex licensing
Isolation	Dedicated hardware for a single customer	Dedicated hardware for a single customer
Management	Requires more management effort	Simpler, less management overhead
Billing	Billed per host	Billed per instance
Use Cases	Regulatory compliance, detailed hardware control, BYOL	Physical isolation without hardware control

- **Capacity Reservation:**
  - ال capacity هنا معناها عدد ال EC2 Instance من type معين فى region او AZ معينه
  - وهنا AWS بتعملك Reservation ل on-demand EC2 Instance Capacity ف Availability zone معينه لمدة بيتم الاتفاق عليها.
  - مفيش time commitment التزام بالوقت .. ابدا وكنسل فى اي وقت براحتك
  - مفيش discount ولكن ممكن تدمجها مع ال reserved instances و ال saving plans بحيث تاخد خصم

💡 هتتحاسب اكنك on-demand وال instance شغالة او مش شغالة هتتحاسب عليها.

ال Usage : هتكون مناسبة لل short-term uninterrupted workloads فى AZ معينه.

- **Scheduled Instances:**
  - ودي من اسمها تقدر من خلالها انك تشغل ال Instances بتاعتك بجدولة ياعني خلال عدد ساعات معينه مثلا ف اليوم او الاسبوع او الشهر ودي بتكون لمدة سنة.

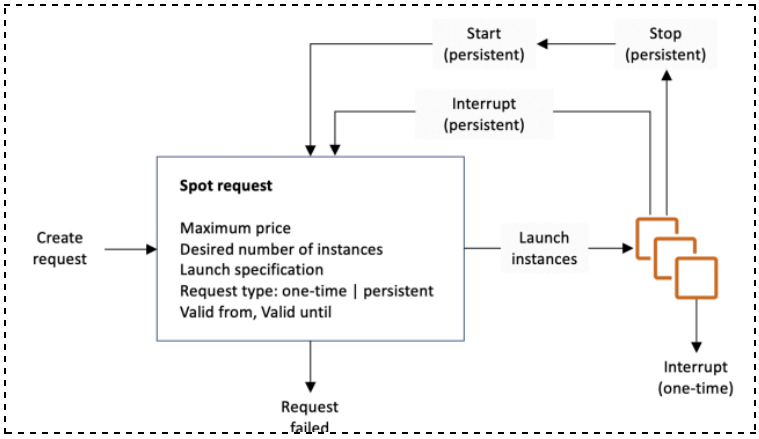
- **Price Comparison - Example – m4.large – us-east-1**

Price Type	Price (per hour)
On-Demand	$0.10
Spot Instance (Spot Price)	$0.038 - $0.039 (up to 61% off)
Reserved Instance (1 year)	$0.062 (No Upfront) - $0.058 (All Upfront)
Reserved Instance (3 years)	$0.043 (No Upfront) - $0.037 (All Upfront)
EC2 Savings Plan (1 year)	$0.062 (No Upfront) - $0.058 (All Upfront)
Reserved **Convertible** Instance (1 year)	$0.071 (No Upfront) - $0.066 (All Upfront)
Dedicated Host	On-Demand Price
Dedicated Host Reservation	Up to 70% off
Capacity Reservations	On-Demand Price

➥Spot Instances & Spot Fleet:

- لو السعر تخطي السعر اقصي سعر انت هتدفعو قدامك دقيقتين تقرر فيهم هل هتعمل stop او terminate للـ instance .. لو هتعمل stop لما الـ spot price ينزل عن الـ max price تقدر تعملها start من تاني.

● How Spot Request Work :



- بتحدد كام عدد الـ instances و الـ max price اللى هتدفعو والـ launch specifications

---

● How to terminate spot instances?

- عندك نوعين من الـ request :

● One-time request

- بمجرد ما بتعمل الـ request الـ instances بت launch ولو حصلها interrupt or termination مش هتقوم تاني الا لما انت تعمل request جديد **دة مناسب** لل work loads اللى بتحتاج one single run او مش محتاجة continuous execution

- **Example**

You have a data processing job that you run periodically. You submit a one-time Spot Instance request to complete this job. Once the job is finished, or if the instance is terminated, no further action is taken automatically.

- **Persistent request**
  - Persistent request بيحاول دايما يحافظ على عدد ال instances لو اي instance حصلها interrupt او termination علطول بي launch واحدة جديدة كبديل **ودة مناسب** لل work loads اللى بتحتاج continuous execution or high availability
  - **Example**

You have a long-running data analysis application that needs to run continuously. You submit a persistent Spot Instance request for 3 instances. If any of these instances are interrupted, AWS will automatically attempt to launch new instances to maintain the count of 3.

💡 يبقي فى النوع دة لو محتاج ت terminate ال instances خالص لازم الاول توقف ال persistent request لانك لو موقفتوش هيحصل relaunch لل instances بحيث يحافظ على عددها اللى انت محددهوله.

- **Spot Fleet:**

- **Example** : لو عندك web application وعاوز تهندل ال variable traffic اللى بيجيلك خلال اليوم وتحافظ على انك تدفع اقل تكلفة وانو يكون available علطول هتعمل الآتي :

- **Define Target Capacity :**

- بت define عدد ال instances اللى عاوزها وال cpu وهكذا

- **Choose the allocation strategy:**

  - **Lowest price**

- ودي لو محتاج اقل cost

  - **Diversified or Capacity Optimized**

- لو محتاج Availability عالية وأنك تقلل the risk of interruption

  - **priceCapacityOptimized (recommended) :**

- ميكس بين الاتنين high capacity Available و lowest price

- **Choose instances types and AZs**

- هت list ال types وال AZs اللى ال application بتاعك can run on it

- **Mix Purchase Options**

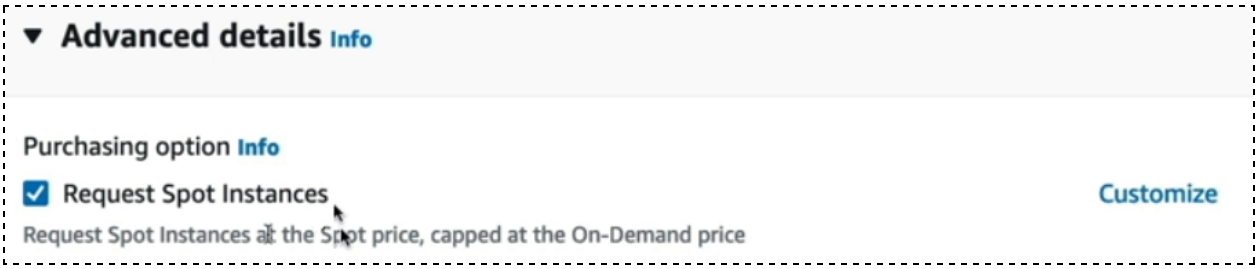
- Optionally include a mix of Spot and On-Demand Instances to balance cost and availability.

- **Launch Spot Fleet**

➥EC2 Instance launch types hands on:

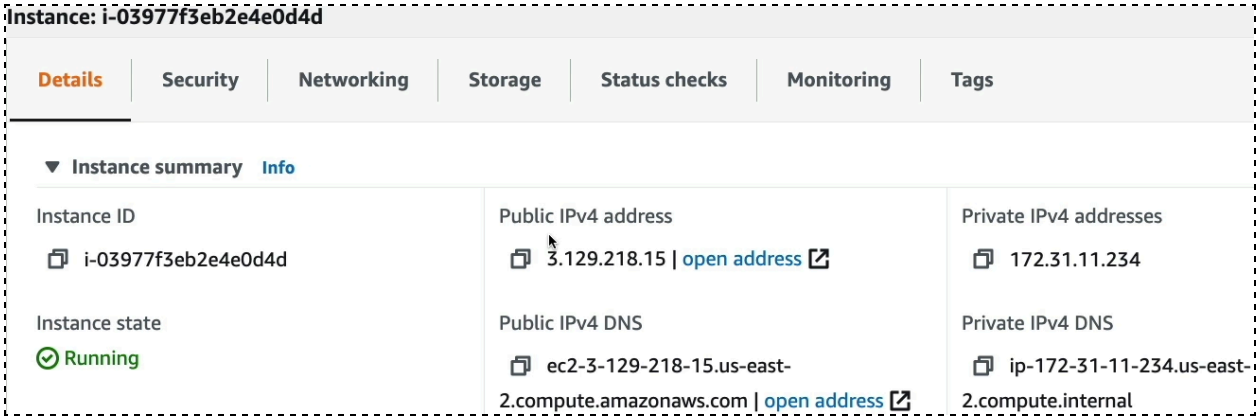
Instances >> spot request >> request spot instances

ممكن من هنا او وانت بت launch ال instance عادي هتلاقي دي ..



CH-06 - EC2 - Solutions Architect Associate Level

➥001 Private vs Public vs Elastic IP:



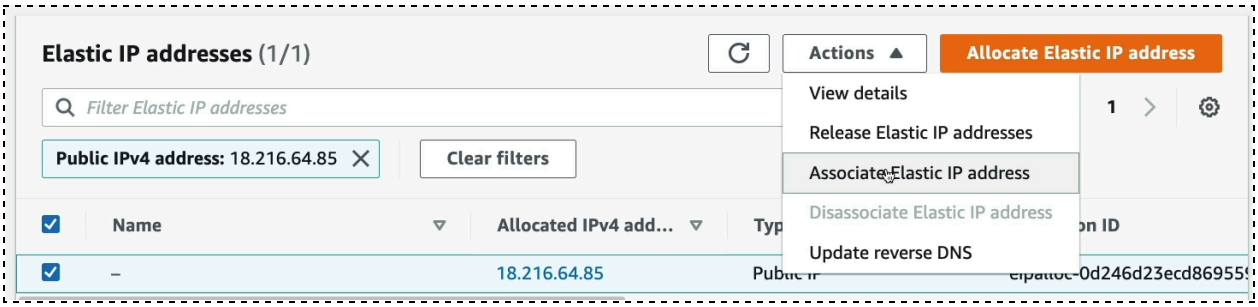
- ال instance اللى انت بتأجرها بيكون ليها public & private ip
- لما بت ssh عال instance بنستخدم ال public ip مينفعش ال private لاننا مش فى same network ال
- ال instance لو عملتلها stop وبعدين start ال public ip دة بيتغير

➥002 Private vs Public vs Elastic IP Hands On

● Create Elastic IP:

Network & Security >> Elastic IPs >> Allocate Elastic IP Address

بيكون ليه تكلفه خاصه لواحدة بحوالي 0.005$ فى الساعه

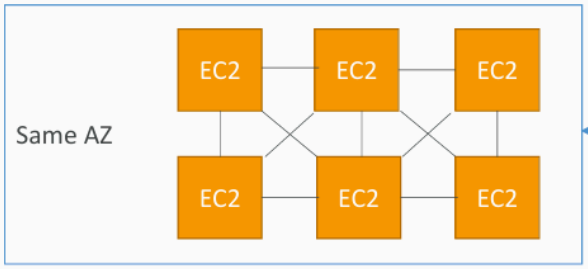


- **بعد ما كريتت ال elastic ip .. بتربطة بال instance بتاعتك وبتستفاد الآتي :**

- لو عملت stop لل instance وبعدين start مش هيتغير ودة بيفيد فى حاجات كتير منها ال DNS record مثلا
- بعد ما بتربطة .. تقدر ت ssh عليه .. بتلاقي ان ال elastic ip , public ipv4 بقو واحد .. هما نفس ال elastic
- تقدر بعد ما ربطته ب instance تشيله وتربطة بواحدة تانية

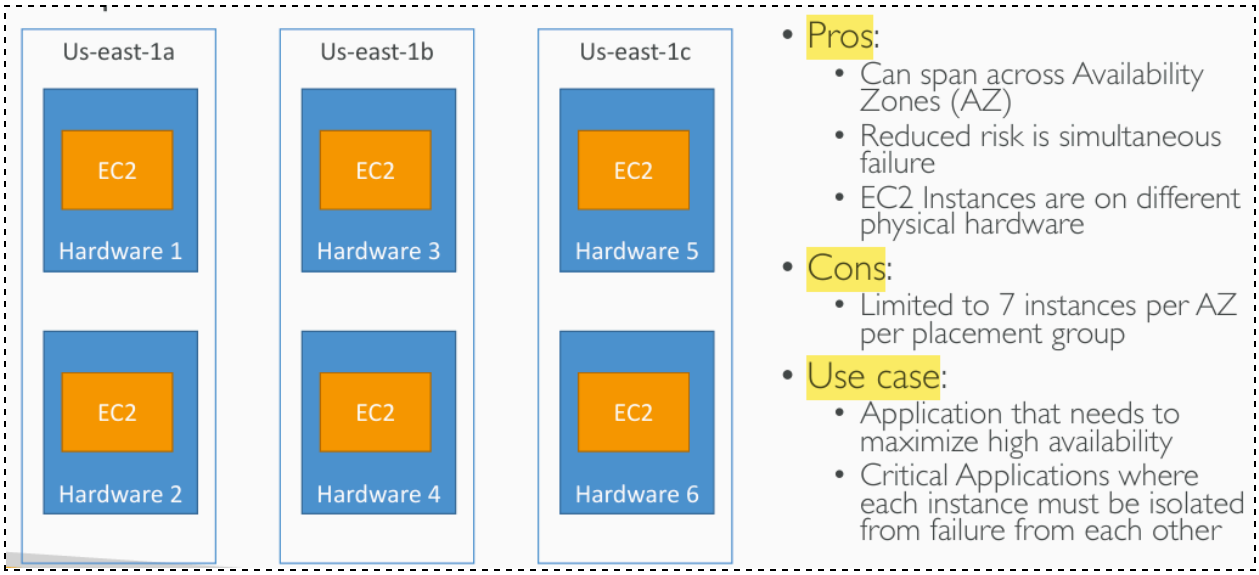
➥003 EC2 Placement Groups

- **Cluster**

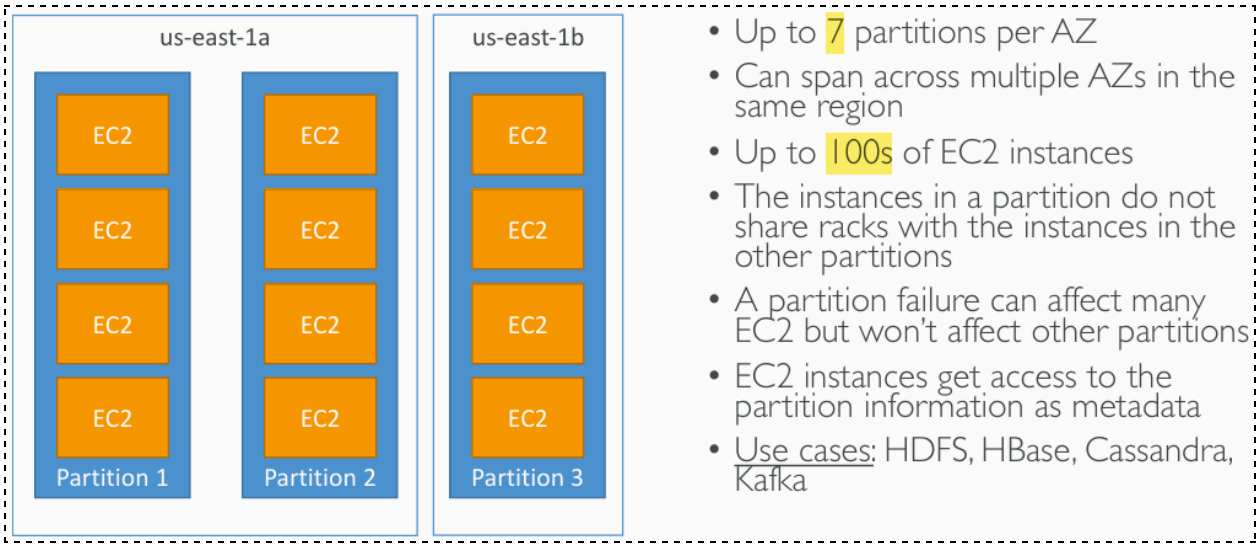


- Pros: Great network (10 Gbps bandwidth between instances with Enhanced Networking enabled - recommended)
- Cons: If the AZ fails, all instances fails at the same time
- Use case:
  - Big Data job that needs to complete fast
  - Application that needs extremely low latency and high network throughput

- **Spread**



- **Partition**



- highest networking performance >> **Cluster**
- maximum availability when there's an AZ failure >> **Spread**

➥004 EC2 Placement Groups - Hands On

Network & Security >> Placement group >> Create Placement group >> name,type

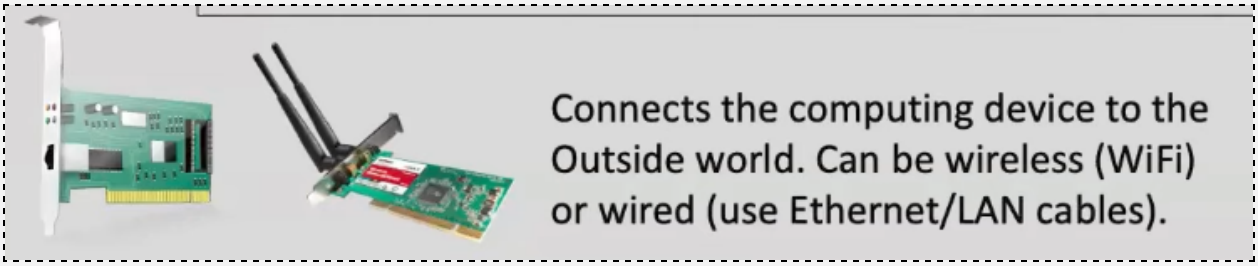
- بعد ما بتكريت ال placement group بتروح تكريت ال instances اللى هتحطها فيها

Instances >> Create Instance >> Advanced details >> Placement groups >> choose the placement group

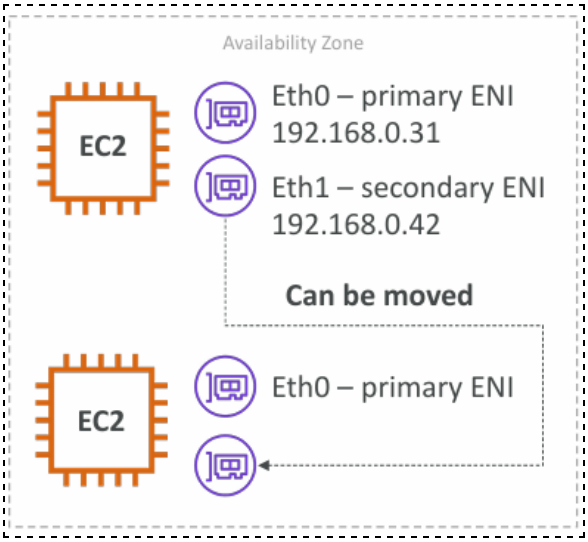
---

➥005 Elastic Network Interfaces (ENI) - Overview

- **Network Cards / Interface**



- عن طريقها بنوصل لل internet سواء wireless زي اللاب كدا او لو كمبيوتر ممكن بكابل او ممكن ب وصلة تخليه يلقط وايفاي.. فى الحالة بتاعتنا ال instance بتوصل للانترنت من خلالة.



- **ENI** : عباره عن virtual network card

- **ENI Have :**

- Primary private IPv4, one or more secondary IPv4
- One Elastic IP (IPv4) per private IPv4
- One Public IPv4
- One or more security groups
- A MAC address

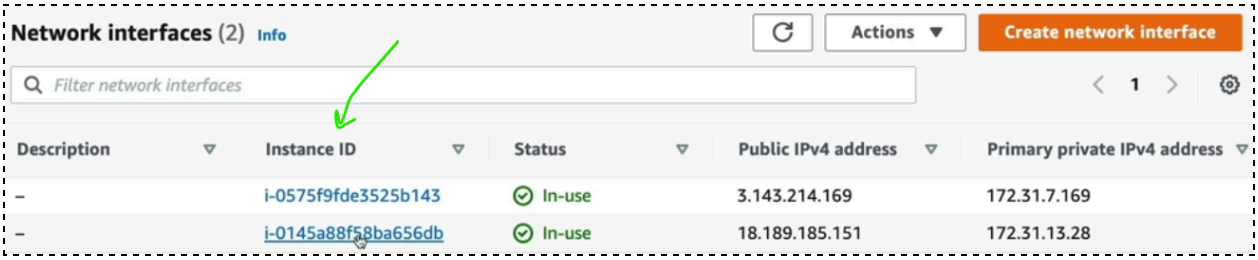
- عبارة عن interface virtual بتكريتها وتعملها attach لل instance بتاعتك
- انت اول مبت launch ال EC2 Instance بتاعتك بتاخد private ip جوا ال subntet اللى انت حددت انها تكون فيها
- جوا ال EC2 Instance هتلاقي option من خلاله تقدر تحدد هل هتدي ال EC2 Instance public ip ولا لا
- ال ENI زي مقولنا بيتعملها attach عال EC2 Instance ف بتكون جزا من ال EC2 Instance دي .. ف ساعتها ال EC2 Instance هيكون ليها 2 ip ال primary اللى هو ال private وال secondary بتاع ال ENI
- ممكن تشيلها من instance وت attach it لواحده تانية
- محكومة بال AZ اللى هيا فيها

➥006 Elastic Network Interfaces (ENI) - Hands On

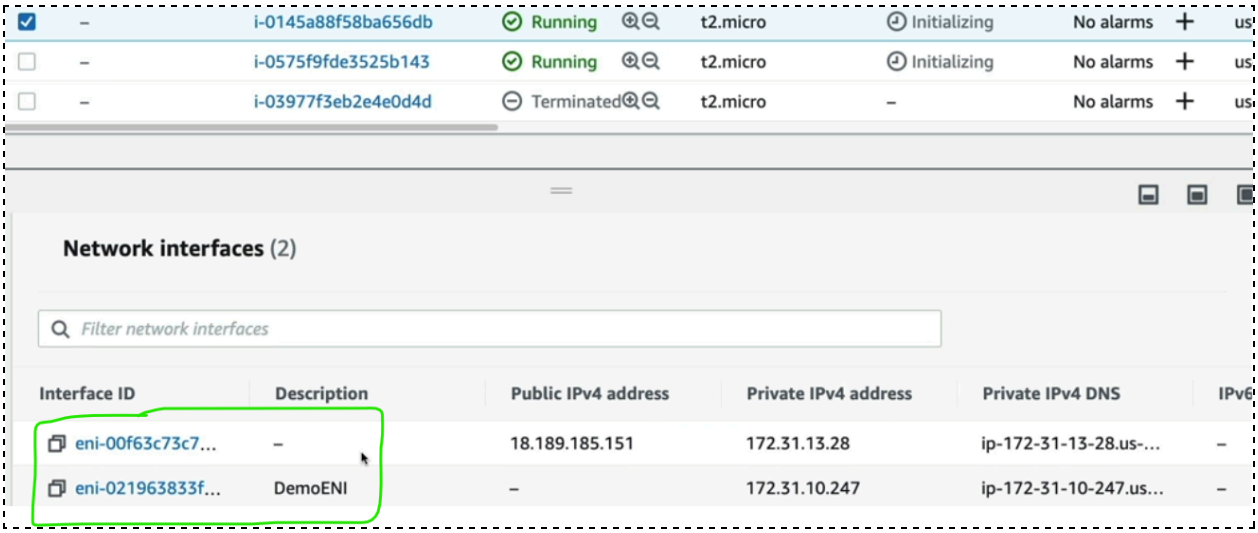
هنعمل 2 instances كل واحدة هتلاقي عندها network interface بيكون ليها id

ومتوضح ال network interface دي مربوطة بانهي instance id وهكذا

Network & Security >> Network Interface



بمجرد ما تكريت network interface هتحددها ومن action تقدر ت attach ال network interface دي لاي instance من الاتنين ف هتروح عليها هتلاقي عليها ال network interface الاصلية والجديدة



وتقدر بعد كدا تنقلها من واحدة للتانية

- **Example: Initial Configuration**
  - **Instance A:**
    - Attached ENI: `eni-12345678`
    - Private IP: `10.0.0.5`
    - Elastic IP: `54.123.45.67`
  - **Instance B:**
    - No ENI initially attached
  - **User Connection:**
    - Users connect to the application using the Elastic IP `54.123.45.67`
- **How It Works Internally**
  - **Before Failover:**
    - Users access the application on Instance A via Elastic IP `54.123.45.67`.

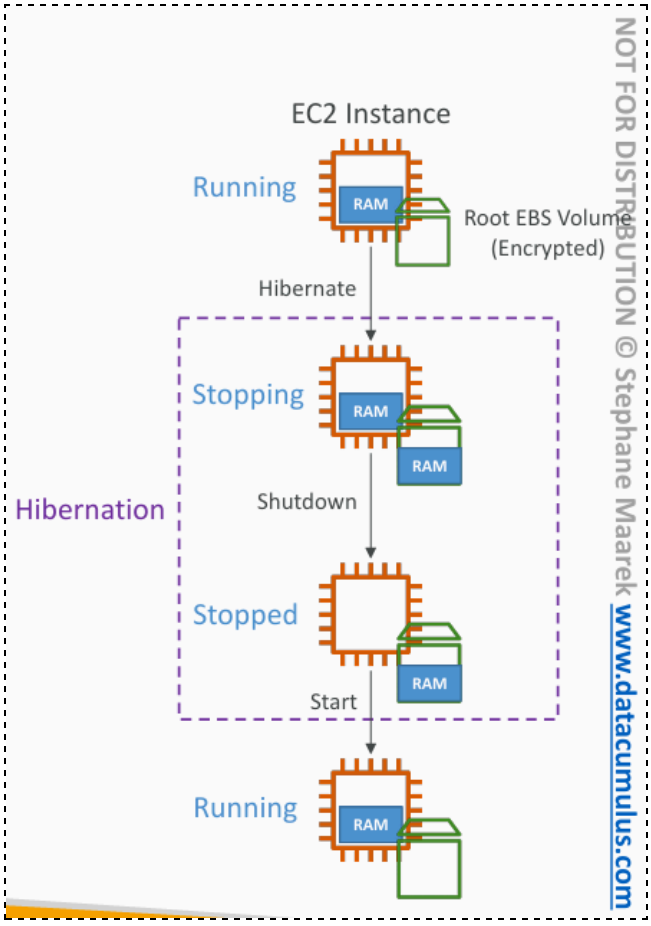
- The traffic is routed through the Elastic IP to the private IP `10.0.0.5` of Instance A.

- **During Failover:**
  - You detach the ENI from Instance A.
  - The ENI retains its IP configuration.
- **After Failover:**
  - You attach the ENI to Instance B.
  - Instance B now uses the same IP configuration.
  - Users access the application on Instance B via Elastic IP `54.123.45.67.`
  - The traffic is routed through the Elastic IP to the private IP `10.0.0.5` of Instance B.

يبقي انت رابط ال ENI عال instance A لو بعمل صيانة مثلا لل app اللى عليها وعاوز انقل علي instance B بنقل ال ENI على B ولما ال user ي access ال Elastic ip هينقلو لل private ip الخاص ب instance b.

➥008 EC2 Hibernate

- **Hibernate** ال instance اكنك بتعمل لل computer بتاعك sleep
- فى الطبيعي لو انت قفلت الكمبيوتر بتاعك shutdown وفتحته من تاني بي boot from scratch وبيبتدي يشغل البرامج من تاني وبتفتح ال files من تاني وهكذا
- انما لو عملتله Sleep الحوارات دي كلها اللى بتكون موجودة فى ال RAM بتتخزن فى ال SSD او فى الHDD على حسب انت عامل الOS بتاعك على مين فيهم ولما تشغله من تاني وبتفتح ال data دي بترجع من تاني من ال SSD or HDD لل RAM



- دة نفس اللى بيحصل عال instance بنفعل ليها ال Hibernate mode واحنا بنكريتها ف لما بنعمل ليها Hibernate بتخذن الdata اللى كانت فى ال RAM عندها

فى ال EBS root volume لحد ما تقوم من جديد وترجع تاني ال data دي لل RAM بتاعتها تاني.

- **Use Cases:**
  - لو process بتاخد وقت طويل ومش عاوزها تفصل او لو فيه services شغالة عال instance وبتاخد وقت طويل على ما تفتح ومش عاوزها تفتح from scratch

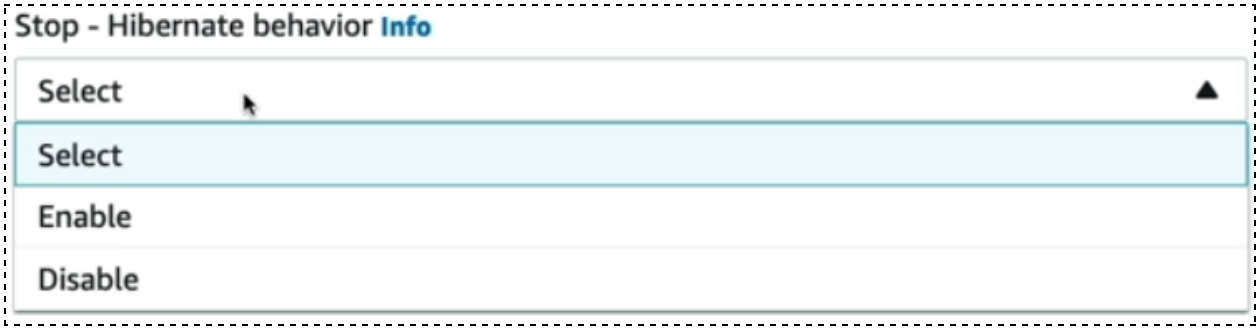
- **فيه شويه حاجات بتحكمك لو هتستعمل ال Hybernate :**

- Supported Instance **Families** – C3, C4, C5, I3, M3, M4, R3, R4, T2, T3, …
- **Instance RAM Size** – must be less than 150 GB.
- **Instance Size** – not supported for bare metal instances.
- **AMI** – Amazon Linux 2, Linux AMI, Ubuntu, RHEL, CentOS & Windows…
- **Root Volume** – must be EBS, encrypted, not instance store, and large
- Available for **On-Demand**, **Reserved** and **Spot Instances**

💡 An instance can NOT be hibernated more than **60 days**

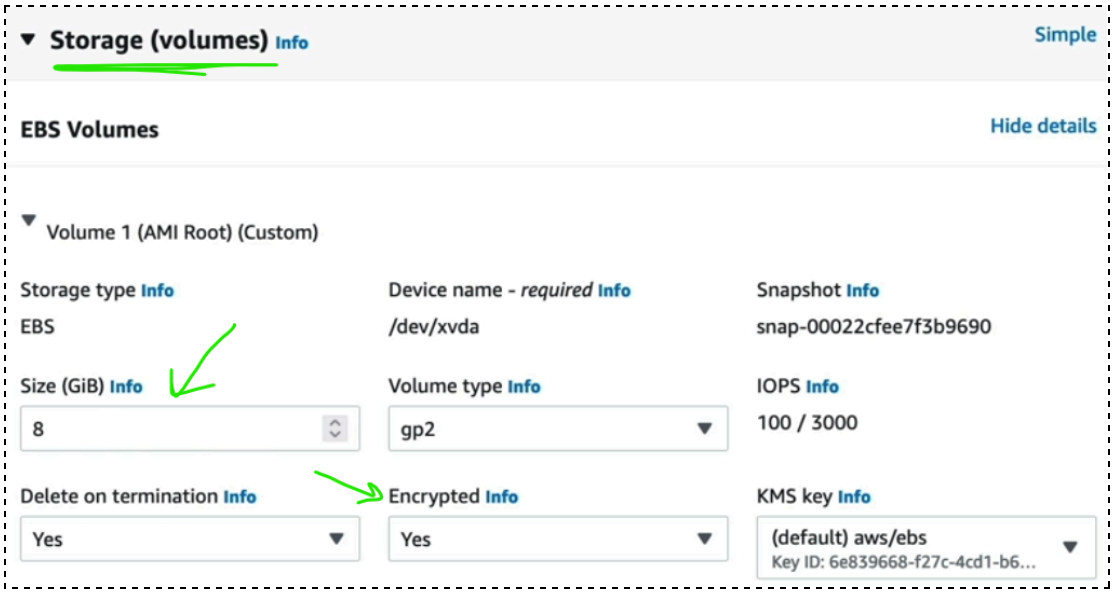
➥009 EC2 Hibernate Hands On:

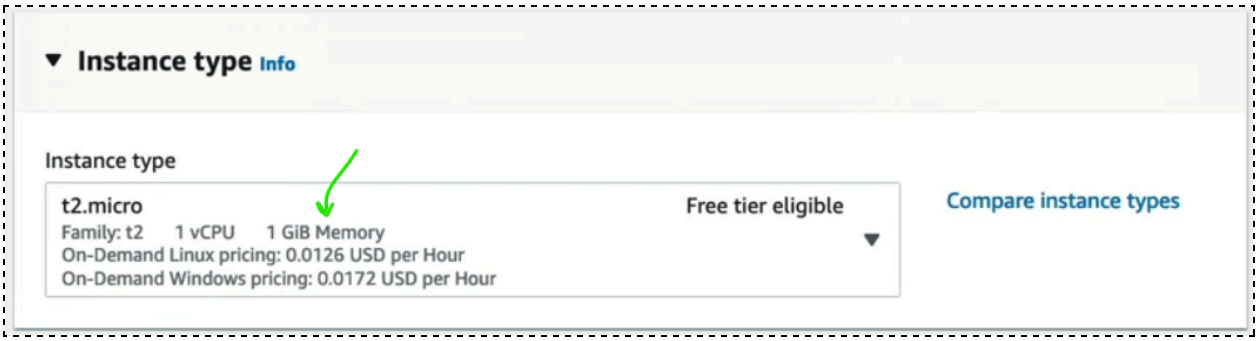
Create EC2 Instance >> Advanced Details >> Hibernate Behavior



- **الحاجات دي لازم تحصل :**

لازم يكون ال root volume عليه مساحة كافيه تكفي ال RAM Content اللى هيتخزن عليه لحد ما ال instance ت start من جديد.. هنا عندنا ال 1 instance والـ GiB Memory ف كدا دة مناسب EBS root volume 8GiB.





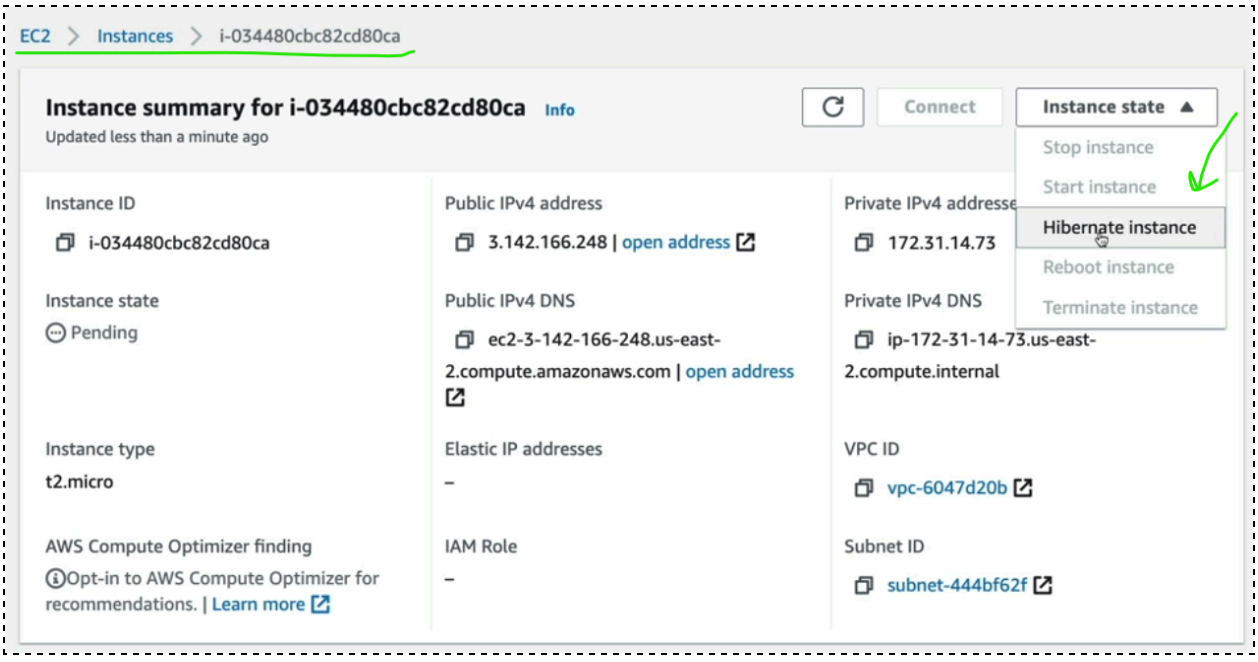
لازم يكون ال root volume encrypted

- **TEST**

علشان نشوف الدنيا دي شغالة ولا اية هنستخدم command اسمو uptime بيعرفنا ال instance دي بقالها قد اية شغالة.

```
[ec2-user@ip-172-31-14-73 ~]$ uptime
 23:34:14 up 1 min,  1 user,  load average: 0.30, 0.12, 0.04
[ec2-user@ip-172-31-14-73 ~]$ uptime
```

ظاهرلنا دلوقتي بقالها شغالة لمدة دقيقة .. ف انت لو مش مفعل ال Hibernate وعملتلها stop & start المفروض لو جيت تنفذ نفس ال command يظهرلك 0 ولكن بما انك مفعل ال Hibernate ف هتلاقي العداد دة متصفرش وكمل تاني ودة معناه ان الداتا دي اللى كانت عال RAM اتنقلت عال EBS root volume لحد ما ال instance تقوم من جديد ورجعتله تاني وكملت.




```
[ec2-user@ip-172-31-14-73 ~]$ uptime
 23:35:57 up 2 min,  1 user,  load average: 0.23, 0.12, 0.05
[ec2-user@ip-172-31-14-73 ~]$
```

I'm Here

CH-07 - EC2 Instance Storage

➥001 EBS Overview

- عباره عن Virtual hard drive بتعمله attach لل instance بتاعتك, اكنك عندك كمبيوتر ومعاك هارد خارجي بتوصله بيه وبتحفظ عليه الداتا بتاعتك.

- **مميزاتها :**

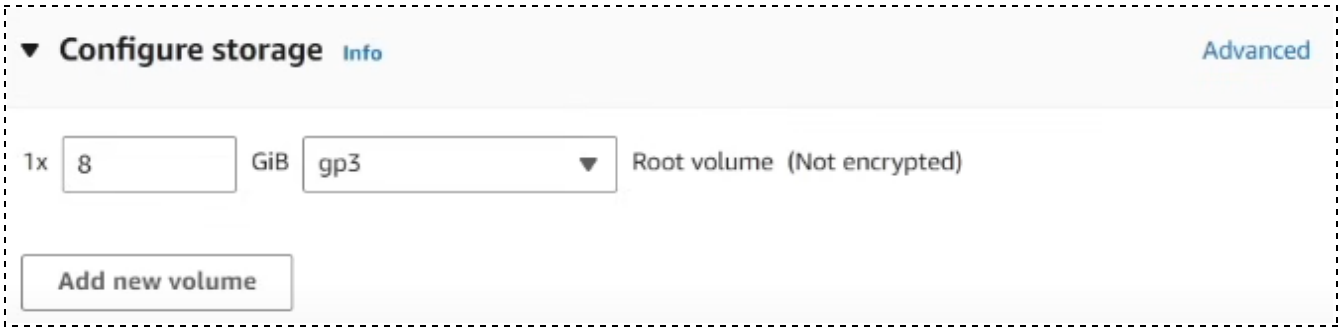
- **Persistent Storage**: الداتا هتفضل موجودة حتي لو عملت stop او restart لل instance بتاعتك .. لو مسحت ال instance هتلاقي ان ال EBS volume لسة موجودة ممكن بعد كدا تعملها attach ل instance تانية.
- **Different Types**: فيه انواع كتير منها على حسب الغرض من استخدامك محتاج اية بالظبط (general-purpose, high-performance, or cost-effective storage).
- **Backup and Restore**: تقدر تاخد (backup snapshots) لل EBS volumes دي بحيث تقدر تـ restore ال data
- **Resize and Change**: تقدر ت resize , change ال EBS volumes دي من غير ما تفقد ال data بتاعتك

💡 متقدرش ت attach ال EBS Volume دي لاكتر من instance الا فى حالة معينه هنعرفها بعدين.

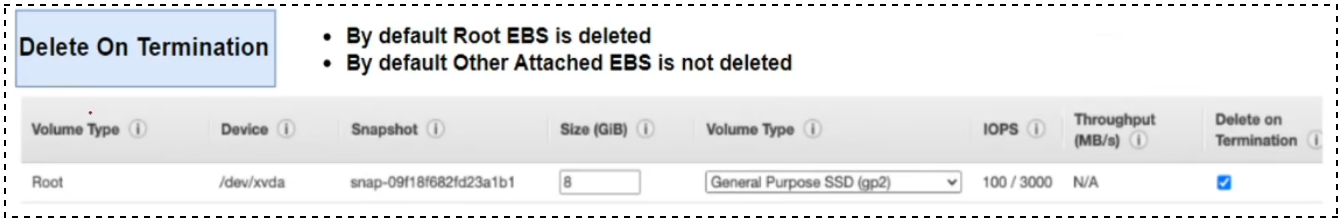
💡 ال EBS Volume محكومة بال AZ اللى هيا فيها بمعني ..ال EBS Volume انت بتكريتها وتعملها attach عال instance وال instance بتكون موجودة وليكن ف AZ1 ف لو انت ف AZ2 تانية مش هتشوف ال EBS Volume اللى ف AZ1

💡 بيبقي عندك EBS root volume الاساسية واي واحدة تانية انت بتكريتها بتبقي EBS volume اضافيه

- **Delete On Termination:**

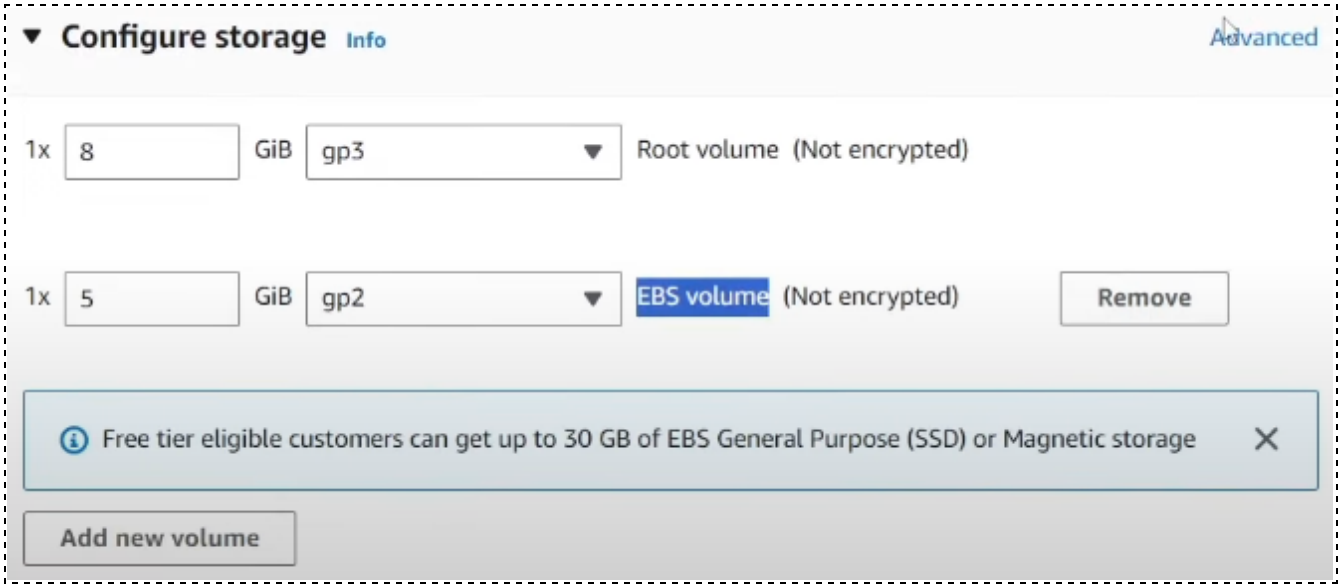


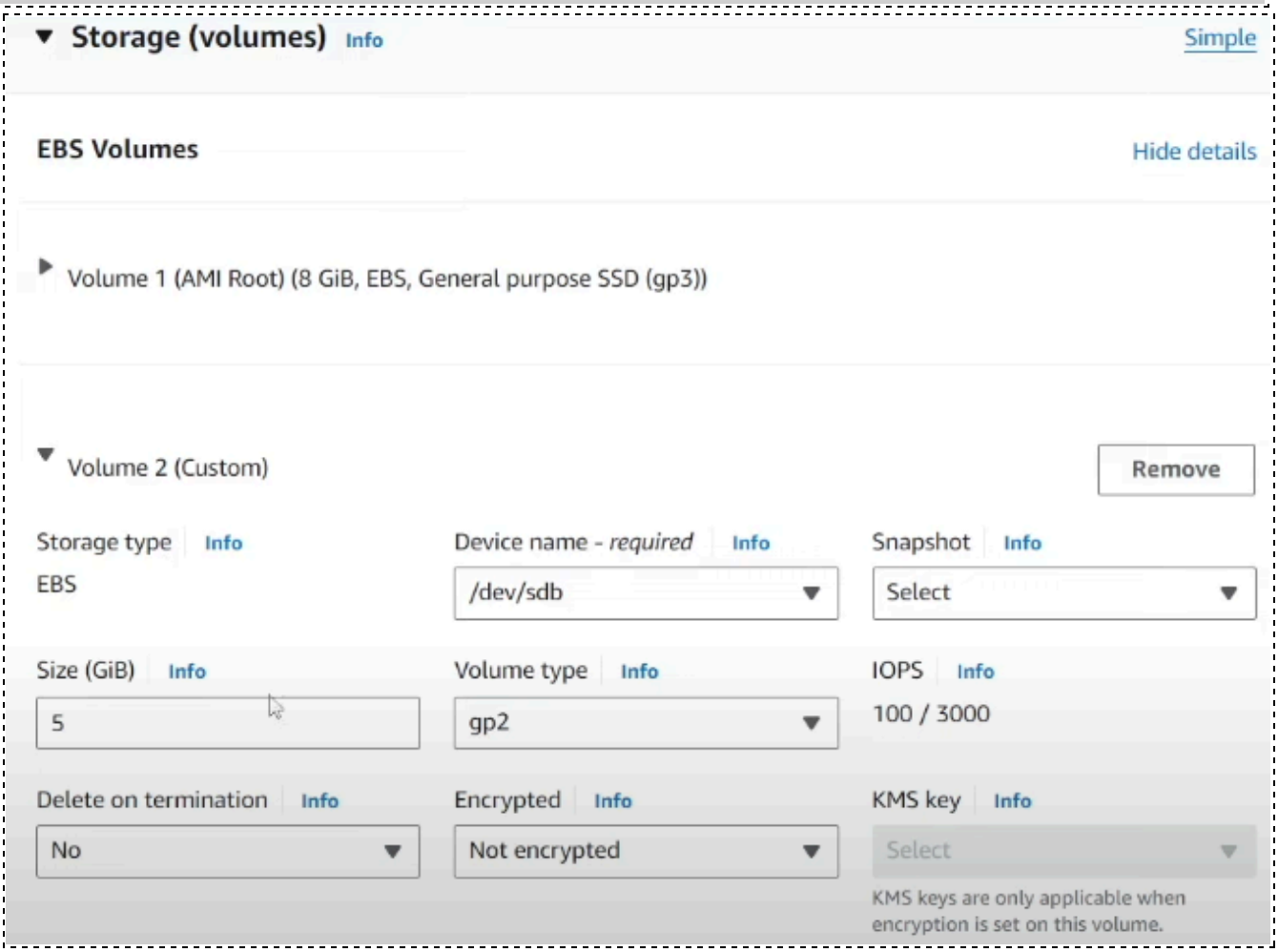
- وانت بتكريت ال instance بتاعتك كنت بتخصص root volume بعدد giga معينه .. ف انت لو عملت delete لل EC2 instance بتاعتك ال root volume دة هيتمسح باللي فيه ودة ال default behavior
- علشان اغير ال default behavior دة وميتمسحش ال root volume لو جيت اعمل delete لل instance بشيل ال **check** من هنا



- ف ال screen اللى فى الصفحة اللى فاتت هتلاحظ انك ممكن تـ add new volume .. يقصد EBS volume ياعني او انت تكريتها بعدين مش لازم وانت بت configure ال instance وبعد دة ت attach ال EBS volume دي لل instance عادى .. ف بكدا بقت ال instance بتاعتك عندها volume 2 .. الاولي root volume والتانية ال EBS Volume اللى انت كريته .. وزي مقولنا قبل كدا لو مسحت ال instance دي ال EBS Volume مش بتتمسح
- أو وانت بتكريت ال EBS Volume تخش علي **Advanced** وبعدين تخلي **delete on termination = no** علشان ال EBS متتمسحش وكذلك لل root volume

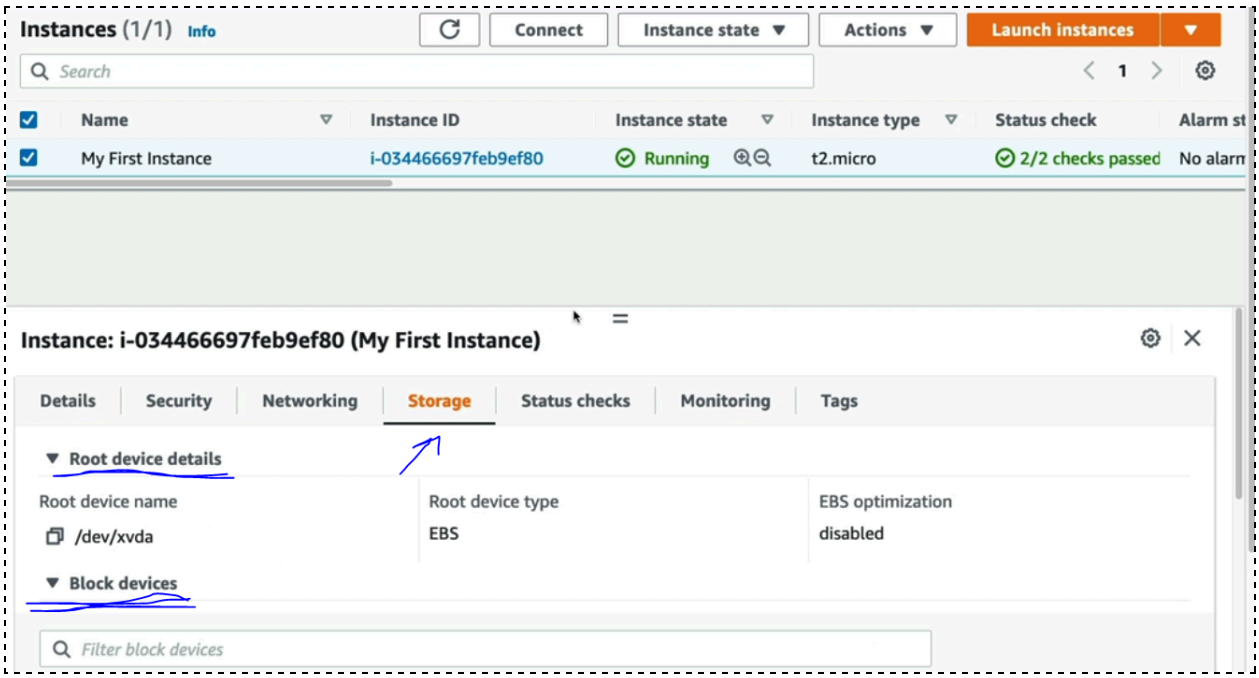
تقدر تخلي بردو ليه **delete on termination = no** علشان لما ال instance تتمسح ال root volume دة يفضل موجود

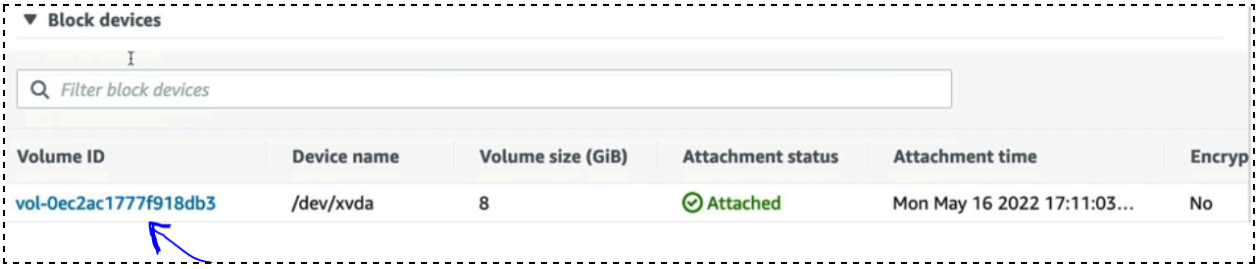

Configure storage Info
Advanced
1x 8 GiB gp3 Root volume (Not encrypted)
1x 5 GiB gp2 EBS volume (Not encrypted) Remove
Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage
Add new volume


Storage (volumes) Info
Simple
EBS Volumes
Hide details
Volume 1 (AMI Root) (8 GiB, EBS, General purpose SSD (gp3))
Volume 2 (Custom)
Remove
Storage type Info
EBS
Device name - required Info
/dev/sdb
Snapshot Info
Select
Size (GiB) Info
5
Volume type Info
gp2
IOPS Info
100 / 3000
Delete on termination Info
No
Encrypted Info
Not encrypted
KMS key Info
Select
KMS keys are only applicable when encryption is set on this volume.

➥002 EBS Hands On

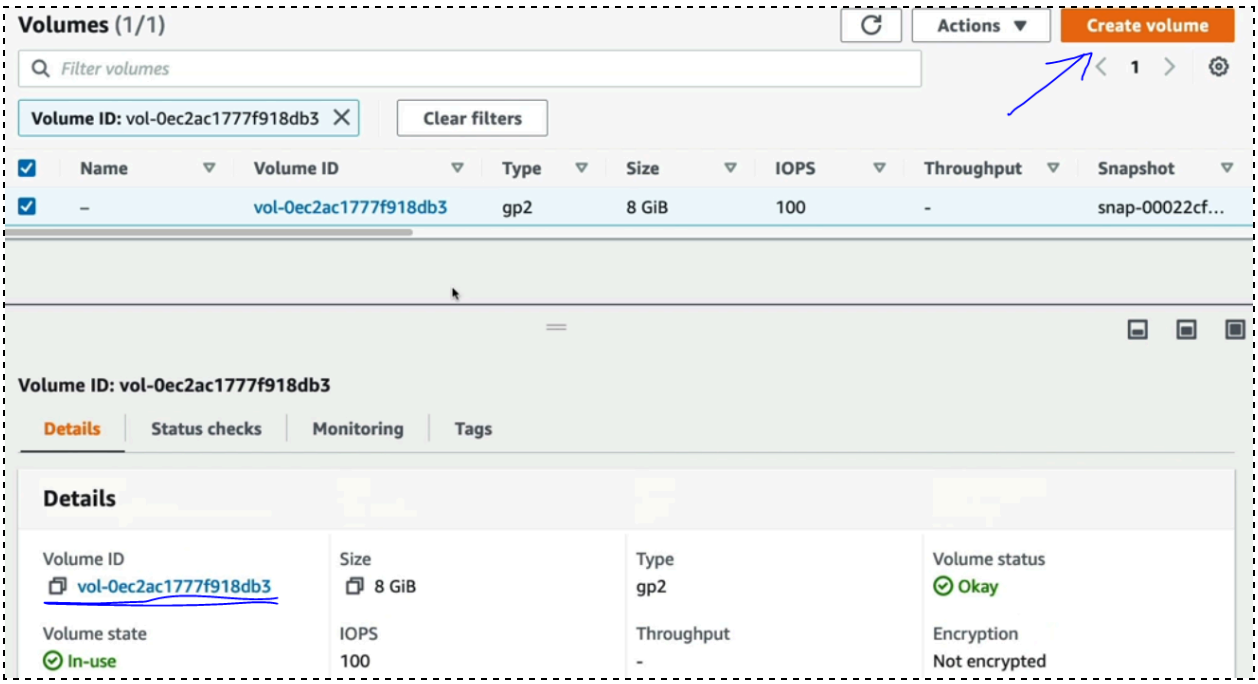
لو حددت ال instance من تحت هتعرف اية ال volumes اللى مرتبطة بيها سواء EBS root او EBS اضافيه



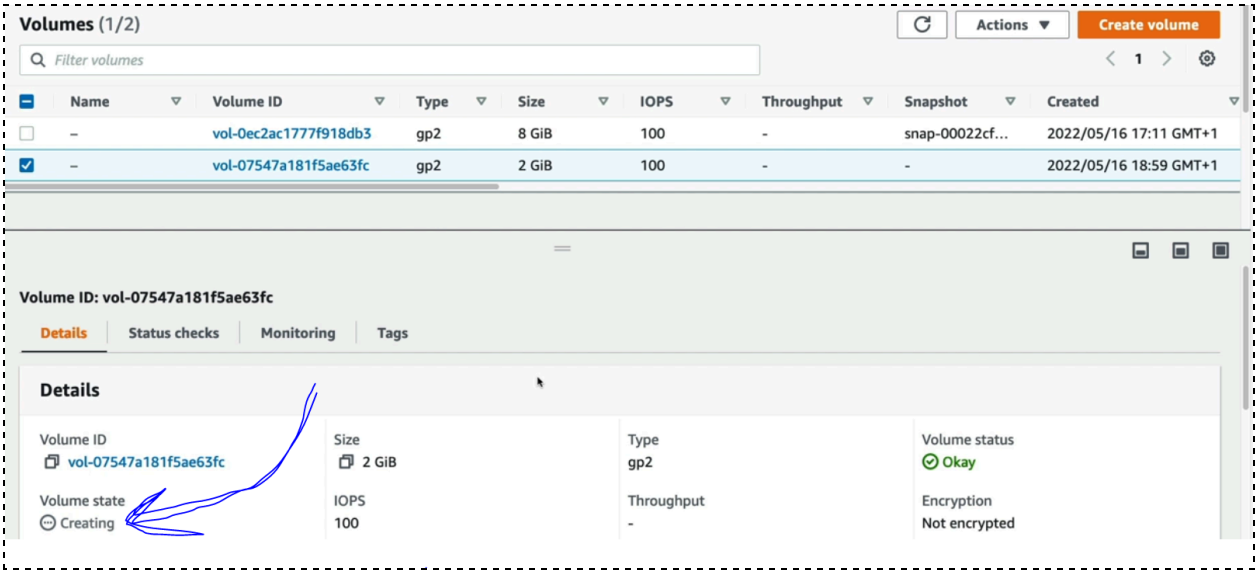


لو ضغطت عال volume دة هيدخلك عالصفحة دي .. هتلاقي معلومات عنه ومرتبط بانهي intsance وهكذا ومن هنا تقدر ت create volume .. او من هنا

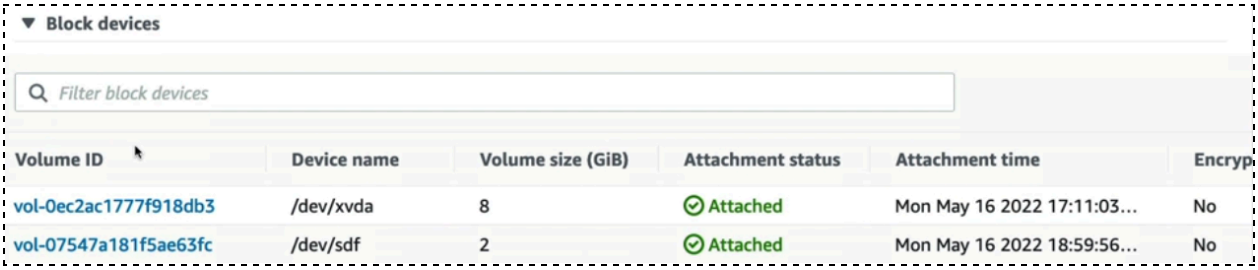
Elastic Block Store (EBS) >> Volumes



بعد ما بتكريت ال EBS volume بتاخد وقت على ما تكون Available ف لازم دي تكون Available علشان تقدر تعملها attach على instance



بعد ما عملت attach لو رجعت لل instance ودخلت على ال storage هتلاقي ان ال EBS Volume دة Attached بالشكل دة

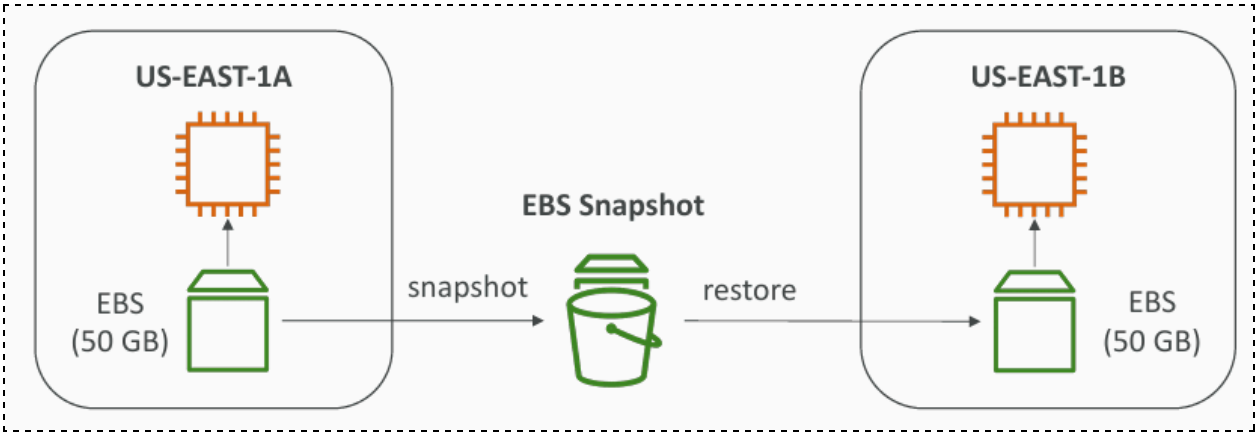


لو عملت terminate لل instance هتلاقي ان ال root volume اتمسح اما ال EBS volume by default لسة موجود .. دة بناءا على انك معملتش check على

ال delete on termination وانت بتكريته زي ما وضحنا قبل كدا فى ال CCP1

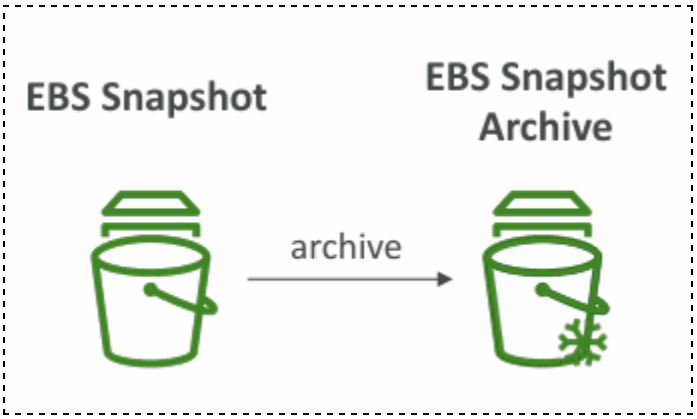
ودة بيكون متوضح على اليمين من نفس الاسكرين اللى فاتت

➥003 EBS Snapshots



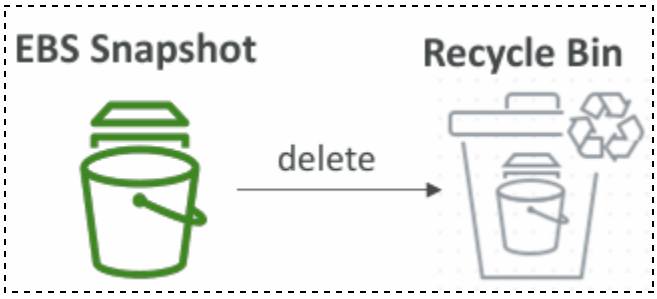
- وليكن انت محتاج تاخد backup من AZ1 ل AZ2 .. زي مقولنا مينفعش ال EBS1 تعملها attach ل instance ف AZ2 تانية .. وبالتالي الحل هنا انك بتاخد snapshot لل EBS1 .. ووانت بتكريت volume لل instance فال AZ2 بتروح عاملrestore لل snapshot دي… ف كدا كل ال data اللى فال snapshot هتروح عال EBS2 volume.

- **EBS Snapshots Features**
  - **EBS Snapshot Archive**



- ممكن ت move ال snapshot دي ل "Archive tier " ودة بيكون ارخص بنسبة 75% ولكن بتستني حوالي 24:72 ساعة لحد ما ترجع ال snapshot دي.

  - **Recycle Bin for EBS Snapshots**



- بت setup rules علشان لو مسحت snapshot بالغلط هتلاقيها فى ال recycle bin وبتفضل جوا لمدة سنة

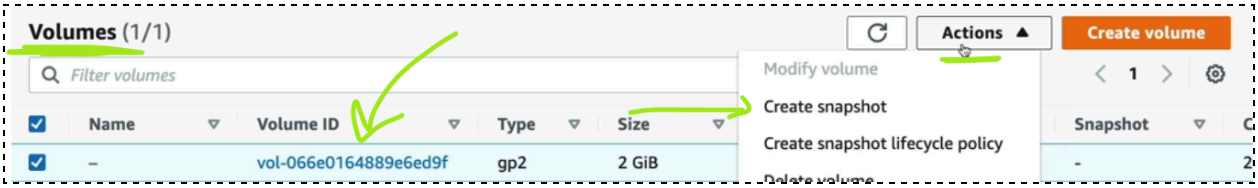
  - **Fast Snapshot Restore (FSR)**

- لو هتكريت من ال snapshot instance وعاوزها فى اسرع وقت ممكن ف دة انسب نوع ليك.

➥004 EBS Snapshots - Hands On

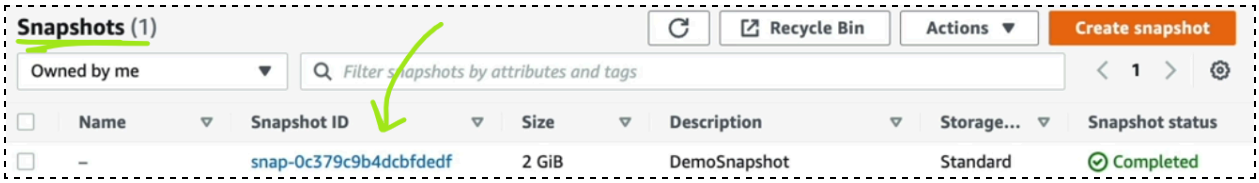
علشان تكريت snapshot

Elastic Block Store (EBS) >> Volumes



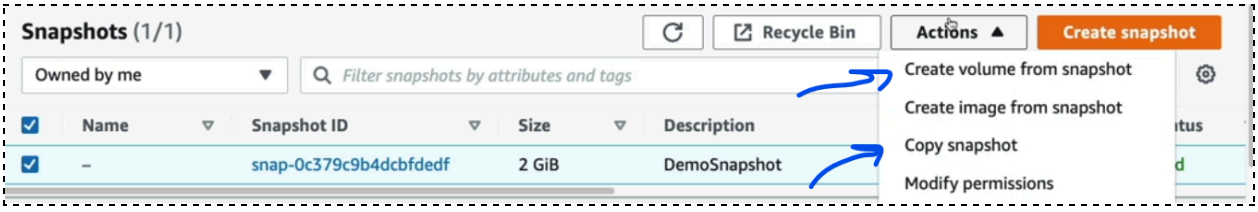
هتلاقيها هنا ..

Elastic Block Store (EBS) >> Snapshots



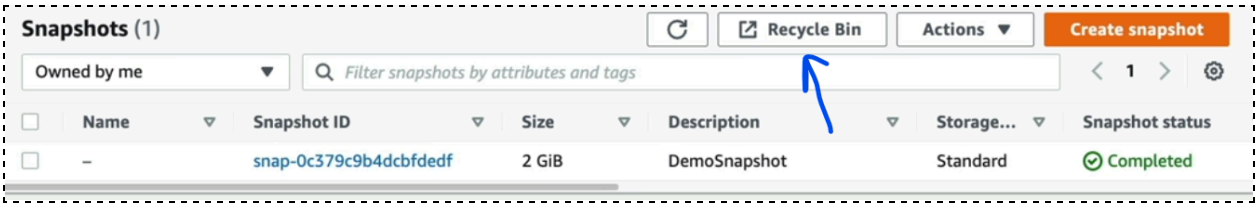
علشان ت copy ال snapshot دي ل region تانية

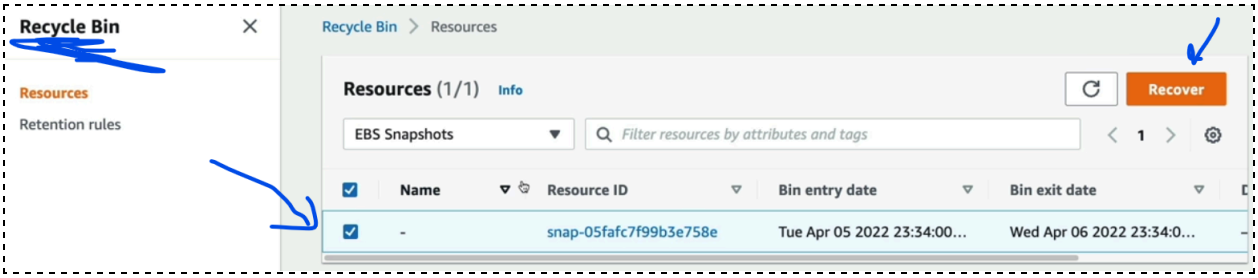
وعلشان تcreate volume منها.. وال volume دة ممكن تخليه فى اي AZ



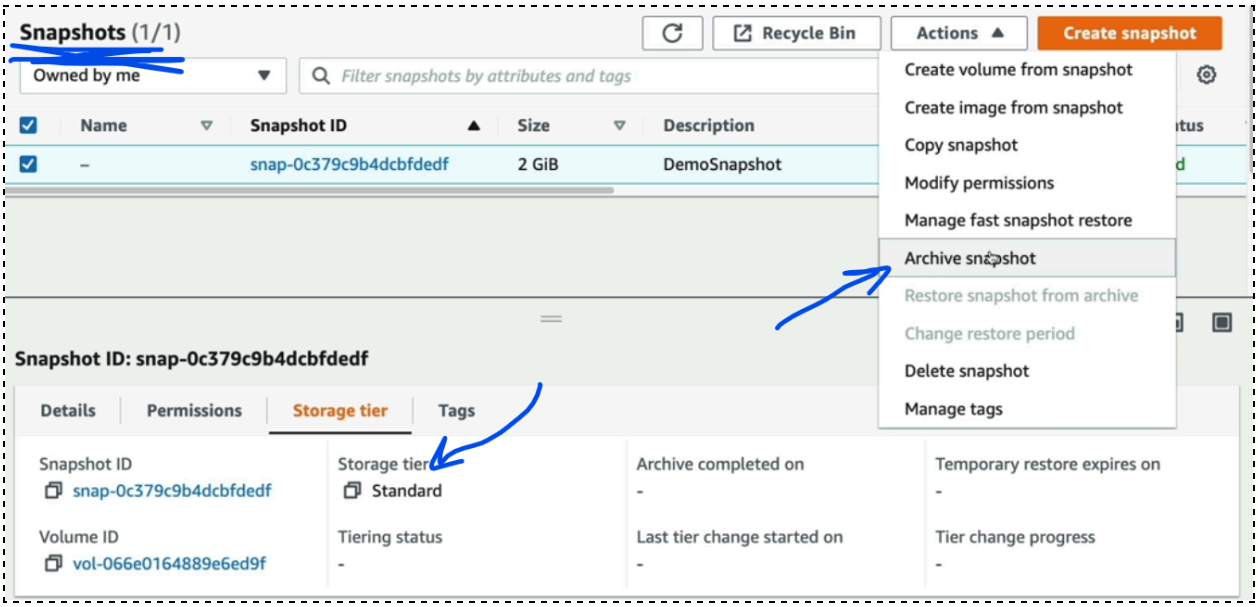
جوا ال snapshots هتلاقي ال recycle bin اللى اتكلمنا عنها منها بتحدد عدد الايام

اللى تقدر فيها تسترجع ال snapshots الممسوحة

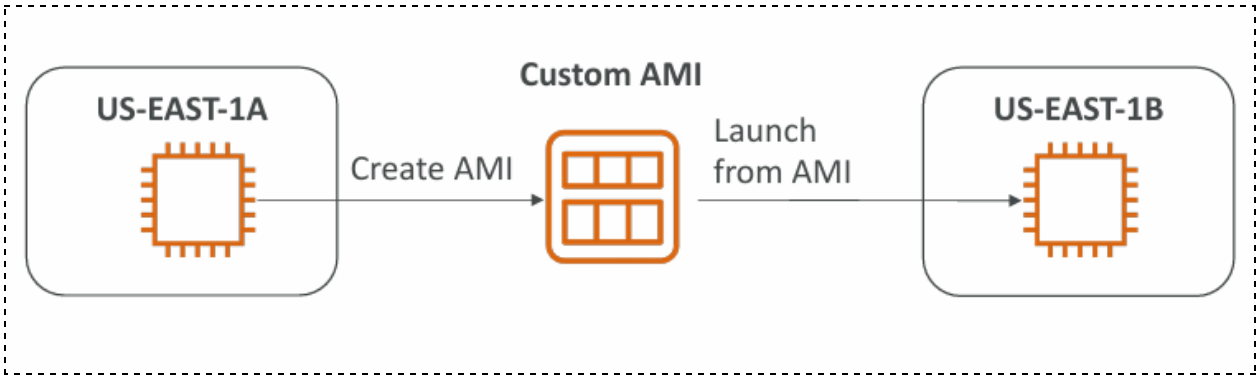




من هنا تقدر ت archive ال snapshot زي ما شرحنا



➥005 AMI Overview (Amazon Machine Image)

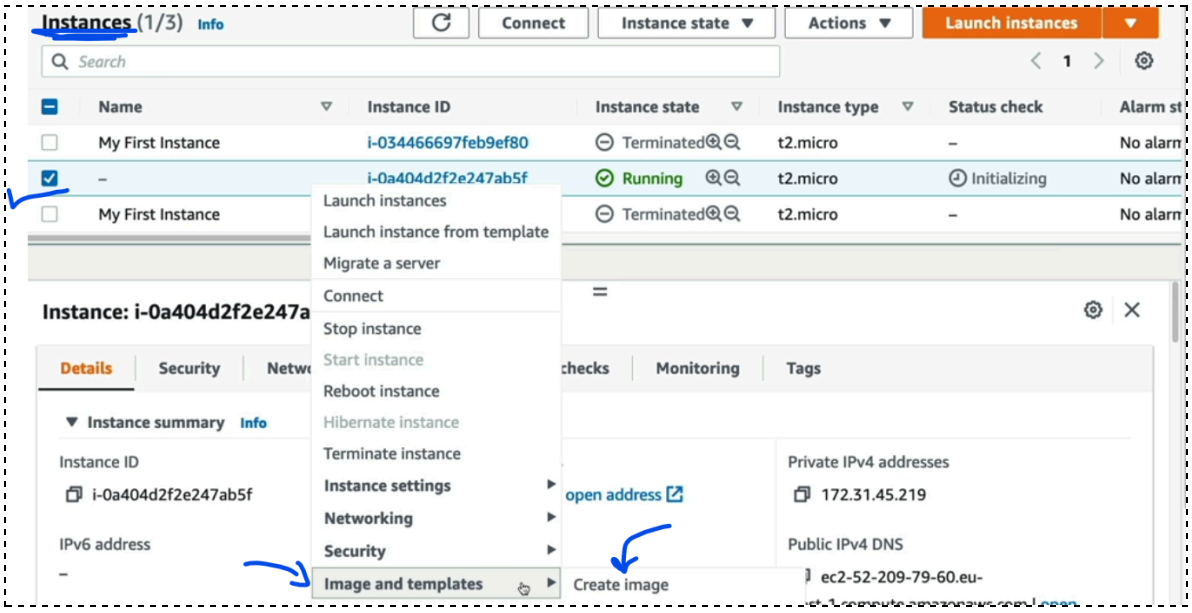


- ودي عباره عن Pre Configured Instances ياعني عباره عن Instance Template جاهزة معملوها كل ال configuration اللى كنا بنعملها ف اول محاضرة وبـعملها Save as AMI image وفيما بعد لو عاوز اعمل نفس ال task بختار ال image دي علطول مش هفضل كل شوية اعمل نفس ال Configuration

- **أنواع ال AMI :**

- **Public AMI:**
  - ناس عملتها ومشيراها Public وتقدر انت تستخدمها علطول
- **Private AMI (Yours):**
  - بتاعتك انت اللى عاملها
- **AWS Market Place AMI:**
  - ممكن يكون حد عامل Image معينه وبيبيعها مثلا.

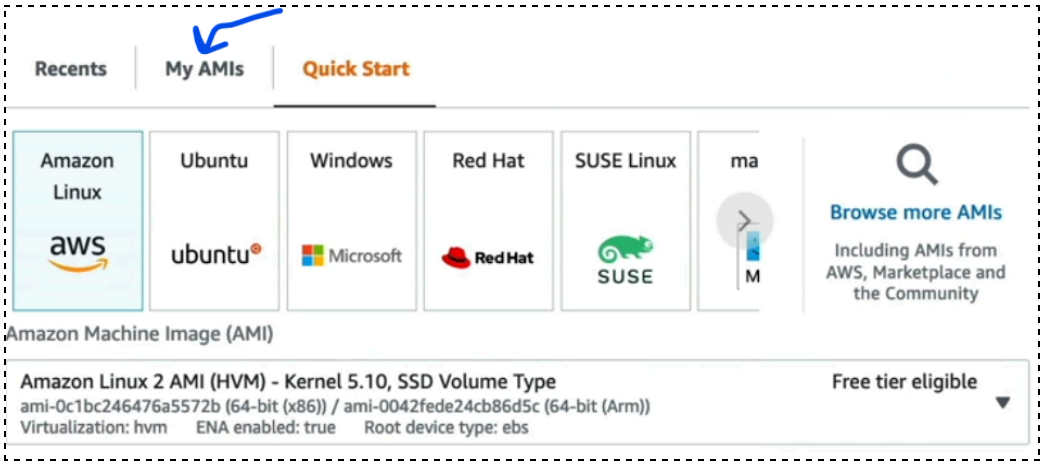
➥006 AMI Hands On



بمجرد ما بتعمل ال instance تقدر تعمل image بالشكل دة وهتلاقي ال image هنا تقدر ت launch instance منها وال instance الجديدة هتلاقيها بالظبط زي القديمة بكل ال configuration بتاعتها

Images >> AMIs

او تقدر من ت launch ال instance دي بال AMI دي وانت بتعمل ال instance بالشكل العادي وبدل ما تختار AWS linux هتختار My AMI وتختار ال image



➥007 EC2 Instance Store

- احنا قولنا ان ال EBS volume بيبقي عباره عن Virtual hard drive بتعمله attach لل instance بتاعتك.. هي كويسة ولكن ال performance بتاعها بيكون limited شوية .. ف لو محتاج حاجة ب Performance اعلي يبقي ال EC2 Instance Store

● Cons:

- فيها عيب وهو انك لو عملت terminate لل instance ال instance store دي هتطير.. على عكس ال EBS Volume لما كنت بتمسح ال instance كانت بتفضل موجودة وكنت تقدر ت attach it ل instance تانية إلا بقا لو كنت عاملها delete on termination

● Use Case:

- Short term storage : Buffer, Cache, Temporary Data

💡يبقي لو محتاج Short long storage دي تمام .. لو محتاج Long term storage ال EBS افضل.

➥008 EBS Volume Types

- **GP2/GP3 (SSD) - General Purpose**
  - دة ssd volume تقدر تستخدمه لل general-purpose وتكلفته قليلة فيه balance بين ال performance , price
  - ودة اللى غالبا هنستخدمه واحنا بنجرب لانه بيستخدم فى boot volumes, Virtual desktops, Development and test environments
- **io1/io2 (SSD)**
  - دة ssd volume تقدر تستخدمه لو محتاج high-performance و low latency و high throughput
  - **high throughput** : ياعني هينقل داتا بشكل اسرع
- **st1 (HHD)**
  - دة HHD volume تقدر تستخدمه لو عاوز high throughput و low cost ومش فارقلك ال latency لان هنا ال latency بتكون عالية شوية
- **sc1 (HHD)**
  - زيو زي اللى قبله دة ولكن هنا لو انت مش بتـ access الداتا بتاعتك كل شوية ياعني كل فتره بتـ access الداتا يبقي دة انسب نوع ليك واكيد سعرو اقل من اللى فوقة دة.

- بيتم تصنيف ال EBS Volumes علي حسب ال IOPS , throughput , size
- **IOPS** : Operation Per Second

- **General Purpose SSD :**

- 1 GiB - 16 TiB
- System boot volumes, Virtual desktops, Development and test environments

- **Gp3:**

- Baseline of 3,000 IOPS and throughput of 125 MiB/s
- Can increase IOPS up to 16,000 and throughput up to 1000 MiB/s independently

- **Gp2:**

- Small gp2 volumes can burst IOPS to 3,000
- Size of the volume and IOPS are linked, max IOPS is 16,000
- 3 IOPS per GB, means at 5,334 GB we are at the max IOPS

💡اقدر اتحكم فى ال IOPS , throughput منفصلين فى Gp3 علي عكس ال Gp2

- **Provisioned IOPS (PIOPS) SSD Use Cases**

- Critical business applications with sustained IOPS performance
- Or applications that need more than 16,000 IOPS
- Great for databases workloads (sensitive to storage perf and consistency)
- **io1 (4 GiB - 16 TiB)**
  - Max PIOPS: 64,000 for Nitro EC2 instances & 32,000 for other
  - Can increase PIOPS independently from storage size

- **io2 Block Express (4 GiB – 64 TiB):**
    - Sub-millisecond latency
    - Max PIOPS: 256,000 with an IOPS:GiB ratio of 1,000:1
- **Supports EBS Multi-attach**

● Hard Disk Drives (HDD)

- Cannot be a boot volume
- 125 GiB to 16 TiB
- **Throughput Optimized HDD (st1)**
    - Big Data, Data Warehouses, Log Processing
    - Max throughput 500 MiB/s – max IOPS 500
- **Cold HDD (sc1):**
    - For data that is infrequently accessed
    - Scenarios where lowest cost is important
    - Max throughput 250 MiB/s – max IOPS 250

💡 يبقي gp2,gp3 لو محتاج اقل من IOPS 16000

و io1 لحد 64000 واكتر من كدا لحد 250000 يبقي io2

💡 مش محتاج تعرف كل ال details دي هتلاقي كل التفاصيل دي هنا

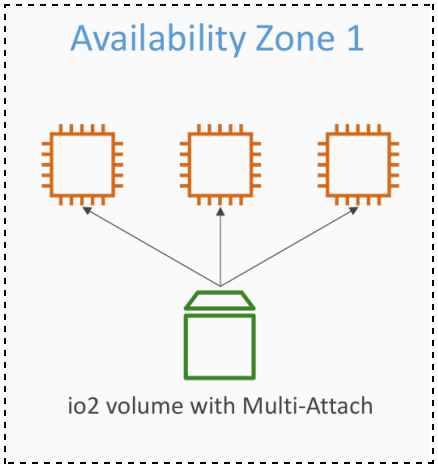
• Solid state drive (SSD) volumes

	General Purpose SSD volumes		Provisioned IOPS SSD volumes	
Volume type	gp3	gp2	io2 Block Express [3]	io1
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)		99.999% durability (0.001% annual failure rate)	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
Use cases	• Transactional workloads • Virtual desktops • Medium-sized, single-instance databases • Low-latency interactive applications • Boot volumes • Development and test environments		Workloads that require: • Sub-millisecond latency • Sustained IOPS performance • More than 64,000 IOPS or 1,000 MiB/s of throughput	• Workloads that require sustained IOPS performance or more than 16,000 IOPS • I/O-intensive database workloads
Volume size	1 GiB - 16 TiB		4 GiB - 64 TiB [4]	4 GiB - 16 TiB
Max IOPS per volume	16,000 (64 KiB I/O)	16,000 (16 KiB I/O)	256,000 (16 KiB I/O) [5]	64,000 (16 KiB I/O)
Max throughput per volume	1,000 MiB/s	250 MiB/s [1]	4,000 MiB/s	1,000 MiB/s [2]
Amazon EBS Multi-attach	Not supported		Supported	
NVMe reservations	Not supported		Supported	Not supported
Boot volume	Supported			

• Hard disk drive (HDD) volumes

	Throughput Optimized HDD volumes	Cold HDD volumes
Volume type	st1	sc1
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	
Use cases	• Big data • Data warehouses • Log processing	• Throughput-oriented storage for data that is infrequently accessed • Scenarios where the lowest storage cost is important
Volume size	125 GiB - 16 TiB	
Max IOPS per volume (1 MiB I/O)	500	250
Max throughput per volume	500 MiB/s	250 MiB/s
Amazon EBS Multi-attach	Not supported	
Boot volume	Not supported	

➥009 EBS Multi-Attach



EBS Multi-Attach is a feature of Amazon EBS that **allows you to attach a single Provisioned IOPS SSD (io1 or io2) volume to multiple EC2 instances in the same Availability Zone.** Here's an overview of EBS Multi-Attach:

- **Key points:**

💡متاحة فقط للنوع io1/io2

- تقدر تربط ال EBS Volume دة بعدد 16 Nitro-based EC2 instances فى نفس الوقت
- لازم ال instances تكون فى نفس ال AZ
- تستخدم فى clustered Linux applications ودي بتشتغل applications across multiple servers
- Requires a cluster-aware file system (not standard file systems like ext4 or XFS).

- **Benefits:**

- High availability : لو instance حصلها fail الباقي بيشتغل
- بتمكنك ت build clustered applications

➥010 EBS Encryption

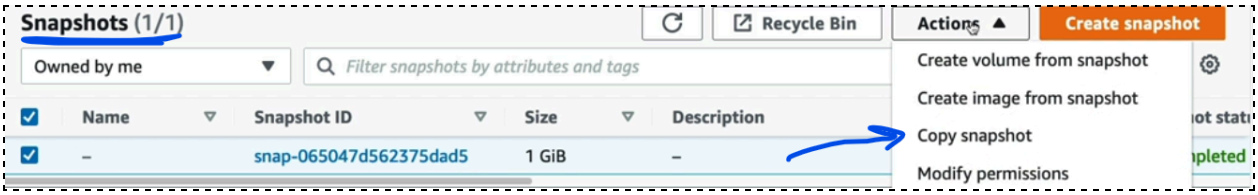
- **لو عملت EBS Volume encrypted هتحصل على الآتي :**
  - كل ال data اللى فى ال EBS Volume دي بتكون encrypted
  - ال data اللى بتروح وتيجي بين ال EBS Volume وال instances بتكون encrypted
  - كل ال snapshots وال instances اللى هتتعمل من ال snapshots دي بتكون encrypted
  - ال encryption بيسبب latency طفيف جدا مش هتحس بيه

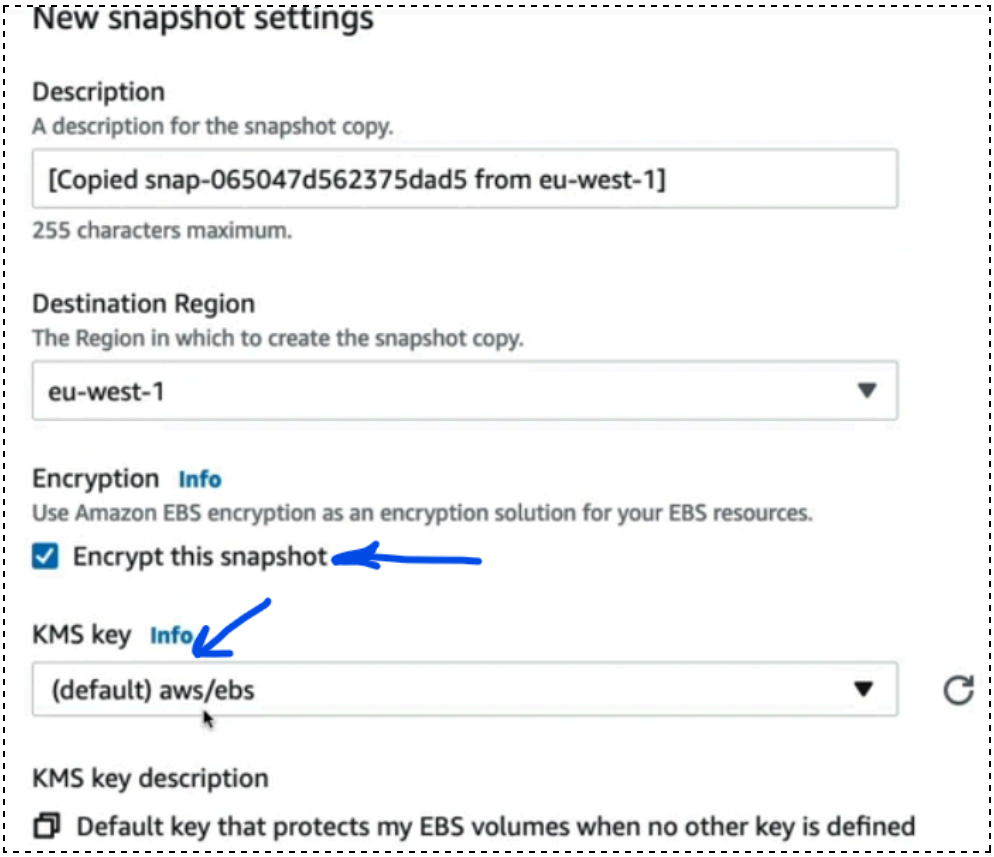
- **EBS Encryption leverages keys from KMS (AES-256)**
  - **KMS** stands for AWS Key Management Service.
  - **AES** stands for Advanced Encryption Standard, a widely used encryption algorithm, 256 refers to the key size in bits.
  - يبقي ال encryption بيتم من خلال AWS KMS والنوع ال default اللى بيتم استخدامه هو (AES-256)

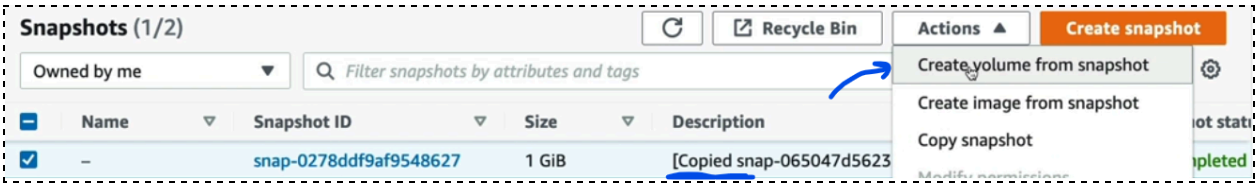
- **Encryption: encrypt an unencrypted EBS volume**
  - Create an EBS snapshot of the volume
  - Encrypt the EBS snapshot (using copy)
  - Create new EBS volume from the snapshot (the volume will also be encrypted)

- Now you can attach the encrypted volume to the original instance

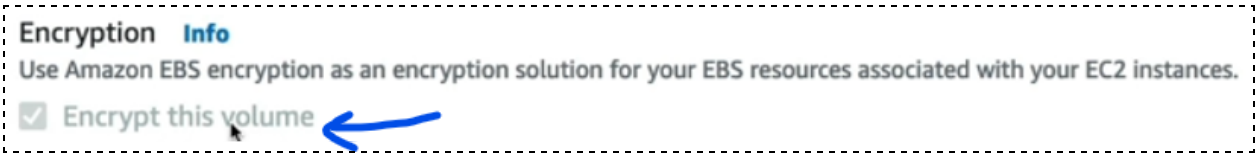
- ياعني كان عندنا unencrypted EBS volume وكنا عاوزين نخليه encrypted



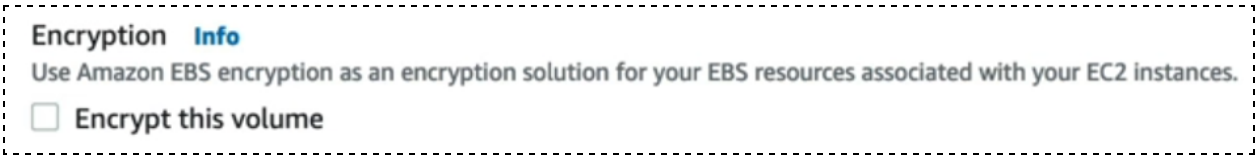




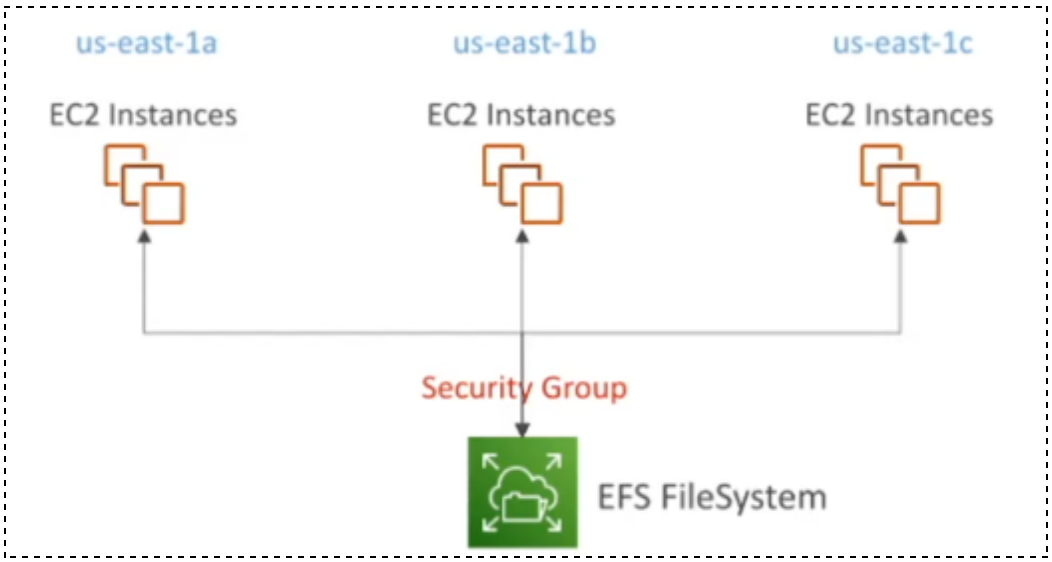
- هتلاقيه already encrypted بما انه معمول من encrypted snapshot



- أما لو بتعمل EBS Volume من جديد ف ممكن تخليه encrypted direct



➥011 Amazon EFS (Elastic File System)



- ال EBS كان storage أما هنا ال EFS عباره عن File System
- بتدفع عال storage اللى بتستخدمها فقط

- **Key Points**
  - **Use Cases: Content management, web serving, data sharing, Wordpress**

  - **Uses NFSv4.1 (Network file system) protocol**
    - ودة استخدامه انو بيمكنك من انك تـ access files over the internet سواء هتـ read or write عليها

  - **Uses security group to control access to EFS**
    - اللى داخل واللى خارج من ال file system دة بنعرف نتحكم فيه

  - **Compatible with Linux based AMI (not Windows)**
    - شغالة مع Linux فقط

  - **Encryption at rest using KMS**
    - الداتا اللى عال file system دة بتكون encrypted باستخدام KMS اللى هيا AWS Key Management Service

  - **POSIX file system (~Linux) that has a standard file API**
    - بتشتغل اكنها file system عادي زي بتاع linux دة معني انها بتتبع ال POSIX Standards اللى هيا بتتصرف بالطريقة العادية زي ما كنت فى لينكس بنفس طريقة ال commands وال file handling ياعني تقدر تستخدم معاها ls , mv , cp وهكذا

- **File system scales automatically, pay-per-use, no capacity planning!**

- "ال EFS بتـ grow , shrink .. ياعني هتـ write data هتلاقيها بتـ grow هتمسح data هتلاقيها بتـ shrink .. ياعني "pay for what you use"

EFS – Performance & Storage Classes

- **وانت بتـ Create ال EFS هتلاقي عندك:**
  - **Performance Mode:**
  - **General Purpose** (default): latency-sensitive use cases (web server, CMS, etc…)
  - ودة بيكون حاجة balanced كدا من جميع النواحي
  - **Max I/O**: higher latency, throughput, highly parallel (big data, media processing)
  - دة لو محتاج high throughput بيكون اغلي شوية أكيد .. هتفهم معنى ال throughput دلوقتي

  - **Throughput Mode:**
  - **Bursting** : 1 TB = 50MiB/s + burst of up to 100MiB/s
  - عبارة عن Traffic عالي فجأة عال EFS ف هل عاوز ال EFS يهندل ال Bursting اللى حصل دة ولالا
  - Bursting انفجار
  - **Provisioned** : set your throughput regardless of storage size, ex: 1 GiB/s for 1 TB storage
  - بتهيئها .. ياعني بتحدد انت عاوز ال throughput بتاعك يكون قد اية وليكن مثلا 1 GB/sec

- **Elastic** : automatically scales throughput up or down based on your workloads
- ودة بيـ Scale Automatic مع ال work load اللى جاي وبيعمل throughput up and down ودة بيكون مفضل فى ال unpredictable workloads

💡يبقي مثلا لو عندك بيتبعت وليكن 1 GB عال file system دة وفجاة بقو 20 GB ف دة bursting (انفجار) .. اما لو انت عارف ال throughput rate ف خليك provision .. اما لو عاوز الموضوع dynamic ف خليك ف elastic وكلو بحسابه😂.

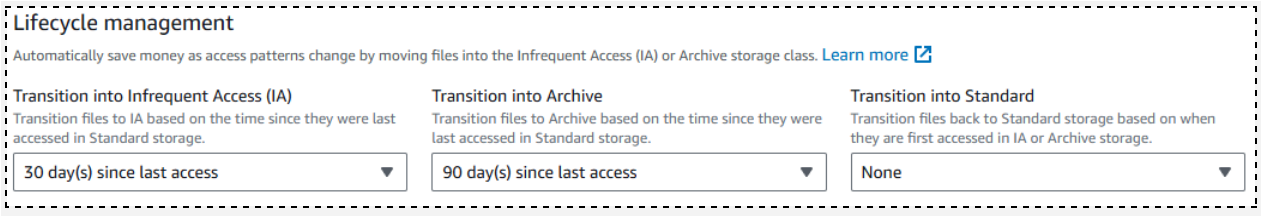
● Storage Tier:

- بن manage بيها ال lifecycle بتاعه ال file system دة .. من خلالها تقدر تحدد هنعمل اية فى الfile system دة بعد فتره معينه .. فى عندك فيها option 3

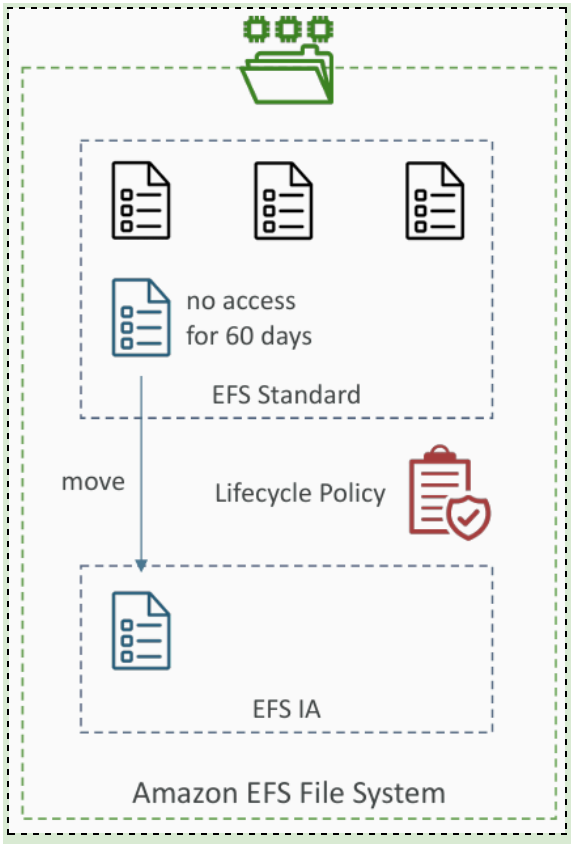
● Frequently Accessed:

- مُصمم لل active data workloads اللى بيحصلها access باستمرار

- **Infrequently Accessed(EFS IA):** cost to retrieve files, lower price to store.
- مناسبة لو ال data بتاعتنا مش بيحصلها access كتير .. بتكون هنا التكلفة اقل ولكن فيه تكلفه على استرجاع الملفات
- **Archive:** rarely accessed data (few times each year), 50% cheaper
- "لو ال data بيحصلها access اقل ك عدد مرات من الoptions 2 اللى فوق وطبعا ارخص منهم"



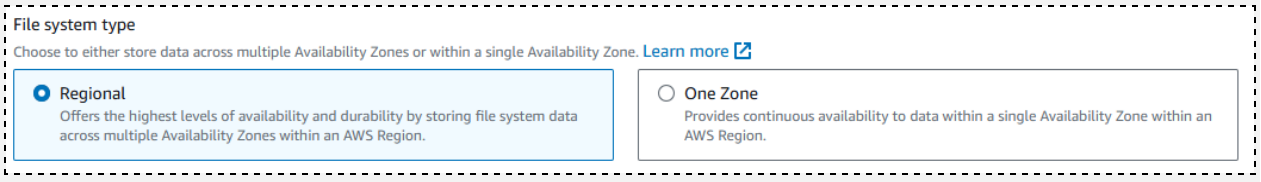
💡 علشان ت move ال data بتاعتك between ال storage tiers دي بيتم دة من خلال انك بت set lifecycle polices تحققلك دة.



- **Availability & Durability:**
  - **Regional:** Multi-AZ, great for prod

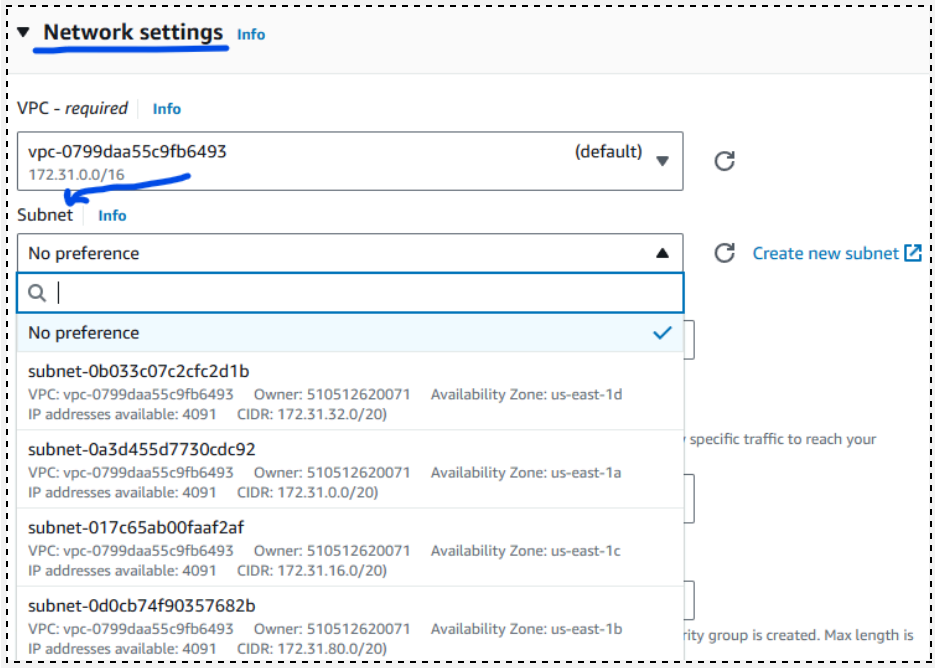
- متاح ف اكتر من AZ .. ياعني لو واحدة وقعت ال EFS file system دة يفضل شغال اغلي .. مناسب لل production environment"

- **One Zone:** One AZ, great for dev, backup enabled by default, compatible with IA (EFS One Zone-IA)
  - يشتغل ف zone واحدة فقط .. ودة مناسب اكتر لل development or backup or "testing



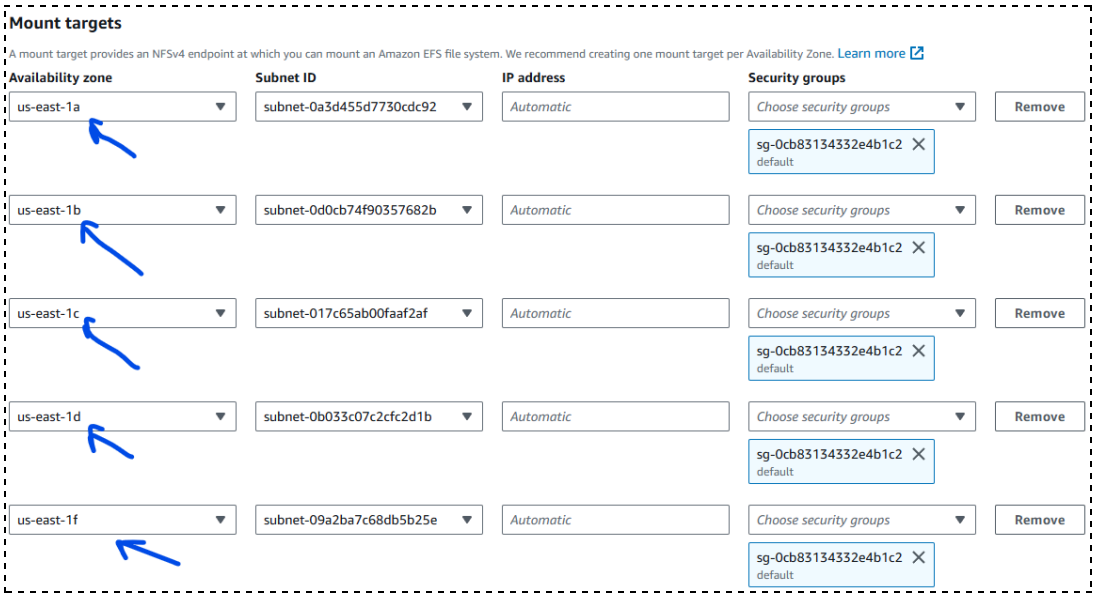
➥012 Amazon EFS - Hands On

"Search على EFS وهتحدد ال setting اللى عاوزها بناءا على اللى اتشرح فوق وتختار security group ولو معملتش security group مخصص لل EFS اعملو وارجع اختاره وبعدها تكريت ال instances وتربط ال instances بال EFS."

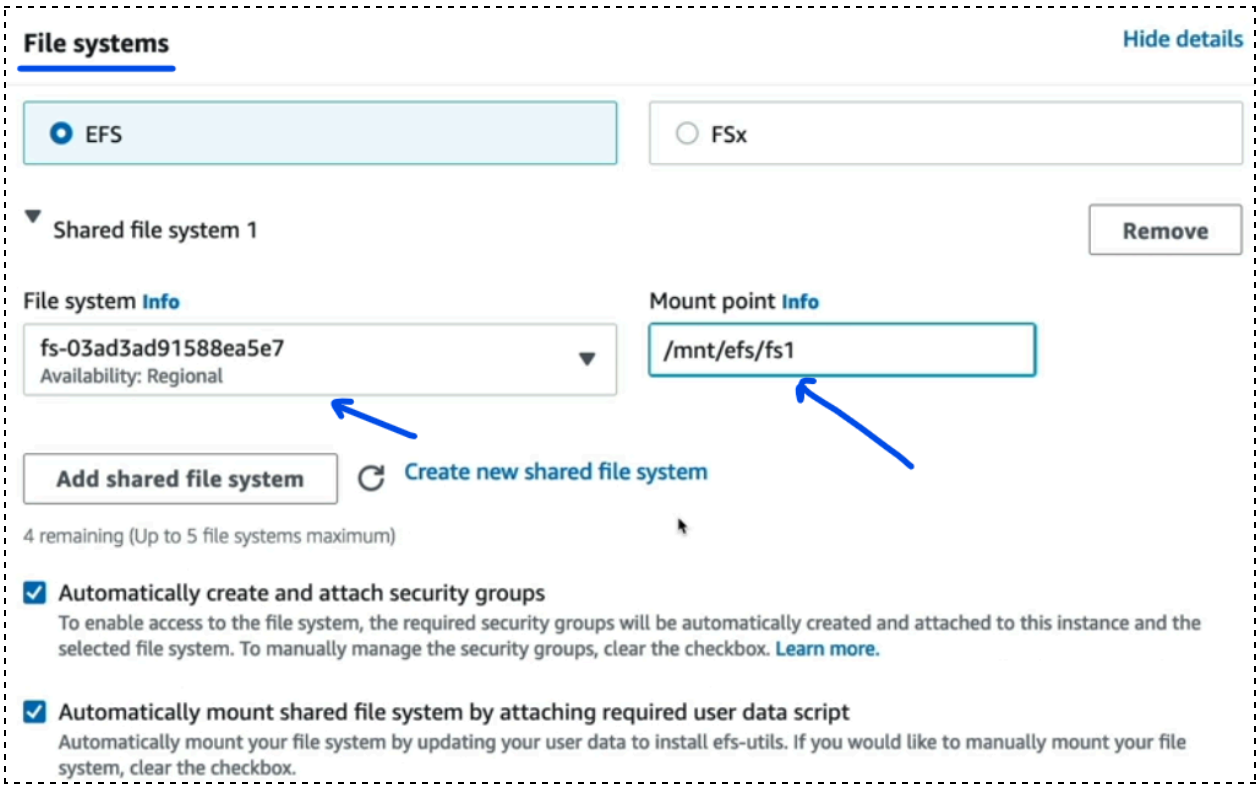


" هكريت مثلا instance A مجرد اسم اول واحدة ياعني .. من ال network
setting هختارلها subnet طب انهي subnet ؟ هختار ال subnet اللى هيا
us-east-1a وهكذا instance B هختارلها us-east-1b"

"دي كدا وانت بتكريت ال EFS بتختار ال security group وال subnet وكدا"

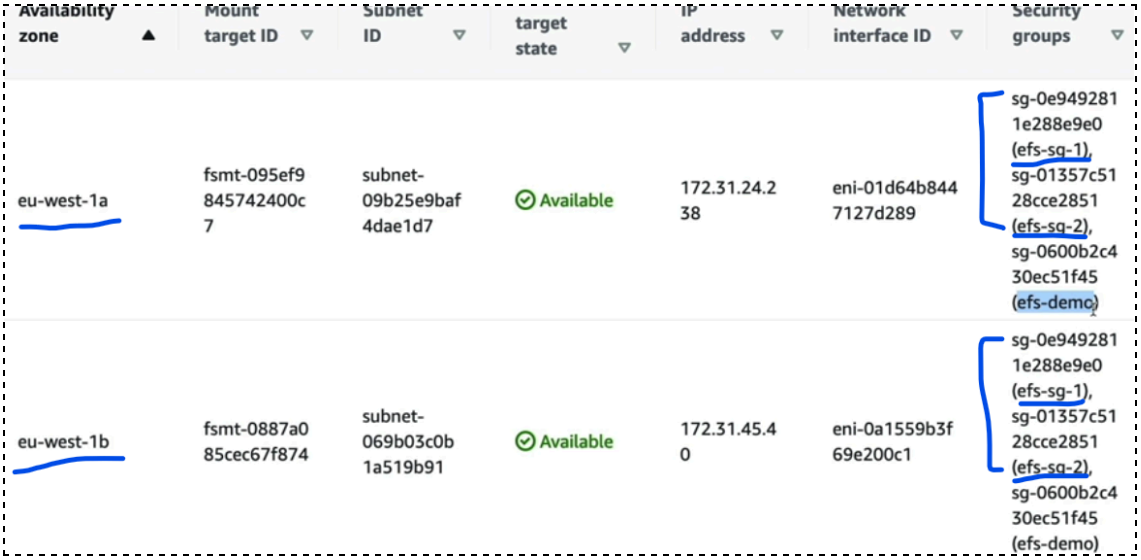


💡 بتختار نفس ال file system لل instances لان ممكن يكون عندك اكتر من file system وبتختار نفس ال mount point

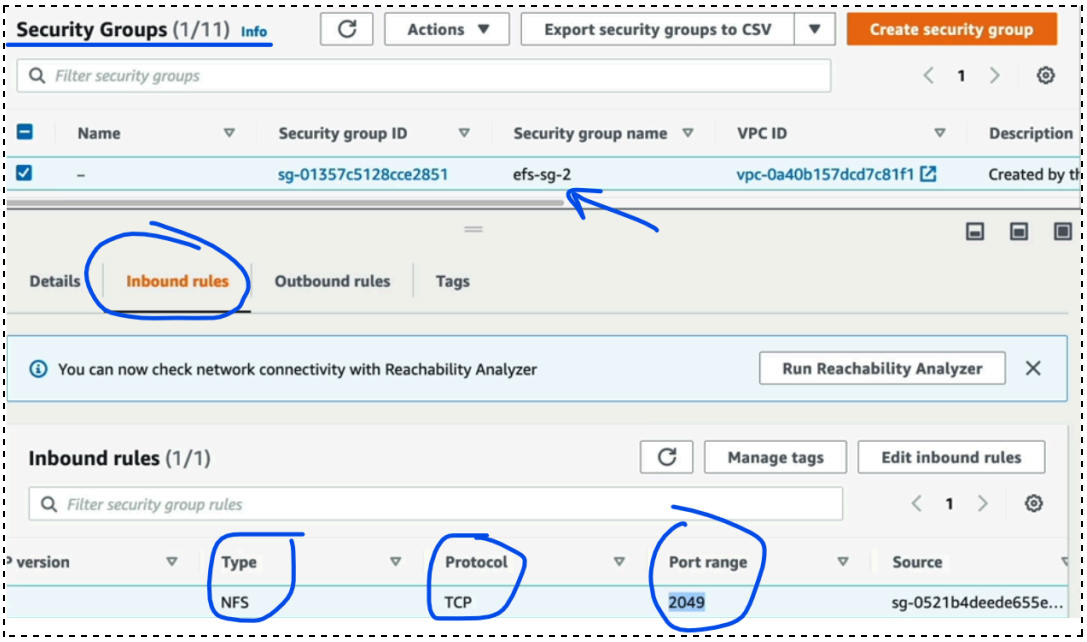


💡 تلقائي بيتعمل security groups عال AZs اللى بتحتوي عال instances بتاعتك اللى مرتبطة ب file system واحد .. ياعني ال3 instances اللى عملناهم دلوقتي كل واحدة كانت فى AZ مختلفة ف لو دخلت على ال EFS هتلاقي ان فيه بالاضافة لل security group اللى عملناه فيه by default security groups اتعملت ال security groups اللى اتعملت تلقائي دي ولو دخلت عال security groups دي هتلاقي انها فى ال inboung rules بتسمح

ب protocol NFS اللى اتكلمنا عنه قبل كدا.



Availability zone	Mount target ID	Subnet ID	target state	IP address	Network interface ID	Security groups
eu-west-1a	fsmt-095ef9845742400c7	subnet-09b25e9baf4dae1d7	Available	172.31.24.238	eni-01d64b8447127d289	sg-0e9492811e288e9e0 (efs-sg-1), sg-01357c5128cce2851 (efs-sg-2), sg-0600b2c430ec51f45 (efs-demo)
eu-west-1b	fsmt-0887a085cec67f874	subnet-069b03c0b1a519b91	Available	172.31.45.40	eni-0a1559b3f69e200c1	sg-0e9492811e288e9e0 (efs-sg-1), sg-01357c5128cce2851 (efs-sg-2), sg-0600b2c430ec51f45 (efs-demo)



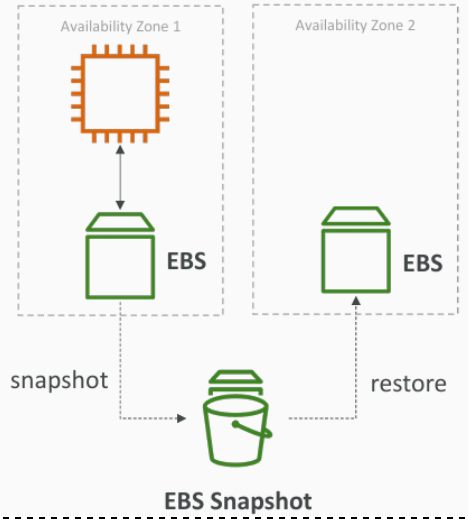
💡لو روحت بقا دخلت على اول instances وكريتت file فى الmount point اللى هيا mnt/efs/fs1/ وسجلت خروج ودخلت عال instance التانية او التالته ودخلت على نفس ال mount point هتلاقي نفس ال file هناك .. ف ال 3 مشتركين فى نفس ال mount point على نفس ال file system رغم انهم فى different AZs.

➥013 EBS vs EFS:

- إتكلمنا ف كل دة قبل كدا دة بس ملخص سريع.

- **EBS**

- EBS volumes…
  - one instance (except multi-attach io1/io2)
  - are locked at the Availability Zone (AZ) level
  - gp2: IO increases if the disk size increases
  - gp3 & io1: can increase IO independently
- To migrate an EBS volume across AZ
  - Take a snapshot
  - Restore the snapshot to another AZ
  - EBS backups use IO and you shouldn't run them while your application is handling a lot of traffic
- Root EBS Volumes of instances get terminated by default if the EC2 instance gets terminated. (you can disable that)

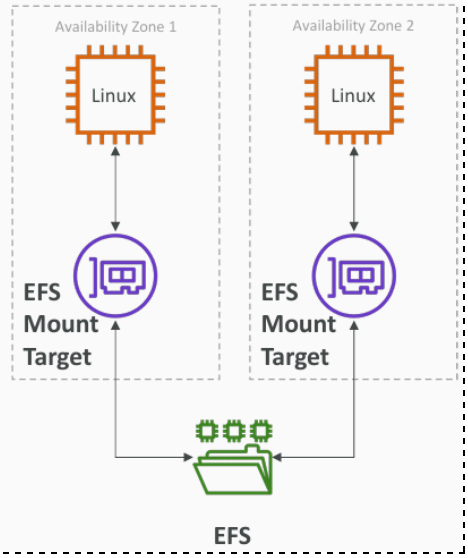


• EFS

- Mounting 100s of instances across AZ
- EFS share website files (WordPress)
- Only for Linux Instances (POSIX)

- EFS has a higher price point than EBS
- Can leverage Storage Tiers for cost savings

- Remember: EFS vs EBS vs Instance Store



➥014 EBS & EFS - Section Cleanup

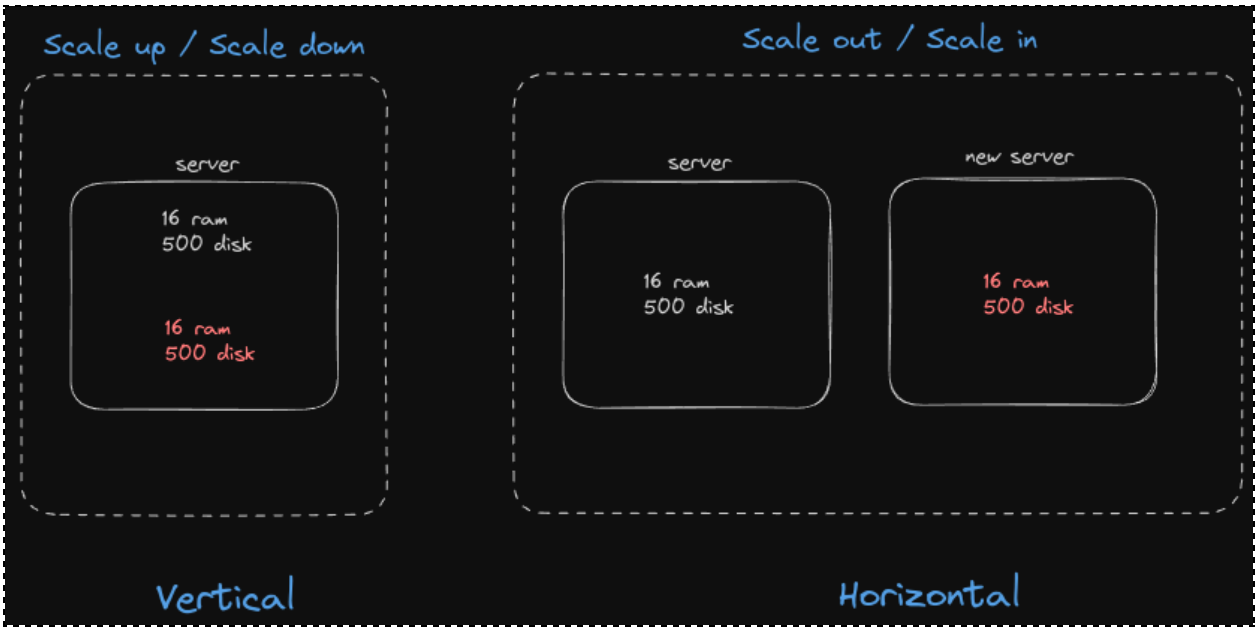
- هتمسح كل حاجة instances,volumes,snapshots.. كدا كدا انت المفروض بتمسح اول باول علشان التكلفه.

CH-08 - High Availability and Scalability ELB & ASG

➥001 High Availability and Scalability

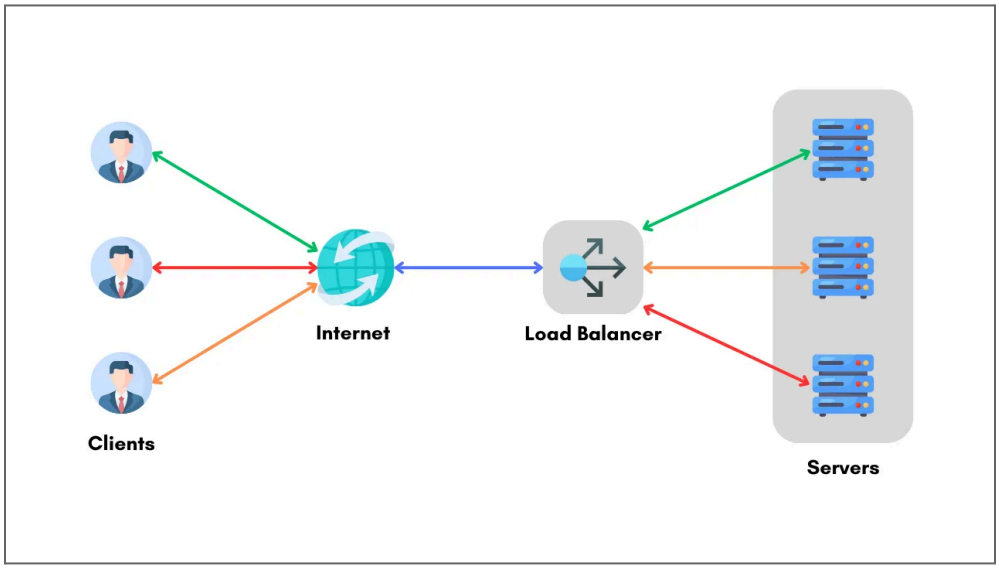
- **Scalability:**
  - قدرة ال system على انو يهندل ال increased load اللى جايله عن طريق adding resources **وعندك منها نوعين :**

- **Vertical Scaling (Scaling Up/down):**
  - اضافة power CPU,RAM على نفس ال Machine
- **Horizontal Scaling (Scaling Out/in)**
  - اضافة Machines زيادة



- **Availability:**
  - قدرة ال system على انو يفضل up and running ودة بيتم عن طريق انو يكون ليه redundancy, failover mechanisms, and reliable infrastructure.
    موجود ف كذا AZs مثلا علشان لو instance وقعت التانيه تشتغل وهكذا.

➥002 Elastic Load Balancing (ELB) Overview



- Load Balancer : عباره عن server او مجموعة من ال servers وظيفتها انها توزع ال load او ال traffic اللى جايلها على مجموعة من ال back end servers بنسميهم downstream instances .. موضوع ال load balancing دة AWS بتقدمه ك service اسمهاElastic Load Balancer

Key Diff Between AWS LB & Nginx LB

Key Differences

1. **Management and Maintenance:**
   - **NGINX:** Requires manual setup, configuration, and maintenance.
   - **ELB:** Fully managed by AWS, reducing operational overhead.
2. **Scalability:**
   - **NGINX:** Manual scaling and configuration required.
   - **ELB:** Automatically scales to handle traffic fluctuations.
3. **Availability and Redundancy:**
   - **NGINX:** Requires additional setup for high availability and failover.
   - **ELB:** Built-in high availability and fault tolerance.
4. **Customization and Control:**
   - **NGINX:** Greater control and customization options.
   - **ELB:** Limited to AWS-provided configurations but sufficient for most use cases.
5. **Cost:**
   - **NGINX:** Potentially lower costs but higher management overhead.
   - **ELB:** Pay-as-you-go pricing, potentially higher costs but lower management overhead.

Types of load balancer on AWS

- **Classic Load Balancer (v1 - old generation) – 2009 – CLB**
  - **Support Protocols:** HTTP, HTTPS, TCP, SSL
- **Application Load Balancer (v2 - new generation) – 2016 – ALB**
  - **Support Protocols:** HTTP, HTTPS, WebSocket
- **Network Load Balancer (v2 - new generation) – 2017 – NLB**
  - **Support Protocols:** TCP, TLS (secure TCP), UDP

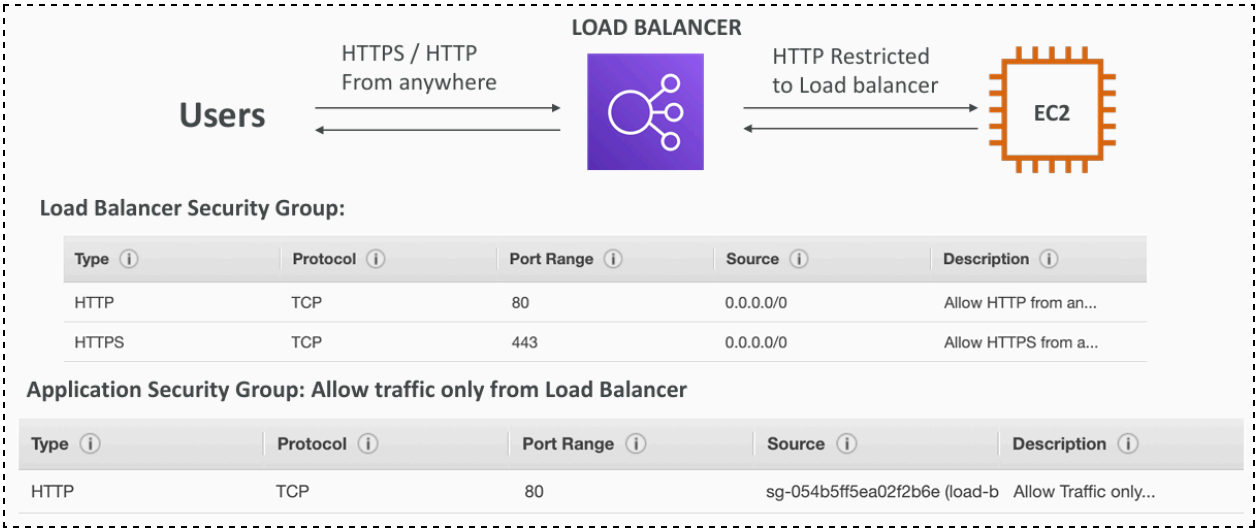
- **Gateway Load Balancer – 2020 – GWLB**
- Operates at layer 3 (Network layer) – IP Protocol
- هنتكلم عن كل نوع بالتفصيل تحت.

💡 Recommended انك تستخدم ال new generations بيبقي فيهم features اكتر.

💡 Some load balancers can be setup as internal (private) or external (public) ELBs

Load Balancer Security Groups

- من ال user لل load balancer بتسمح فى ال security group بال http/https requests اللى جايه من anywhere
- أما من ال load balancer لل instances ف بتسمح فقط بال requests اللى جايه عن طريق ال load balancer ودة بيحصل عن طريق انك بتحدد فى ال security group بتاع ال instance ان ال source هو ال security group بتاع ال load balancer



Load Balancer Security Group:

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	0.0.0.0/0	Allow HTTP from an...
HTTPS	TCP	443	0.0.0.0/0	Allow HTTPS from a...

Application Security Group: Allow traffic only from Load Balancer

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	sg-054b5ff5ea02f2b6e (load-b	Allow Traffic only...

➥003 Note About the 1-Classic Load Balancer (CLB)

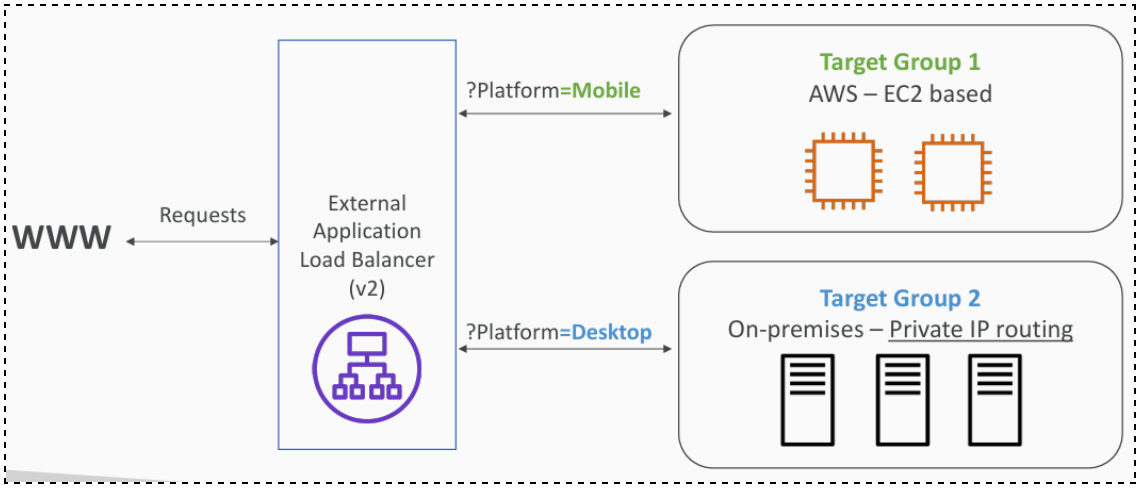
- **The Classic Load Balancer** is **deprecated** at AWS and will soon not be available in the AWS Console.
- The exam also has removed any references to it, so we will not cover it in depth in the course, nor the hands-on.

➥004 2-Application Load Balancer (ALB)

● Key Features

- بيشتغل ف layer 7 وبيهندل ال traffic اللى جاي عليه بناءا على rules كتيره specifications
- بيدعم ال SSL/TLS Termination

● Target Groups:



- عباره عن Backend Component بتكون خلف ال load balancer وممكن تكون اي حاجة من دول :

- **EC2 instances** (can be managed by an Auto Scaling Group) – HTTP
- **ECS tasks** (managed by ECS itself) – HTTP
- **Lambda functions** – HTTP request is translated into a JSON event
- **IP Addresses** – must be private IPs – must be private IPs

💡 ALB can route to multiple target groups

💡 Health checks are at the target group level

- **Listeners and Rules:**

- ال **Listener Rules** دي عباره عن ان ازاي ال load balancer هيتعامل مع ال requests اللى جايه من ال user ويعملها routing على ال Target groups ودة بيعتمد علي بعض ال conditions :

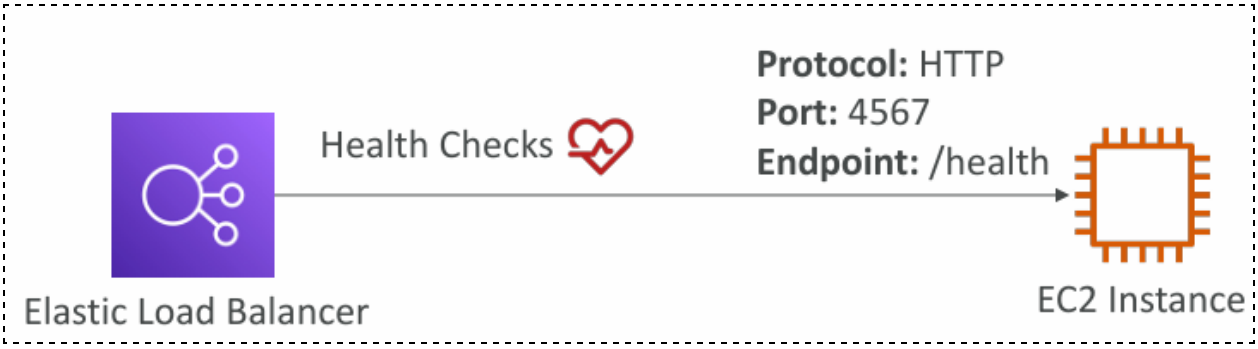
- **Route based on URL PATH**

```
If the URL path is /images/*, >> route traffic to Target Group 1.
If the URL path is /videos/*, >> route traffic to Target Group 2.
```

- **Route Based on Host Name**

```
If the hostname is api.example.com,>>route traffic to Target Group 1.
If the hostname is www.example.com,>>route traffic to Target Group 2.
```

- **Health Check**



- عباره عن process ال load balancer بيعملها بانتظام علشان يتأكد ان ال Targets قادرة انها تـ respond ال requests اللى هو باعتهالها ولا لا.
- سواء بتستخدم aws load balancer او load balancer software على سيرفرك زي nginx فانت بت configure ال options الخاصة بال health check زي interval, timeout وكدا
- لما بتـكريت Targets group بتقدر انك تعمل .. health check configuration والخطوة الجاية هيتوضح ياعني اية Health Check اكتر.

- **Health Check Configuration:**
  - **Configuration:**
    - **Target Type**: EC2 instances
    - **Health Check Protocol**: TCP
    - **Protocol**: HTTP
    - **Path**: /healthcheck
    - **Port**: 80
    - **Interval**: 30 seconds
    - **Timeout**: 5 seconds
    - **Healthy Threshold**: 2

- **Unhealthy Threshold**: 2
  - فـ المثال دة ال load balancer هيبعت كل 30 ث request لـ

```
http://<target-ip>:80/healthcheck
```

  - لو الـ request دة اتعملو respond 200 ok فـ خلال 5 ث ..

```
It is considered healthy.
```

  - لو الـ request دة اتعملو fails فـ خلال 5 ث أو response with error ..

```
It is considered unhealthy.
```

  - Unhealthy Threshold: لو ال Target حصله fail فـ 2 health checks متتابعين ..

```
It will be marked as unhealthy and stops receiving traffic.
```

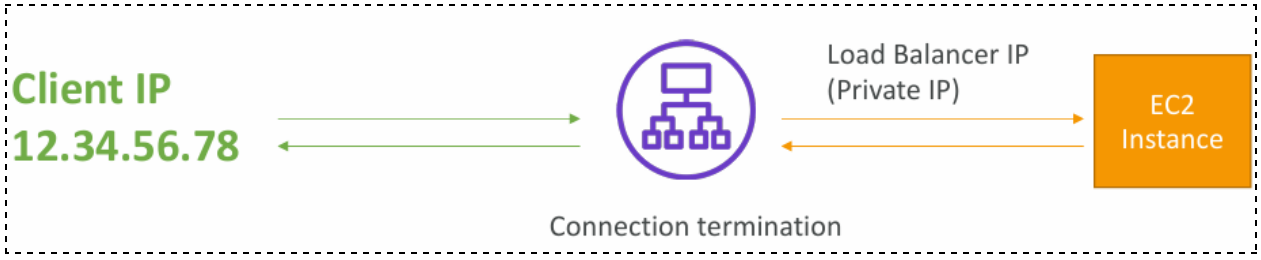
  - Healthy Threshold: لو ال Target حصله success فـ 2 health checks متتابعين .. وبلاش 1 دة مش مؤشر قوي ان ال instance تستحمل ولالا

```
it is marked as healthy and starts receiving traffic again.
```

💡 if the health check found the instance not working it will be terminated then replaced

💡 Health Checks support the TCP, HTTP and HTTPS Protocols.

- **‘X-Forwarded-For’ Header (How it works?):**



1. الـ client لما بيعمل request بيروح لل >> Application Load Balancer
   ALP دة بيشتغل ف layer 7 وبيهندل ال HTTP , HTTPS Traffic
2. الـ ALP بيضيف للـ request ال X-Forwarded-For Header
3. الـ ALP بيـ forward الـ request بعد ما انضافلة ال Header للـ backend targets

```
GET / HTTP/1.1
Host: www.example.com
X-Forwarded-For: 192.0.2.1
```

4. دا كدا شكل ال request اللى رايح لل backend targets بعد ما انضافلة ال Header

💡 الفايدة من الموضوع دة إن ال backend target بيـ

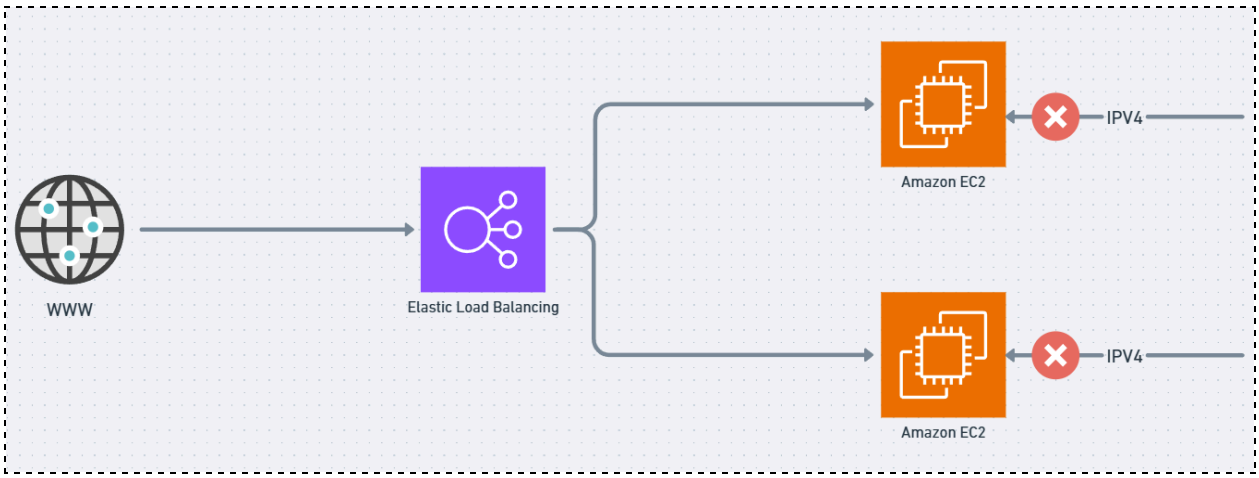
Can identify the original client's IP address for logging, analytics, and security purposes, Helps in tracing the request path through multiple proxies.

➥005 Application Load Balancer (ALB) - Hands On - Part 1

هتكريت 2 instances وال SG بتاعهم هتسمح فيه ان ال connections http
تيجي عن طريق any where وهتكريت ال load balancer وهتسمح ال
connection اللى داخلاله انها تكون http من any where واللى خارجة منه بردو
all all وبالتالي كدا قادر ال load balancer يستلم ويوزع عال ec2 instances
وتقدر تخش عليهم من خلال ال loadbalancer DNS او ال ip addresses بتاع
ال instances

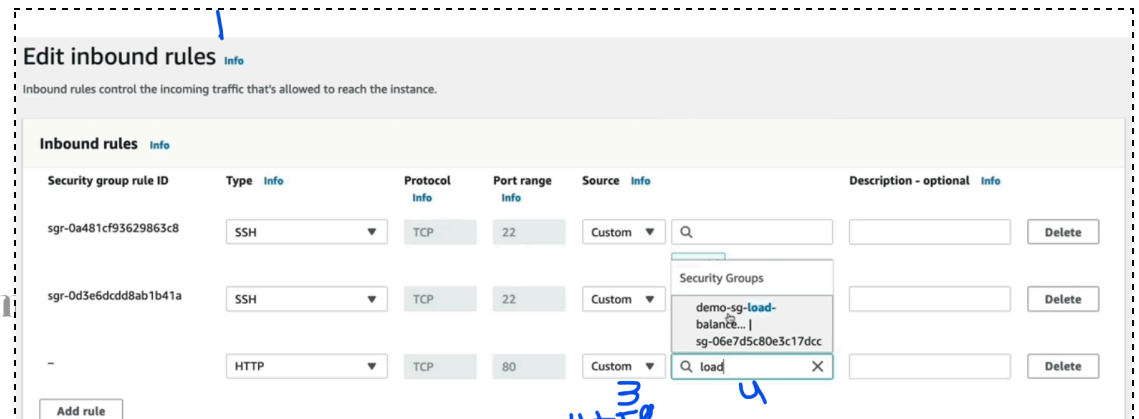
➥006 Application Load Balancer (ALB) - Hands On - Part 2

- **Tip1-** Enter the instances through ALP only



لما اشتغلنا فى Part 1 كنا نقدر ن access ال EC2 Instances سواء من ال IPv4 بتاعهم او تخش عال Load Balancer DNS وهو بيوزعك عليهم .. ف احنا عاوزين نقفل جزأيه ان حد يقدر يخش direct عال EC2 Instances عاوزين بس من خلال ال load balancer

دة هيتم عن طريق ال security group قولنا انها اكنها عباره عن firewall بت control ال inbound , outbound traffic ف هنخش عال security group



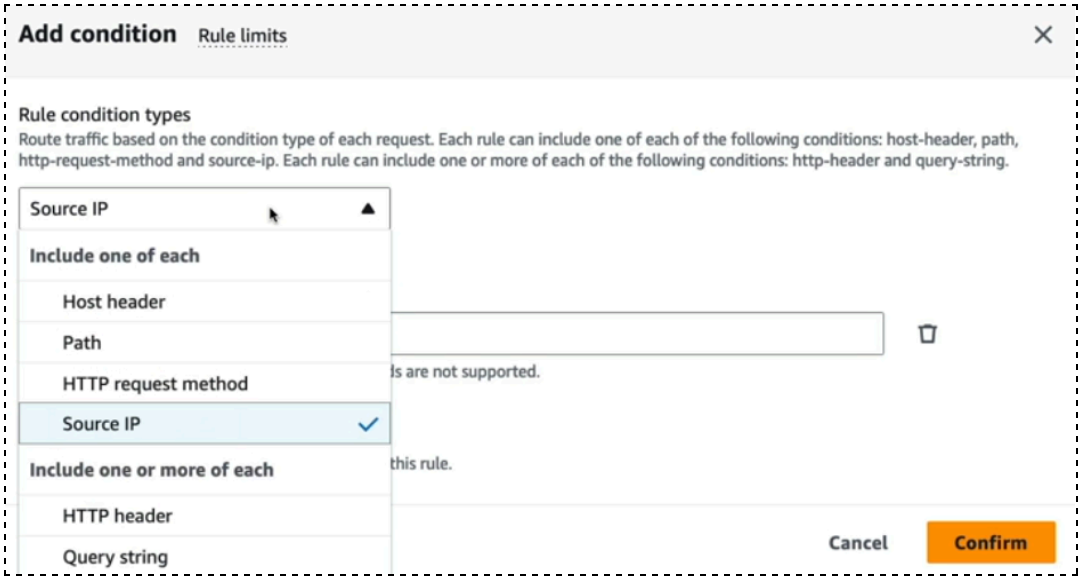
بتاع ال instances ون allow فى ال inbound rules ال HTTP يجيلك فقط من

ال Security group بتاع ال load balancer

دلوقتي لو جيت ت Access ال instances من خلال ال IPv4 هت timeout معاك.

- **Tip2 -** Modify the ALP Listeners & rules

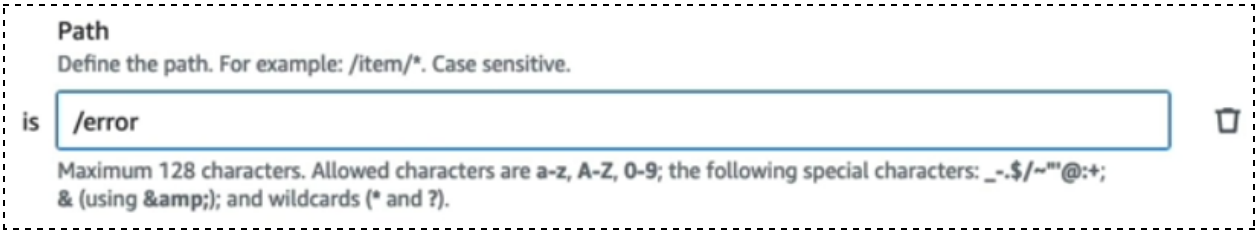
```
Go to ALP >> Listeners >> HTTP-80 >> Listeners rules >> add rules
```

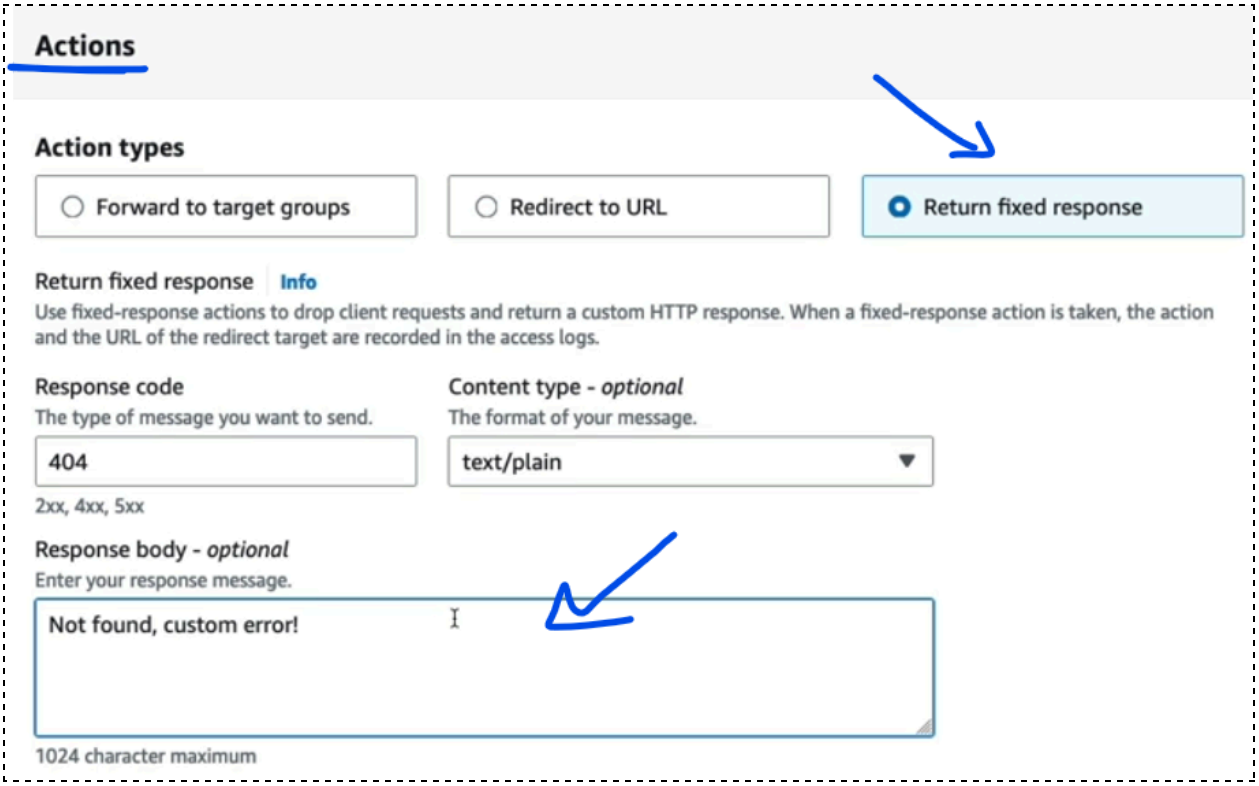


تقدر تحط عدد ال rules اللى انت عاوزها وكل rule بال condition بتاعها .. وبعدين

بتحدد ال action اللى هيحصل بناءا عال condition دة

هنا احنا هنعمل action معين يحصل لو ال user دخل عال path دة

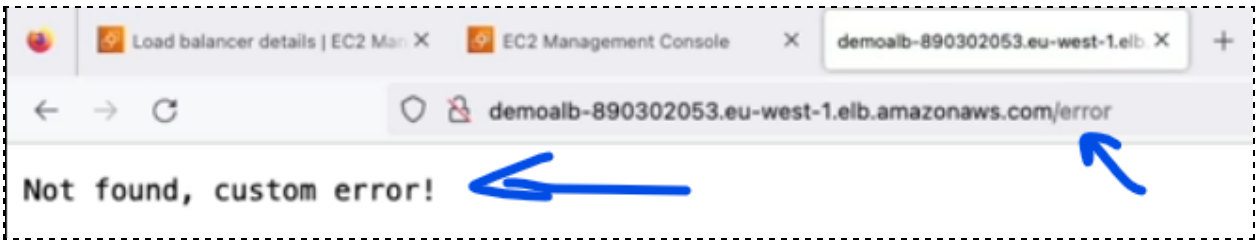




هتلاقي عندك حاجة اسمها priority معناها الأولوية .. فايدتها أننا لو عندنا rule 100

مثلا ال rule اللى الأولوية بتاعتها اعلي هي اللى هتتنفذ

ودي كانت النتيجة



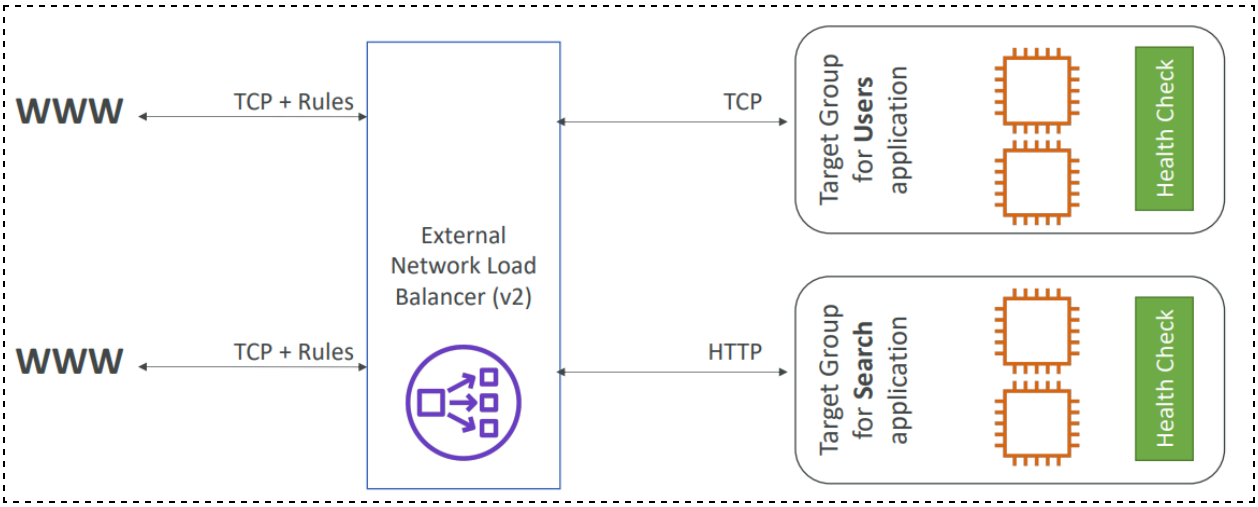
➥007 Network Load Balancer (NLB)

- **Key Features**

- بيشتغل فى Layer 4 وبي forward **TCP** & **UDP** traffic عال targets group
- بيهندل ال high volume of connections .. ودة بيخليه مناسب اكتر لل real-time gaming, media streaming, or financial services.
- ال latency قليلة هنا 100 ms مقارنه بال ALP اللى كان 400 ms

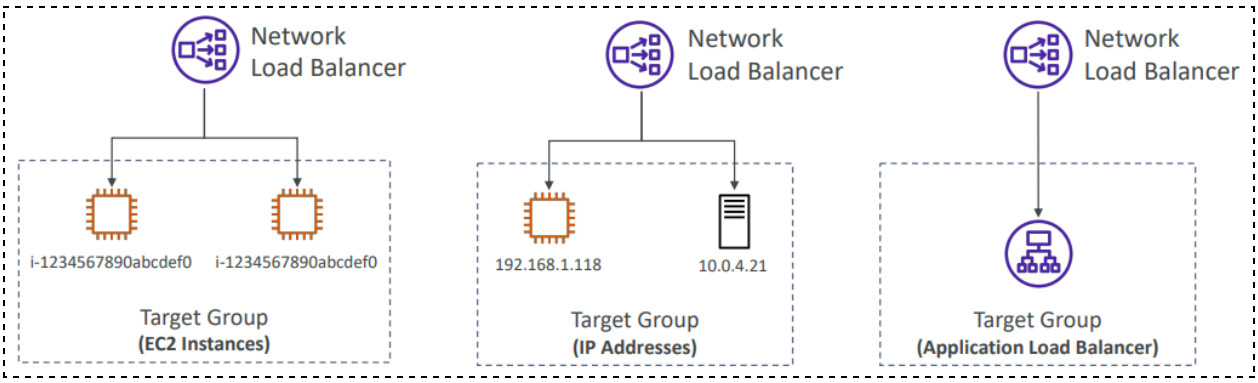
NLB has one **static IP per AZ**, and **supports assigning Elastic IP** (helpful for whitelisting specific IP)

- ميزة ان ال NLP ليه Static IP بتخليك تستخدمها بشكل كويس فـ مثلا لو عندك data base او اي backend services وعاوز تخليها تـ Accept ال Traffic اللى جاي عليها only from ال NLP ف بتروح فالـ firewall بتاعها ت allow ال Traffic اللى جاي من الـ Static IP دة وبس كدا ضمنت ان مفيش اي Traffic هيجيلك غير منه فقط..نفس الكلام وانت بتـ assign ال DNS عال Static IP .. مفيش تغيير ف ال IP ف دة كويس.
- مش متاح ف ال free tier



• Network Load Balancer –Target Groups

- نفس اللى قولناه فوق ممكن تكون EC2 instances او IP Addresses وهنا **الزيادة ممكن يكون ALP**



• IP Addresses

- فى حالة لو ال target group كان ip addresses لازم تكون **Private IPs**

• NLP With ALP As a target group

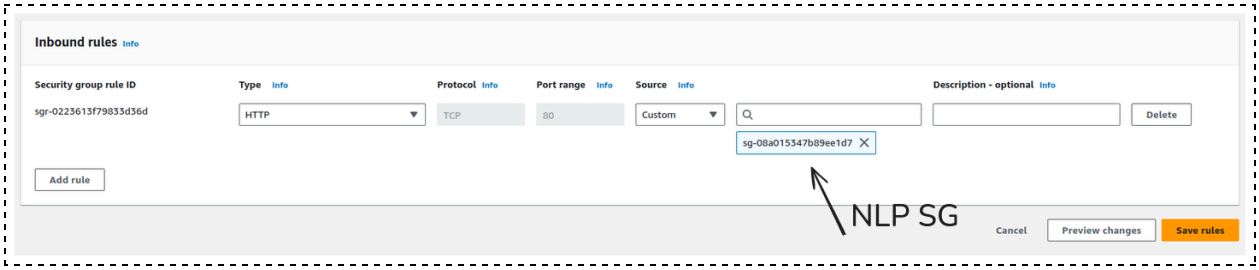
- ال scenario دة بيحصل لما اكون محتاج ال **NLP** يهندل ال TCP/UDP traffic العالي اللى جايلي ويكون فيه low latency وال ALP يديني اوبشنز كتيره علشان ي Handle HTTP/HTTPS traffic

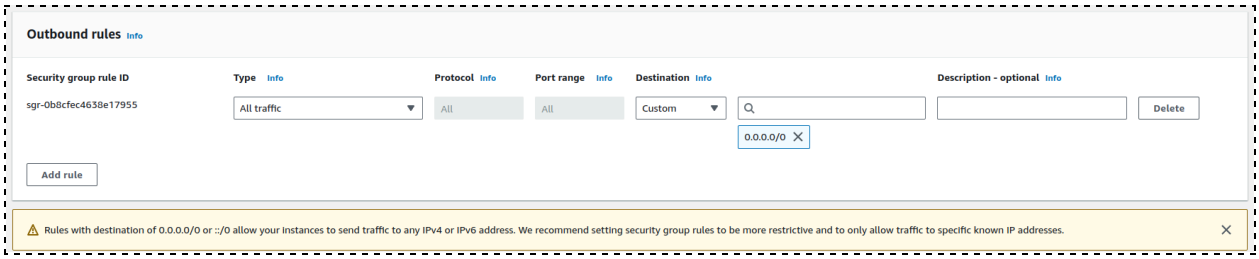
➥008 Network Load Balancer (NLB) - Hands On

- Create >> NLP SG , Instances SG , Instances with user data , target group (wait for the instances to be healthy) , NLP

1. Instances SG >>

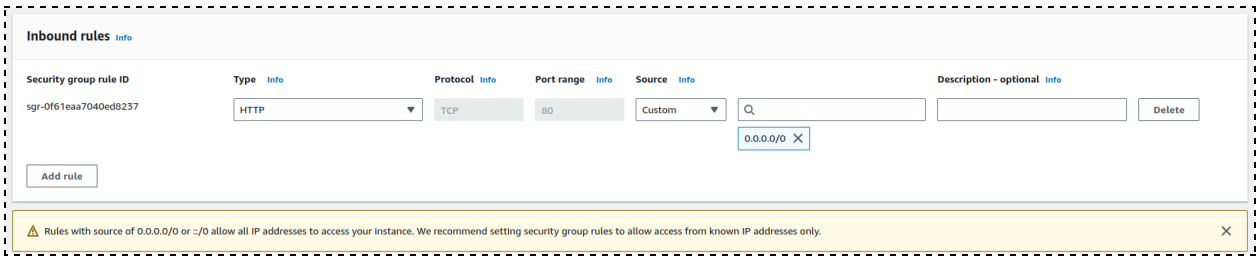
عندنا 2 instances وانت بتكريتهم هتروح الاول تكريت SG تسمح فيه بال http من inbound NLP SG وتحط فى ال user data الكود بتاعك اللى هيعرفك انت بتاكسس انهي instance

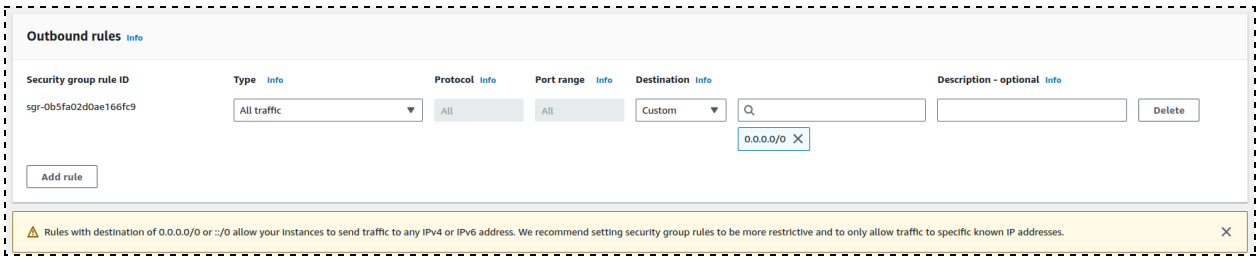




2. NLP SG

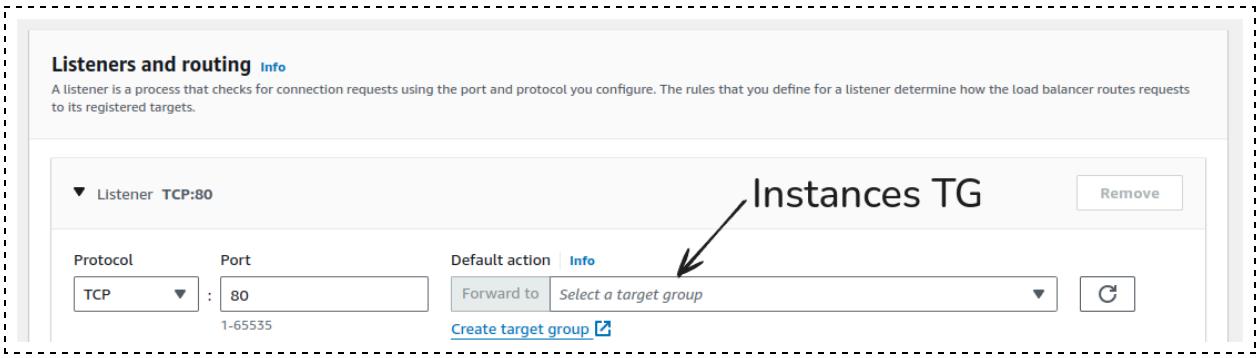
دي ال inbound , outbound الخاصة بيه





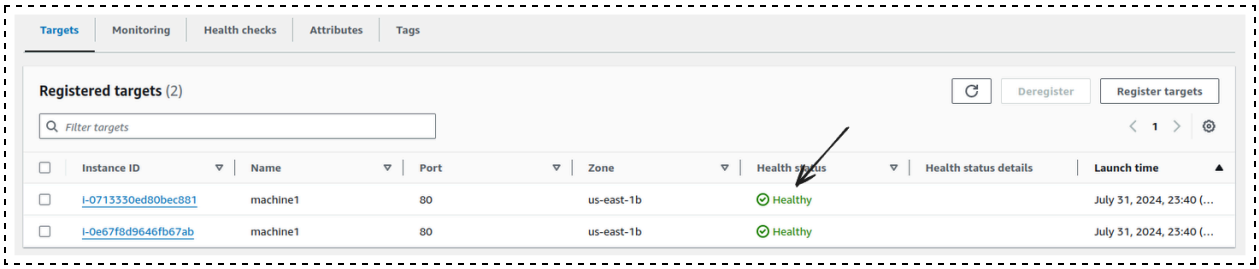
3. NLP

هتختار ال instances target جروب اللى هتوزع عليه



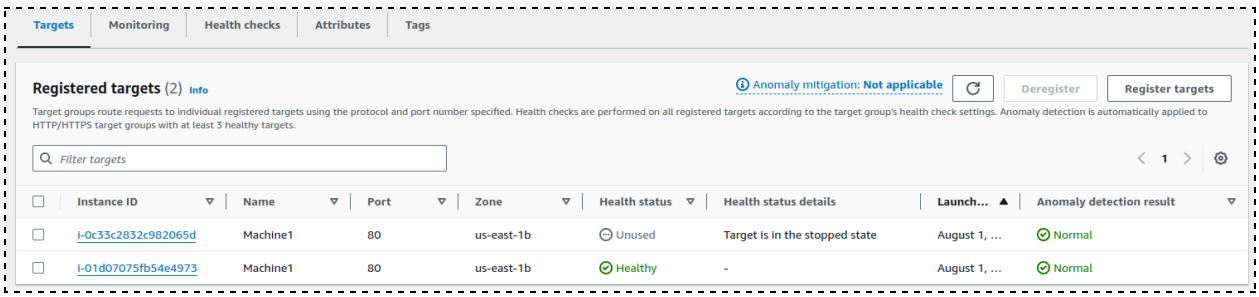
زي مقولنا تستني ال target group توضحلك ان ال instances healthy .. لان

فى ا لبداية بتكون **unhealthy cause of bootstrapping**



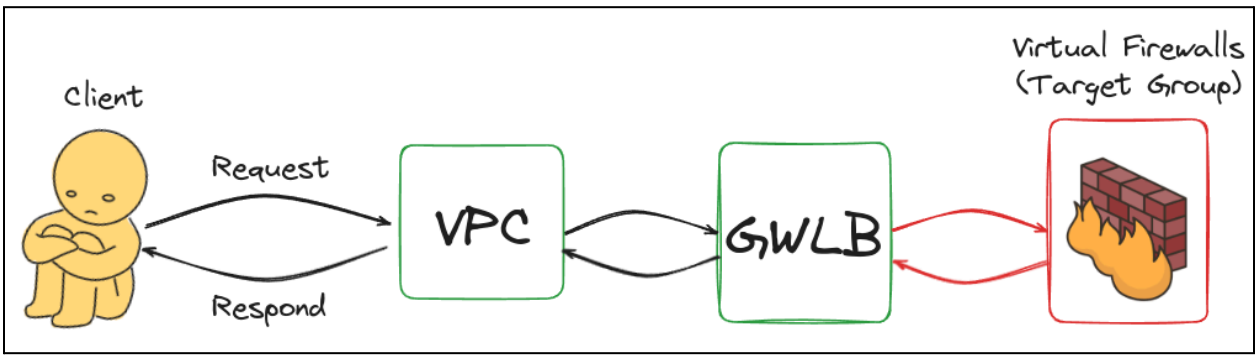
لو عملت stop ل instance وروحت عال target group هتلاقيها بالشكل دة انها فى

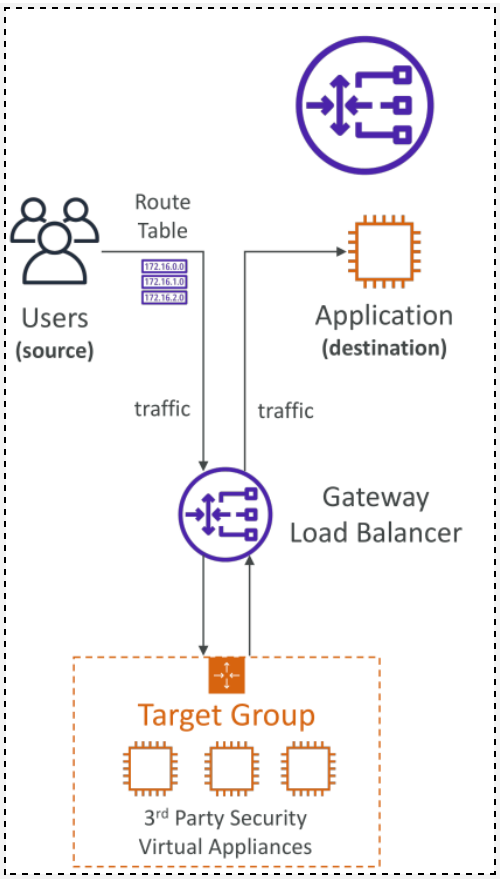
ال stopped state



I'm Here

➥009 Gateway Load Balancer (GWLB)



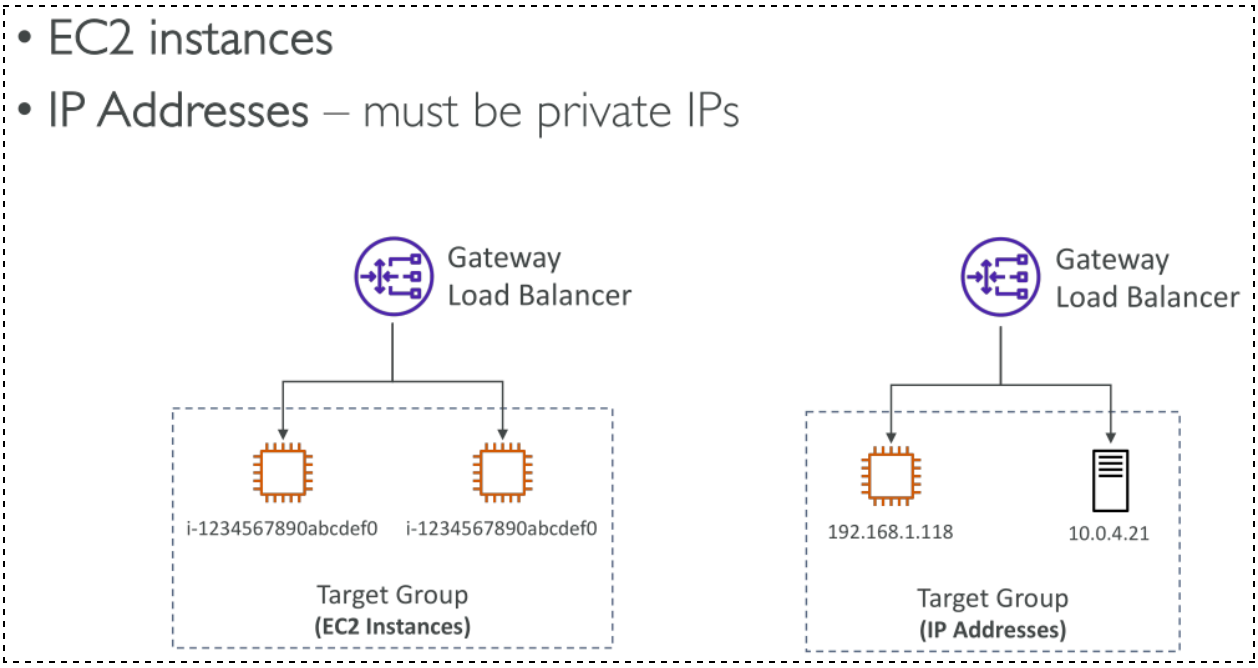


- **النوع ثالث من أنواع ال load balancers .. محتاجين نعرف عنها هنا طريقة عملها :**
  - بيكون فيه Traffic داخل عالAWS infrastructure

- الـ GWLP بتستلم ال Traffic دة وتعمله route علي firewall instance او اكتر من واحد بيكونو داخل target group
- بيتم فلترة ال Traffic دة من خلال ال firewall وبيشوف مين هيتعملو accept ومين reject
- بعدها ال firewall بي route ال accepted Traffic لل GWLB وبترجعه لل AWS infrastructure

هنعمل Hands on ولكن بعدين لان الموضوع معقد شوية.

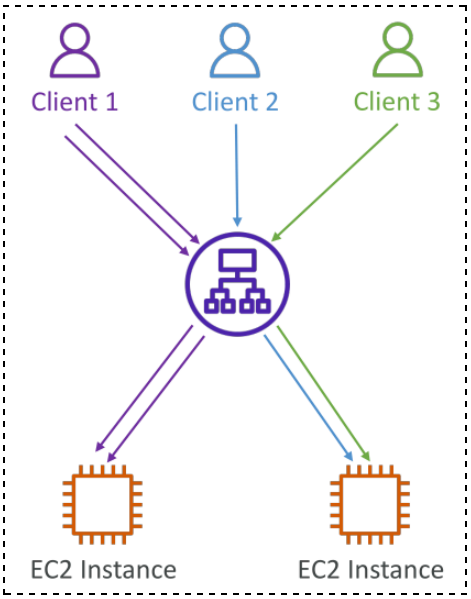
• Gateway Load Balancer –Target Groups



- نفس اللى قولناه فوق ممكن تكون EC2 instances او **Private** IP Addresses

➥010 Elastic Load Balancer - Sticky Sessions

- **Overview:**



- فى الطبيعي ال load balancer سواء ال classic , network , application بيوزعو ال traffic عال instance الاولي ثم التانية ثم التالته وهكذا .. فى الحالة دي لو استخدمت ال sticky sessions اللى هيتم ان ال client لو بعت 2 request لل load balancer ف هيوديه على نفس ال instance مش هيوزعو

- **Cycle Scenario:**

- الuser بيبعت لل load balancer ال request ف بيحولو على اي server فال server بي generate session ID ويبعتها لل browser على هيئة cookie وال session ID دة بيكون unique لل user حتي ال load balancer من خلال ال session ID الموجود فى ال cookie بيعرف هيودي ال same requests على انهي server

- **عندنا بقا نوعين من ال cookies :**

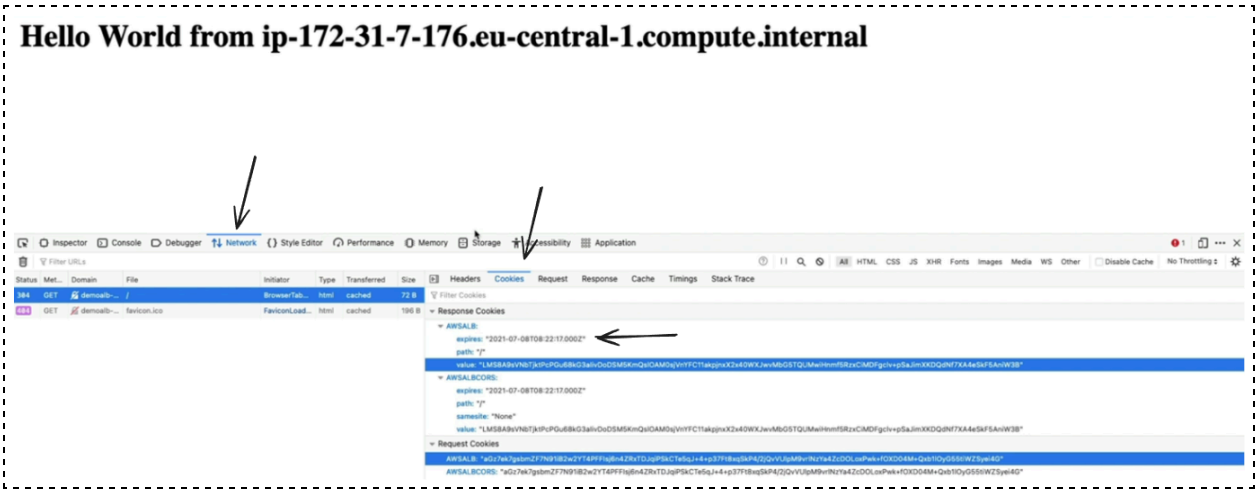
1. Application-based Cookies

- ال user لما بيبعت request لل server وال server ي generate cookie with session ID ال browser بيخذن ال cookie دة بال session id اللى جواه ويفضل الموضوع مستمر ان ال user دة بال session id بتاعة مرتبط ب specific server بيتحول عليه ودة مش هيتغير .
- طب اية بيحصل لما بمسح cookies ال browser ؟ اللى بيحصل ان ال session دي خلاص اتقفلت ف هترجع بقا من تاني تبعت request وسيرفر جديد تتحول عليه دا بمزاج ال load balancer بناءا عال load وال server هي generate cookie & session id جداد وتبدا من جديد حتي تلاقي ان ال account عمل sign out من الموقع اللى بتستخدمه او ان ال themes بتاعة المتصفح اتمسحت او ال shop cart فضيت وهكذا

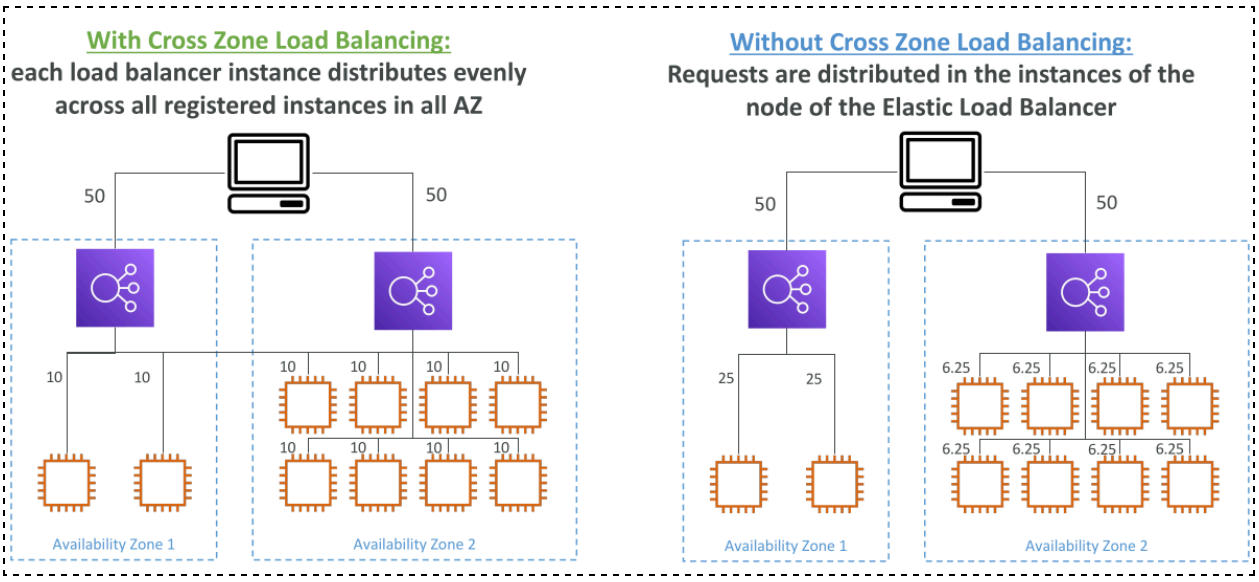
2. Duration-based Cookies

- لمدة معينه انت بتحددها وليكن مثلا لمدة 30 دقيقة ال user هيفضل يستخدم نفس ال cookie مش هتتجدد وهيفضل يتحول على نفس ال instance

هتلاقي حتي فى ال requests , responses فى جزأ ال network & cookies فى ال browser بتنتهي امتي وهكذا.

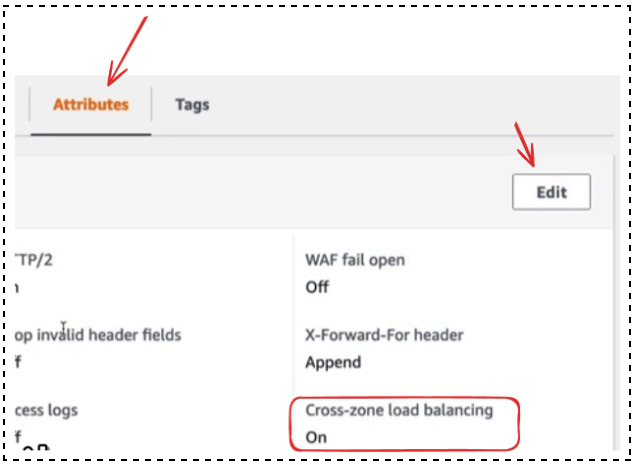


➥011 Elastic Load Balancer - Cross Zone Load Balancing

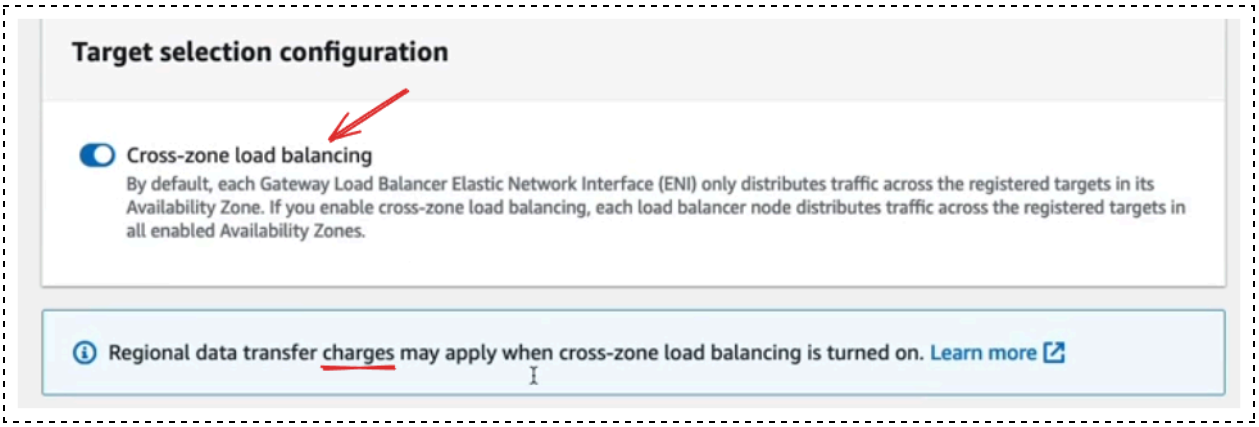


- ال Traffic اللى جاي لو انت مفعل ال Cross zone LB فى الحالة دي هيتوزع عال 10 instance بالتساوي اما لو مش مفعله ف ال traffic اللى جاي على AZ معينه هيتقسم عال instances اللى جواها فقط ومش هيخرج براها.
- بنستخدمه علشان الدنيا تمشي بالتساوي وعلشان مفيش instance تقع مننا.
- بتلاقيه **enabled by default** فى حالة ال **ALB**
- بتلاقيه **disable by default** فى حالة ال **NLB , classic**
- بالنسبة للدفع لو متفعل هتدفع فقط فى ال **NLB**

تقدر بعد ما تخش عال load balancer تعرف هو enable ولا disable من هنا

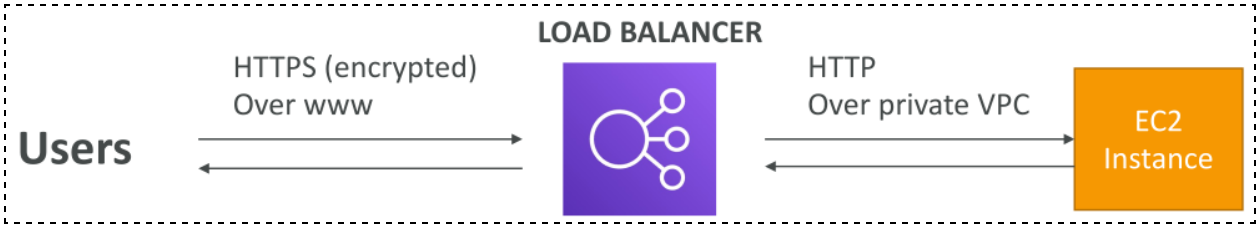


ومن edit تقدر تخليه enable or disable



➥012 Elastic Load Balancer - SSL Certificates

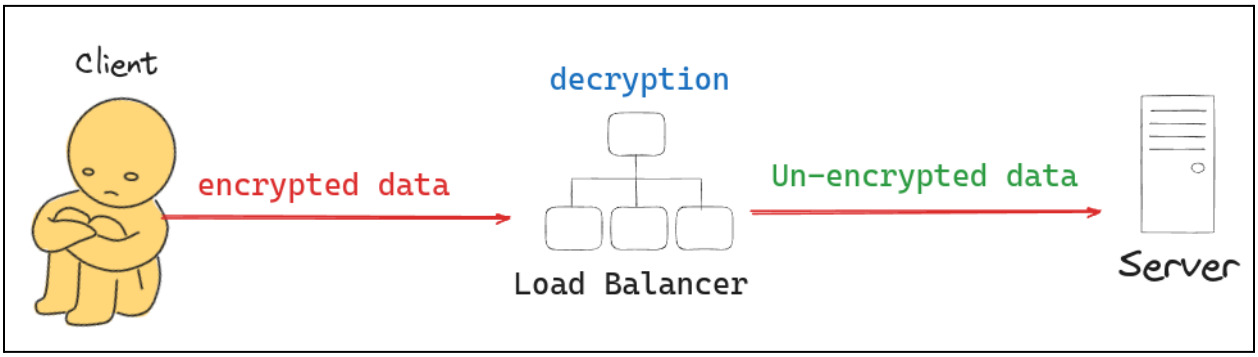
- **Load Balancer SSL Certificate (Secure Sockets Layer):**



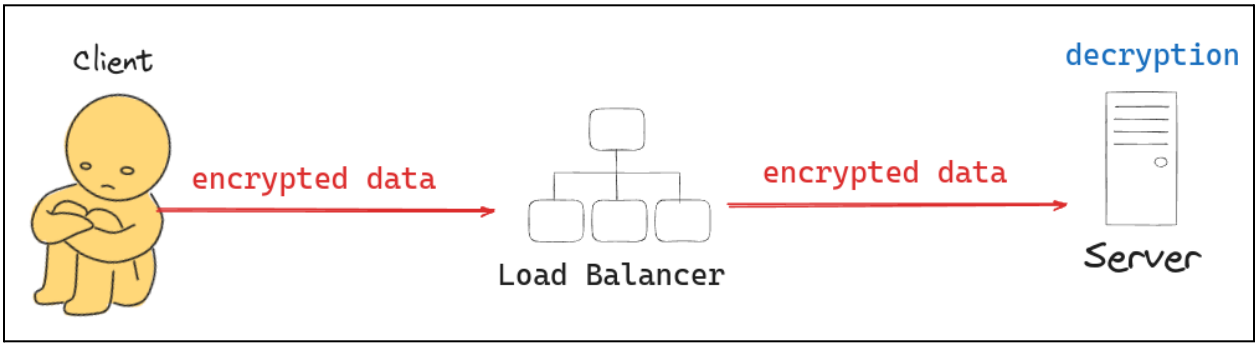
- بنستخدمها علشان نـ encrypt , protect ال connection بين ال user والـ load balancer

- الـ **SSL** : هو older protocol بيـ secure الـ internet connection
- الـ **TLS** : الأحدث و more secure وحاليا مستخدم أكتر ولكن شائع اننا بنقول عليه SSL

- **SSL Termination:**



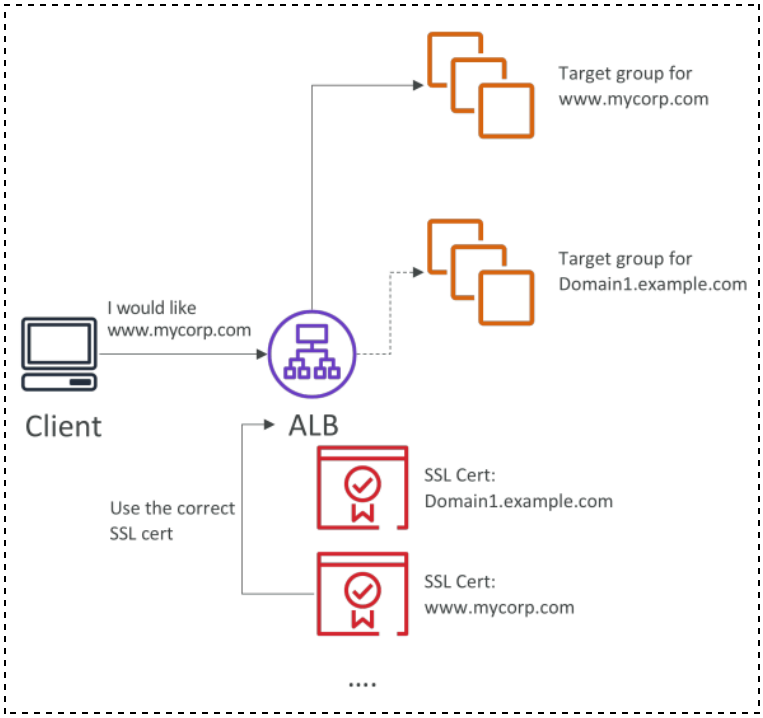
- الـSSL/TLS traffic بيجي من ال user معموله encrypt فالـ load balancer بيعملو decrypt وبيبعت الـ unencrypted data دي لل backend servers ودة فايدته انو بيخفف الحمل عال backend servers انو ميعملش decrypt وكمان بتكون محتاج SSL Certificate واحدة بس عال load balancer مش لكل back end server واحدة.



- **SSL Passthrough:**

- الـSSL/TLS traffic بيجي من ال user معمولة encrypt وبيفضل كدا لحد ما يوصل لل backend servers .. ودة فايدته ان انت حققت End-to-End Encryption على طول المسار ال data protected.

- **AWS Certificate Manager (ACM):**
  - عباره عن AWS Service بتـ simplify الـ provisioning, managing, and deploying SSL/TLS certificates for use
  - او ممكن انت manual ت upload ال certificate details بتاعتك
- **Server Name Indication (SNI):**



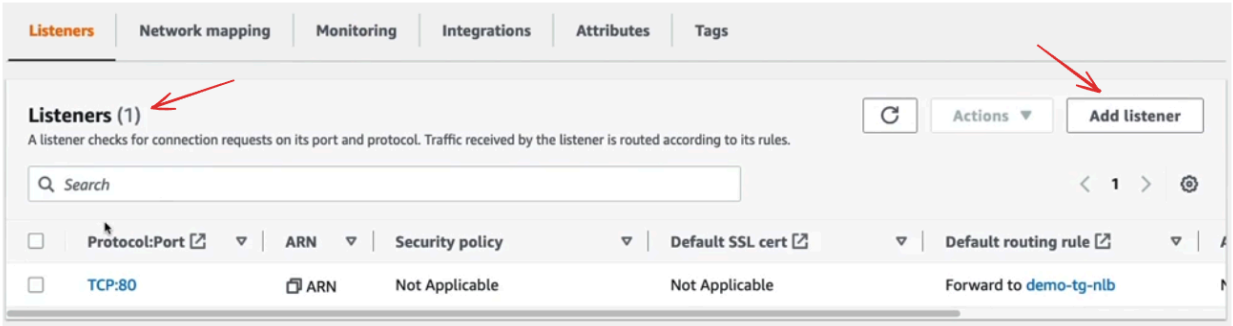
- لو عندي server واحد فقط وعليه كذا website وليكن mycorp.com و Domain1.example.com وهكذا .. وكل website ليه certificate **ف ازاي ال load balancer هيعرف ان ال certificate الفلانية تبع الموقع الفلاني؟**

1. ال client بيبدا SSL/TLS handshake اللى بيكون فيه ال host name اللى عاوز يوصله وضمن ال client hello message بردو بيكون فيه ال SNI Field
2. ال load balancer بيستلم ال client hello message ومن خلال ال SNI Field اللى فيها بيعرف هو يقصد أنهي Certificate
3. بيروح ال load balancer عال domain names اللى معمولها configure عنده ويعرف ال certificate دي تبع انهي موقع (host name) ويوصل ال client بيه

4. بمجرد ما بيعرف ال load balancer بياخد بقا ال data دي سواء هطبق عليها مبدأ ال ssl termination or ssl passthrough ويبعتها لل backend servers

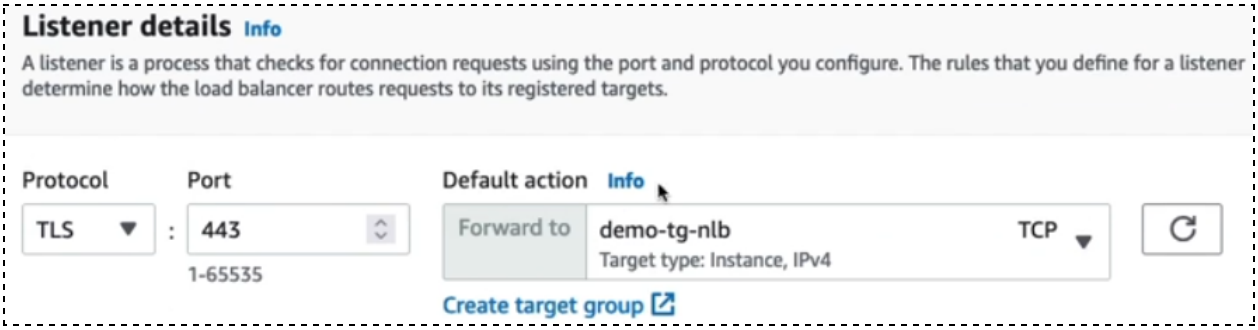
- **Only works for** ALB & NLB (newer generation), CloudFront, Does not work for CLB (older gen).

➥013 Elastic Load Balancer - SSL Certificates - Hands On

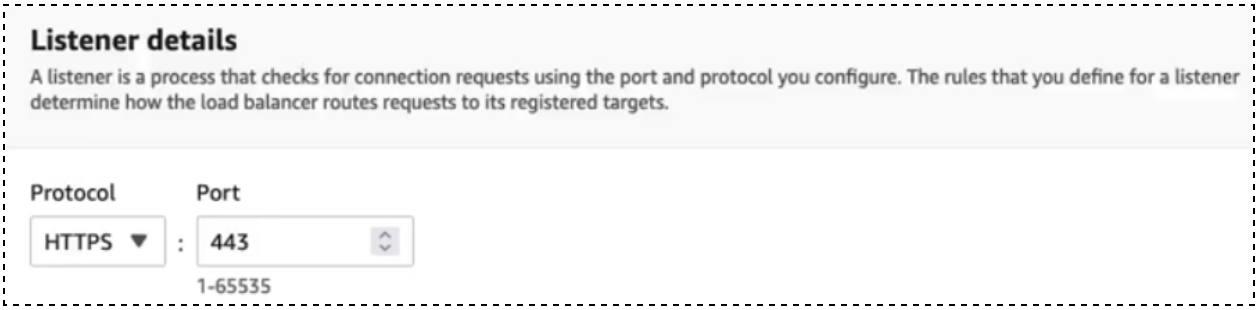


بتخش عال load balancer بتاعك سواء ALB , NLB وت Add listener

فى حالة **NLB** هتختار TLS بدل ما كنا بنختار TCP



وفى حالة ال **ALB** هتختار HTTPS بدل HTTP



➥014 Elastic Load Balancer - Connection Draining

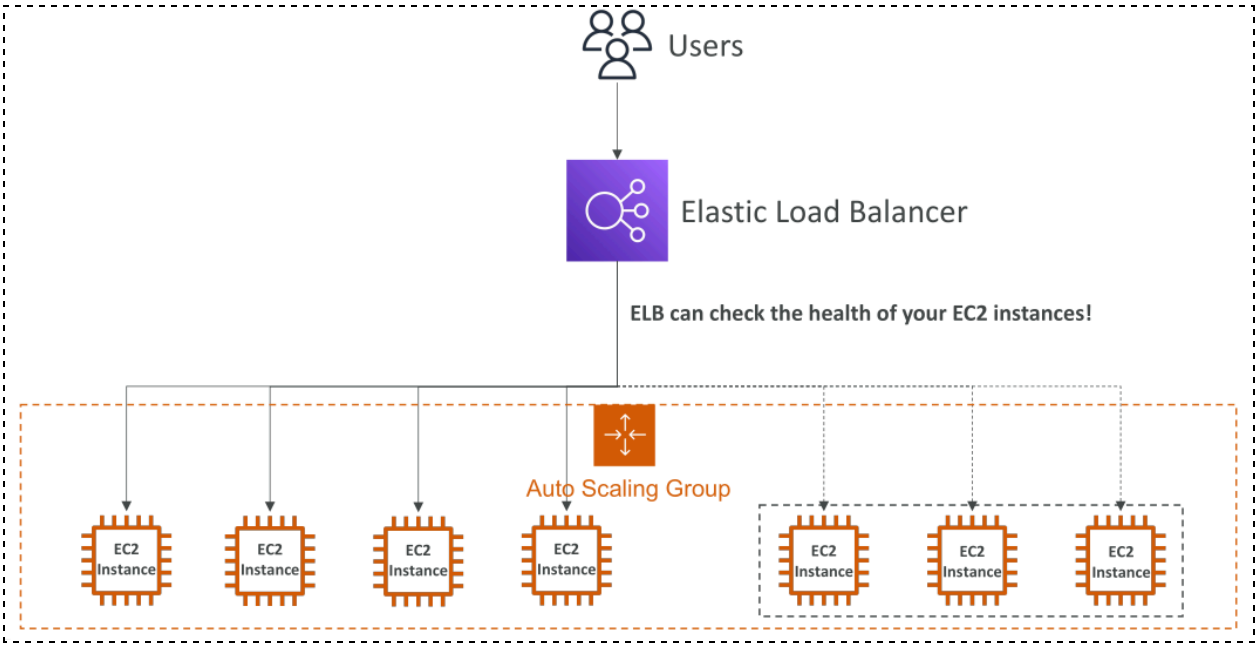
- محتاج مثلا تعمل updata لل APP بتاعك وال old instances عاوز تغيرها مثلا ف لما تغيرها ال Load Balancer هيوقف اي connections بين ال user & instances ولكن باستخدام ال Connection Draining ال connection حتي لو ال instance حصلها stop او حصل فيها اي مشكلة خلتها Unhealthy هيفضل ال connection مفتوح لمدة معينه انت بتحددها اسمها time out

- يبقي ال connection وليكن هيفضل لمدة 5 دقايق 300 ث شغال لحد ما ال instance تتبدل وال connections الجديدة تروح عليها

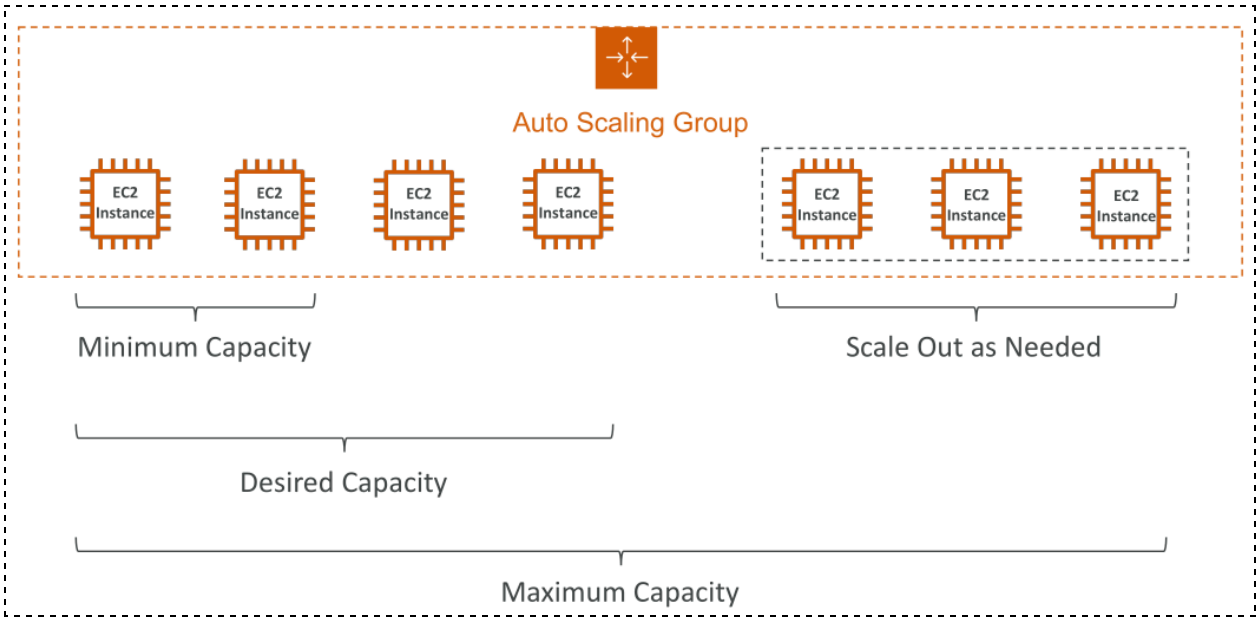
- ال Time out بتحدده فى ال ELB Elastic Load Balancer ايا كان نوعه وبيكون من 1 : 3600 ث .. علي حسب نوع ال requests بتاعتك بقا لو هيا بتاخد وقت كبير بتكبر المدة لو بتاخد وقت صغير بتصغر المدة على حسب ال app ونوعيه ال requests

➥015 Auto Scaling Groups (ASG) Overview

- **Overview:**



- عباره عن **Free** AWS feature وظيفتها انها تحافظ عالـ application بتاعك انو يكون علطول Available من خلال انها بتزود وتنقص EC2 Instances علي حسب ال Traffic اللى بيبعته ال load balancer عالجروب دة
- لو فيه traffic عالي بتـ scale out Automatic ولو فيه traffic قليل بترجع تاني تعمل scale in Automatic علشان تقلل التكلفة لان التكلفة اكيد زادت بما انك زودت instances



- **Auto Scaling Group Capacity:**

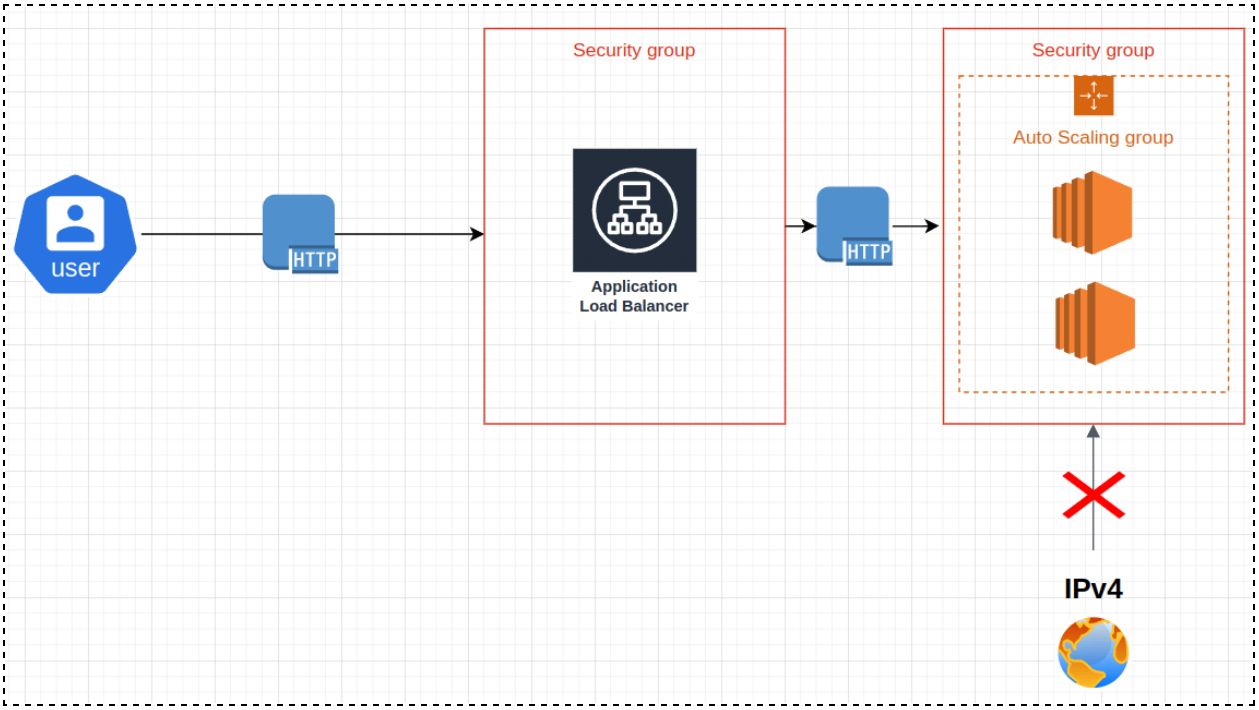
1. **Minimum Capacity** : أقل عدد Instances انا عاوزها تكون موجودة
2. **Maximum Capacity** : أكبر عدد Instances انا عاوز اني اوصله
3. **Desired Capacity** : العدد المتوسط اللى انا عاوزة

- ممكن أتحكم فى عدد ال instances ازود او انقص بناءا على ال cloud watch alarm .. مثلا جالي alarm ان فيه استهلاك عالى فال CPU ف نزود EC2 Instances وهكذا

- **Create Launch Template:**

- شوية parameters بتعملها set بحيث اي EC2 Instance هتكريتها هتقوم بال parameters دي.. يا عين لما يحتاج يـ Scale out هياخد نفس ال Parameters دي .. مثلا عملت set لل Subnet = 1A معني كدا ان اي EC2 Instance هتقوم من ال Zone 1A = وهكذا

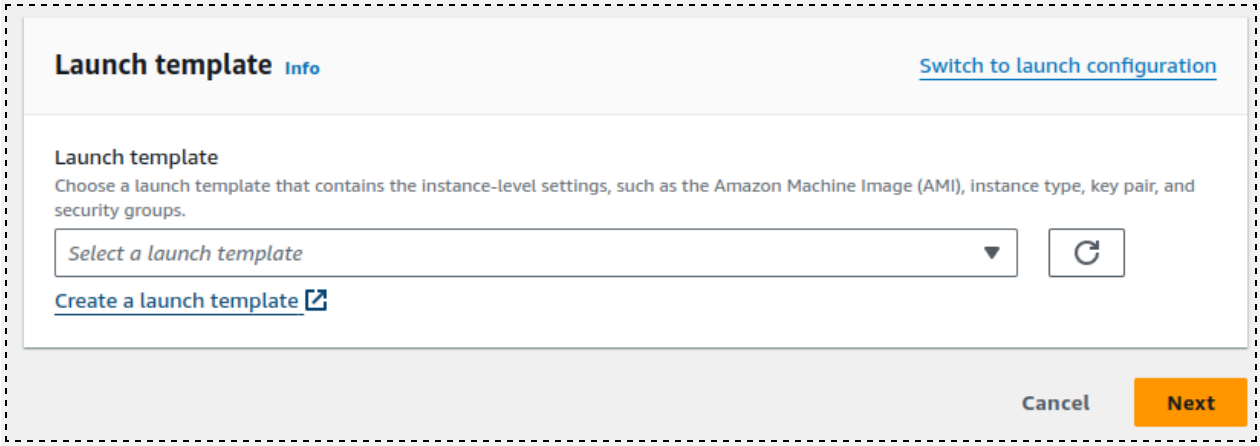
➥016 Auto Scaling Groups Hands On



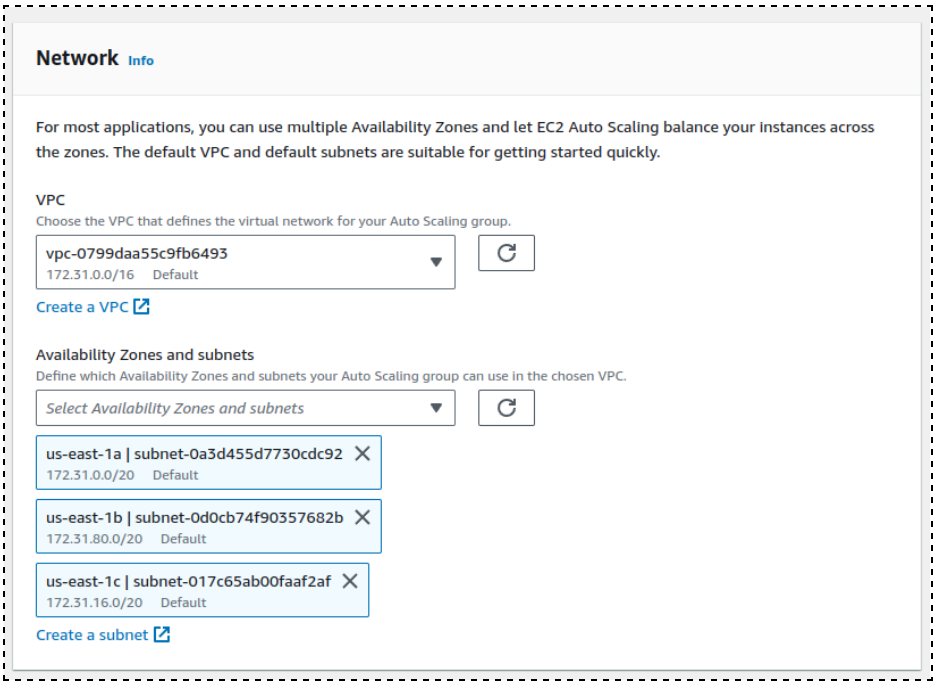
أول حاجة هتعملها ال launch template علشان اي instance تقوم تقوم بنفس ال

configurations

ASG >> Launch Template

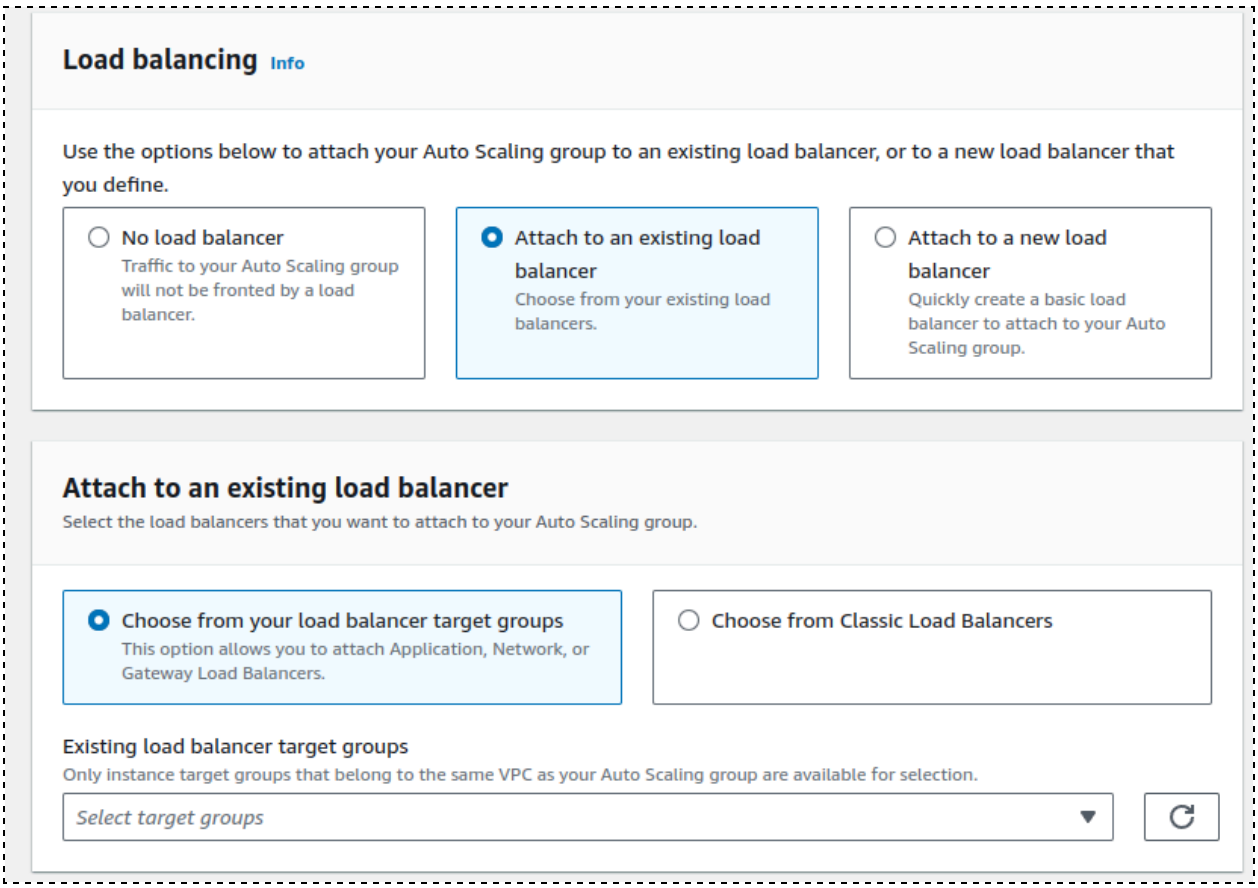


هتختار مثلا 3 AZ علشان هنقوم حوالي 3 instances



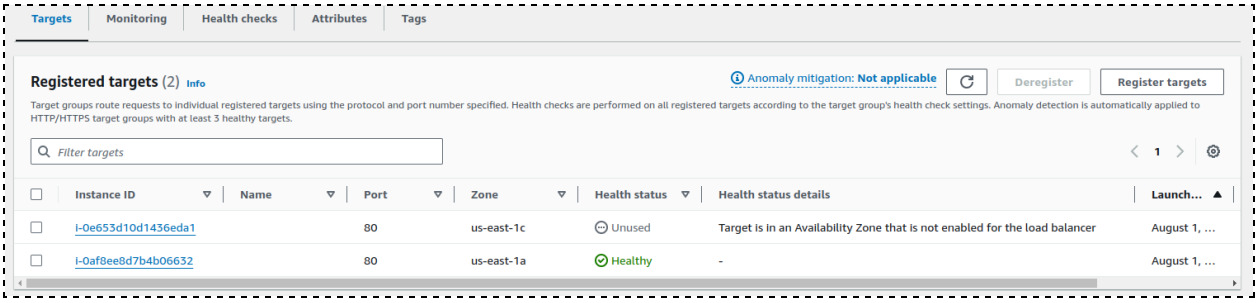
هت attach بقا ال ASG دة بال load balancer هنكريت احنا ALB زي ما عملنا قبل كدا

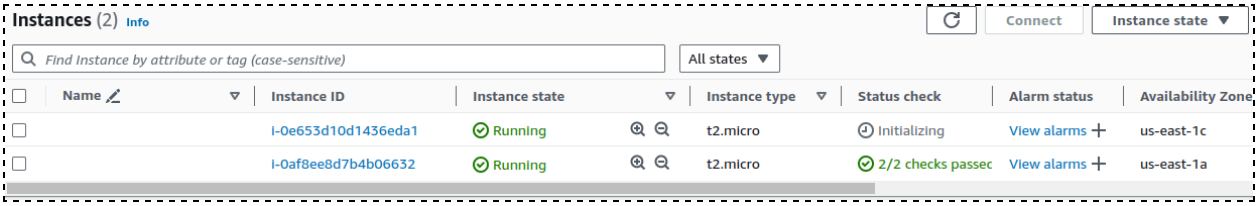
وبعدين نربطه بال ASG دة

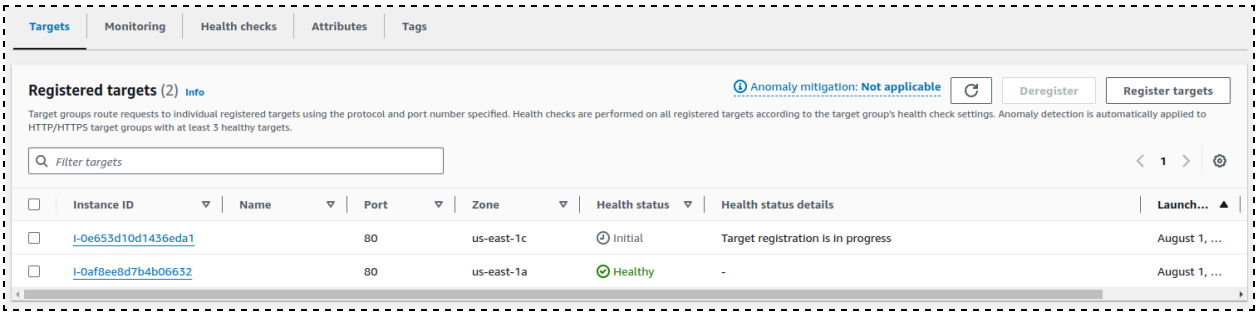


تزود ال desired وليكن ل 2 بدل 1 تلاقي فال instance واحدة بتقوم تروح عال ASG تلاقي نفس الكلام ان فيه واحدة بتقوم تروح عال target جروب بعدها تلاقيها بتوصل وبقت unhealthy علشان ال bootstrapping ثم بقت healthy تيجي تأكسس ال load balancer بدل ما كان بيوجهك على واحدة بدأ يبدل عالاتنين وهكذا

ظهرلي مشكلة وهي اني لما جيت اعمل ال load balancer حددت zone 2 بس ولكن ال instance لما جت تقوم قامت ف 1c وانا كنت محدد 1a & 1b ف روحت عال load balancer عدلتها .. ولقيتها بدل ما كانت واقفة فى ال instances بدات تقوم وفى ال target group كذلك







💡 ال ASG مربوط بال TG وال TG مربوط بال ALB

➥017 Auto Scaling Groups - Scaling Policies

- **Dynamic Scaling Policy:**
- **Target Tracking Scaling:**
  - مثلا أحددلو ان أخرك ف استهلاك الـ CPU انو يكون 60% زيادة عن كدا زودلي EC2 instances .. وطبعا هيلتزم بال maximum اللى احنا حددناه فوق

- **Simple/Step Scaling:**

- لو فيه cloud watch alarm معين نفذ كذا .. لو مثلا عندي alarm ان ال cpu usage بقي فوق ال 70% ف فى الحالة دي زودلي EC2 instances وليكن زود 2 ولو اقل من 70% قلل 1 وهكذا

- **Scheduled Scaling:**

- بنعمل دة لو فيه event ف وقت معين بيكون فيه traffic عالي .. رأس السنة مثلا عيد الأم يوم الجمعة كدا ياعني .. فالاوقات دي فية traffic اعلي من الطبيعي ف زودلي EC2 instances ولما الايام دي تنتهي ارجع للطبيعي.

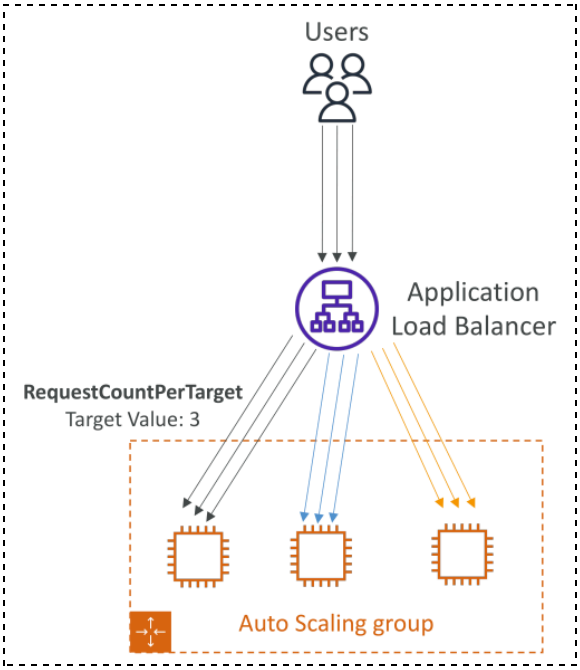
Good Metrics to Scale On in AWS:

- حاجات ممكن ت Scale in/out بناءا عليها:

- **CPU Utilization:**

- **Request Count Per Target:**

- بناءا على عدد ال requests اللى جايه عال instances بتاعتي هنا مثلا 3



- **Average Network In / Out:**

- Measures the average network traffic in and out of your instances.

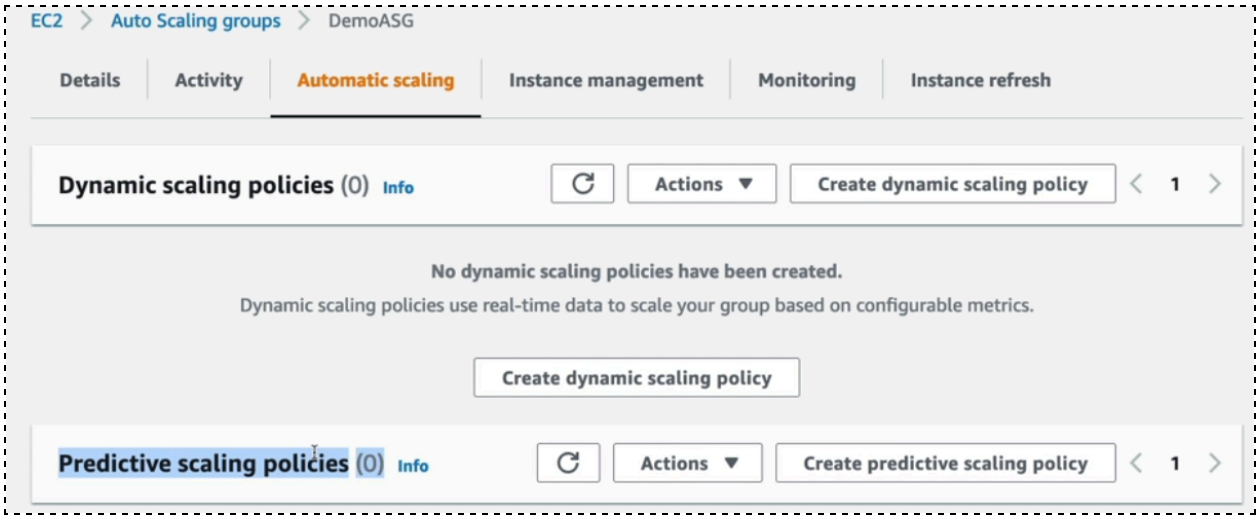
- **Custom Metrics:**

- اي specific metrics انت عاوزها وبتدخلها عال cloud watch وتستخدمها.

- **Scaling cooldown:**

- فتره فى خلالها ال ASG مبيقومش باي actions تخص ال scaling سواء in/out بحيث ي stabilize وميكونش فيه rapid scaling و by default بيكون 300s = 5 min لو معملتوش ال rapid scaling دة هيكلفك كتير excessive costs

- **Predictive Scaling:**



- عباره عن feature موجودة فى ال ASG بتحلل ال traffic data اللى حصلت قبل كدا وبناءا عليها بتعرف امتي ال traffic بيكون عالي وامتي اقل وكدا وبناءا عليه بت scale فى اي عطلة او مناسبة او كدا

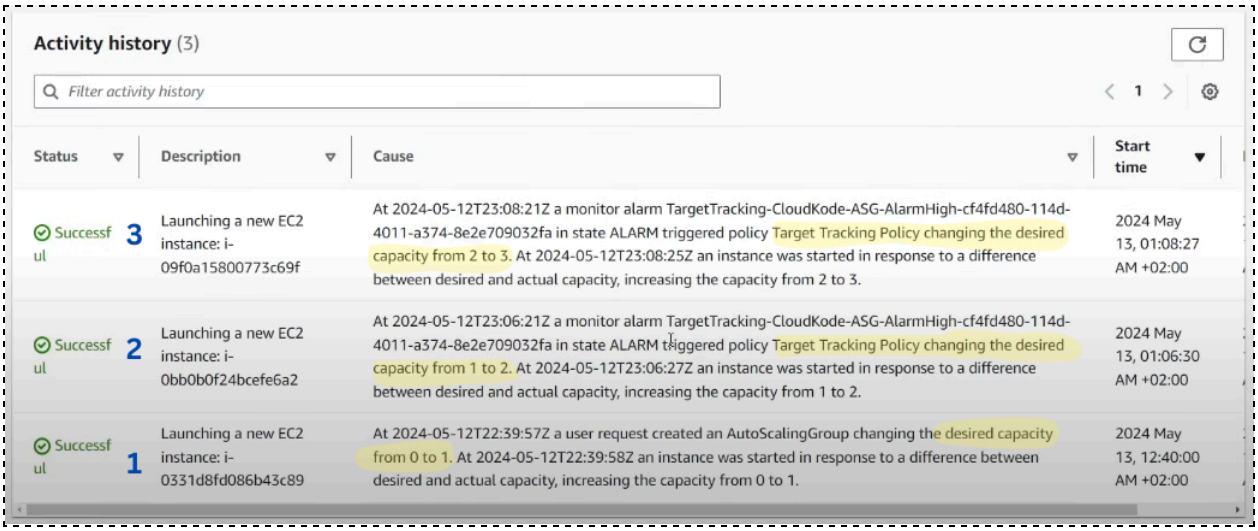
- بنستخدمها لان من غيرها ال ASG ممكن متشتغلش بكفاءة ممكن يكون فيه مثلا lag time على ما ال instances تقوم اما لما بنستخدمها بت schedules scaling actions مقدما

- **Activity History Example :**

1. كنت محدد ف ال desired capacity = 1 ف هتلاقيه زود instance من 0 إلي 1

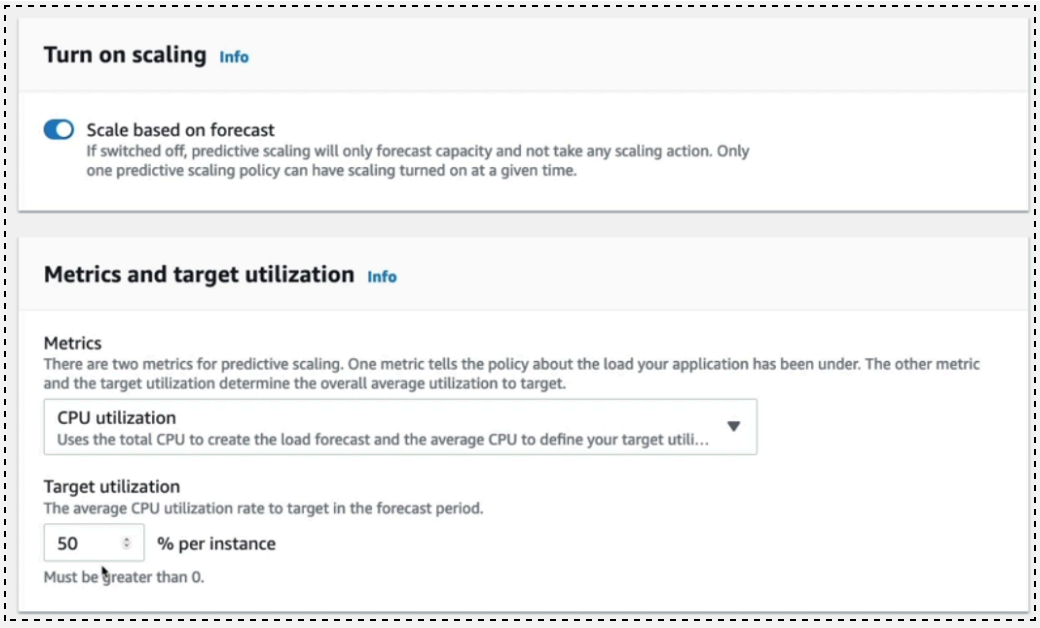
I'm Here

2. كنت محدد ف ال target tracking = 50% ف لما ال cpu زاد عن 50% زود 1 instance لان زي مقولنا معني ال target tracking (ان أخرك ف استهلاك الـ CPU انو يكون 50% زيادة عن كدا زودلي EC2 instances .. وطبعا هيلتزم بال maximum اللى احنا حددناه فوق)
3. بعد ما زود instance وبقي عندنا 2 لقي ان فيه ضعط جاي تاني عال CPU وبيزيد عن 50% تاني فبالتالي زود واحدة كمان ووصلنا ل3 اللى هو ال maximum



➥018 Auto Scaling Groups - Scaling Policies Hands On

ال setup بتاع ال predictive scaling بسيط بتحدد بس ال metrics وخلاص



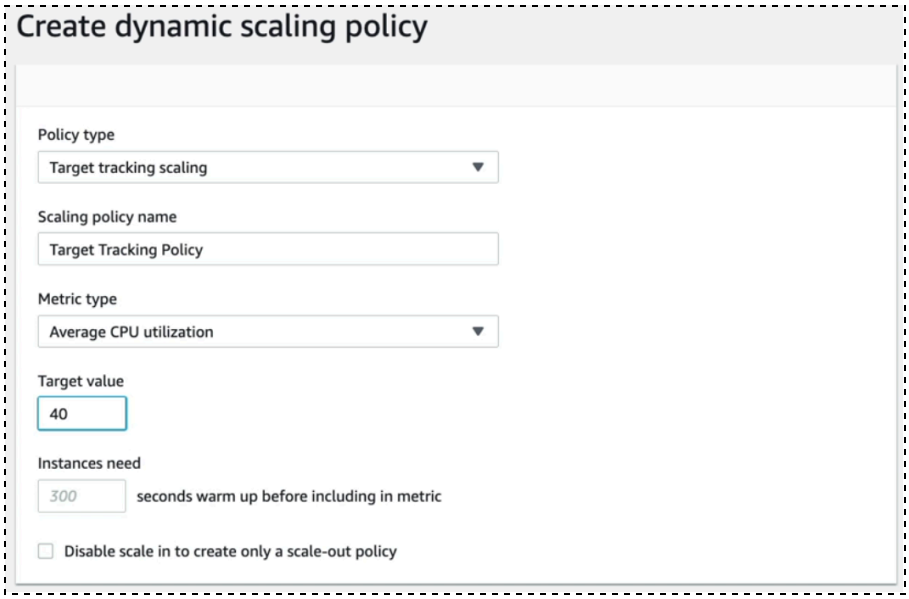
بالنسبة لل Dynamic scaling policy بتحدد بقا نوعها ولو فيه cloud watch alarm

وهكذا بردو setup بسيط

هنا هنكريت target tracking scaling بناءا عال cpu 40%

- خلي بالك بقا انك تغير ال maximum capacity , desired وكدا لان دلوقتي لما تشتغل وال cpu يزيد عن 40% ف هي scale out ف لازم تكون سامح ب دة قبل كدا

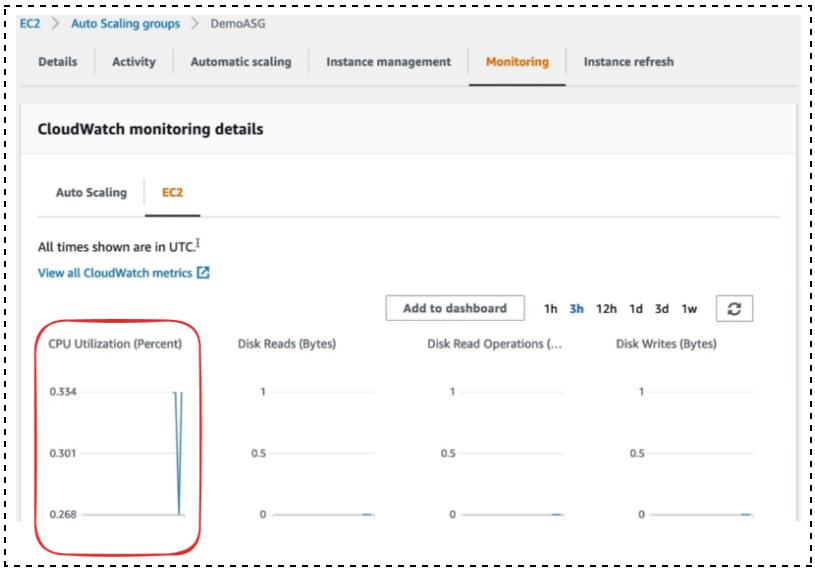
فى ال ASG



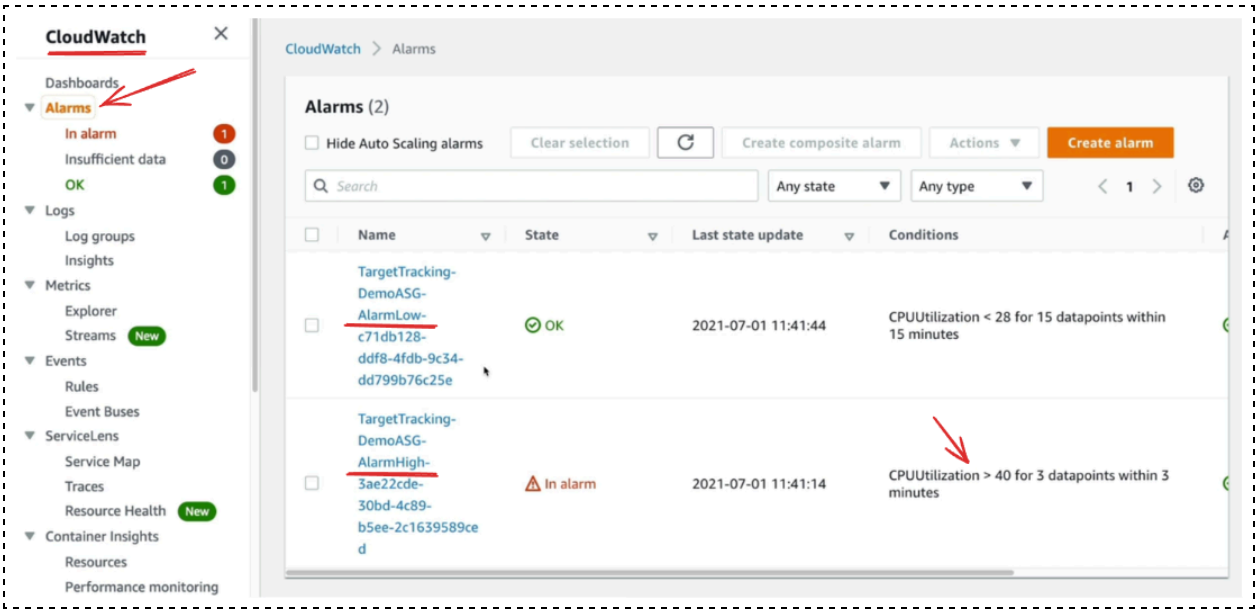
تقدر ت connect عال instance من خلال AWS عادي وبدها command ب دول بحيث ت stress ال cpu انو يرفع عن 40%

```
sudo amazon-linux-extras install epel -y
sudo yum install stress -y
Stress -c 4
```

زي مانت شايف فى ال monitoring ال cpu utilization رفع .. وبالتالي هي scale out ب 1 او 2 instances على حسب الضغط وعلي حسب ال maximum capacity

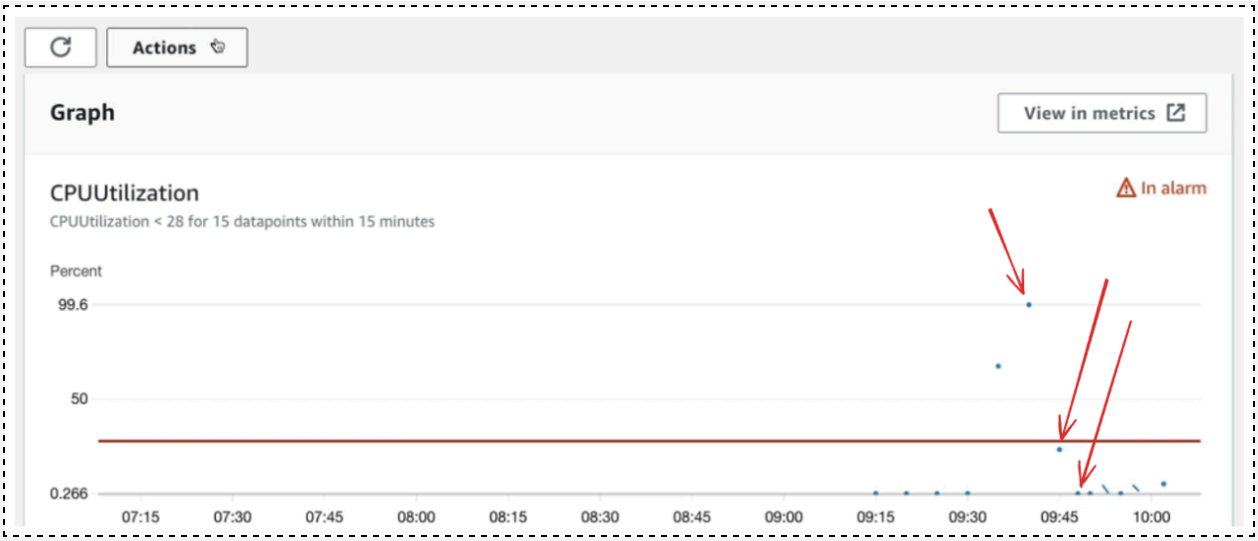


ولو روحت عال cloud watch alarm هتلاقي فعلا ان فيه alarm جايلك من target policy اللى احنا كريتناها **الأول** لما بدانا خالص ان ال cpu اقل من 28 ف تمام هنكريت instance واحدة ودي البداية **والتاني** alarm high بيوضحلك ان ال cpu زاد عن 40% ف كدا هنخش ف instances اكتر



لما تيجي ترجع لل connection بينك وبين ال instance هتلاقيها معلقه علشان استهلاك ال cpu العالي ف روح عال instances dashboard واعملهم reboot يبقي كدا ال cpu هيرجع لل0

هتلاقي بقا graph للى حصل فى ال cloud watch .. طلع ثم نزل تاني لما عملنا reboot



الip بيتغير سواء v4 or v6 اما ال dns ثابت بتخش من خلاله عال instances

Question 3:

Elastic Load Balancers provide a .....

- ○ static IPv4 we can use in our application
- ⊙ static DNS name we can use in our application
- ○ static IPv6 we can use in our application

Submit

Question 11:

For compliance purposes, you would like to expose a fixed static IP address to your end-users so that they can write firewall rules that will be stable and approved by regulators. What type of Elastic Load Balancer would you choose?

- ○ Application Load Balancer with an Elastic IP attached to it
- ⊙ Network Load Balancer
- ○ Classic Load Balancer

Submit

مش هيحصل حاجة لانو على desired = 3

Question 16:

You have an application hosted on a set of EC2 instances managed by an Auto Scaling Group that you configured both desired and maximum capacity to 3. Also, you have created a CloudWatch Alarm that is configured to scale out your ASG when CPU Utilization reaches 60%. Your application suddenly received huge traffic and is now running at 80% CPU Utilization. What will happen?

- ○ Nothing
- ○ The desired capacity will go up to 4 and the maximum capacity will stay at 3
- ⊙ The desired capacity will go up to 4 and the maximum capacity will stay at 4

Submit

Question 20:

You have an ASG and a Network Load Balancer. The application on your ASG supports the HTTP protocol and is integrated with the Load Balancer health checks. You are currently using the TCP health checks. You would like to migrate to using HTTP health checks, what do you do?

- ○ Migrate to an Application Load Balancer
- ⊙ Migrate the health check to HTTP

Submit

Question 21:

You have a website hosted in EC2 instances in an Auto Scaling Group fronted by an Application Load Balancer. Currently, the website is served over HTTP, and you have been tasked to configure it to use HTTPS. You have created a certificate in ACM and attached it to the Application Load Balancer. What you can do to force users to access the website using HTTPS instead of HTTP?

- ○ Send an email to all customers to use HTTPS instead of HTTP
- ⊙ Configure the Application Load Balancer to redirect HTTP to HTTPS
- ○ Configure the DNS record to redirect HTTP to HTTPS

CH-09 - AWS Fundamentals

RDS + Aurora + ElastiCache

➥001 Amazon RDS Overview (Relational Database Service)

- عباره عن service بتقدمها AWS بتسهلك ال setup, operation, and scaling of relational databases in the cloud

- **Key Features**

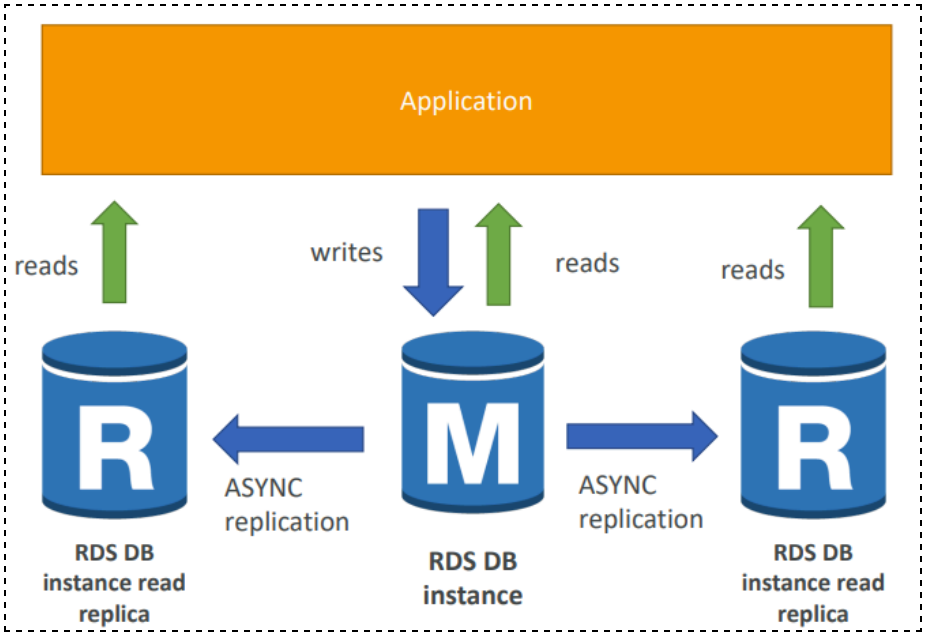
- **Supports multiple database engines**, including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle, and Microsoft SQL Server.
- **Automates tasks** such as hardware provisioning, database setup, patching, and backups.
- Provides **easy scaling** of database instance types and storage, Supports read replicas for read-heavy applications.
- **High Availability and Durability**: Multi-AZ (Availability Zone) deployments for automated failover and redundancy.
- **Automated backups** and **point-in-time recovery** options.

- Amazon **CloudWatch integration for monitoring** database performance and setting alarms.
- **Pay-as-you-go** pricing model.

BUT you can't SSH into your instances 💡

➥002 RDS Read Replicas vs Multi AZ

- **Overview**



- Replicas : نسخ متماثلة
- ASYNC = asynchronous غير متزامن
- SYNC = Synchronization متزامن

- Read Replicas : عباره عن بديل من ال database الاساسية بيحصل عليها read فقط مش read & write زي الاساسية بنستخدمها بحيث نوزع عليها حمل ال read بحيث نخف عن الاساسية.
- لو عندك web app عنده primary database بتهندل ال read , write operations وال app بي grow وبقي بيجيلو heavy traffic ودة بيسبب performance issues ف علشان نحل المشكلة دي بنكريت Read Replicas من ال primary database بحيث انها تاخد ال read traffic وتخفف الضغط عال primary database instance
- ال data عال read replicas بتكون **eventually consistent** "متسقة فى النهاية" معناها ان بيكون فيه delay ف ان ال replicas تاخد نفس ال recent data changes عال primary database
- ليك لحد **replicas 15** فى ال AZ or Cross AZ or Cross Region

• Connection string

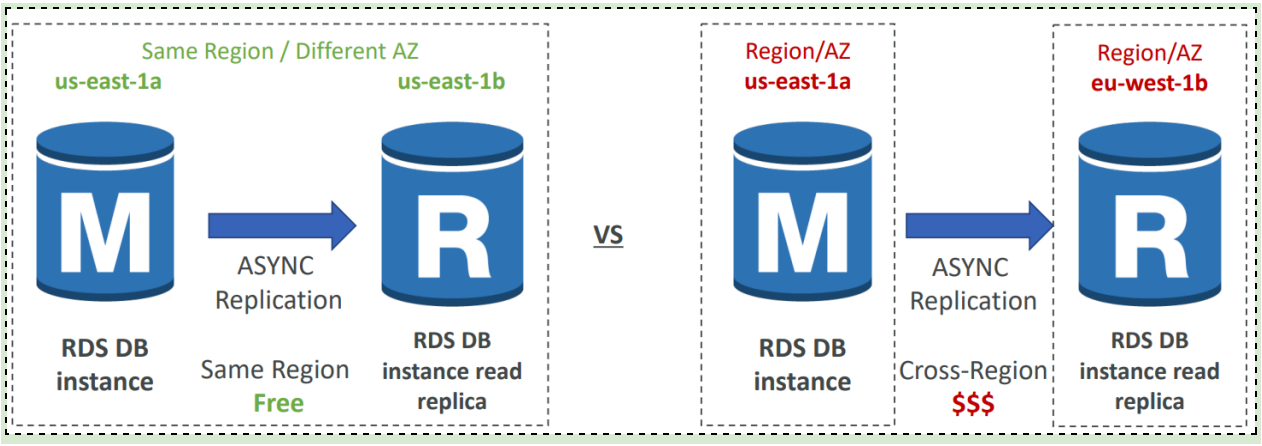
- عندك parameter اسمه **connection string** لازم ت configure it بحيث ان ال read traffic اللى جاي ميروحش عال primary database ولكن يروح عال replicas end point

• Replicas End point

- ال **end point** معناها .. انت لما بتعمل replicas بيكون ل كل واحدة منها hostname & port دول ال end point
- تقدر تحول ت promote ال replicas دي ل standalone database انها تكون primary ياعني يحصل عليها read & write

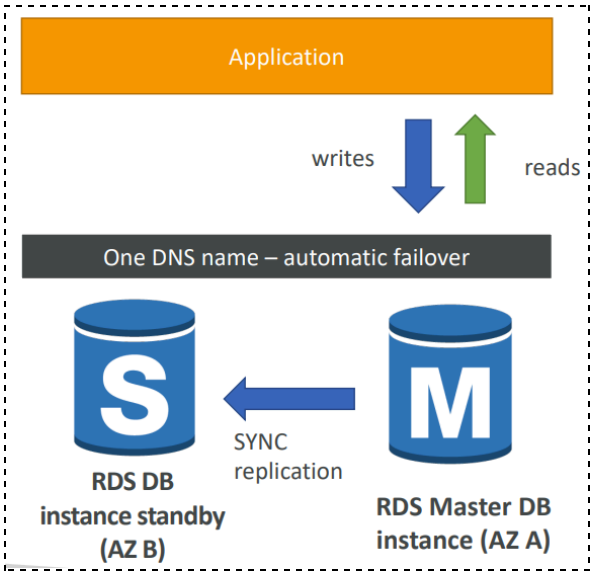
• Cost

💡نقل ال data فى الطبيعي من AZ الى AZ تانية بيندفع عليه cost أما فى حالة ال replicas عندك حالتين



- لو هتكريت replicas فى **نفس** ال Region **مش هتدفع**
- لو هتكريت replicas فى Regions **مختلفة هتدفع**

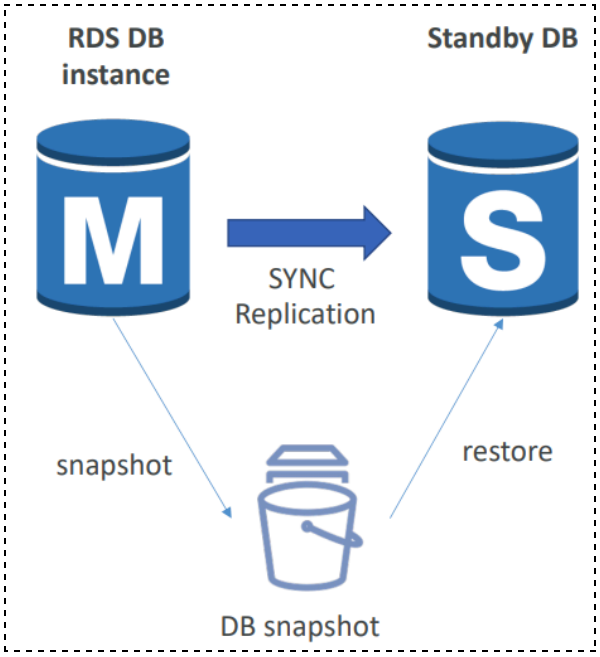
• RDS Multi AZ (Disaster Recovery)



- لو عندك e-commerce application with critical data base علشان تضمن نفسك فى حالة لو ال database دي حصلها مشكلة بت deploy ال data base دي فى multi AZ بحيث لو واحدة وقعت الثانية تستلم مكانها.. بتسمي DB instance Standby
- الاتنين بيكونو مرتبطين بنفس ال DNS Name ف دة شيئ كويس وبدون اي تدخل manual منك الموضوع هيتم من غير ماتحس.
- ال read replicas هيا كمان بيتعملها setup as multi az.

- **علشان تخلي ال RDS multi AZ :**

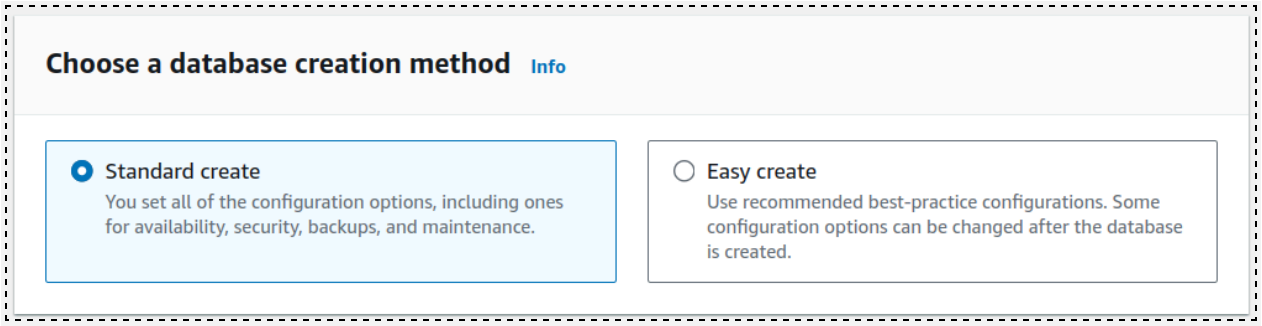
- فقط بت modify ال database وت enable ال multi AZ ولكن اللى بيحصل فى الخلفيه بيكون بالشكل دة



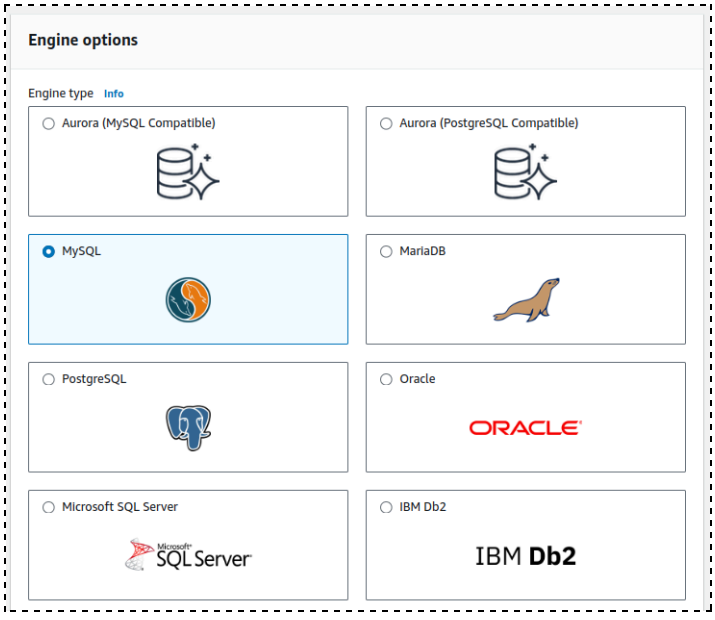
- بيتاخد snapshot من ال primary database وبعدين بيحصل restore ليها ف AZ مختلفة وبيحصل بعدها Synchronization بينهم.

➥003 Amazon RDS Hands On

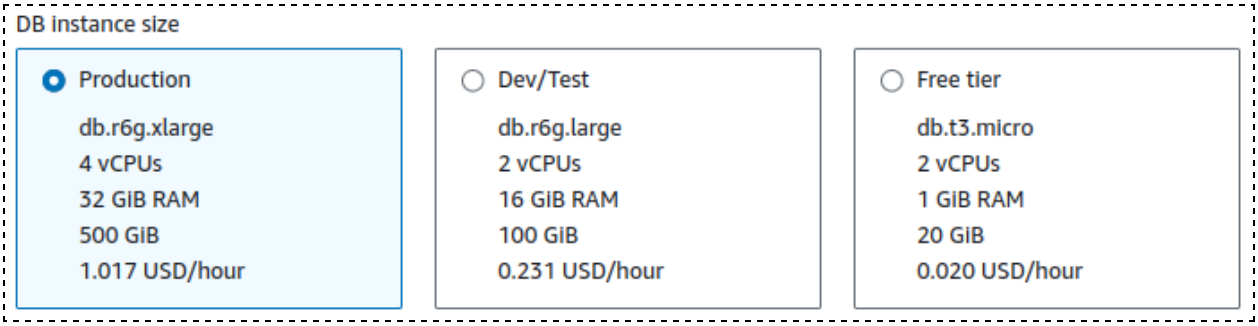
هل محتاج تحط كل ال configuration ولا محتاج حاجة فى السريع



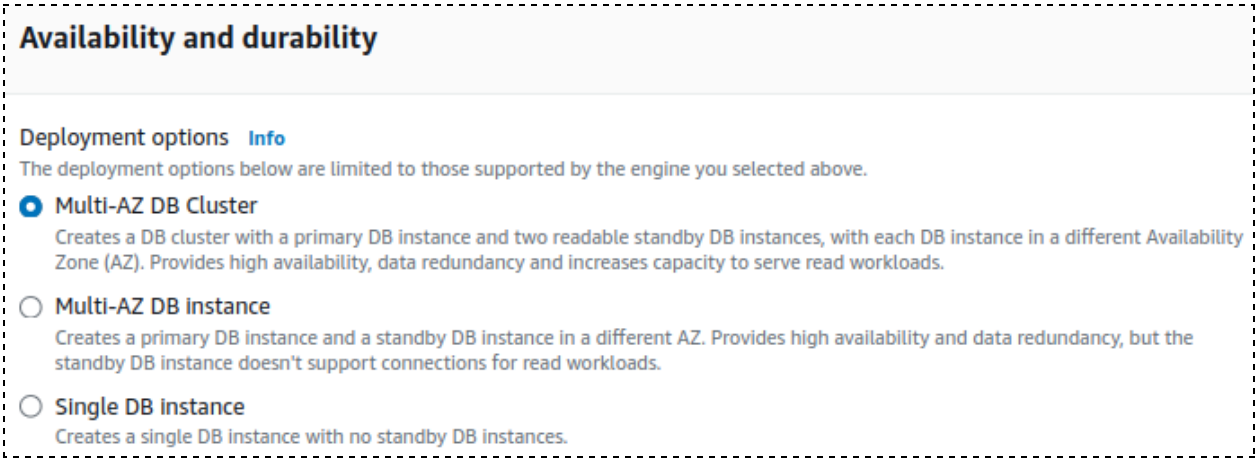
ال engines المتاحة



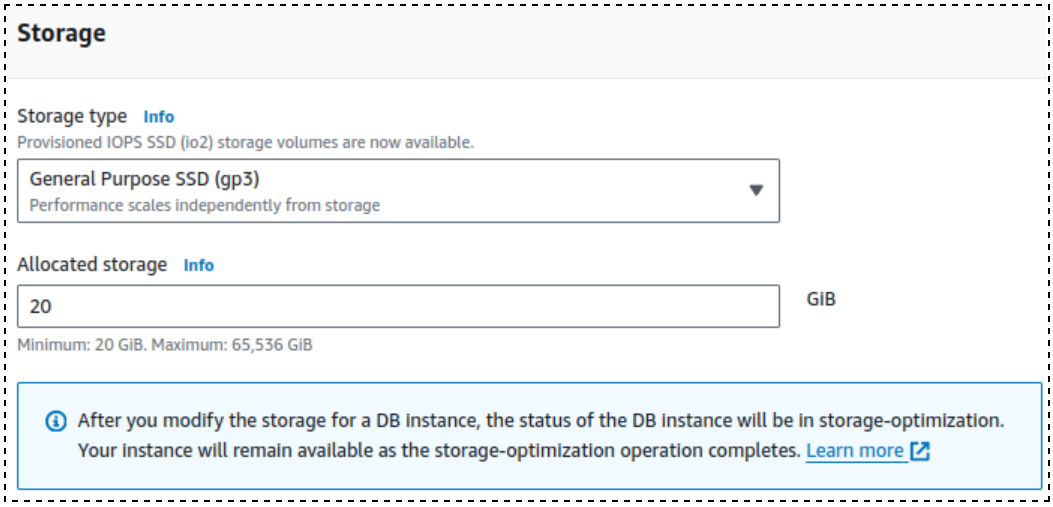
امكانيات ال instance اللى هتقوم عليها ال data base



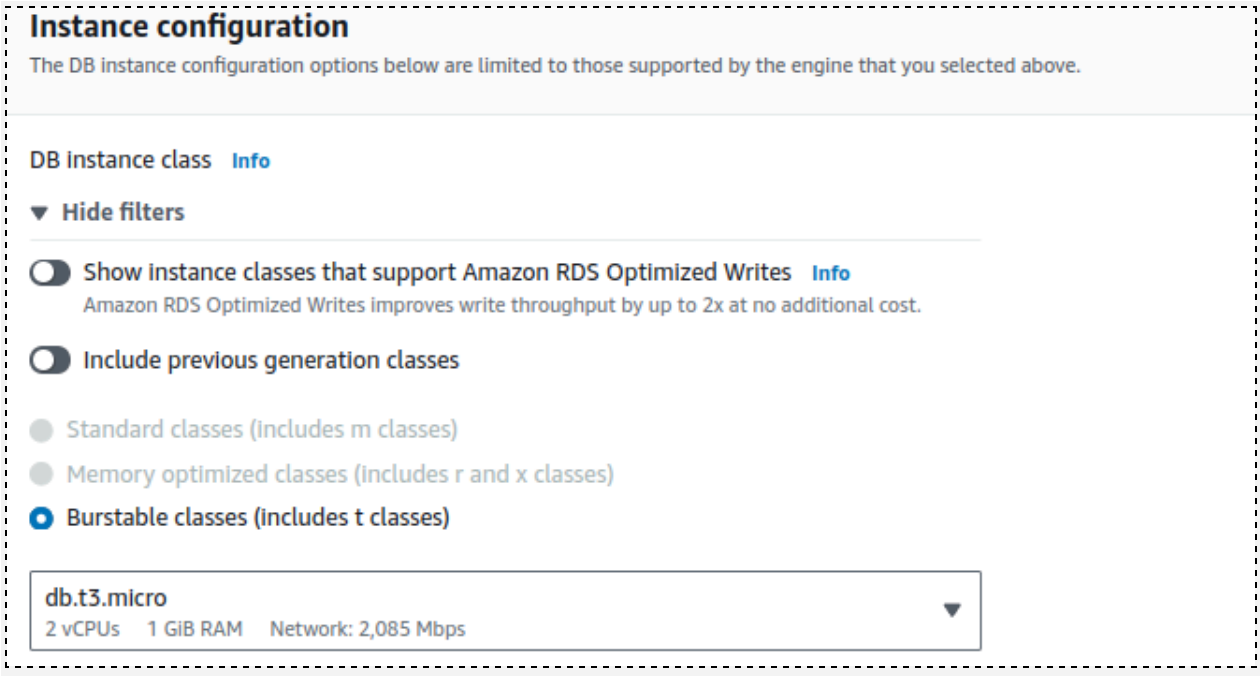
1 primary & 2 standby **or** 1 primary & 1 standby **or** 1 primary only

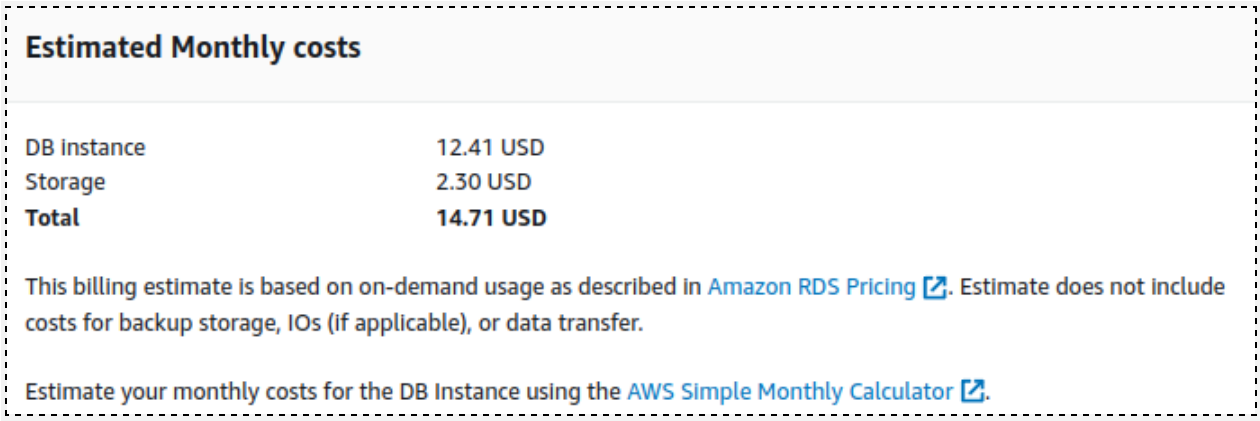


ال storage اللى هتحتاجها نوعها اية وحجمها .. gp3 suitable for free tier

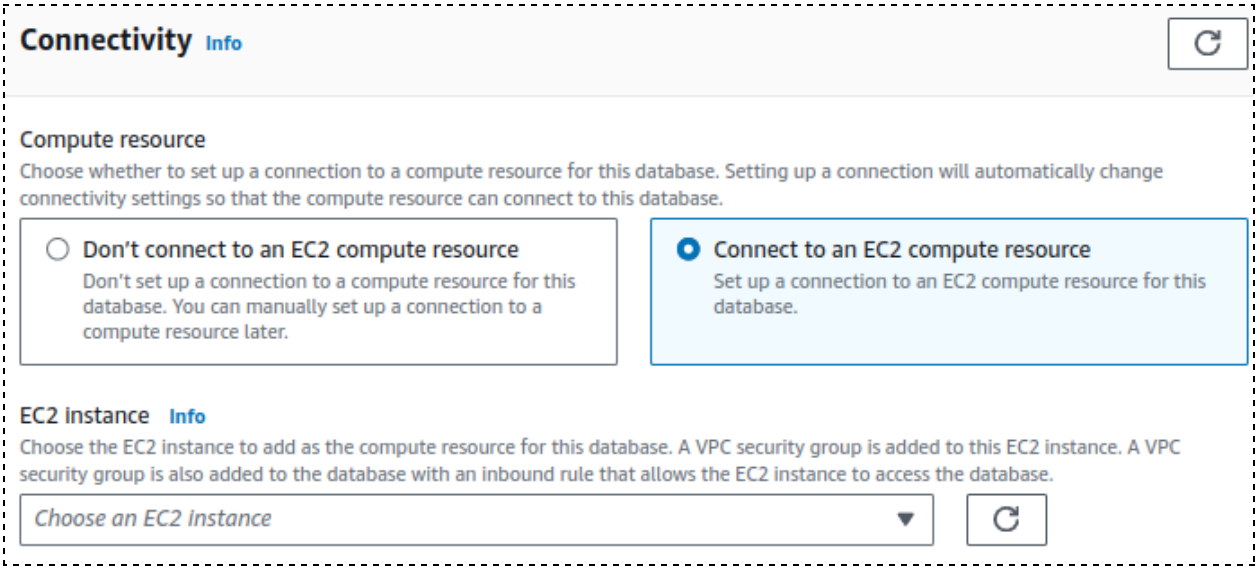


علشان ميكونش فيه تكلفه اختار النوع دة لازم db.t2.micro حتي لو ظهرلك تكلفه تحت ف عادي لازم بس تختار النوع دة

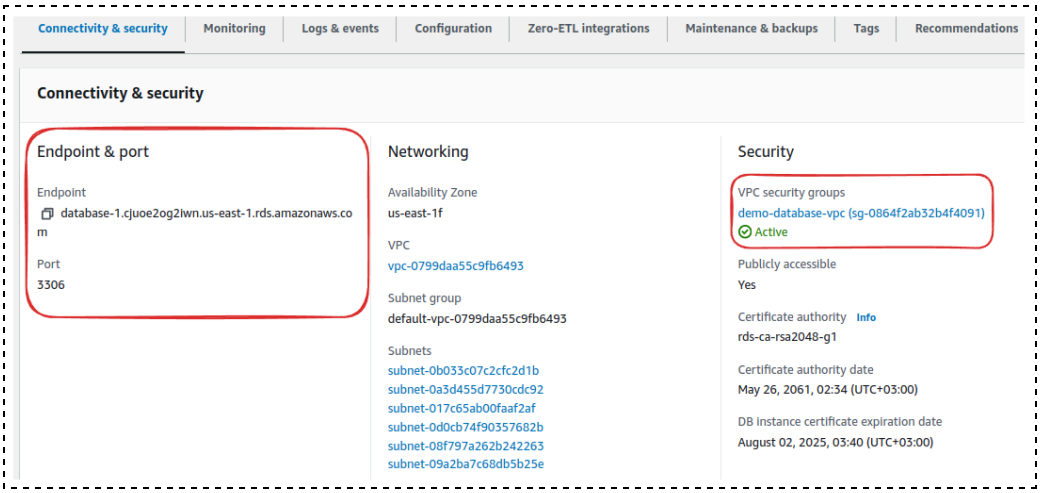


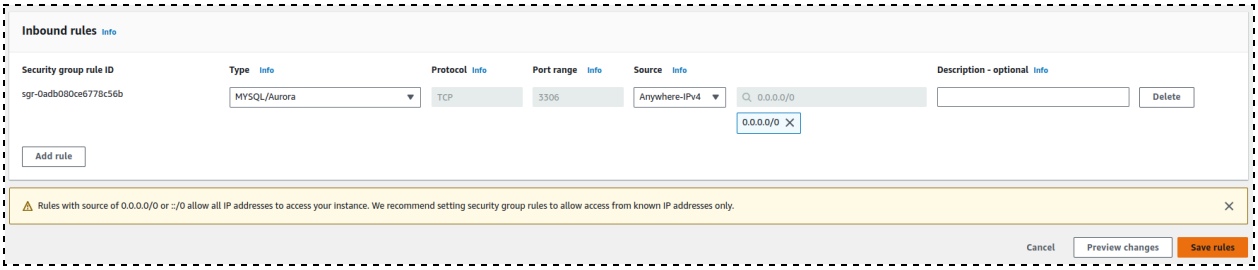


ان ربطتها ب instance موجودة مش محتاج تعمل network configuration هتلاقي ال security groups linked والامور تمام

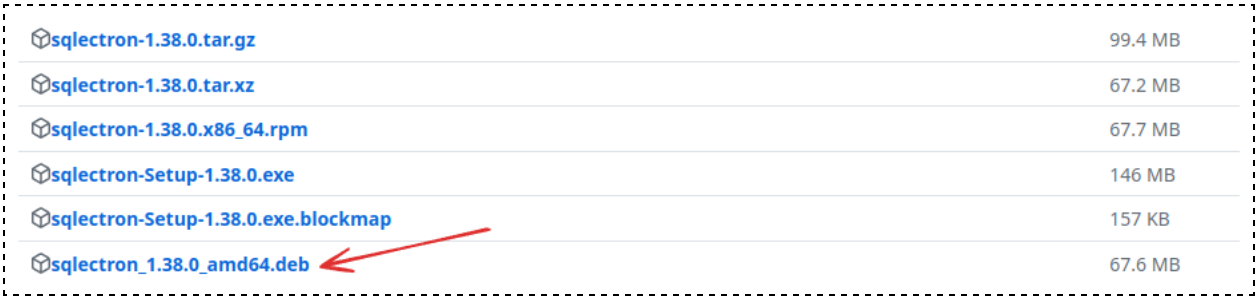


لو دخلت عال security جروب دة هتلاقي فى الinbound rules بتاعته ال income مش من anywhere ف هنعدل دة





بعد ما بتكريت ال database بيكون ليها ال endpoint اللى اتكلمنا عنها من خلالها هتتصل عليها ودة هيكون باستخدام sqlectron عباره عن GUI سهله ف جوجل هتلاقيه

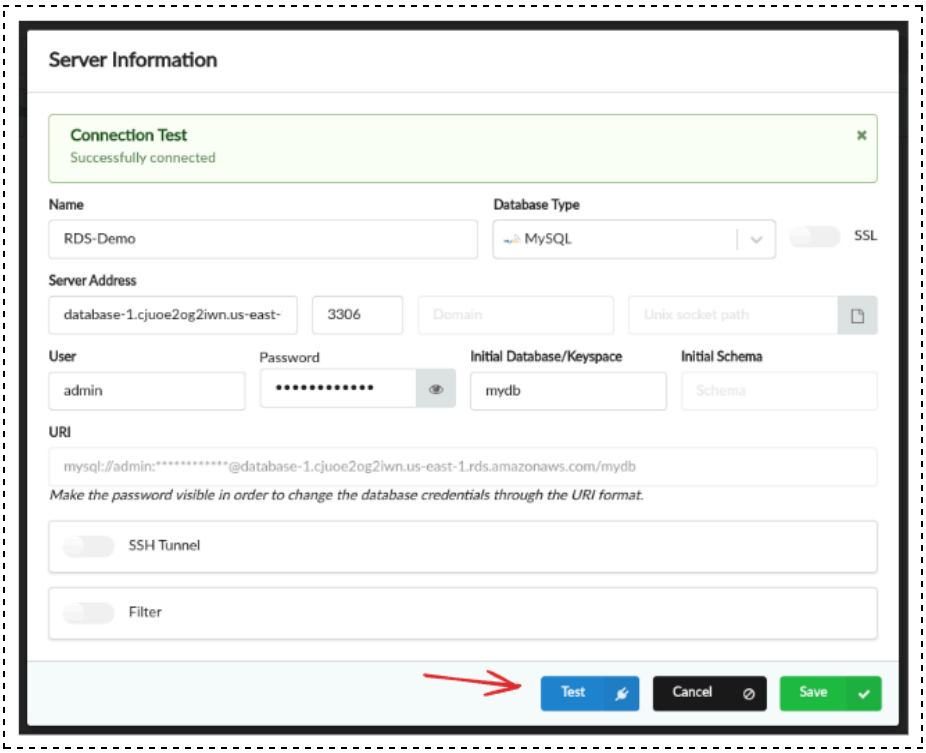


```
Sudo -i dpkg sqlectron_1.38.0_amd64.deb
sudo apt remove sqlectron
```

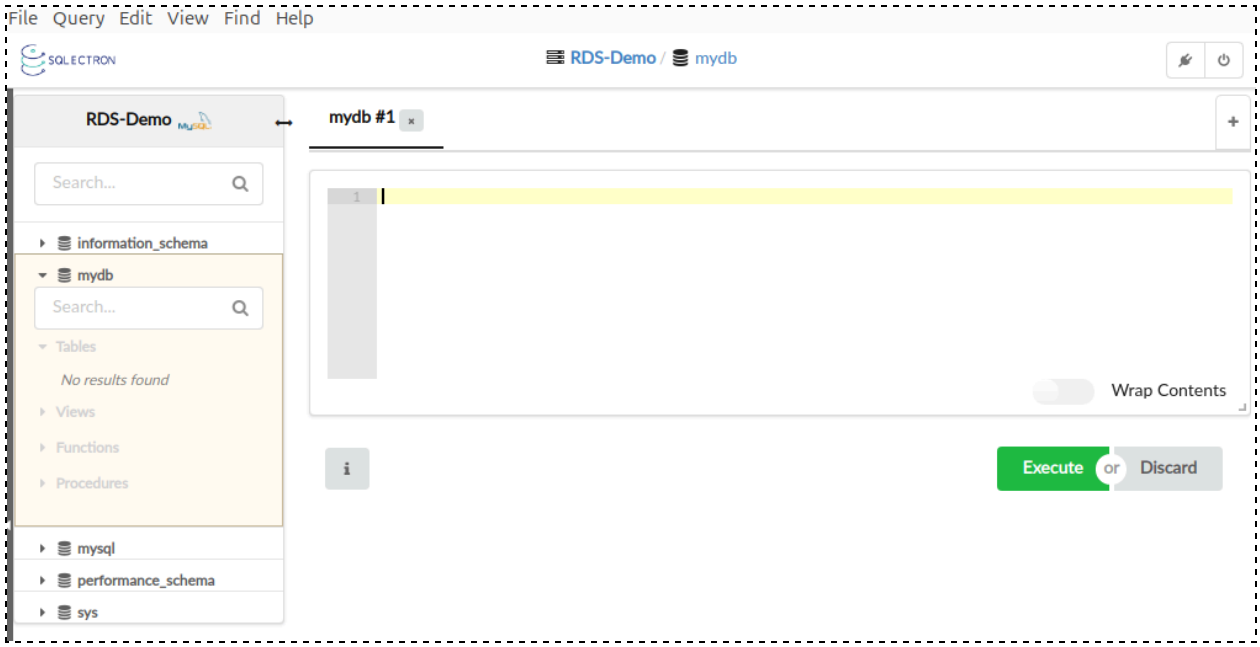
بعد ما تفتحه هت test وبس

بالنسبة لل initial database/keyspace هتلاقيه فى ال configuration وانت بتعمل

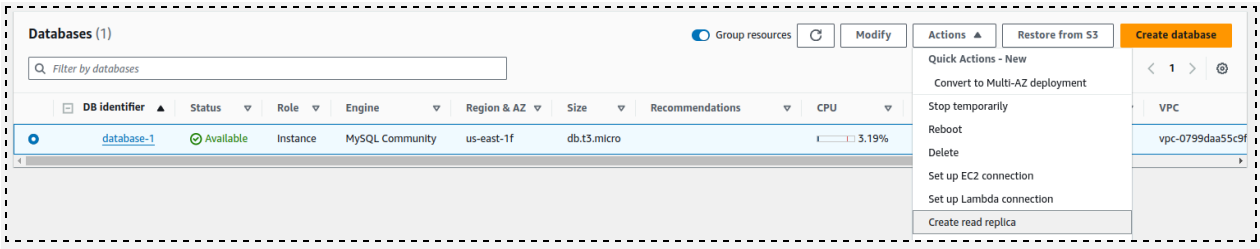
الdatabase او تقدر بعد ما عملتها ترجع تشوفه من الconfiguration tab



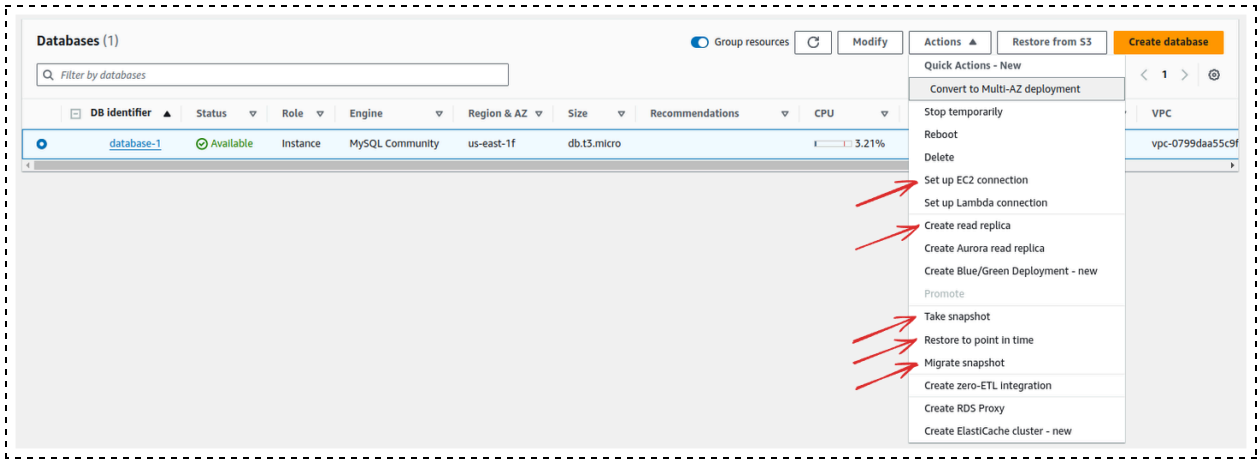
بعد ما ت save خش ب connect وتقدر بقا تعدل عليها ب SQL عادي.



من action تقدر ت create replica وال setting بسيطة.



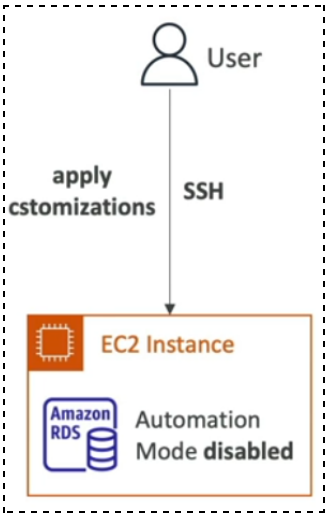
من action هتلاقي كل ال options اللى ممكن تحتاجها



بعد ما تخلص امسحها ولو كنت مفعل ال **Deletion protection** ارجع الغيها.

➥004 RDS Custom for Oracle and Microsoft SQL Server

- **RDS vs. RDS Custom**
- **RDS:** entire database and the OS to be managed by AWS
  - managed By AWS كل حاجة entire database & OS ال
- **RDS Custom:** full admin access to the underlying OS and the database



  - من خلال RDS Custom تقدر يكون ليك full access عال database & OS وال instance ومتنساش تقفل ال Automation mode

➥005 Amazon Aurora

- **Relational Databases:**
  - من اسمها فيه علاقه بين ال tables جوا نفس ال database وال schema fixed
  - اللغة اللى ب query بسال بيها عن ال data بتاعتي اسمها SQL Structured Query Language
  - عكسها ال non-relational database
- **Overview:**
  - عباره عن relational database service و managed by AWS ومصممه علشان توفرلك high performance, high availability, and scalability و compatible مع MySQL & PostgreSQL
- **Key Features**

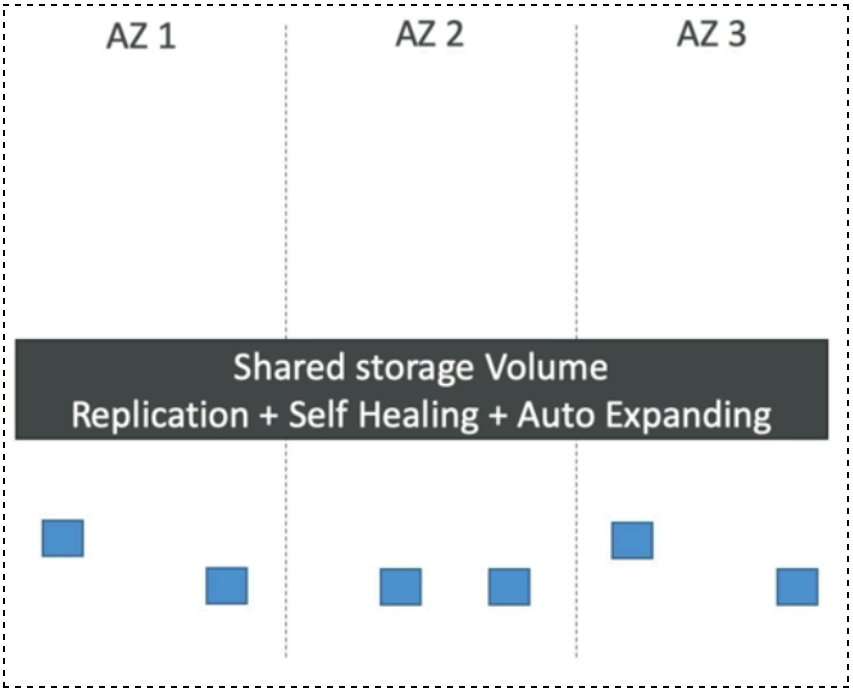
1. High Performance:

- **Optimized Storage:** Aurora uses a distributed, fault-tolerant, self-healing storage system that automatically grows in increments of 10 GB, up to 128 TB per database instance.

- ال Self healing ياعني لو فيه مشكلة حصلت فى الداتا corrupted تلفت هيا هتصلحها من نفسها .. عندك نسخ منها فى اماكن تانيه ف بتاخد منهم تحط مكان اللى تلف مش بتخترع جديد.
- **Low Latency:** It provides low-latency reads and writes, with the ability to deliver up to five times the throughput of standard MySQL and three times the throughput of standard PostgreSQL databases.

2. High Availability and Durability:

- **Fault-Tolerant Storage:** Data is replicated across six copies across three Availability Zones (AZs), ensuring data durability and high availability.



- لو عملت write ل some data هتلاقي عندك 6 نسخ من ال data على AZ 3..يبقي ال aurora شبه ال RDS multi AZ

- **Automatic Failover**: Aurora automatically detects and handles instance failures, with failover typically completing within 30 seconds.
- **Multi-AZ Deployment**: Supports Multi-AZ deployments for automatic failover and increased availability.

3. Scalability:

- **Aurora Replicas**: Supports up to 15 low-latency read replicas, allowing you to scale read operations.
- متاح ليك 15 replicas خلال ال AZs بجانب ال Main database
- **Aurora Serverless**: Automatically scales the database capacity up or down based on application needs, allowing you to pay only for the resources you use.

4. Scalability:

- **Encryption**: Data is encrypted at rest using AWS Key Management Service (KMS) and in transit using SSL.
- **Network Isolation**: Integration with Amazon VPC allows you to isolate your database within a virtual network.
- **IAM Integration**: Allows fine-grained access control and authentication using AWS Identity and Access Management (IAM).

5. Backup and Recovery:

- **Automated Backups**: Continuous backups to Amazon S3 with point-in-time recovery.
- **Snapshot Backups**: Manual snapshots can be taken and restored as needed.

6. Compatibility:

- **MySQL-Compatible:** Aurora MySQL is compatible with MySQL 5.6, 5.7, and 8.0, allowing you to use existing MySQL tools and applications.
- **PostgreSQL-Compatible:** Aurora PostgreSQL is compatible with PostgreSQL 9.6, 10.x, 11.x, and 12.x, providing similar compatibility benefits.

- أغلي بحوالي 20% من ال RDS الباقيين.. ولكن هي more efficient

What if the master database failed for RDS and Aurora:

Aurora

- كدا كدا عندك 15 replicas متاحين اي واحدة منهم تقدر ت promote to be the master ودة بيحصل automatic فى خلال 30ث

Amazon RDS

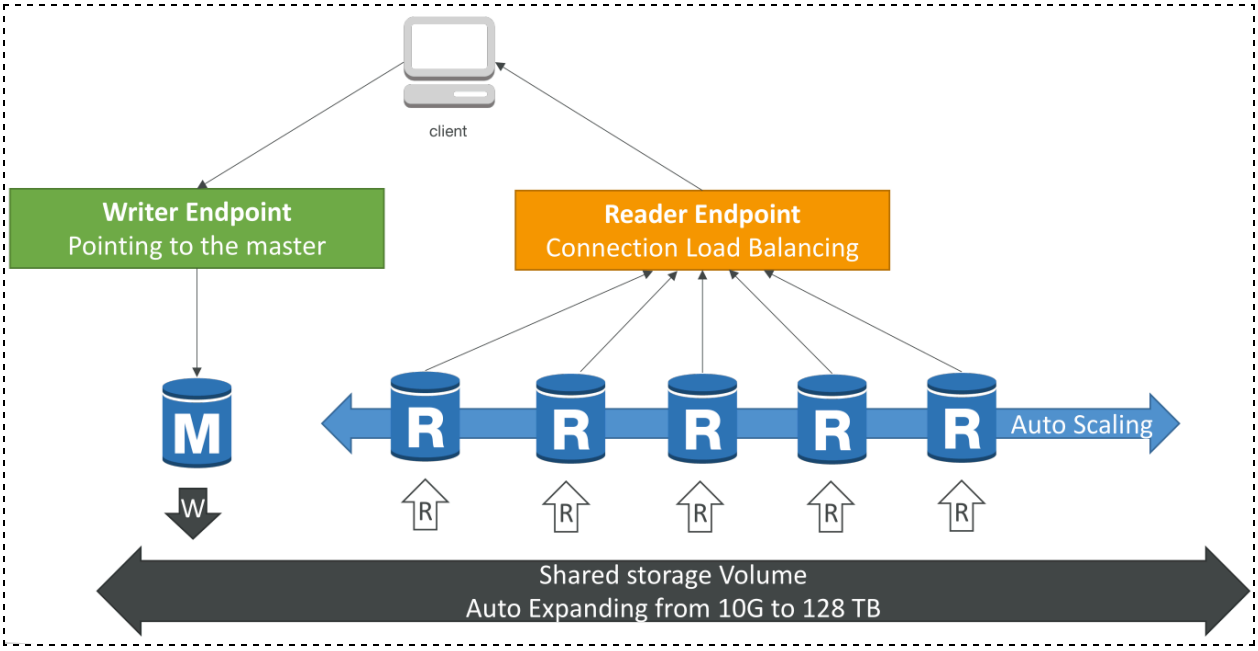
- Have RDS Multi AZ

- لو عندك زي ما شرحنا هيكون عندك بجانب ال master هتلاقي standby instance هتاخد الدور لو ال master failed

- Don’t Have RDS Multi AZ

- لو معندكش بقا ال Down time هيطول على ما ت restore بقا من خلال backup ولا تشوف المشكلة فين وتصلحها

Aurora DB Cluster



- ال master DB بتبقي مرتبطة ب writer endpoint عباره عن DNS name وحتي لو ال master حصلها مشكله ال client بيتوجه عال writer endpoint دي وبتوجهه عال instance اللى حصلها promote انها تبقي ال master

- قولنا ان بيكون عندنا 15 replicas ف صعب عليك ت track ال 15 replicas دول خصوصها انهم بيعملو auto scaling

- **The diff between Aurora Server less and normal one**
- **Server less**
  - مناسبة اكتر لل unpredictable workloads ونظام الدفع pay as you go

---

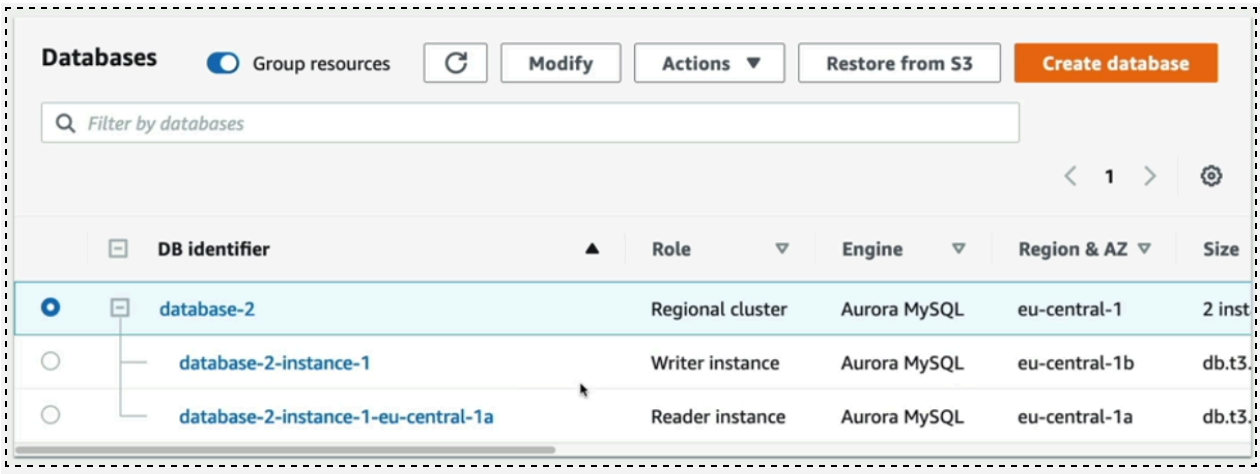
➥006 Amazon Aurora - Hands On

هيكون مكلف ال hands on دة.

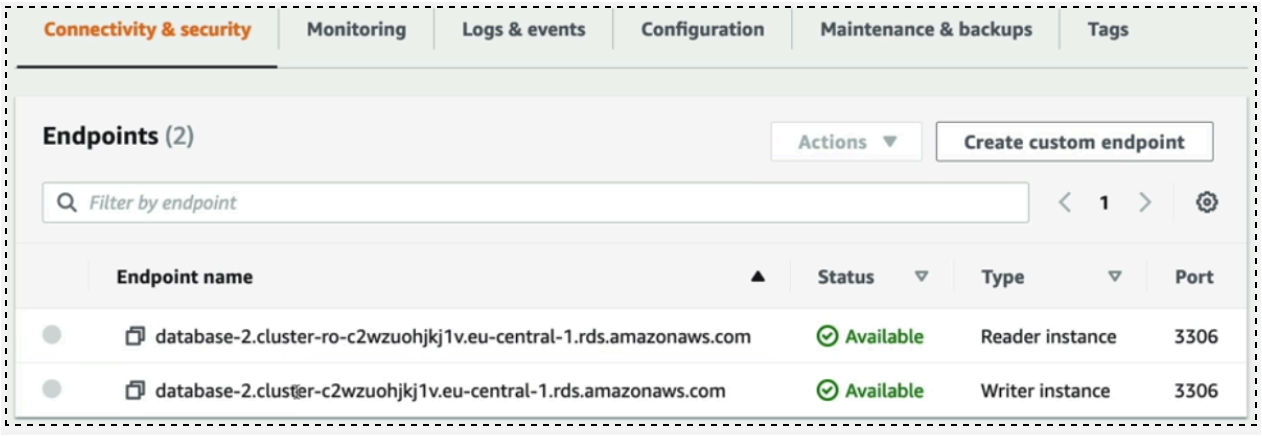
يعتبر كل ال configuration اتكلمنا فيها.

بعد ما كريتناها .. هتلاقيها بالشكل دة

- ال Regional Cluster اللى بيوضحلك انت ف انهي region
- Writer instance ودة ال main data base اللى بيحصل عليه read & write عادي وبيكون عدده 1 فقط
- Reader Instance ودة بيحصل عليه read فقط وبيبقي فيه لحد 15



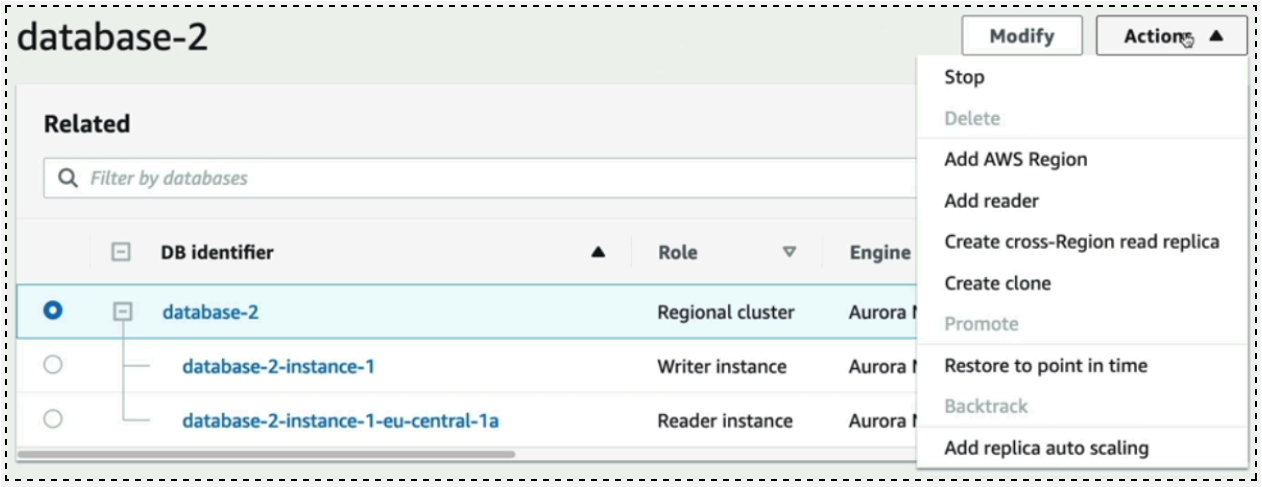
بعد ما ت select ال Regional Cluster هتلاقي ال endpoints الخاصة بال reader & writer وال ports وكدا..وممكن تحدد اي instance لوحدها وهيظهرلك بردو ال endpoint



عندك Actions كتير تقدر تعملها

Reader instance = replica

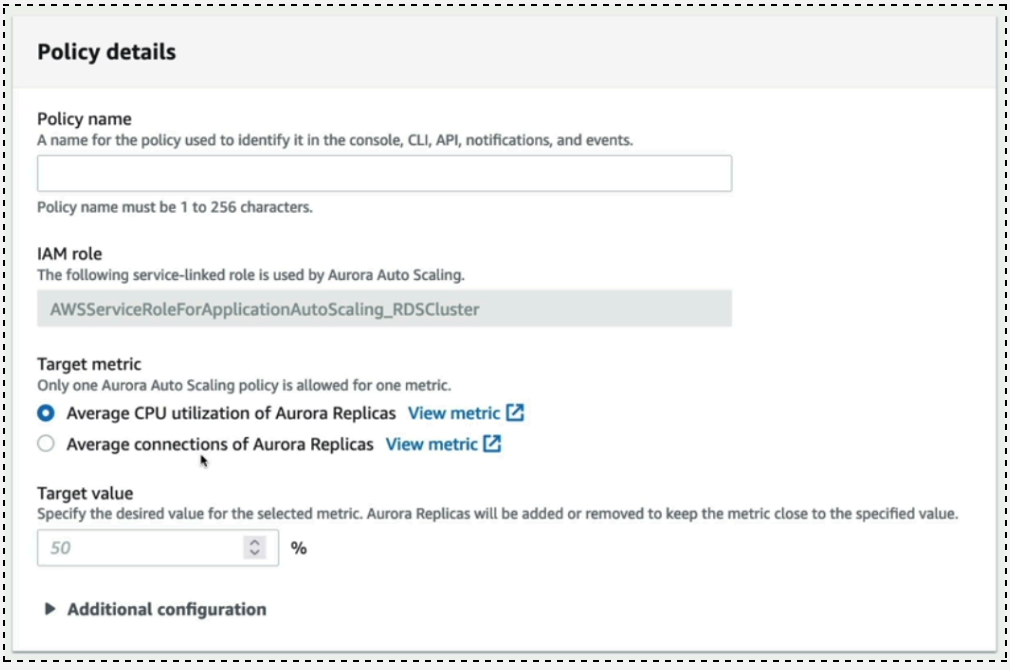
- تزود reader instance فى AZ مختلفه فى نفس ال Region
- تزود replica فى Region مختلفة
- حصل مشكلة فى ال main مثلا ف ت restore to point in time
- تضيف Replica ل Auto scaling



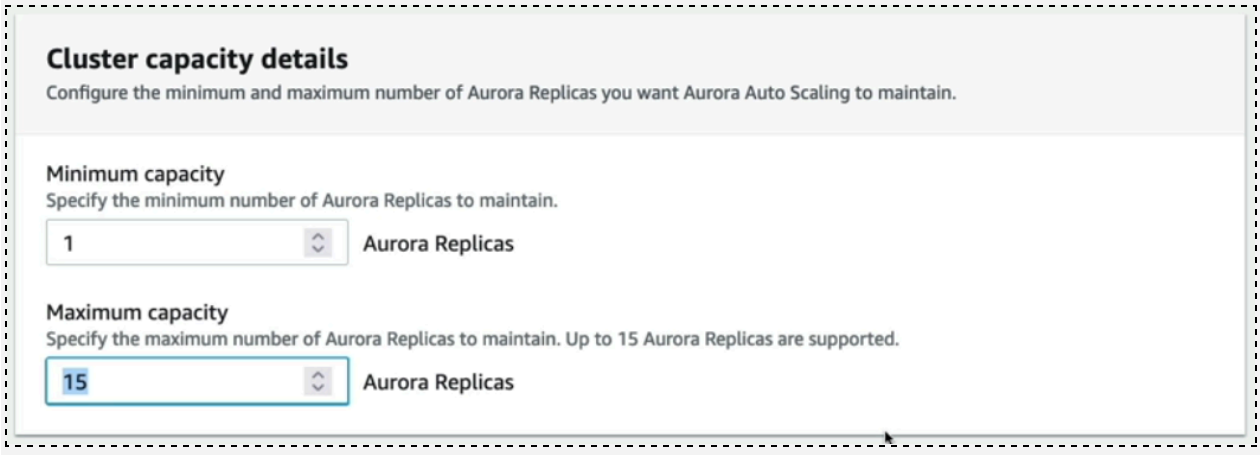
I'm Here

دلوقتي هنضيف **Replica to Auto Scaling Group**

بنكريت policy وظيفتها ان معناها لو ال CPU Utilization زاد عن حد معين وليكن 50% او عدد ال connection زاد عن رقم كذا عال Replicas اللى already موجودة .. كريتلي replica اكتر

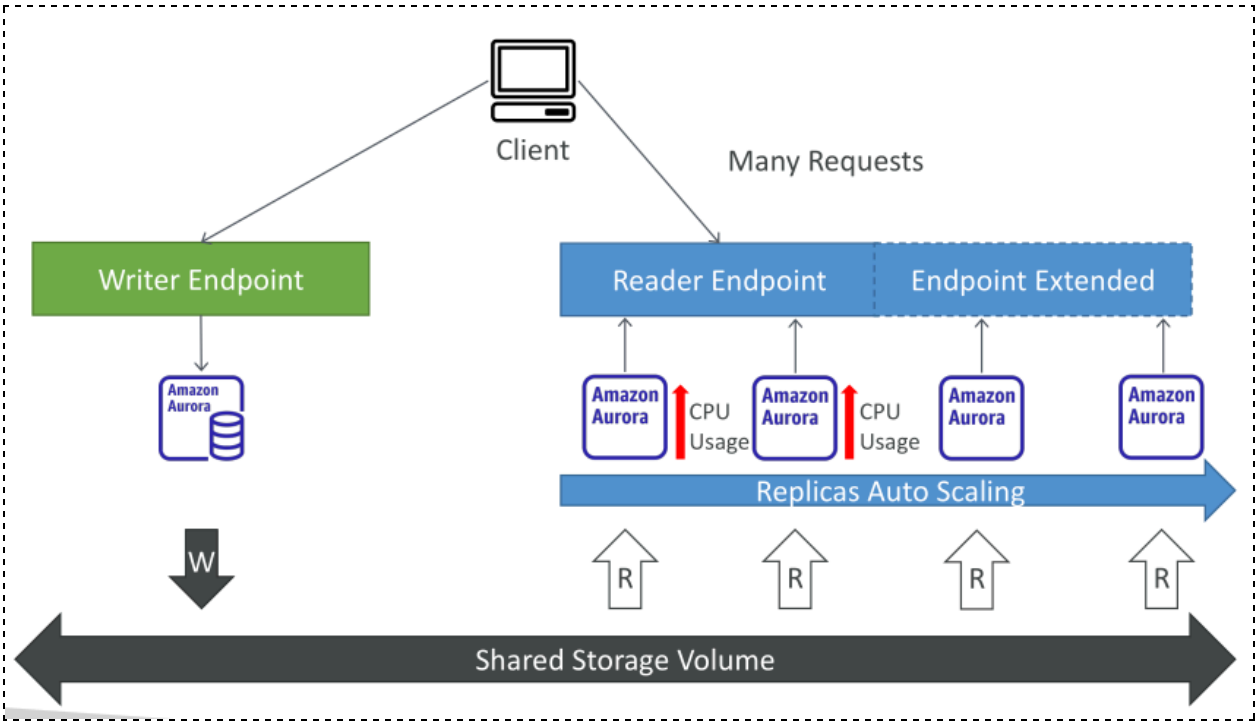


وبتحدد عدد ال Maximum Replicas Number



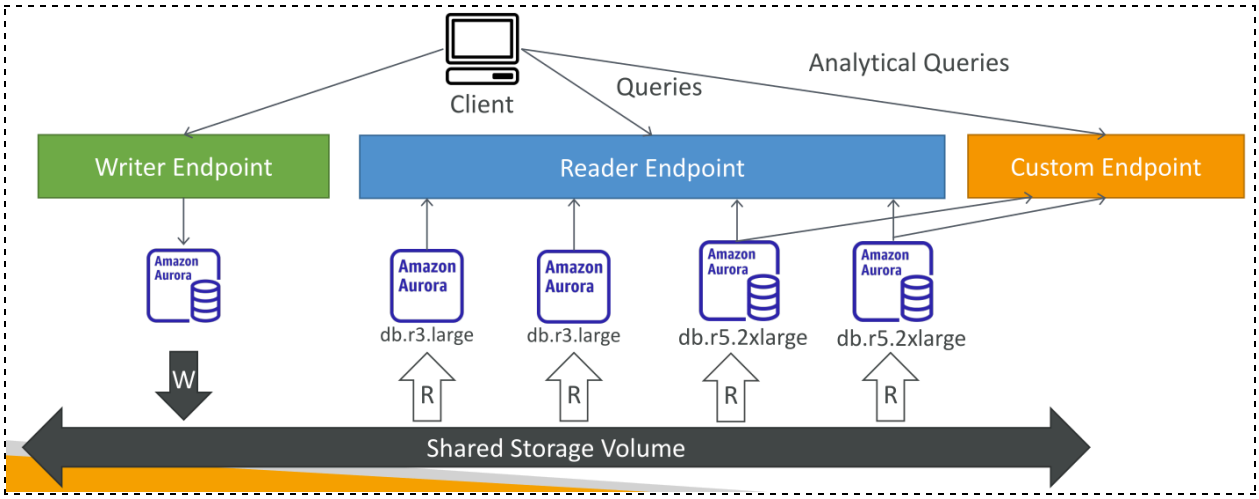
➥007 Amazon Aurora - Advanced Concepts

- **Aurora Replicas - Auto Scaling**



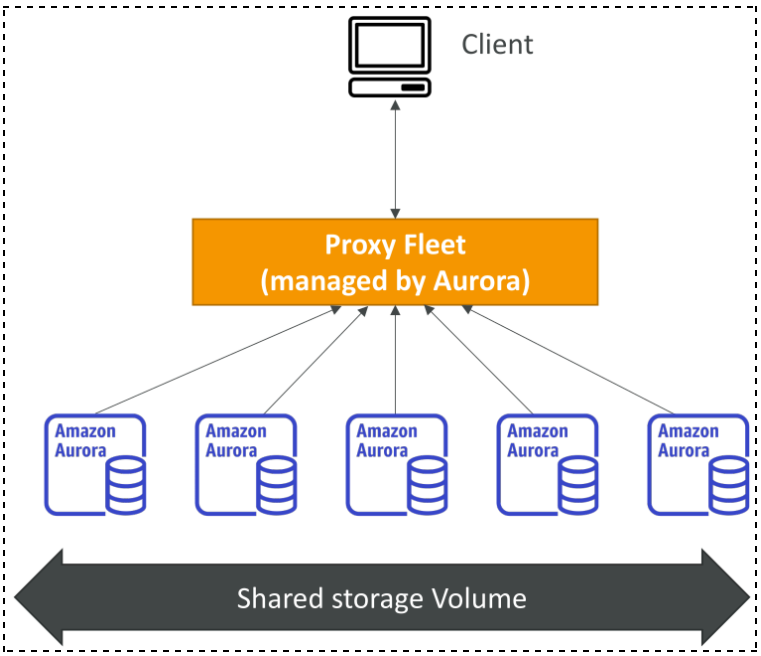
- بعد ما ال CPU utilization يزيد ال replicas بتزيد ب auto scaling وال reader هيا كمان بت extend معاهم endpoint.

• Aurora – Custom Endpoints



- بنكريت ال custom endpoint علشان نوزع traffic معين على instance معينه .. بمعني انك مثلا عاوز ال Analytical queries traffic يروح على instance تستحمل الشغل دة تكون اقوي من ال replicas العادية ف بنعمل isolation بين ال reader & custom endpoint

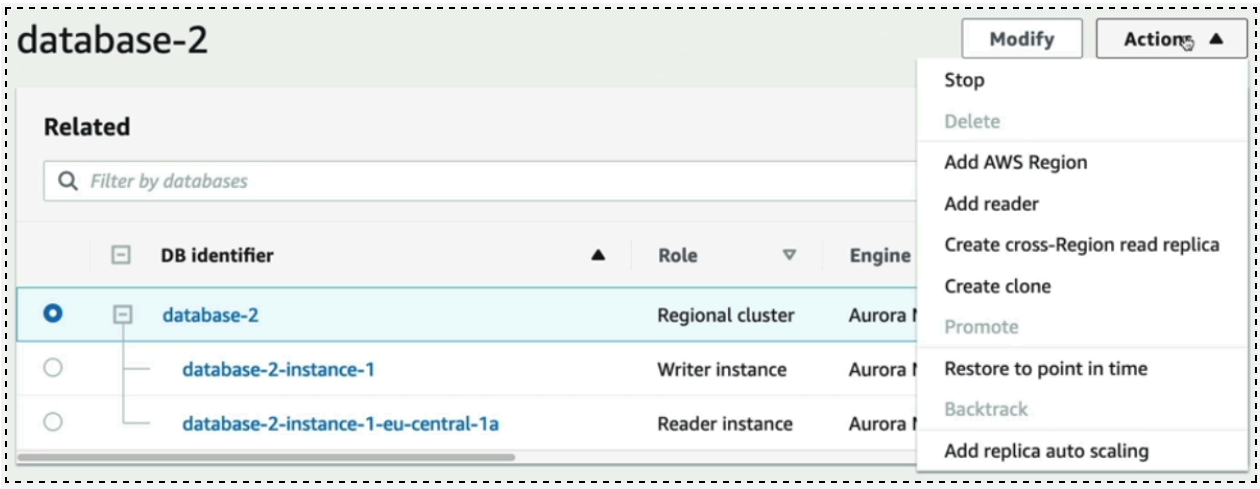
• The diff between Aurora Server less and normal one



- مناسبة اكتر لل unpredictable workloads او ال infrequent access ونظام الدفع pay as you go ومش محتاج تعمل capacity planning

- ال data bases جاهزة بال auto scaling بناءا عال usage بتاعك.
- يبقي ال client بي connect ل proxy fleet ودة managed by Aurora وبيتوزعه عال instances اللى بيكونو محطوطين فى auto scaling وعلي حسب بقا ال usage ال instances بتزيد او بتقل.

- **Global Aurora**



- من **Action** زي مشوفنا تقدر كمان تضيف **AWS Region** بس لازم تكون مختار Aurora Version بيدعم ال Global database feature ونوع instance يكون compatible مع ال Global database feature
- إستخدامها : بيكون عندك ال main + reader replicas فى ال primary region ولكن انت مثلا محتاج ت serve عملاء اوربيين مثلا ف عاوز يبقي عندك replicas ف region تانيه بحيث يكون الموضوع اسرع ف بالتالي بت add region .. يبقي انت كنت ف وليكن us-east-1 (N. Virginia) region وضيفت

eu-west-1 (Ireland) **to serve European users with low-latency read access and to have a disaster recovery solution.**

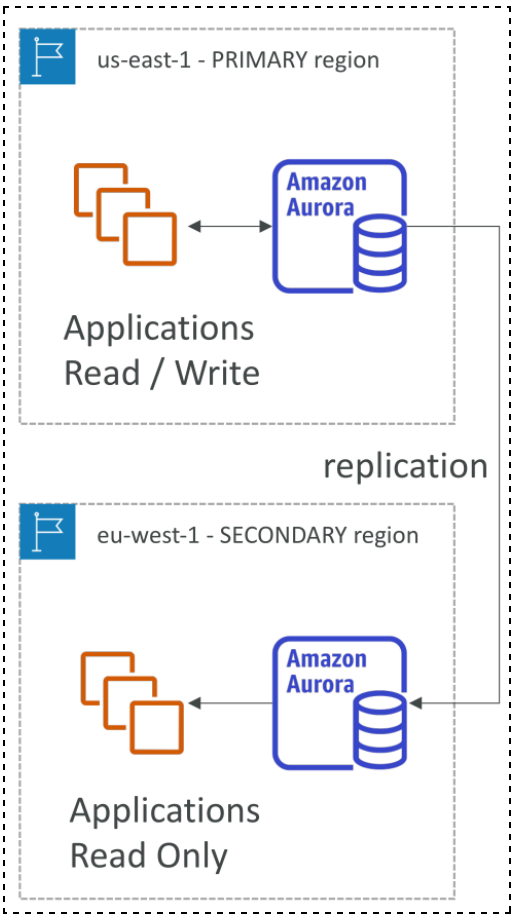
- **Aurora Global Database (recommended):**

- دة اللى تقدر يكون عندك:

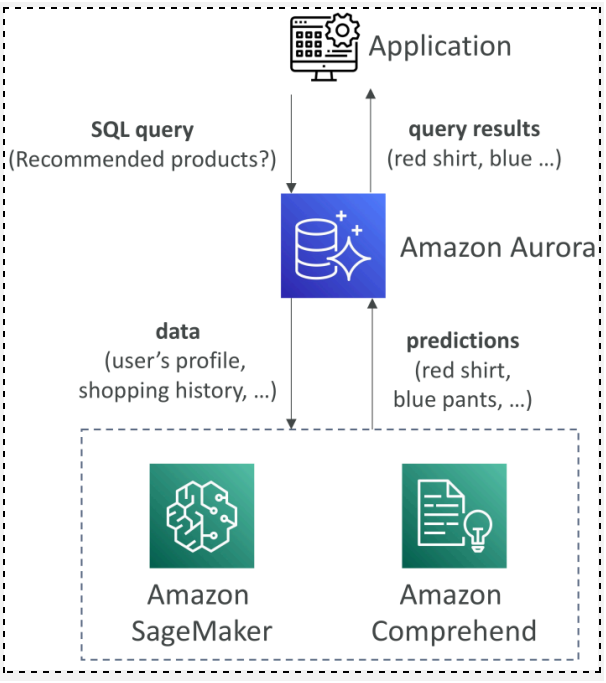
1. **Primary region** >> read & write
2. **Up to 5 secondary (read-only) regions**, replication lag is less than 1 second
3. Up to 16 Read Replicas per secondary region

- ال primary 15 read replicas ياعني + 1 main ياعني توتال 16 .. ف عندك بردو فى ال5 regions الاضافيه اللى تقدر تضيفهم كل واحدة يكون فيها 16 replicas

- Promoting another region (for disaster recovery) has an RTO of < 1 minute
- Typical cross-region replication takes less than 1 second



- **Aurora Machine Learning**



- عباره عن integration بين Aurora و AWS Machine Learning Services
- عندك مثلا Aurora متصله ب ML Services معينه ف تقدر انت من خلال simple SQL Query ي run it ال application وليكن مثلا اية ال recommended products ف Aurora بما انها database تبعت شوية داتا للل ML services زي ال shopping profile , shopping history وهكذا وتشتغل ال ML services دي عالداتا دي وتعرف اية ال recommended products .. وتبعت ال query results دي لل Aurora اللى هتبعتها لل APP.

➥008 RDS & Aurora - Backup and Monitoring

- **RDS Backup:**
- **Automated Backup**

- بيحصل backup كل 5 دقايق بيتجدد ياعني لو عملت restore ل any point in time هت restore اخر نسخة اللى معموله من حوالي 5 دقايق .. وال backups دي بتفضل من 0 : 35 يوم

- **Manual DB Snapshots**

- دي بتعملها manual والفرق بينها وبين ال automated ان دي بتفضل موجودة لحد ما انت تمسحها التانية اخرها 35 يوم

💡لو مش عاوز costs عالية .. لو بتشتغل مثلا على database ساعتين فقط فى الشهر .. بدل ما تشغل ال Automated Backup ويعد عليك كل ساعة ال data بتاعة ال backup دي موجودة فيها .. خد افضل snapshot بعد ما تخلص ساعتين الشغل دول وابقي اعملها restore دة ارخص بكتير..لانك حتي لو عامل stop لل data base بعد ما تخلص الساعتين شغل ف انت هتدفع عليها حتي لو واقفة.

- **Aurora Backup:**
- **Automated Backup**

- بيحصل من 1 : 35 يوم ومتقدرش توقف ال backup دي اقل حاجة هتفضل ال data يوم.. وليك point in time recovery فى اي وقت خلال المدة دي.

- **Manual DB Snapshots**

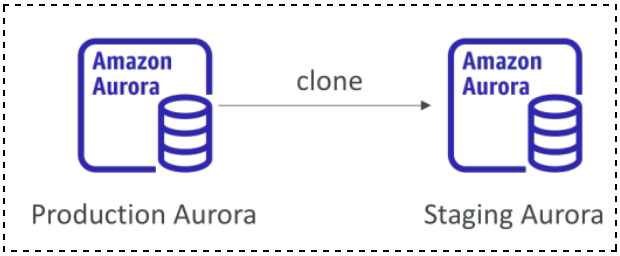
- زيها زي اللى فوق .. بتعملها manual وبتخليها موجودة زي مانت عاوز وتمسحها وقت ما تعوز.

RDS & Aurora Restore options

- لما بت restore ال backup سواء من snapshots , Automated فبتتعمل new database بيها.
- تقدر ت restore database من S3 ودي عباره عن service بنخزن بيها ال objects على AWS Cloud هنتكلم عنها بالتفصيل قدام .

- Restoring MySQL RDS database from S3
  - Create a backup of your on-premises database
  - Store it on Amazon S3 (object storage)
  - Restore the backup file onto a new RDS instance running MySQL
- Restoring MySQL Aurora cluster from S3
  - Create a backup of your on-premises database using Percona XtraBackup
  - Store the backup file on Amazon S3
  - Restore the backup file onto a new Aurora cluster running MySQL

Aurora Database Cloning



- بت clone بتنسخ ال main database اللى عال production لو محتاج تعمل test عليها ف حاجة معينه علشان متضرهاش هيا .. ف بتعمل ال test بتاعك عال cloned database ودة اسرع من انك تاخد snapshot وت restore it
- بتستخدم protocol اسمه copy-on-write

➥009 RDS & Aurora Security

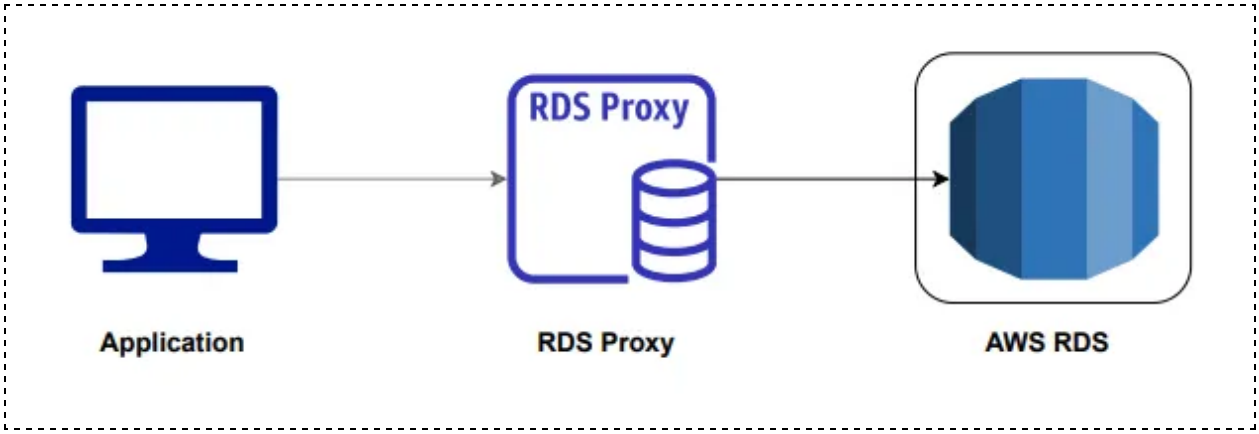
- **Encryption at Rest**
  - لما بت enable ال Automatic Encryption لل RDS or Aurora ف كل ال data المتخزنة بتكون encrypted دة بيشمل ال database instance, automated backups, read replicas, and snapshots.
  - مينفعش ت enable ال encryption على database مش معمولها encryption الحل انك بتاخد snapshot وتعملها restore as new encrypted instance
- **Encryption in Transit**
  - RDS & Aurora بي support encryption in transit using SSL/TLS.
  - بت configure ال database connection to use SSL/TLS بحيث تضمن ان ال data اللى بتتنقل بينك وبين ال app encrypted
- **IAM Database Authentication**
  - ممكن ت create IAM Policies تسمحلك بانك ت access ال RDS & Aurora
  - ممكن تستخدم IAM Tokens بدل من ال data base passwords
  - IAM authentication relies on temporary authentication tokens generated by AWS. These tokens are valid for 15 minutes and can be used in place of a password when connecting to the database.
- **Security Groups**
  - تقدر ت control the network access to your database من خلال انك تقدر تتحكم ف مين ي access ال database ممكن ip معين او ports معينه ..
- **SSH Access**
  - متقدرش ت ssh عال databases دي لانها managed serves by AWS ولكن ممكن من خلال RDS Custom اللى اتكلمنا عليها قبل كدا.

I'm Here

- **Audit logs**

- فيها كل حاجة حصلت على ال data bases وكل ال queries **ولكن** مش بتفضل موجودة لوقت طويل ..ف علشان كدا تقدر تبعت ال audit logs دي لل cloud watch علشان تحفظها هناك.

➥010 RDS Proxy



- **Proxy** باختصار عباره عن وسيط بين ال client & Server فى حالتنا ال , application database server

- **Use Case**

- طب هو وسيط بين ال client , database فى الحالة دي ليه محتاجينه طالما احنا نقدر ن connect directly to the database ؟ فى عندنا e-commerce app مثلا واثناء ال sale days فيه ضغط كبير و connections كتيرة رايحة عال database ف دة طبعا ممكن يوقعها ف علشان كدا بنستخدم ال RDS Proxy بحيث يخف الضغط عنها وهو بيكون عباره عن serverless وmulti AZs وبي scale up and down بناءا على حجم ال connection وبي direct the connections بسرعه سواء ل standby database او replica

- **Supports** RDS (MySQL, PostgreSQL, MariaDB, MS SQL Server) and Aurora (MySQL, PostgreSQL)

• RDS Proxy Enhanced Security

- لو عندك web app وعاوز ي connect to a RDS:

• Without RDS Proxy

- فى الغالب بت save the user name & password فى ال app code ودة خطر.

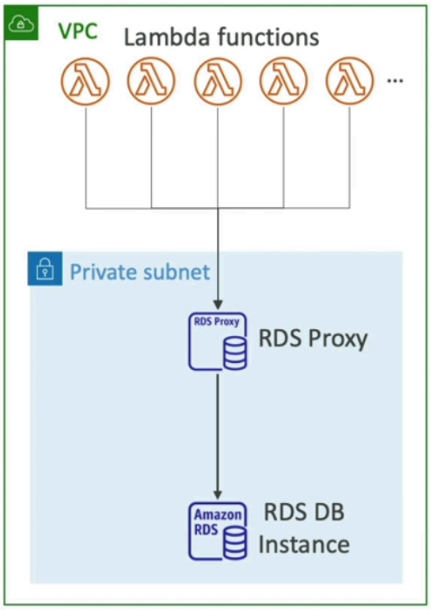
• With RDS Proxy

- تقدر تستخدم IAM roles فى انك تتحكم ف who can connect to the database
- او بي retrieve ال database credentials من ال AWS Secrets Manager وبي Update it automatically ف ب كدا مش محتاج تحط ال password فى ال application

- **AWS Secrets Manager:** عباره عن service بيتخذن فيها ال sensitive informations زي ال database passwords

💡RDS Proxy لازم يحصله access فقط من خلال ال VPC مش من ال public .. ياعني ال resources فقط اللى جوا ال VPC تقدر تأكسسه غير كدا لا

- من اكتر ال services اللى بتستخدم RDS Proxy هي Lambda function وهنتكلم عنها بالتفصيل بعدين ..



➥011 ElastiCache Overview

- **Diff Between Cookie & Cache**
- **Cookie**

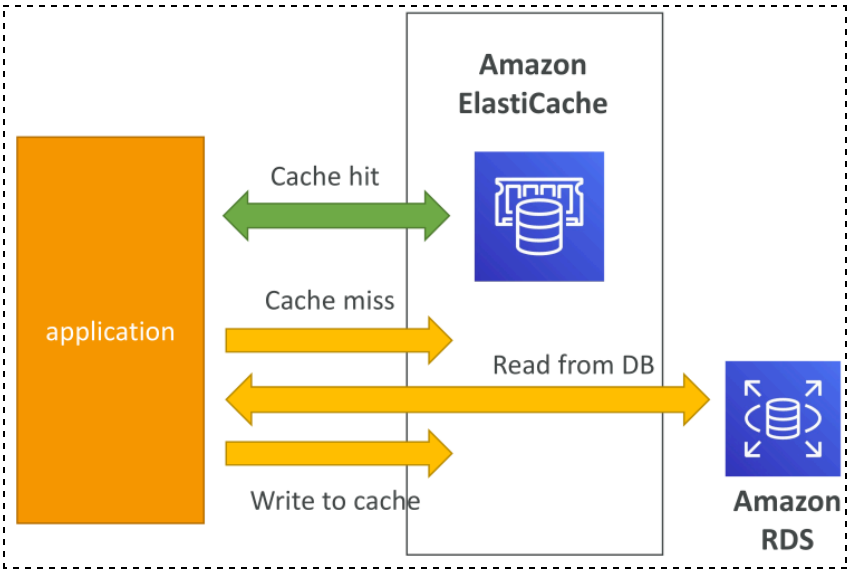
- غرضها انها تخذن ال data اللى ليها علاقه بال user login session وال preferences بتاعته شوية داتا بسيطة ياعني .. وبتتخذن عند ال user's browser وزي ما شرحنا قبل كدا بتتبعت فى ال requests اللى بين ال server وال user .. وال cookie دي ال expiration date بتاعها بنعملو set من خلال ال server ولو انتهي او عملت مسح لل cookies اللى فى ال browser مثلا تلاقي ان ال login sessions عملت log out وكدا زي ما بيحصل ..
- Cookies are sent with every HTTP request to the server.
-

- **Cookie**

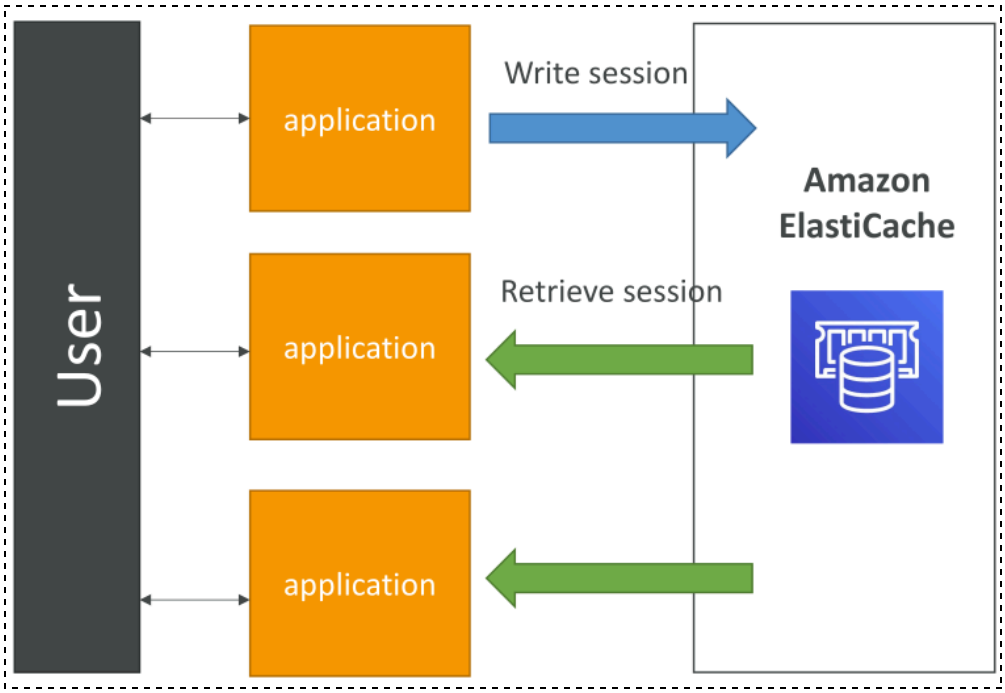
- اكبر شوية من اللى فاتت عباره عن web resources غرضها تقليل ال load times عال server او ال database وتقليل ال bandwidth .. عباره عن files زي HTML pages, images, CSS stylesheets, JavaScript files, and other multimedia content .. بتتخزن locally بردو عال user's computer by the browser وبيحصلها access لو جيت تزور نفس ال website او ال page .. ليها بردو expiration date محطوط فى ال cache headers الكلام دة اتعمله set at the server
- Cached resources are accessed locally by the browser without additional HTTP requests to the server.

- **Amazon ElastiCache**

- عباره عن Managed Service By AWS



- لو ال application عمل ElasticCache query لو ال queries دي already موجودة ومتخزنة فى ال AWS ElasticCache ف دة إسمه **Cache Hit**
- لو ال caches دي مش موجودة فى ال AWS ElasticCache ف هيروح ال App يجيبها من ال DataBase



- مثال كمان : ال user هي login ف اي واحد من ال Apps بتوعك فالApp هي ال Write ال Session data دي فى ال ElasticCache .. ف لو ال user دة حصلو redirecting لاي instance تانية شايله ال App بتاعك ف ال instance دي بت retreive علطول ال session data دي من ال ElasticCache ف بيفضل ال user logged in ومش بيحتاج ي log in again دة كدا اسمه ان ال Application بتاعك **Stateless**

- **Amazon ElastiCache Engines**

Follow ……