AWS CloudWatch

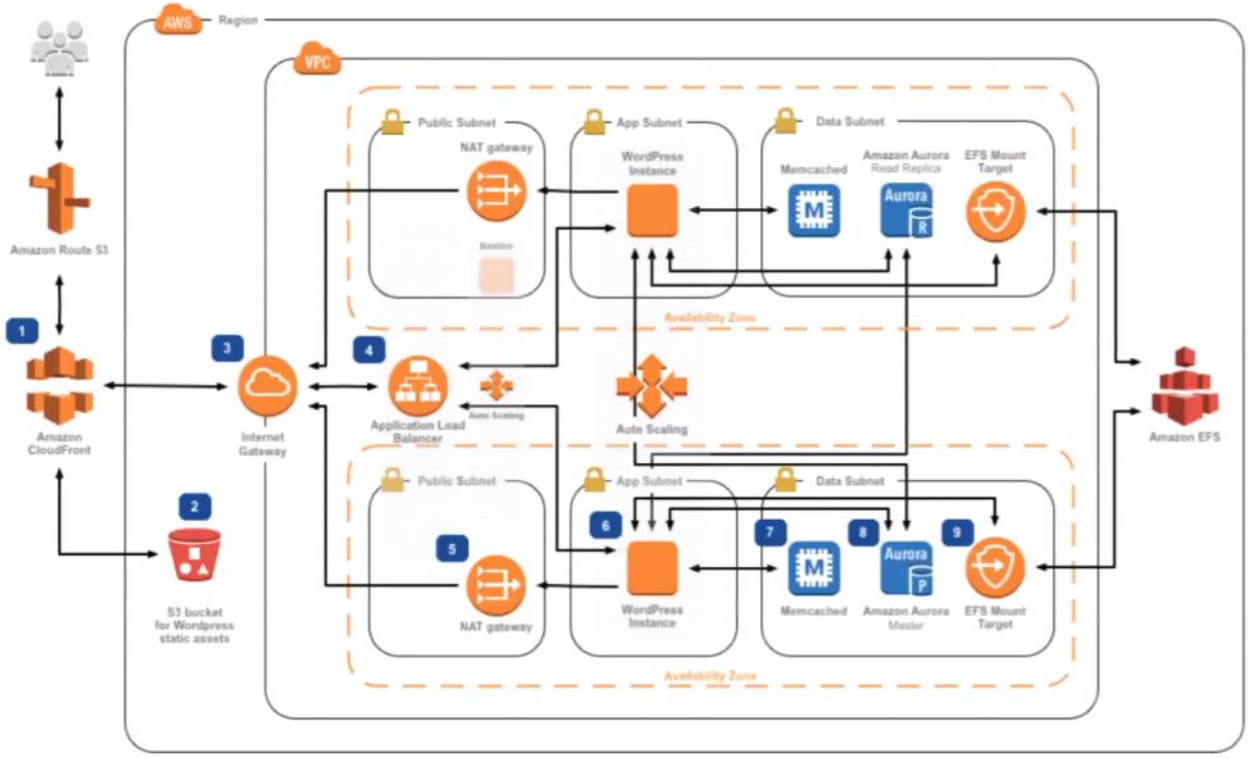
Friday, 13 August 2021 23.30

AWS CloudWatch

Table of Contents

1 What is CloudWatch?

Calling Manager



✓ [What is CloudWatch?]

CloudWatch is the monitoring service of AWS. It enables you to monitor metrics of resources and applications that run on AWS as well as on-premise servers. It also lets you get logs, set alarms, trigger events and analyze utilization.

CloudWatch

What is CloudWatch?

- You can get logs, set alarms, trigger events and analyze utilization by CloudWatch.

AWS nin configurationdan sorumlu monitoring servisi diyebiliriz
Log kayitlari 10 yila kadar metricleri ise 15 aya kadar saklayabiliriz

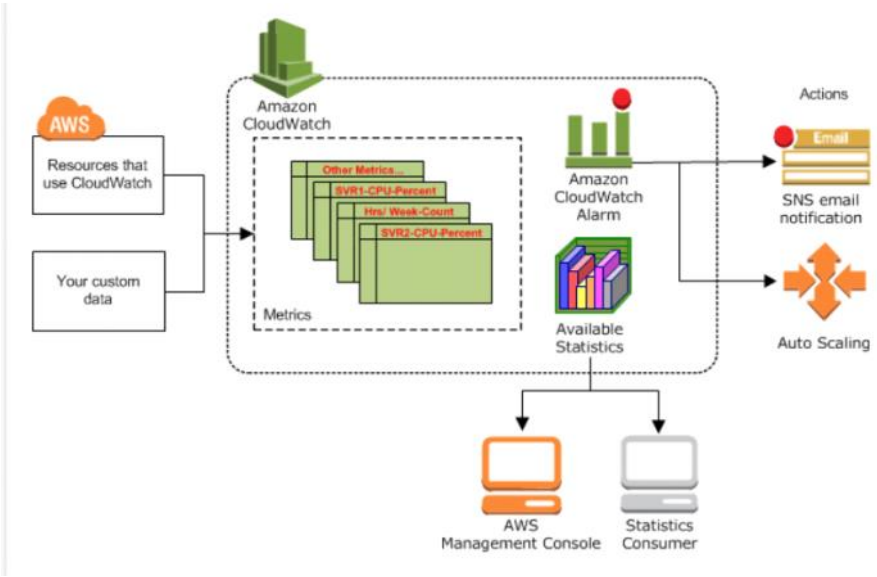
CloudWatch

What is CloudWatch?

- CloudWatch is the **monitoring service** of AWS.
- It enables you to monitor **metrics of resources and applications** that run **on AWS** as well as **on-premise servers**.

✓ How does CloudWatch work?

CloudWatch collects metrics and logs from AWS resources, applications and services running on AWS and on-premises servers. You can visualize applications and infrastructure using **Dashboards**; correlate logs and metrics side by side to troubleshoot and set **Alarms**. It also enables you to automate response to operational changes with **Events** and **Auto-Scaling**. You can leverage metrics (up to 1 second), extended data retention (15 months) and real-time analysis with **Metric Math**.



2 CloudWatch Basic Components

✓ Basic Components of CloudWatch

CloudWatch
Basic Components

- Metrics: Metrics are the **measurable data** about your resources, apps or services like EC2 instance CPU utilization or the number of objects in an S3 bucket.

Namespace : Verilen gorsel bakimdan ayni baslik altinda toplanmasi
Metrics: Metrics are the measurable data about your resources, apps or services like EC2 instance CPU utilization or the number of objects in an S3 bucket.

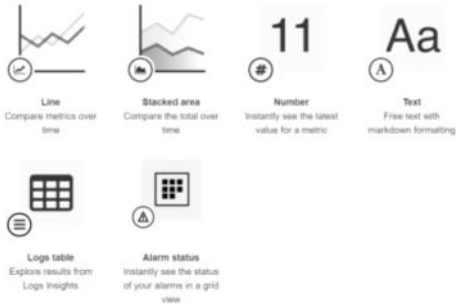
CloudWatch
Basic Components

- Dashboards: Dashboards enable you to create graphs and visualize your cloud resources and applications according to your needs.

Global olarak dusunebiliriz. Farkli regionlardaki resourcelerin metriclerini ekleyebiliriz

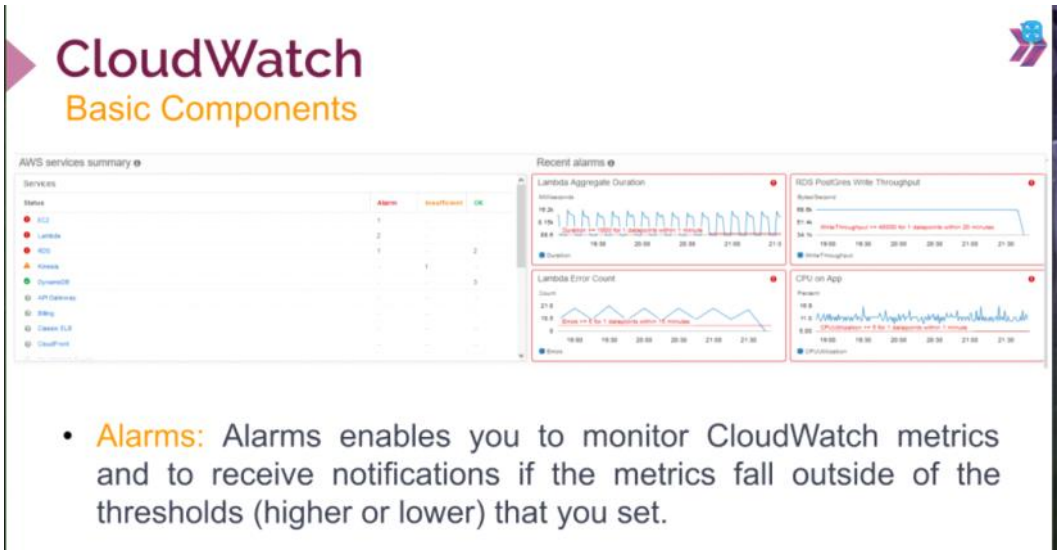
Refresh time : 10 saniyede bir en az olarak guncelleme imkani verebilmektedir

CloudWatch
Basic Components



- Dashboards: You can leverage dashboards as customizable home pages in the CloudWatch console to view metrics, logs and alarms.

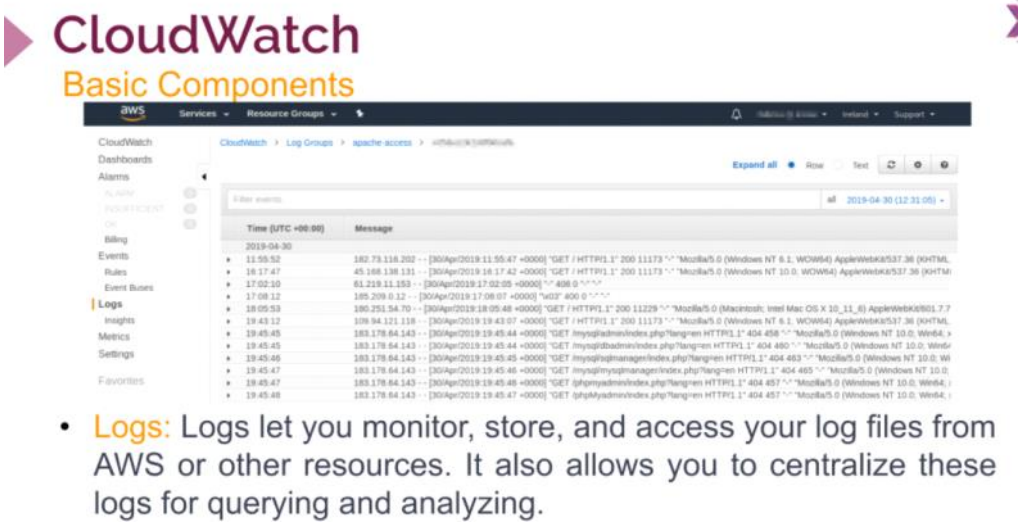
Dashboards: Dashboards enable you to create graphs and visualize your cloud resources and applications according to your needs. You can leverage dashboards as customizable home pages in the CloudWatch console to view metrics, logs and alarms.



Alarms: Alarms enables you to monitor CloudWatch metrics and to receive notifications if the metrics fall outside of the thresholds (higher or lower) that you set.
"Cloudwatch in gucu alarmlardan geliyor denilebilir."
Tresholds: ORN==> EC2 nun CPUkullanimi- %60 in uzerine gectigi zaman alarm gonder diyebiliriz bir nevi **esik** degeri

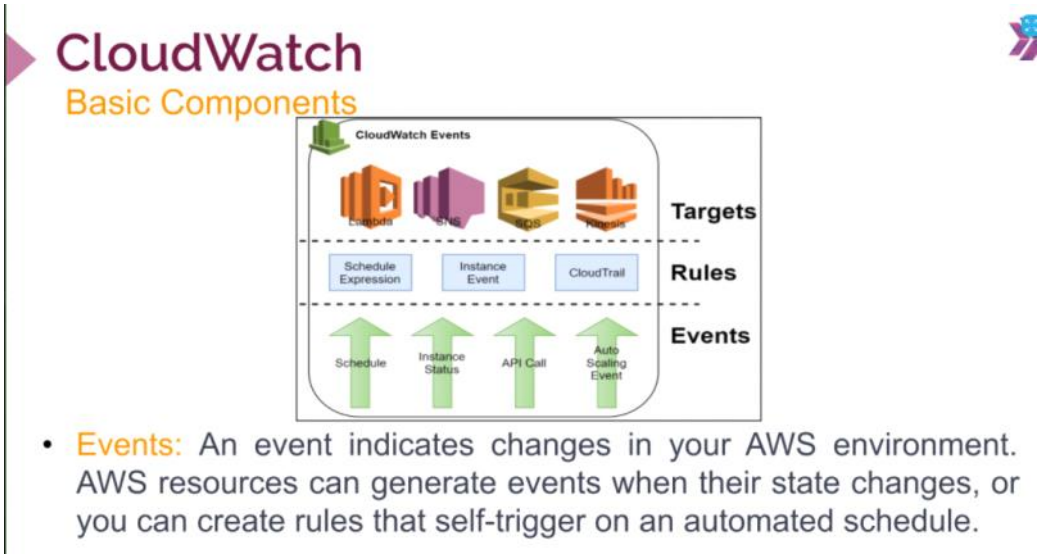
Alarmin **3** seviyesi vardir

Ilk asamasi insufficient
Diger asama OK ==> %60 in altindaki durumlarda(yukaridaki orn)
ALARM ==> %60 in uzeri



Logs: Logs let you monitor, store, and access your log files from AWS or other resources. It also allows you to centralize these logs for querying and analyzing.
Troubleshot ve **analysis** aklimiza gelebilir. EC2 uzerinde app calistiriyoruz ve her bir request log kaydi olarak geldigini dusunebiliriz

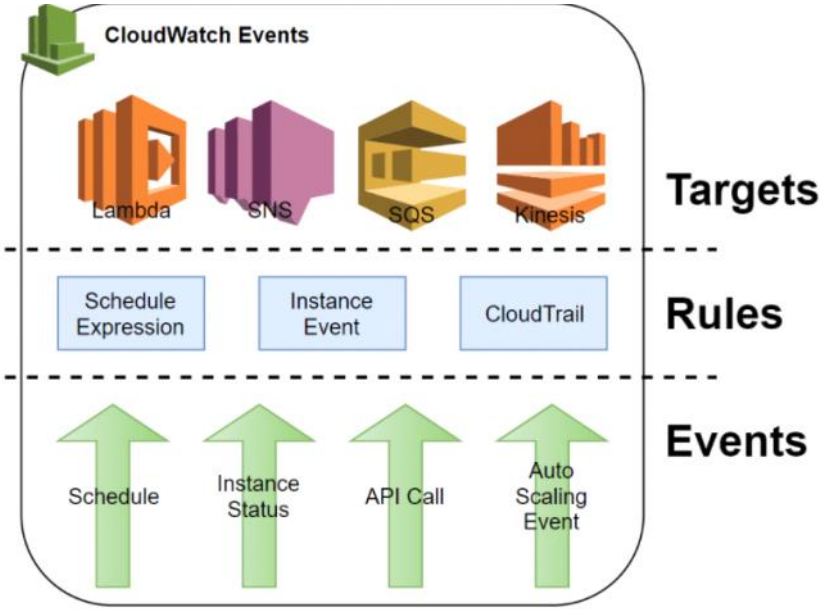
Log incelemesi önemli bir iş onun için piyasada özel toollar var mesela **Splunk** en iyisi ve piyasada çok kullanılanı genel de sitemci ve Siberciler logları inceler



Events: An event indicates changes in your AWS environment. AWS resources can generate events when their state changes, or you can create rules that self-trigger on an automated schedule. For example, you can receive an email via SNS if one of your EC2 instances goes to the "stop" state.

Gercege yakin olarak bizim sistemimizdeki degisikleri tutan **kurallar** butunudur
Mesela *her sabah 8 de lambda fonksiyonu calissin ve verecegimiz kurala gore instance olustursun veya olusani start etsin* gibi

is baslamadan 10 dakka once sistemi acsin bittikten 10 dk sonra kapat gibi



✓ Accessing and Pricing

CloudWatch Pricing

- No **up-front commitment** or **minimum fee**
- Pay for what you use
- Pricing varies by region
- Enables limited **Free Tier** usage

CloudWatch Pricing

Free Tier:

Metrics	Basic Monitoring Metrics (at 5-minute frequency) 10 Detailed Monitoring Metrics (at 1-minute frequency) 1 Million API requests (not applicable to GetMetricData and GetMetricWidgetImage)
Dashboard	3 Dashboards for up to 50 metrics per month
Alarms	10 Alarm metrics (not applicable to high-resolution alarms)
Logs	5GB Data (ingestion, archive storage, and data scanned by Logs Insights queries)
Events	All events except custom events are included
Contributor Insights	1 Contributor Insights rule per month The first one million log events that match the rule per month
Synthetics	100 canary runs per month

CLARUSWAY
WAY TO REINVENT YOURSELF

Accessing:

You can access CloudWatch using any of the following methods:

- Amazon CloudWatch console – https://console.aws.amazon.com/cloudwatch/
- AWS CLI
- CloudWatch API – For more information, see the Amazon CloudWatch API Reference.
- AWS SDKs – For more information, see Tools for Amazon Web Services.

Pricing: here

CloudWatch doesn't require any up-front commitment or minimum fee. As many other Amazon Web Services, you simply pay for *what you use* and will be charged at the *end of the month* for your usage. Pricing varies by region.
Many services vend metrics for free, so that you can leverage

within **Free Tier**:

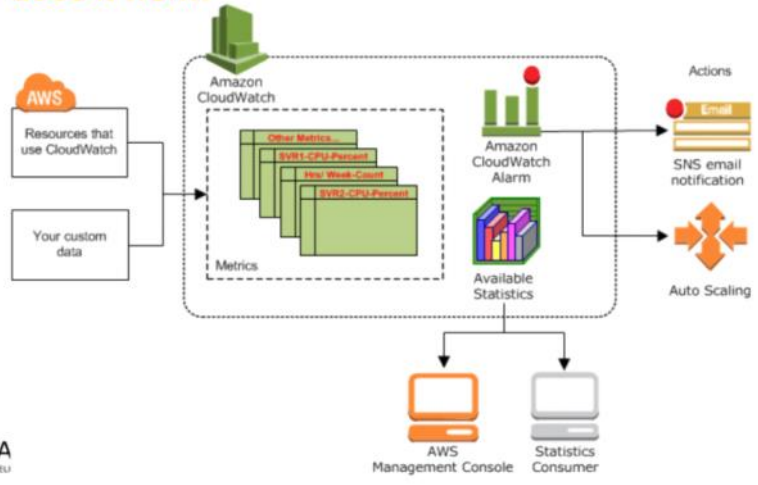
Metrics	Basic Monitoring Metrics (at 5-minute frequency) 10 Detailed Monitoring Metrics (at 1-minute frequency) 1 Million API requests (not applicable to GetMetricData and GetMetricWidgetImage)
Dashboard	3 Dashboards for up to 50 metrics per month
Alarms	10 Alarm metrics (not applicable to high-resolution alarms)
Logs	5GB Data (ingestion, archive storage, and data scanned by Logs Insights queries)
Events	All events except custom events are included
Contributor Insights	1 Contributor Insights rule per month The first one million log events that match the rule per month
Synthetics	100 canary runs per month

3 How does CloudWatch work?

CloudWatch
How does it work?

- CloudWatch collects **metrics and logs** from AWS resources, applications and services running **on AWS and on-premises servers**.
- You can visualize applications and infrastructure using **Dashboards**; correlate logs and metrics side by side to troubleshoot and set **Alarms**.
- It also enables you to automate response to operational changes with **Events**.

CloudWatch
How does it work?



CLARUSWA
WAY TO REINVENT YOURSELF

****
****

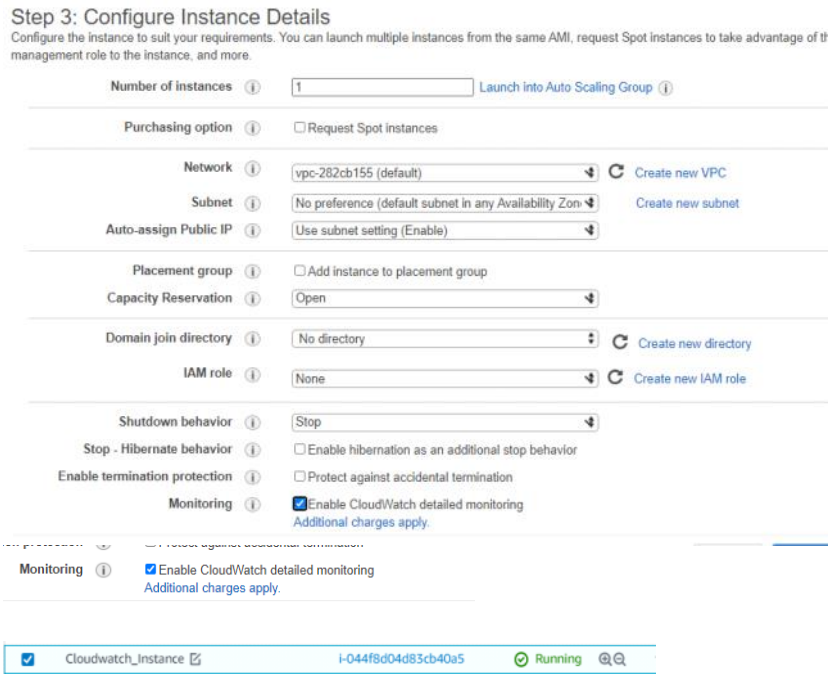
```
## Part 1 - Prep - Launching an Instance

STEP 1 : Create a EC2
```

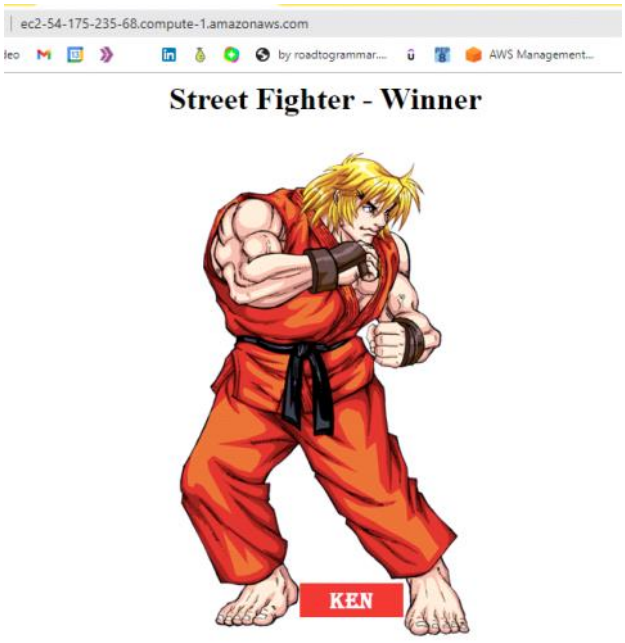
```
AMI             : Amazon Linux 2
Instance Type   : t2.micro
Configure Instance Details:
  - Monitoring ---> Check "Enable CloudWatch detailed monitoring"
Tag             :
    Key         : Name
    Value       : Cloudwatch_Instance
Security Group ---> Allows ssh, http to anywhere
```
- Set user data.

```bash
#! /bin/bash
yum update -y
amazon-linux-extras install nginx1.12
chkconfig nginx on
cd /usr/share/nginx/html
chmod o+w /usr/share/nginx/html
rm index.html
wget https://raw.githubusercontent.com/awsdevopsteam/route-53/master/index.html
wget https://raw.githubusercontent.com/awsdevopsteam/route-53/master/ken.jpg
service nginx start
```

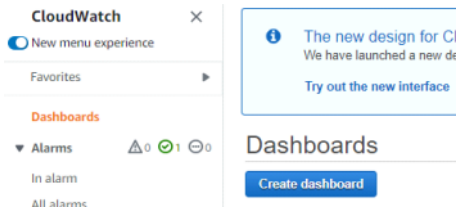
Userdata

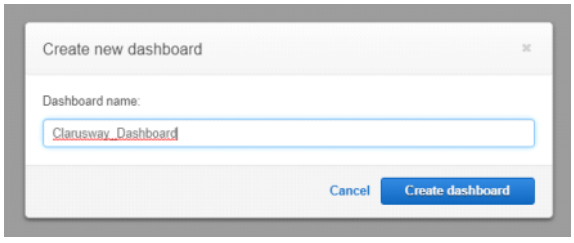
```
#! /bin/bash
yum update -y
amazon-linux-extras install nginx1.12
chkconfig nginx on
cd /usr/share/nginx/html
chmod o+w /usr/share/nginx/html
rm index.html
wget https://raw.githubusercontent.com/awsdevopsteam/route-53/master/index.html
wget https://raw.githubusercontent.com/awsdevopsteam/route-53/master/ken.jpg
service nginx start
```

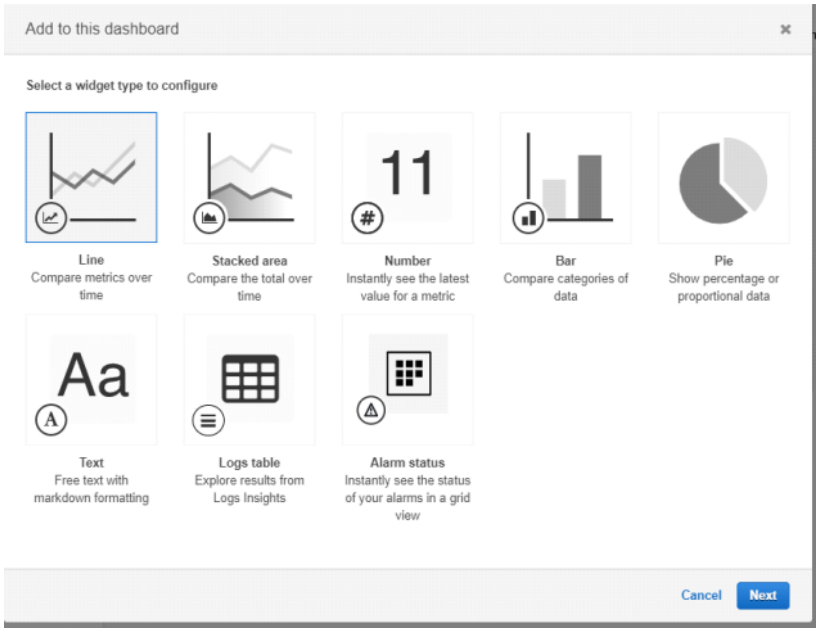


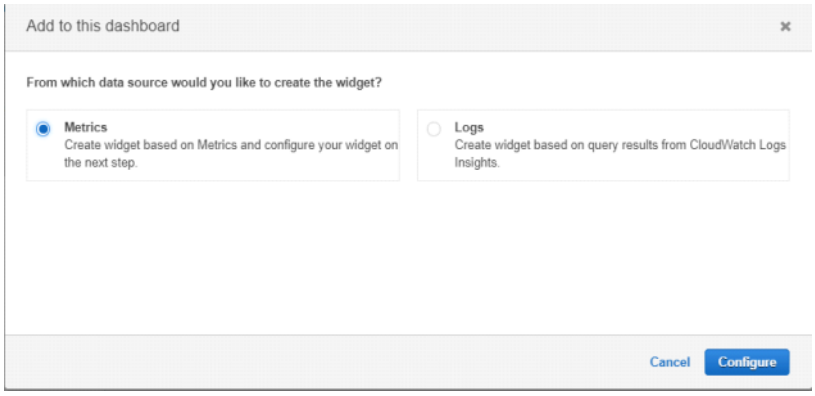


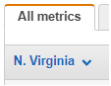

```
## Part 2 - Creating a Cloudwatch dashboard
```



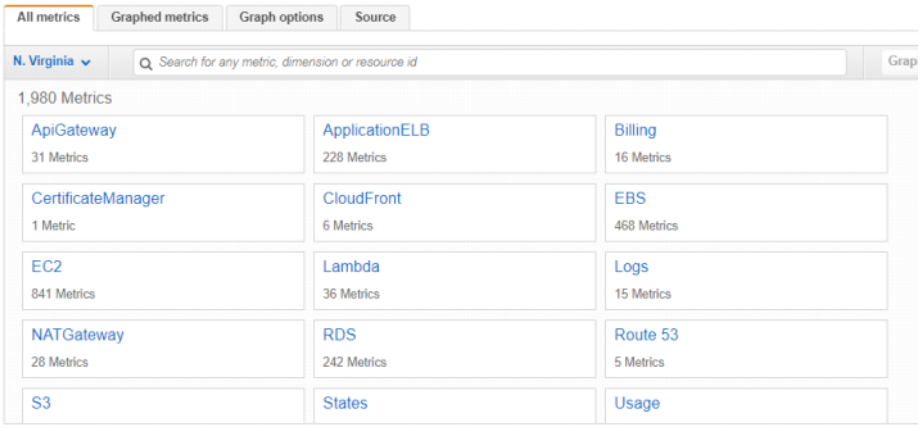


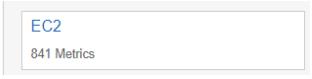




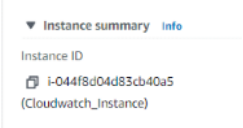


Metricler belirli basliklar altinda toplanmis ==> **namespace**



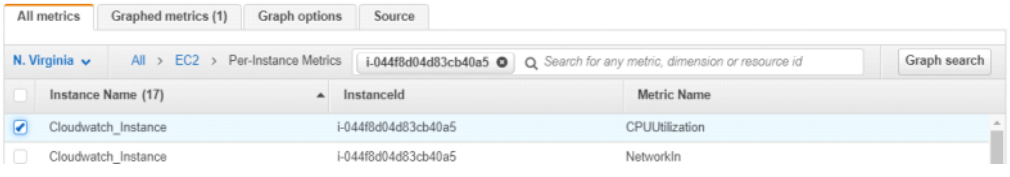


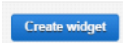
Sonra

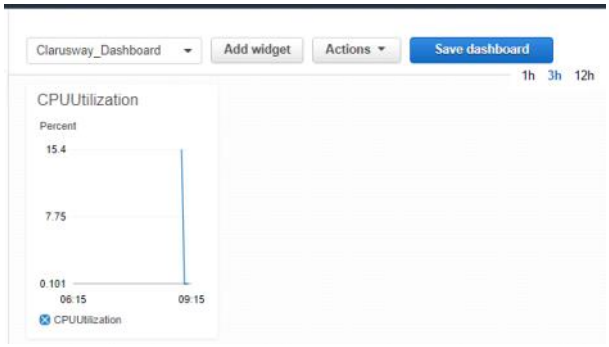


Ec2 id yi kopyalayalim

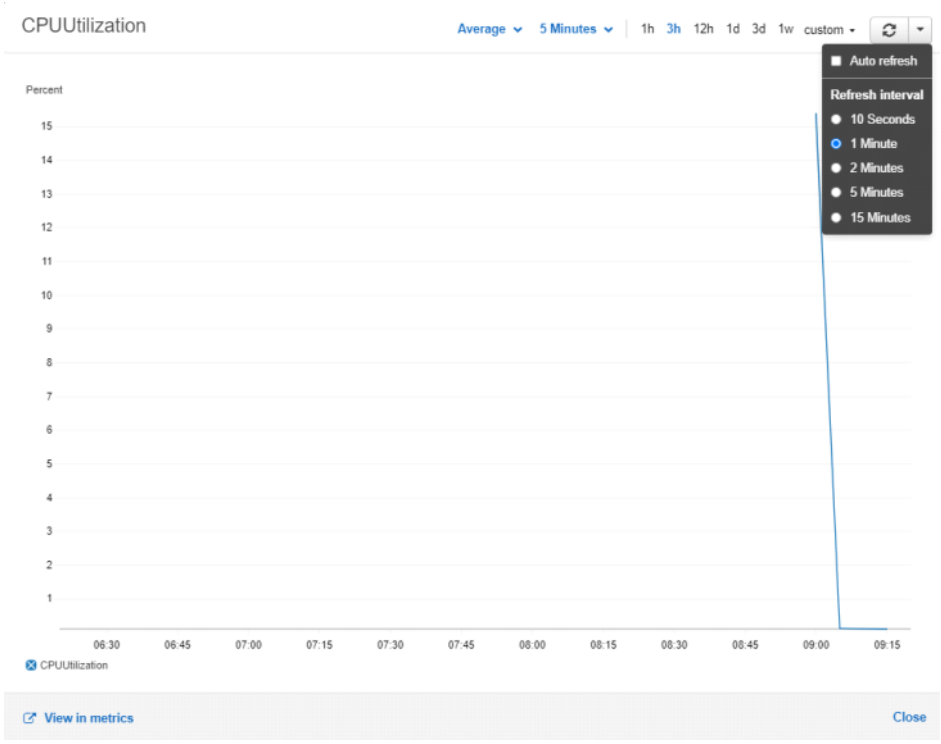
Alana yapistirip enter







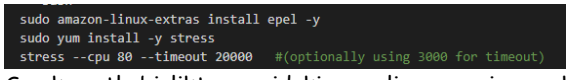
Sag ustten save edelim
Actigimiz ec2 icin cpu utilization basligi altinda metric ayarladik
Cloudwatch ile RAM degerlerini de gosterebiliriz



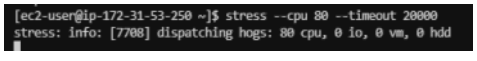
Refresh time i sag ustte gorebiliriz

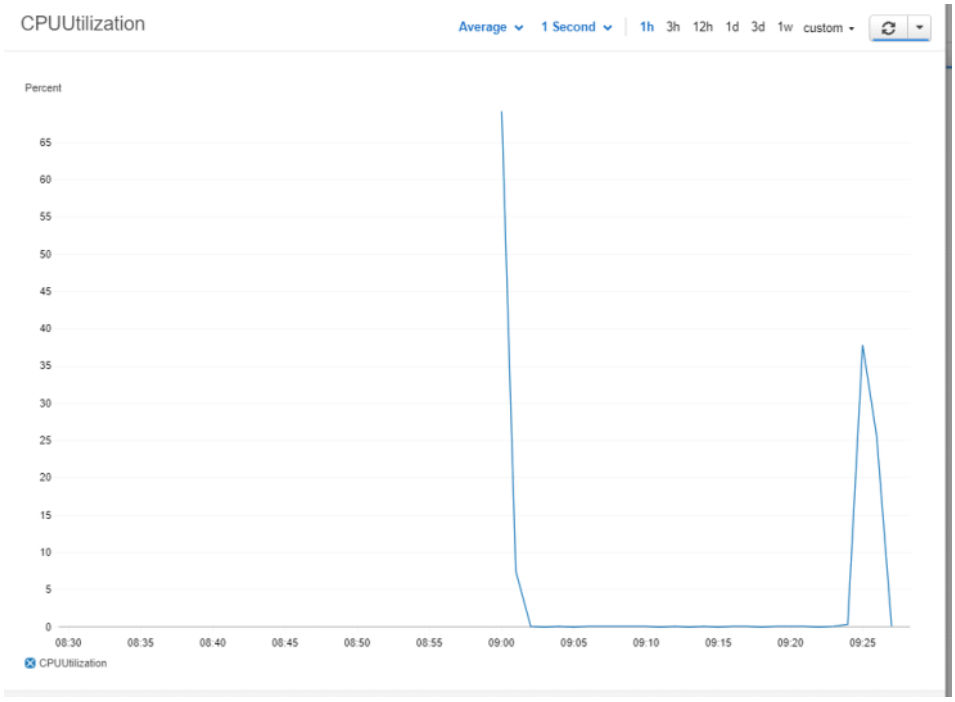
Instance ye ssh ile baglanalim

Asagidaki komutlari yazalim



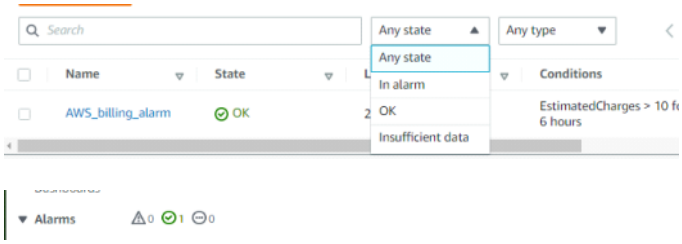
Son komutla birlikte asagidaki gorseli gormemiz gerekmektedir



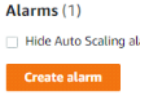


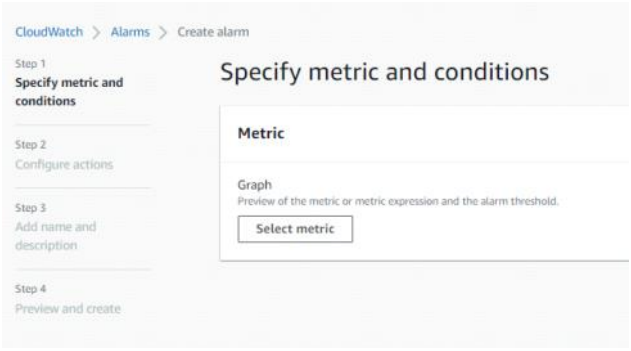
Ctrl c ile terminalden cikalim

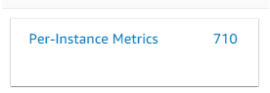
```
## Part 3 - Create an Alarm.
```

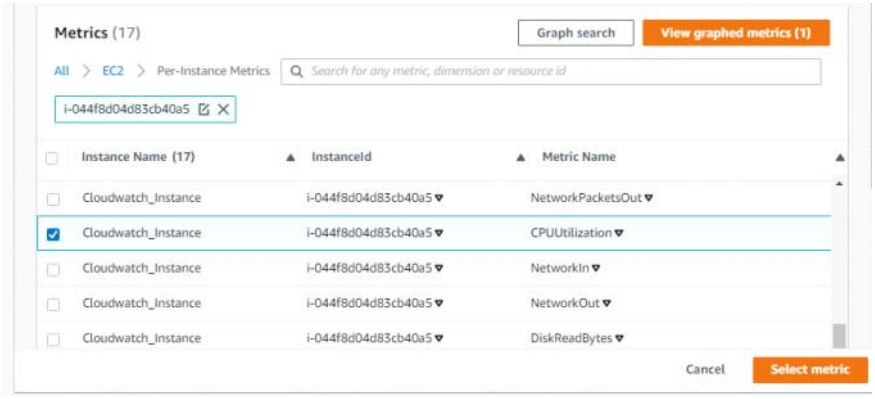


Alarm ile ilgili 3 asamayi gorebiliriz

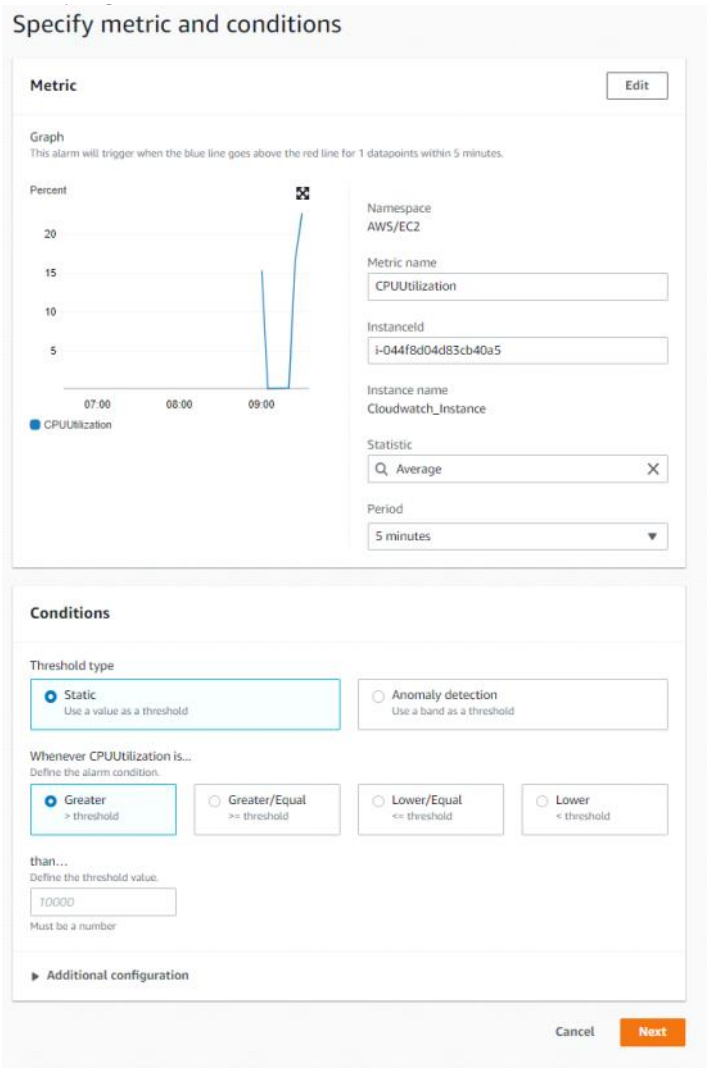




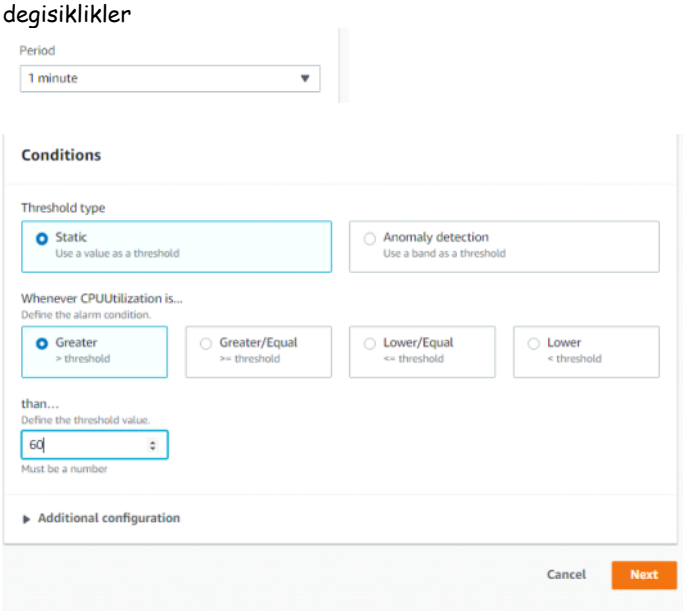


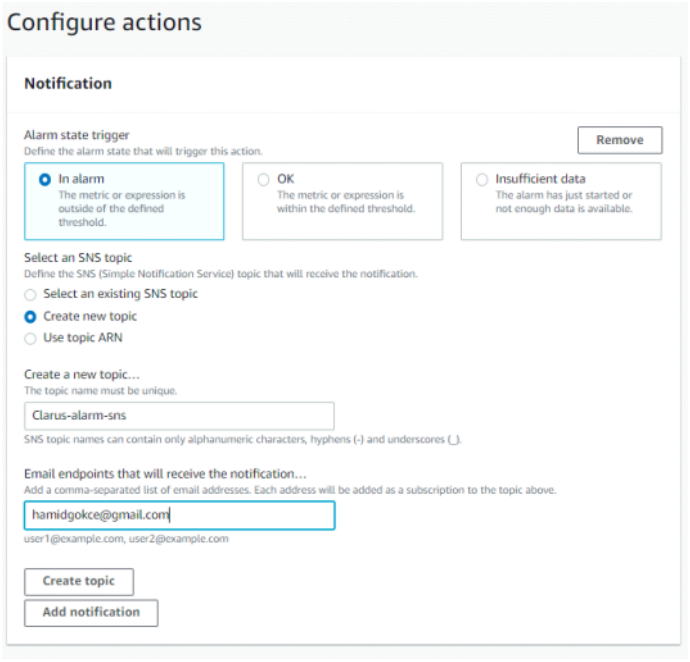


Ilk sayfa gorunumu

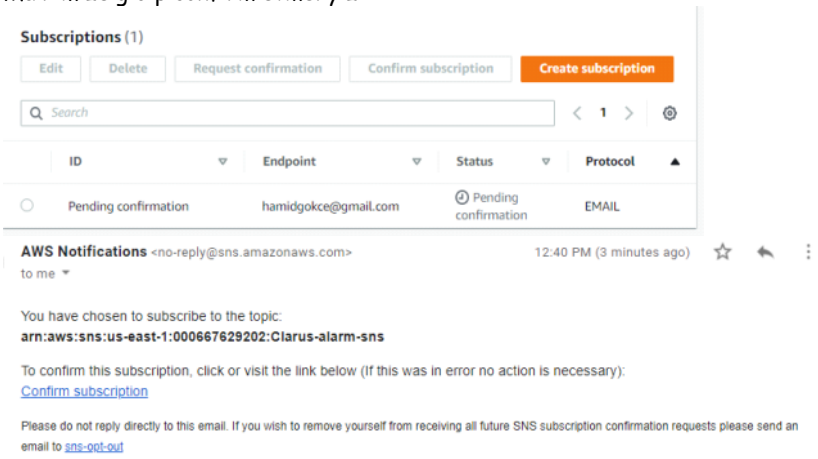


degisiklikler

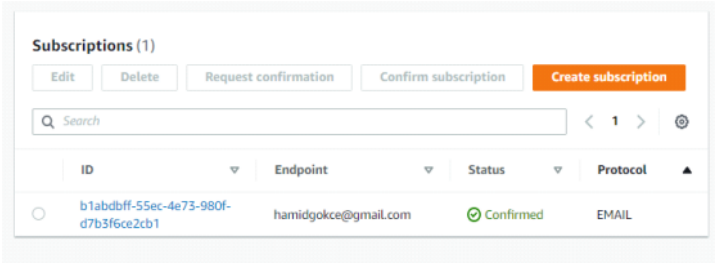


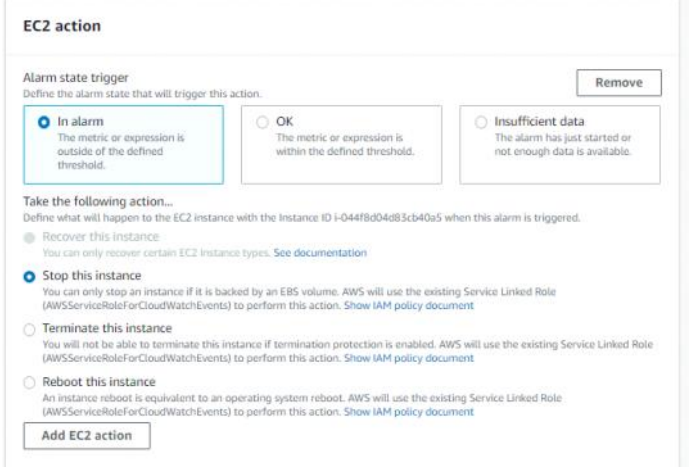


Create topic diyelim
Mail imize gidip confirm etmeliyiz

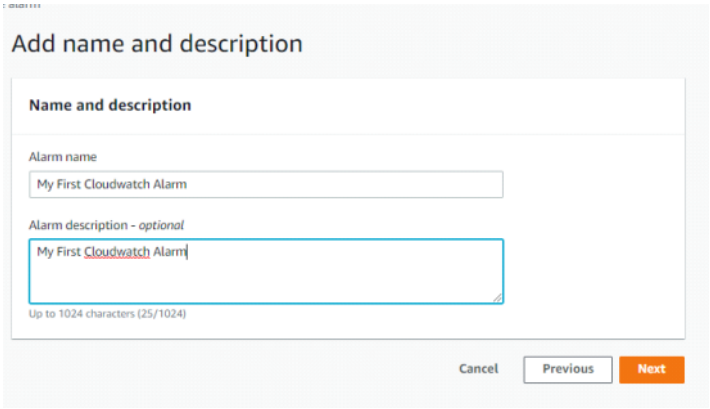


Confirm

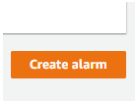


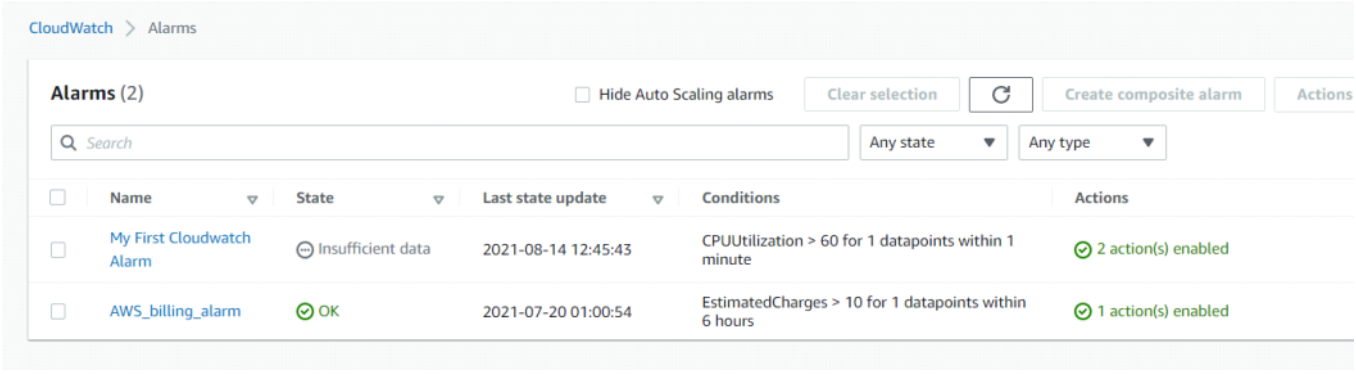


Alarm oldugu zaman instanceyi stop et
Next

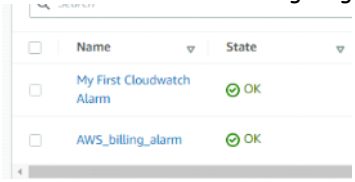


Next

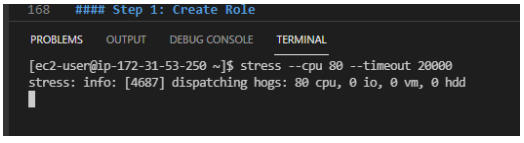


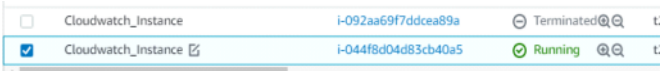


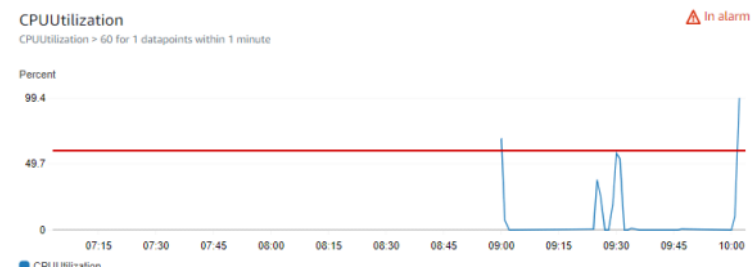
Biraz bekledikten sonra ok geldigini gorebiliriz



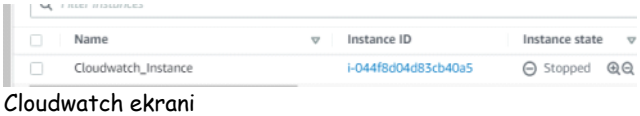
Stress toolun tekrar aktif edelim







Ec2 ekrani



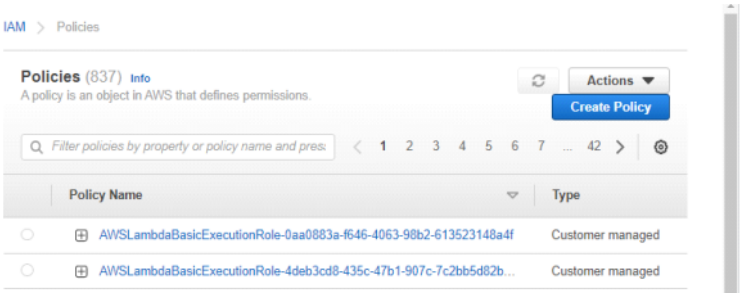
Cloudwatch ekrani

Stresstool u calistirdigimiz icin ec2 stop etti

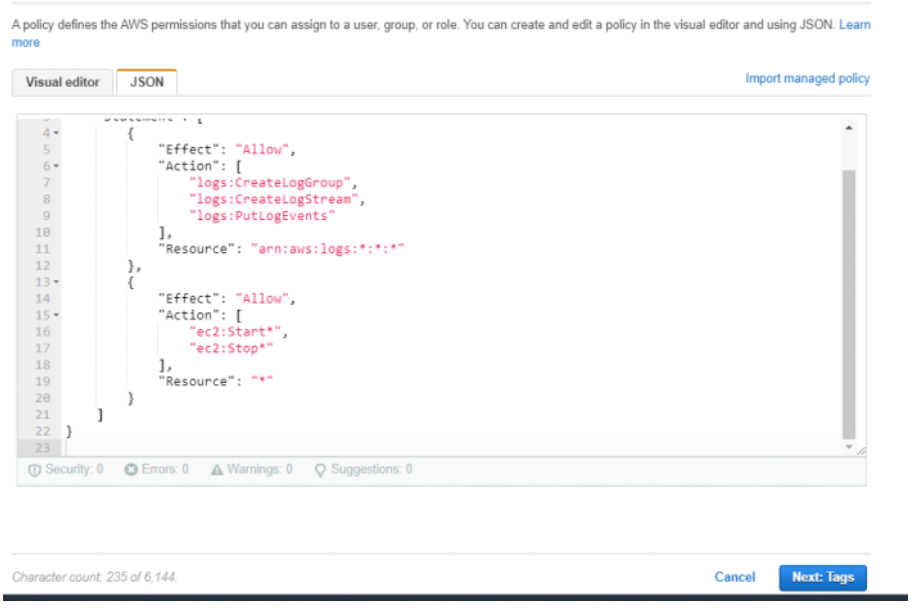
```
### Part 4 - CloudWatch Events with Lambda
```

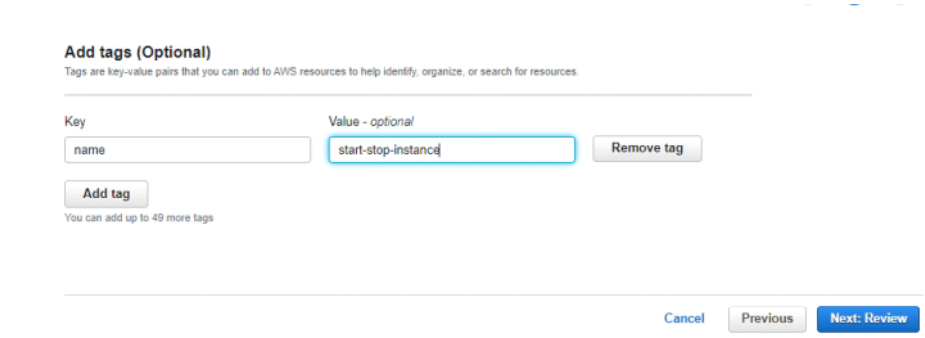
Simdi bir tane event olusturacagiz

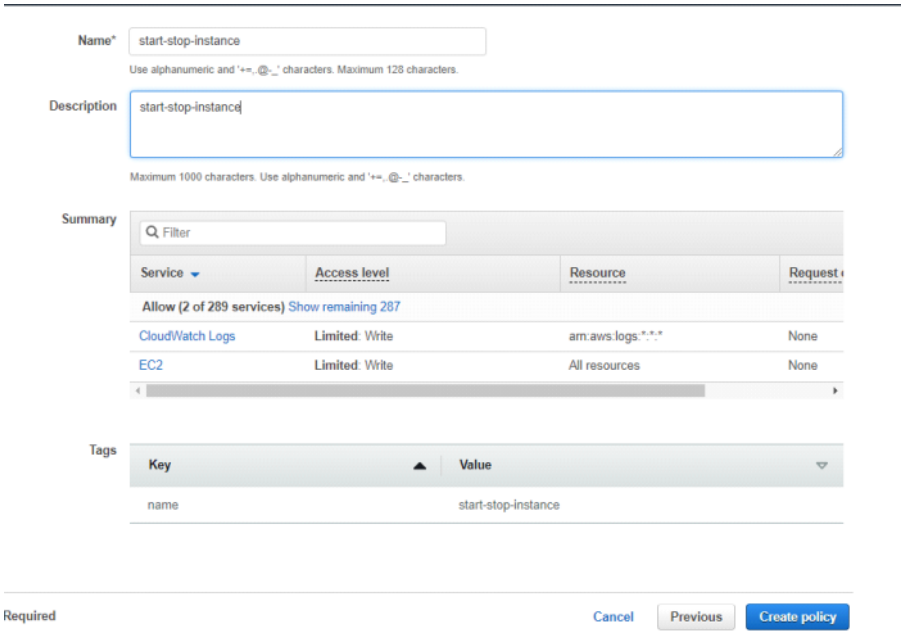
Lambda ve IAM servislerine gidelim
Policy olusturup lambda ya attach edecegiz

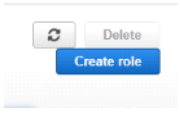


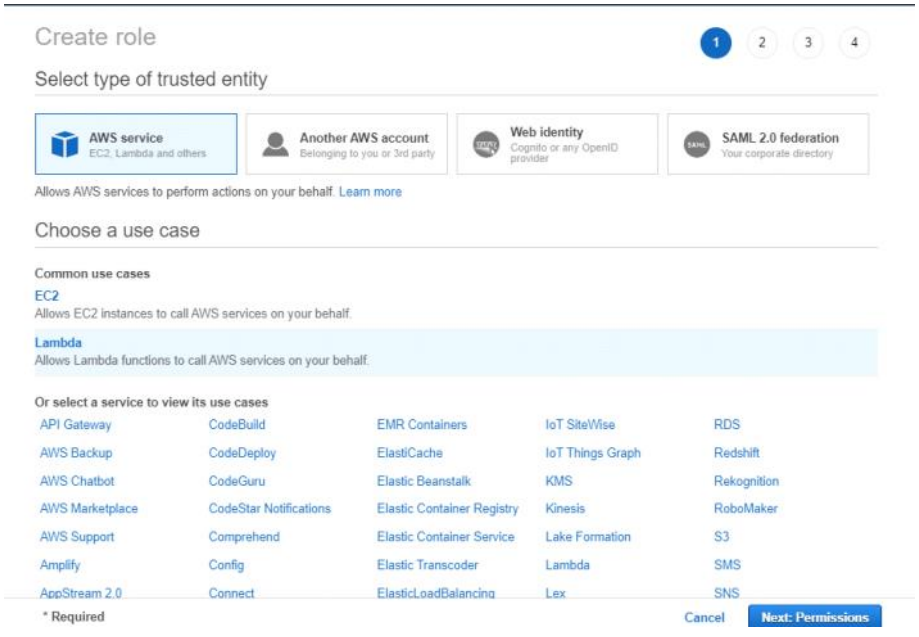

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Start*",
                "ec2:Stop*"
            ],
            "Resource": "*"
        }
    ]
}
```

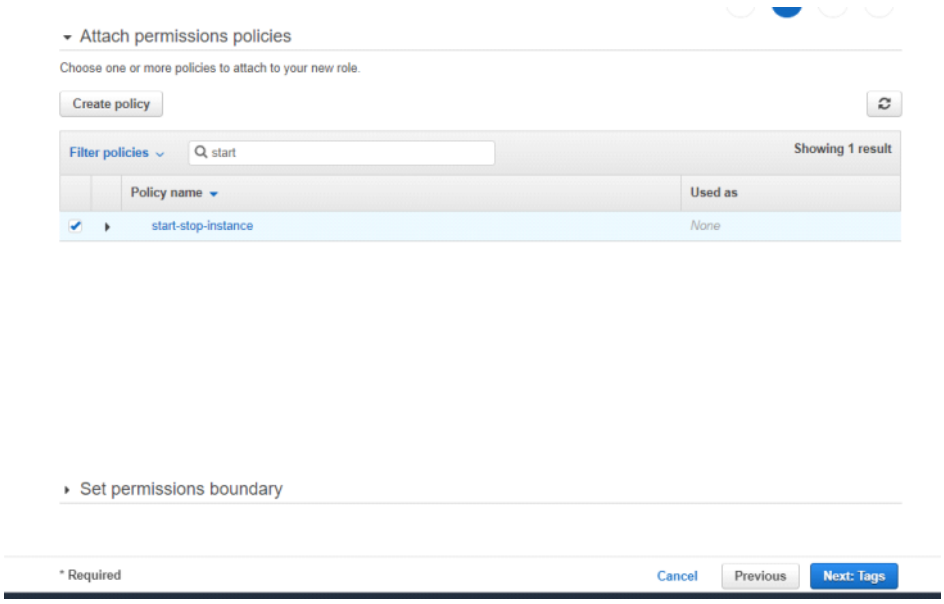


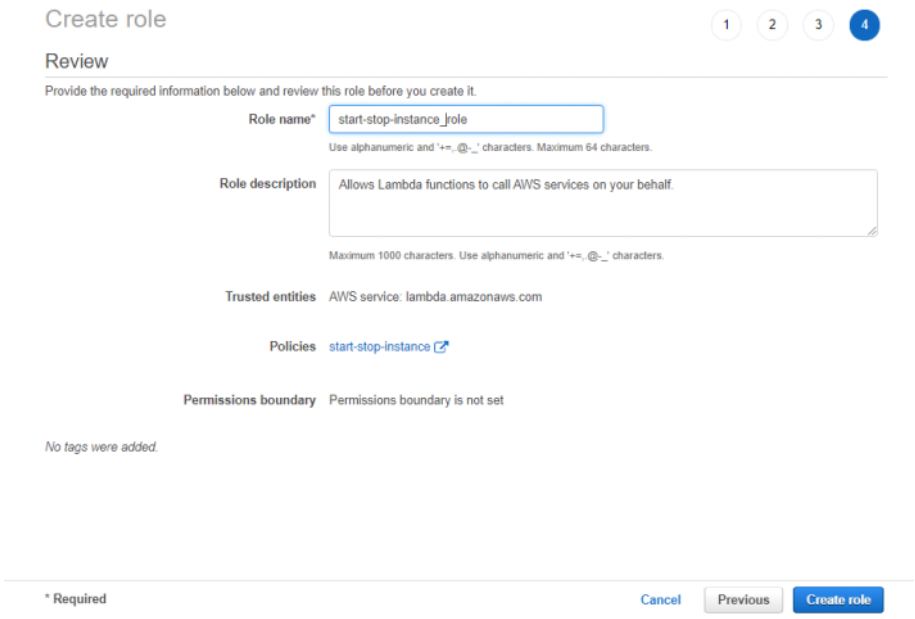

Add tags (Optional)
Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.
Key
Value - optional
name
start-stop-instance
Remove tag
Add tag
You can add up to 49 more tags
Cancel
Previous
Next: Review


Name*
start-stop-instance
Use alphanumeric and '+=,.@-_' characters. Maximum 128 characters.
Description
start-stop-instance
Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.
Summary
Filter
Service
Access level
Resource
Request
Allow (2 of 289 services) Show remaining 287
CloudWatch Logs
Limited: Write
arn:aws:logs:*:*:*
None
EC2
Limited: Write
All resources
None
Tags
Key
Value
name
start-stop-instance
Required
Cancel
Previous
Create policy

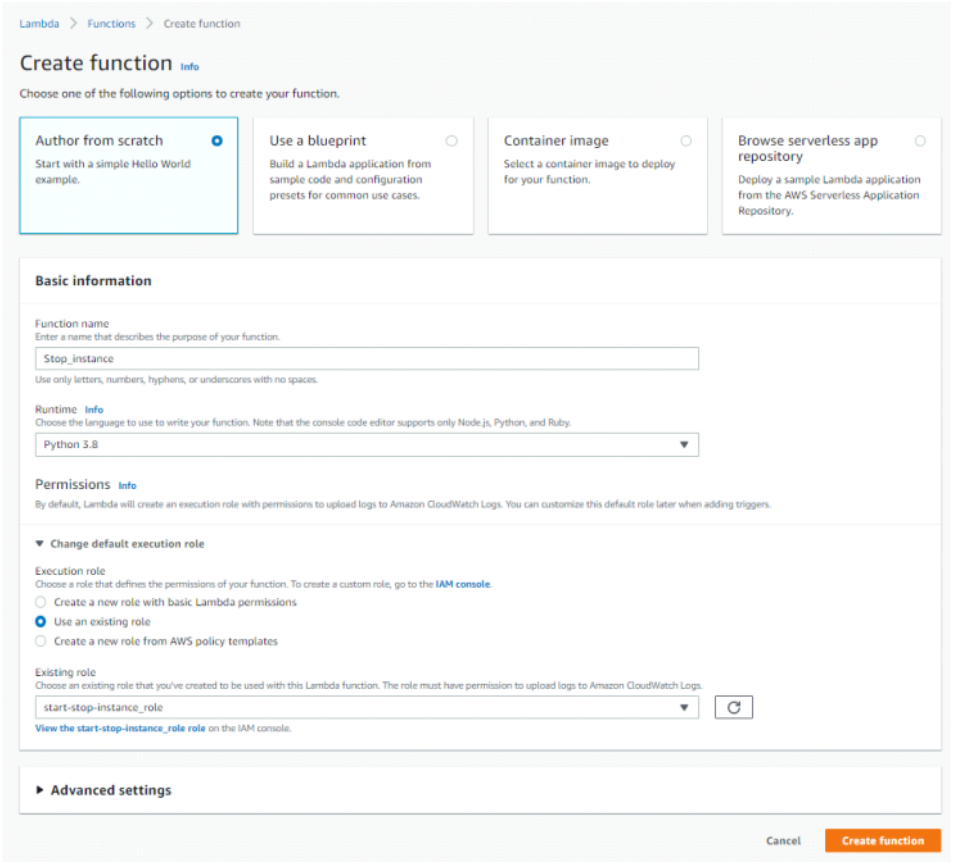

Delete
Create role


Create role
1
2
3
4
Select type of trusted entity
AWS service
EC2, Lambda and others
Another AWS account
Belonging to you or 3rd party
Web identity
Cognito or any OpenID provider
SAML 2.0 federation
Your corporate directory
Allows AWS services to perform actions on your behalf. Learn more
Choose a use case
Common use cases
EC2
Allows EC2 instances to call AWS services on your behalf.
Lambda
Allows Lambda functions to call AWS services on your behalf.
Or select a service to view its use cases
API Gateway
AWS Backup
AWS Chatbot
AWS Marketplace
AWS Support
Amplify
AppStream 2.0
CodeBuild
CodeDeploy
CodeGuru
CodeStar Notifications
Comprehend
Config
Connect
EMR Containers
ElastiCache
Elastic Beanstalk
Elastic Container Registry
Elastic Container Service
Elastic Transcoder
ElasticLoadBalancing
IoT SiteWise
IoT Things Graph
KMS
Kinesis
Lake Formation
Lambda
Lex
RDS
Redshift
Rekognition
RoboMaker
S3
SMS
SNS
* Required
Cancel
Next: Permissions





Lambda ya gecelim

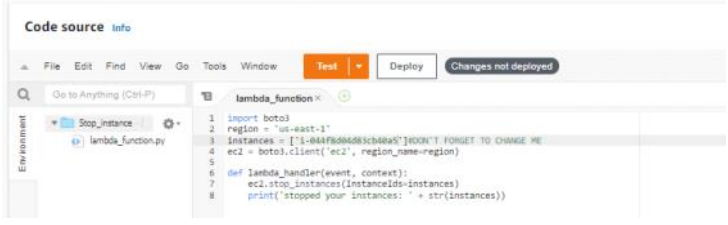


Olusturdugumuz rolu sectik

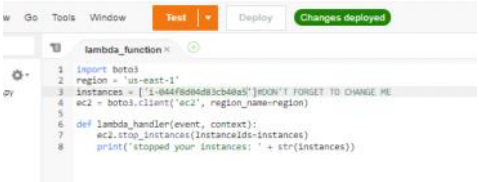
Lambda ekranina python kodunu yapistiralim

```
import boto3
region = 'us-east-1'
instances = ['i-02c107da60f5dad15']#DON'T FORGET TO CHANGE ME
ec2 = boto3.client('ec2', region_name=region)
def lambda_handler(event, context):
    ec2.stop_instances(InstanceIds=instances)
    print('stopped your instances: ' + str(instances))
```

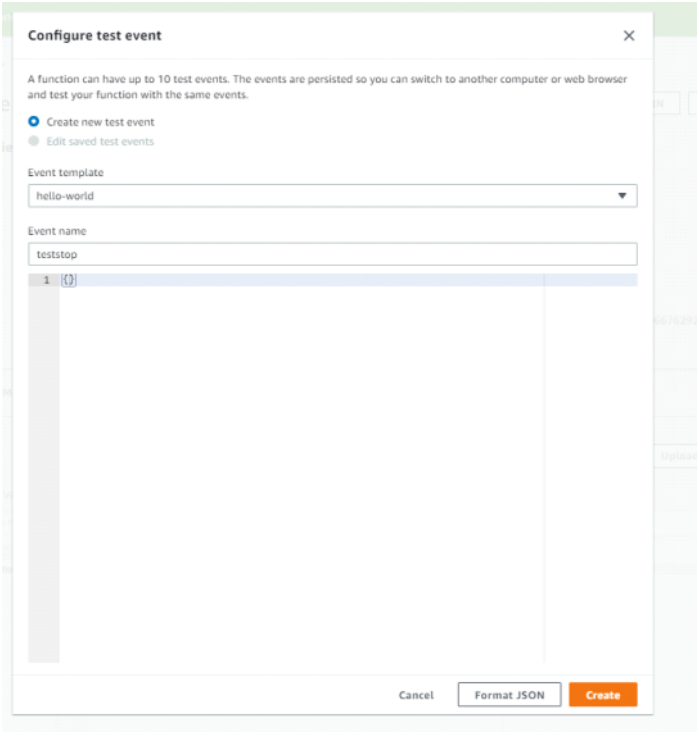
Ec2 id sini unutmayalim



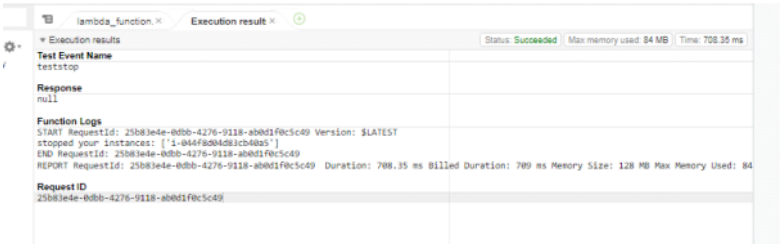
Ve deploy diyelim

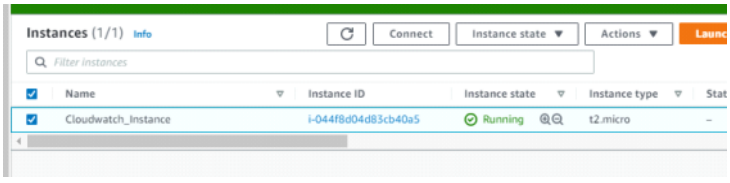


Testi tiklayalim
Ve ec2 yu start edelim test edebilmek icin

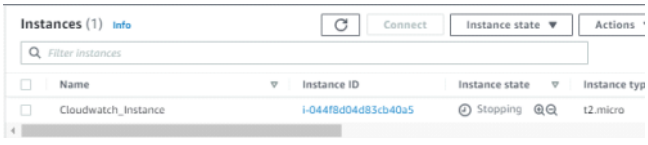


Teste basalim

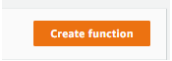


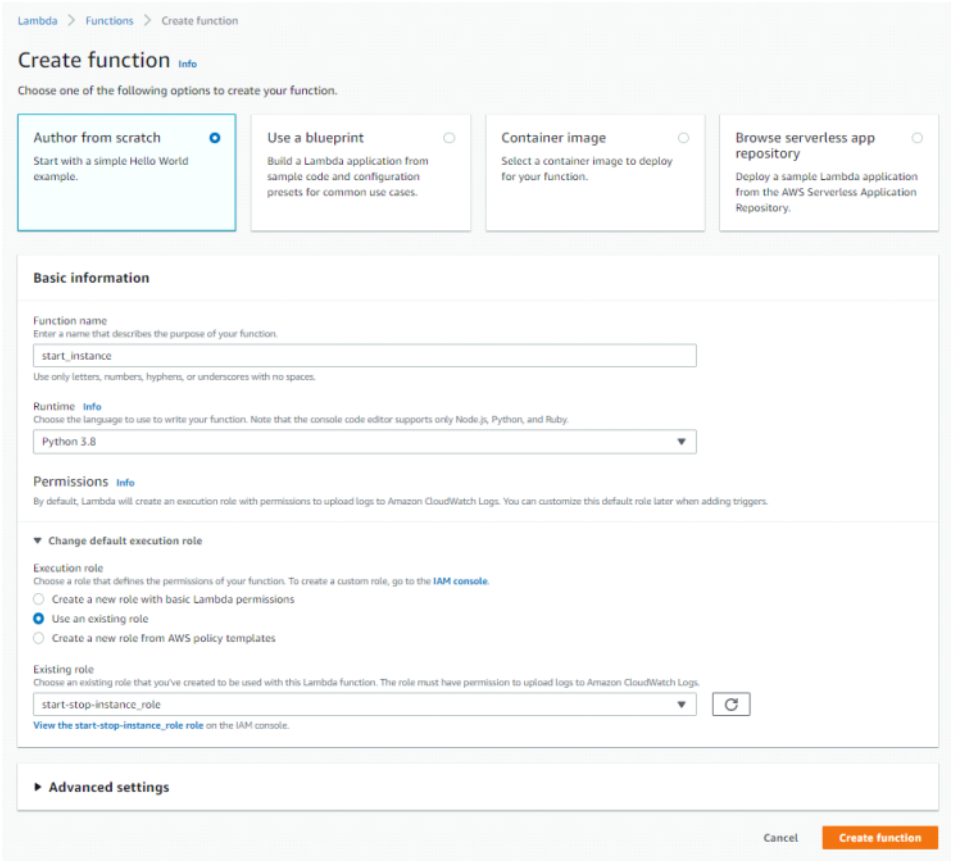


Test otomatik stop etti instanceyi

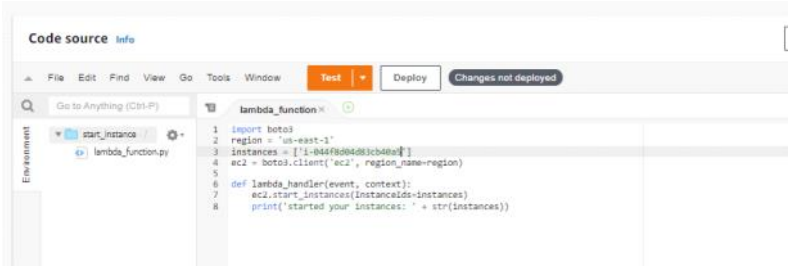


Ayni islemleri start icin de yapalim

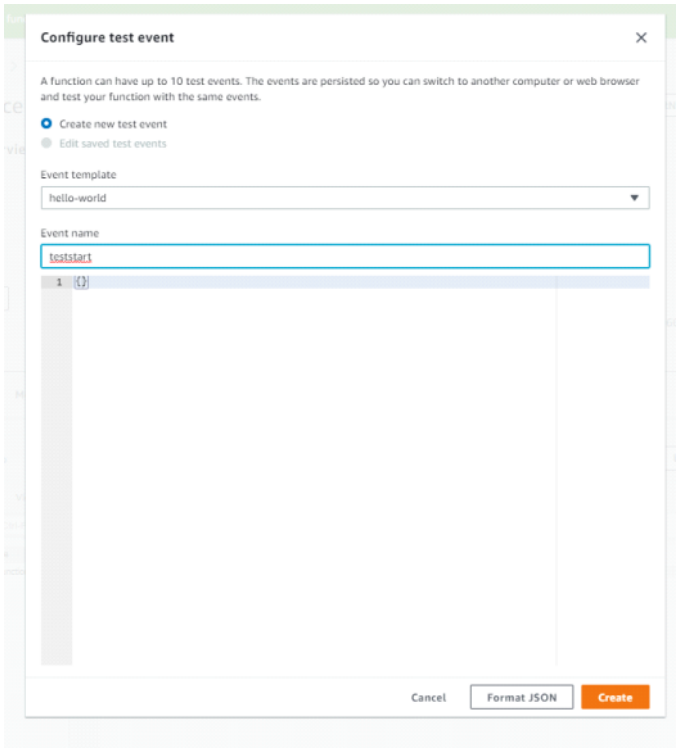


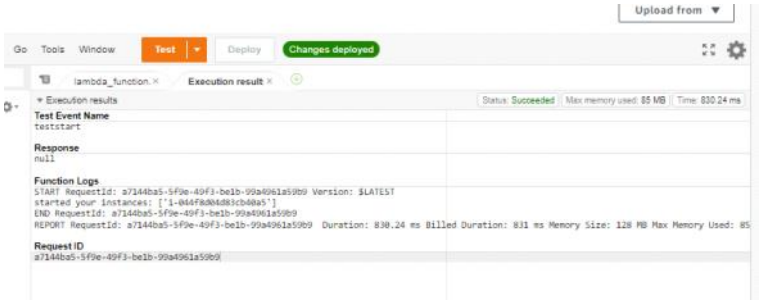


Kodumuzu ekleyelim ve ec2 id yi ekleyelim

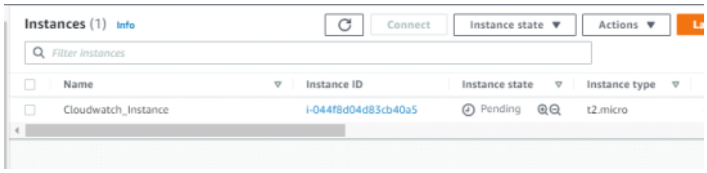


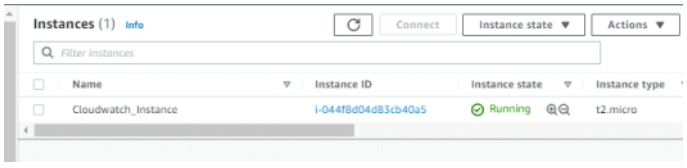
Deploy edelim





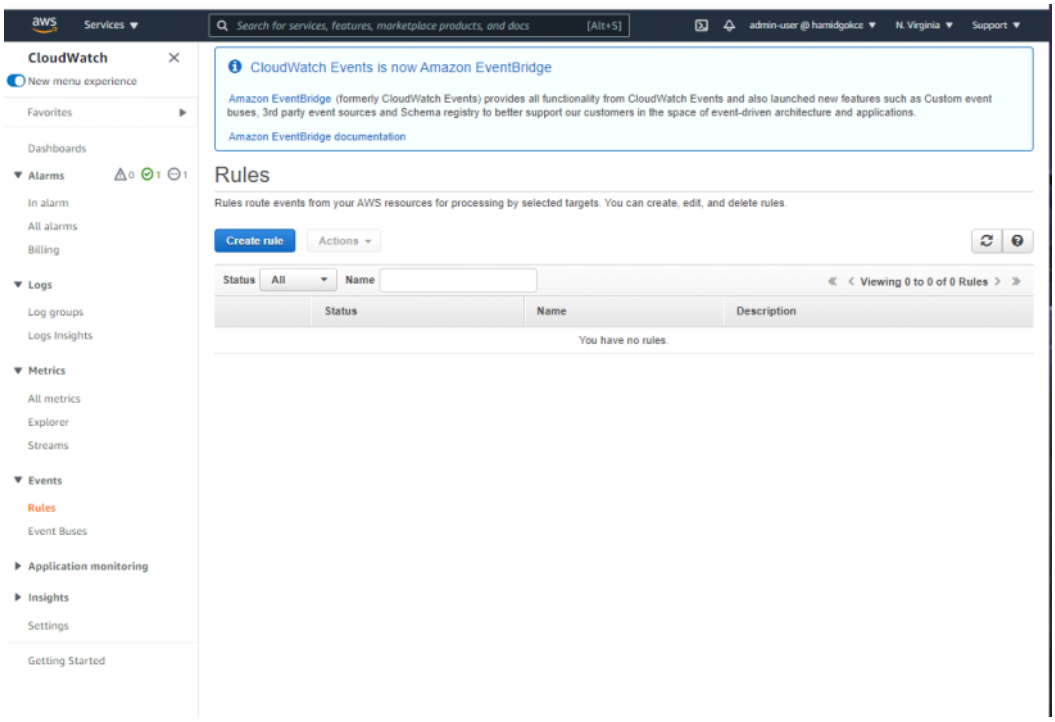
Kapanan instance yinin lambda sayesinde acilabilecegini gorebiliriz





Bu islemlerin herbirini manuel olarak yaptik. Eger biz bunu cloudwatch e tanimlarsak bir zamana endexlemis oluruz ve istedigimiz zamanda acilip kapanmasini saglayabiliriz

Cloudwatch ekranina gelelim



Create rule

```
#### Step 6 Creating Stop-Cloudwatch Event

- Go to the CloudWatch Console and from left hand menu under the Event
sub-section
Rules-------> Create Rule

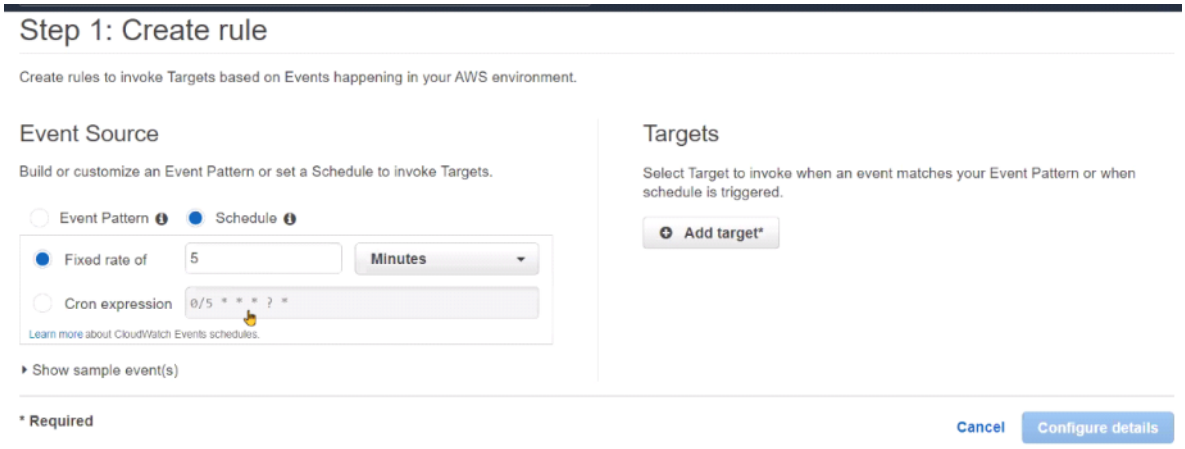
- Select the Event Source parameters:

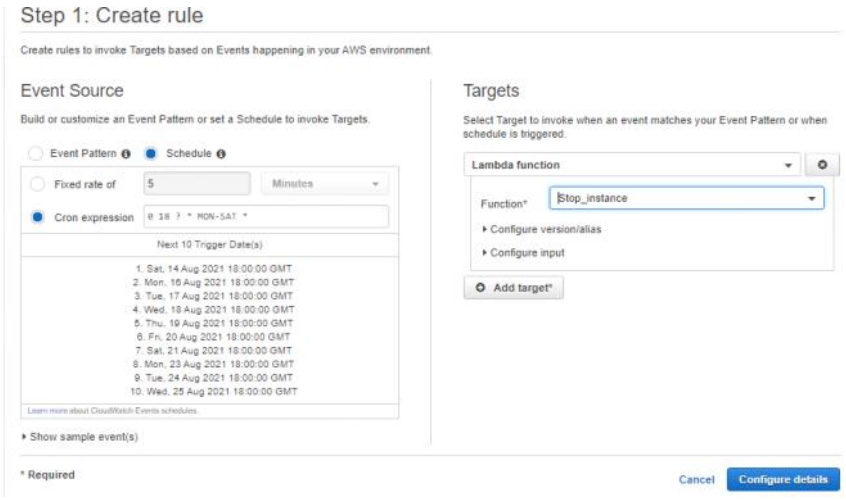
```
Event Source:
  - Select "Schedule"
  - Cron expression: 0 18 ? * MON-FRI *
```
- Explain the cron parameters.

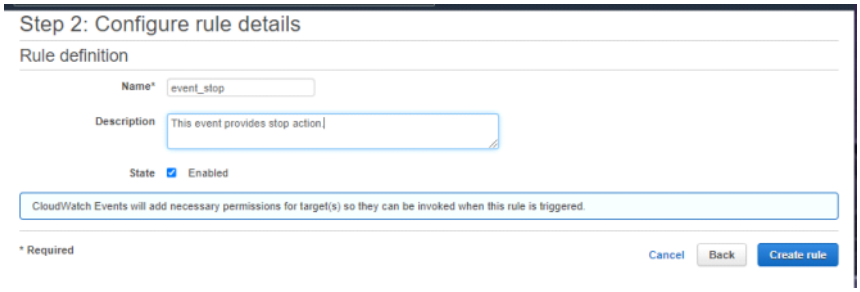
- Select the Target parameters:

```
Targets:
  - Function: Stop_Instance
```
- Click "Configure Details"

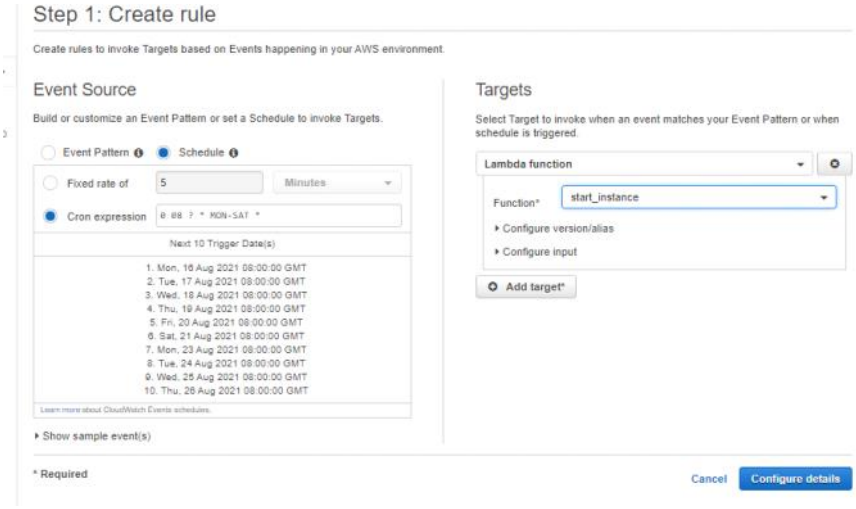
```
- Name: Event_Stop
- Description: This event provides stop action.
- State: Enabled
```
- Click "Create Rule."
```

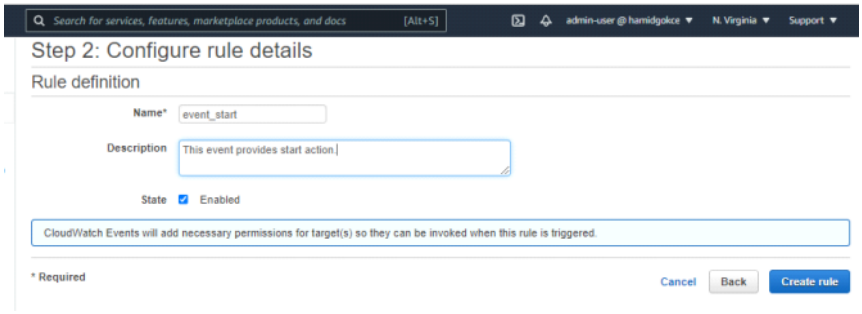




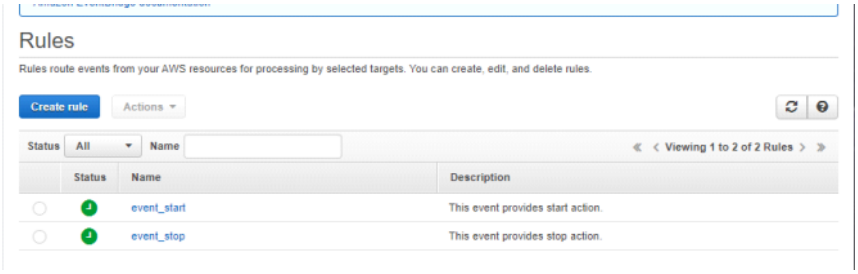


Stop kurali hazir simdi start kurali

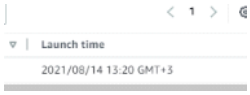




Olusturulan 2 rules

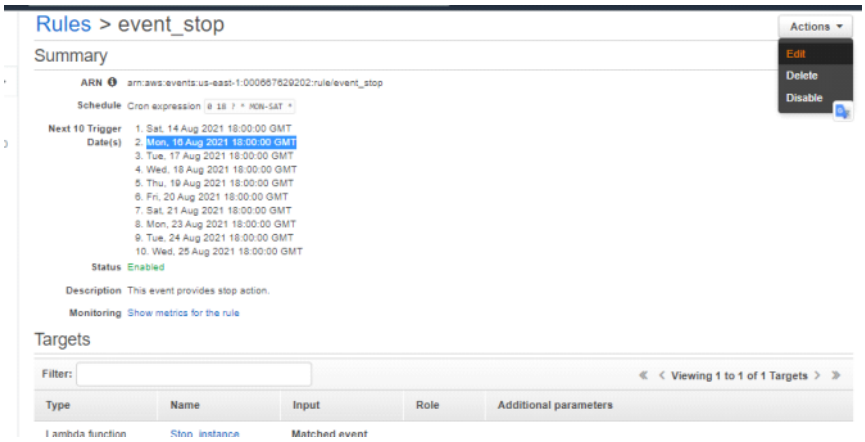


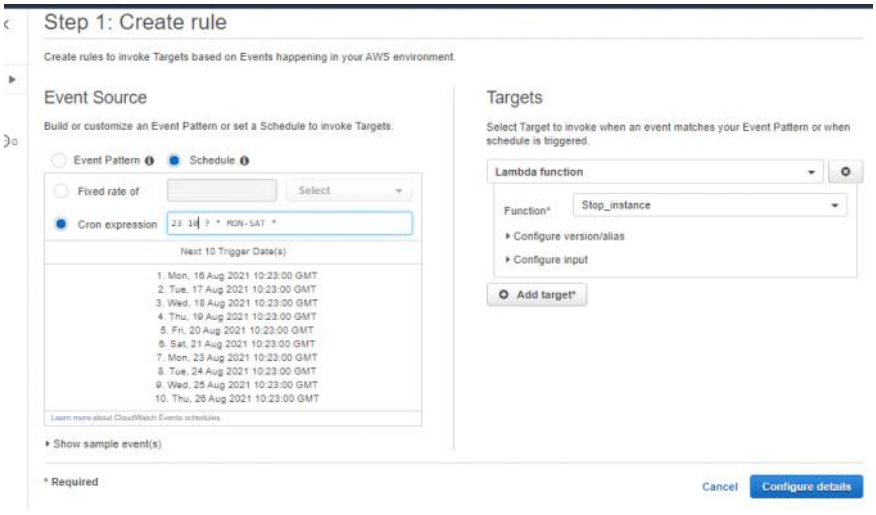
EC2 olusturma saati

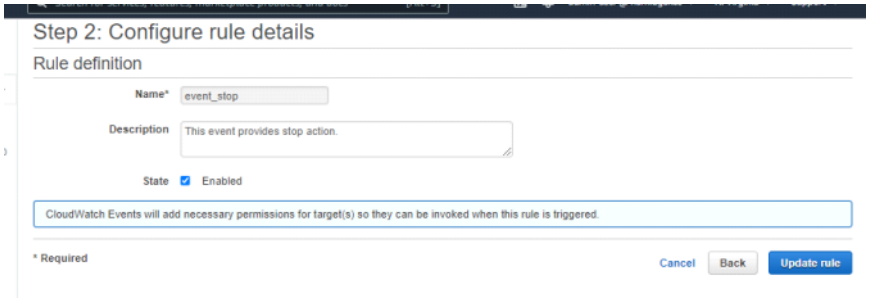


Stop a gelelim

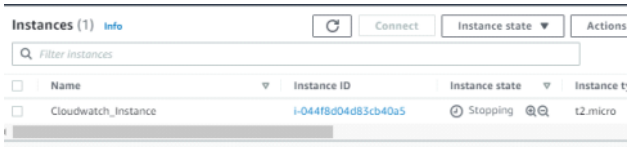
Edit diyelim



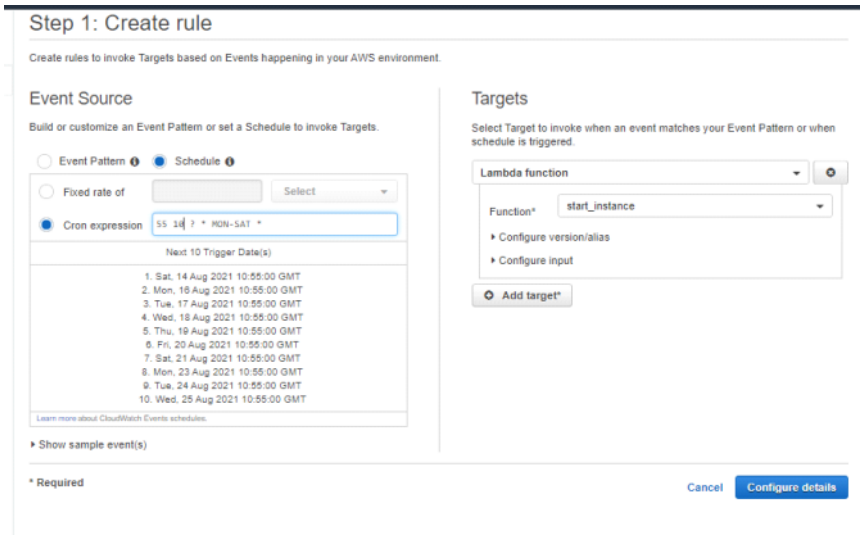




Ec2 yu kontrol ettigimizde lambda sayesinde stop edildigini gorebiliriz

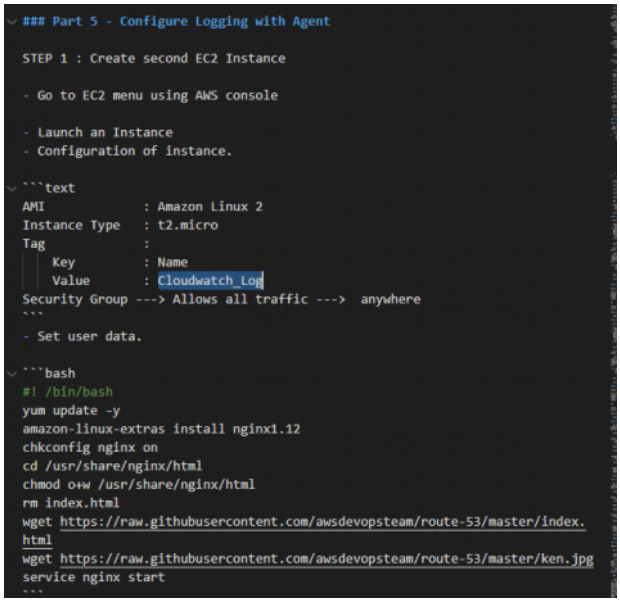


Start rule de degisiklik yapalim gorseldeki gibi / update edelim



https://crontab.cronhub.io/
https://crontab.guru/ ==> tarih formatlarini ayarlayabiliriz

Yeni ec2 olusturalim-



```
### Part 5 - Configure Logging with Agent

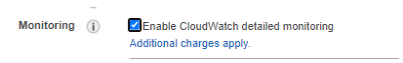
STEP 1 : Create second EC2 Instance

- Go to EC2 menu using AWS console

- Launch an Instance
- Configuration of instance.

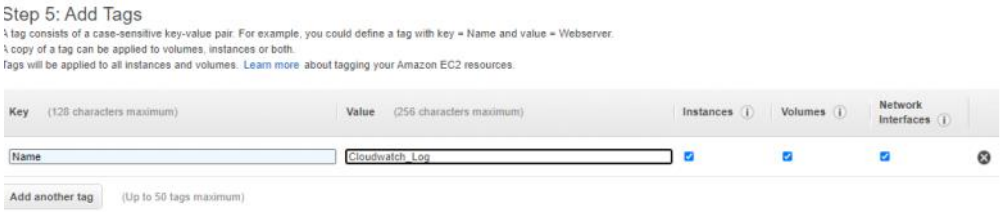
```text
AMI             : Amazon Linux 2
Instance Type   : t2.micro
Tag             :
    Key         : Name
    Value       : Cloudwatch_Log
Security Group ---> Allows all traffic --->  anywhere
```
- Set user data.

```bash
#! /bin/bash
yum update -y
amazon-linux-extras install nginx1.12
chkconfig nginx on
cd /usr/share/nginx/html
chmod o+w /usr/share/nginx/html
rm index.html
wget https://raw.githubusercontent.com/awsdevopsteam/route-53/master/index.
html
wget https://raw.githubusercontent.com/awsdevopsteam/route-53/master/ken.jpg
service nginx start
```
```



Userdata

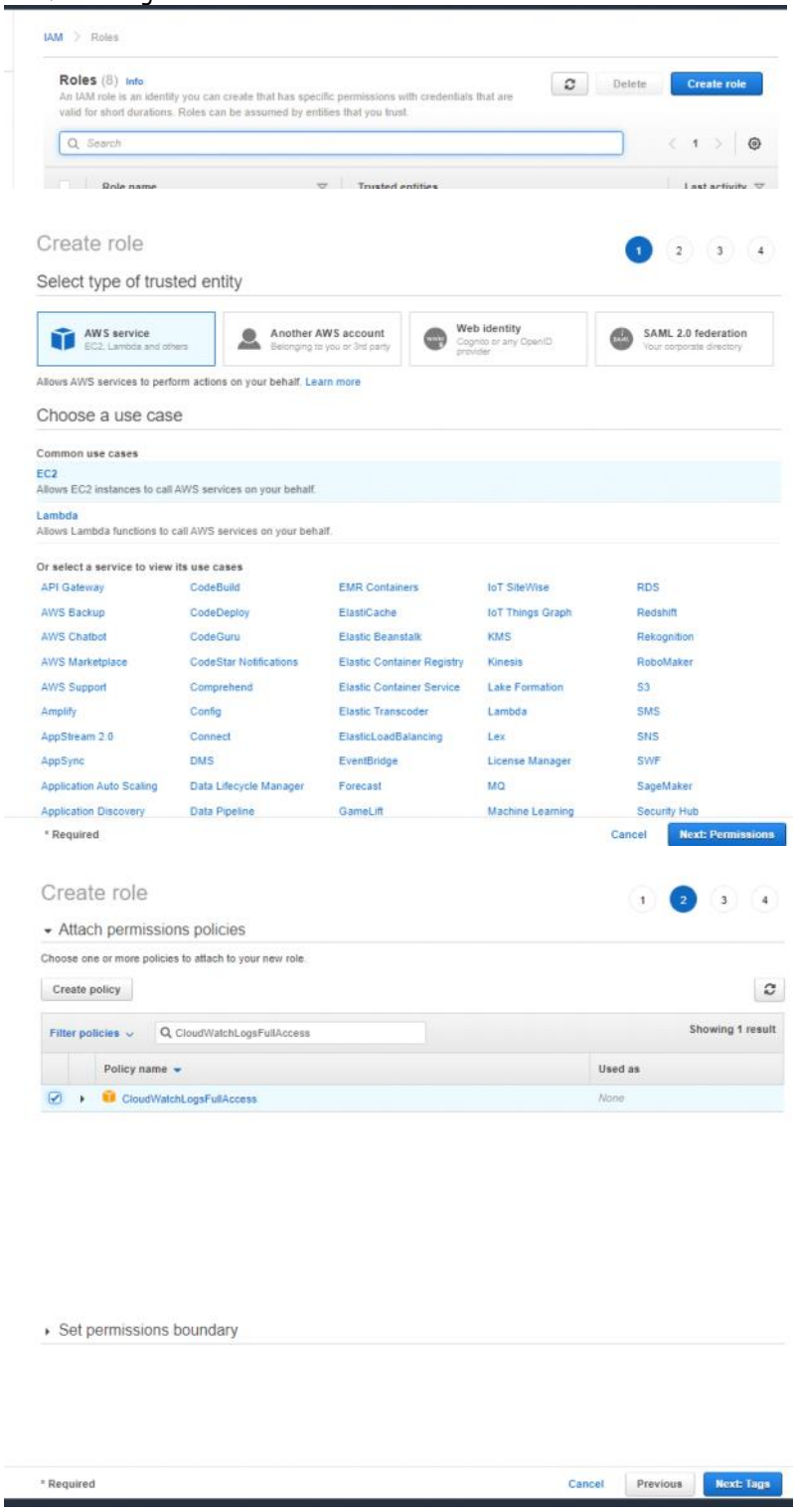
```
#! /bin/bash
yum update -y
amazon-linux-extras install nginx1.12
chkconfig nginx on
cd /usr/share/nginx/html
chmod o+w /usr/share/nginx/html
rm index.html
wget https://raw.githubusercontent.com/awsdevopsteam/route-53/master/index.htm
l
wget https://raw.githubusercontent.com/awsdevopsteam/route-53/master/ken.jpg
service nginx start
```

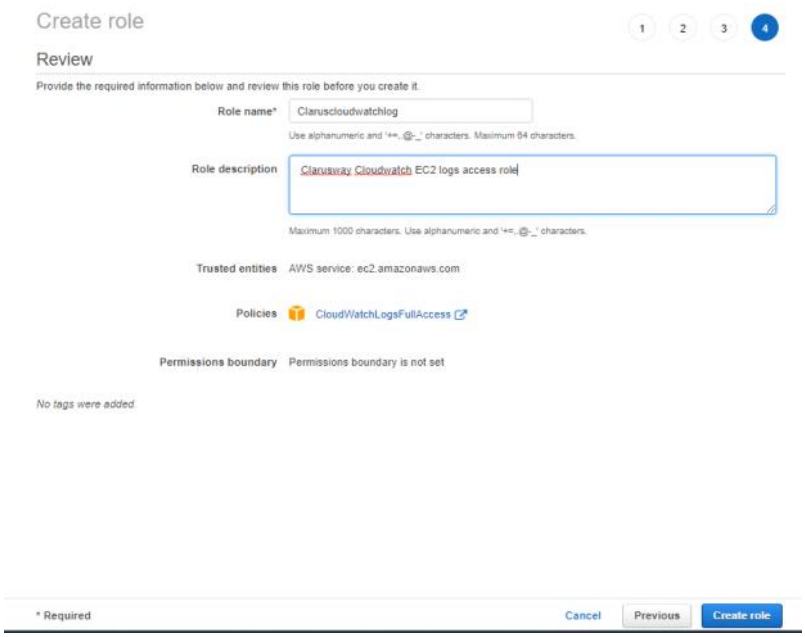


Type	Protocol	Port Range	Source	Descrip
HTTP	TCP	80	0.0.0.0/0	
HTTP	TCP	80	::/0	
SSH	TCP	22	0.0.0.0/0	
SSH	TCP	22	::/0	

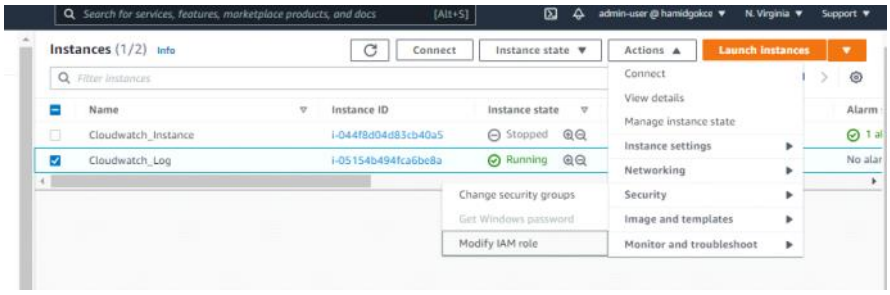
Bu instancenin cloud watch e log kayitlarini tanimlayacagimiz rol ile gonderecegiz

IAM servise gidelim

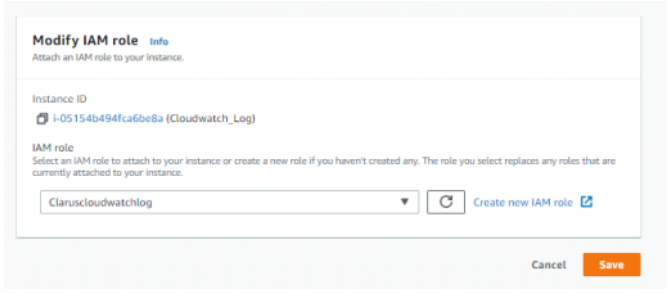




Create role

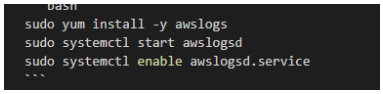


Rolu attach edelim



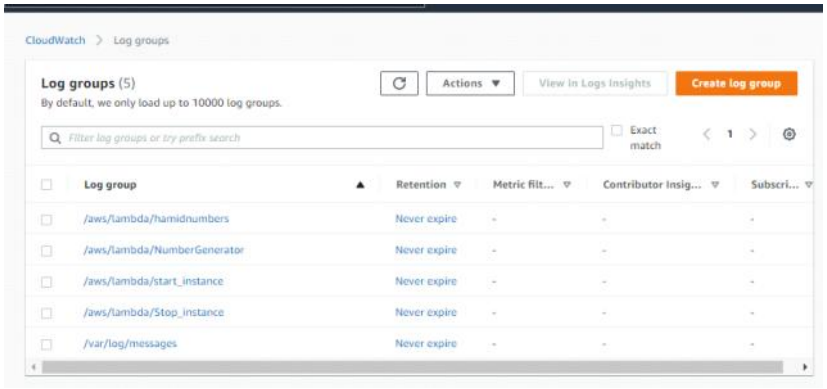
Olusturdugumuz rolu secelim ve save

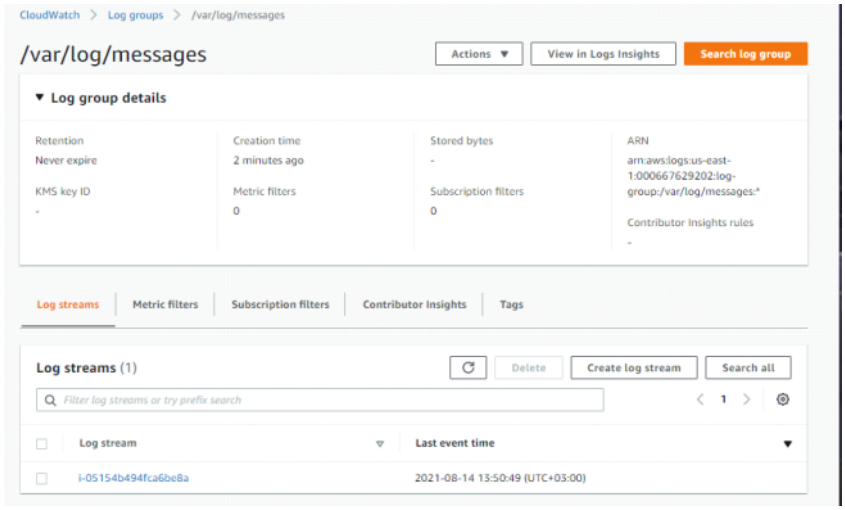
Ec2 ya baglanalim



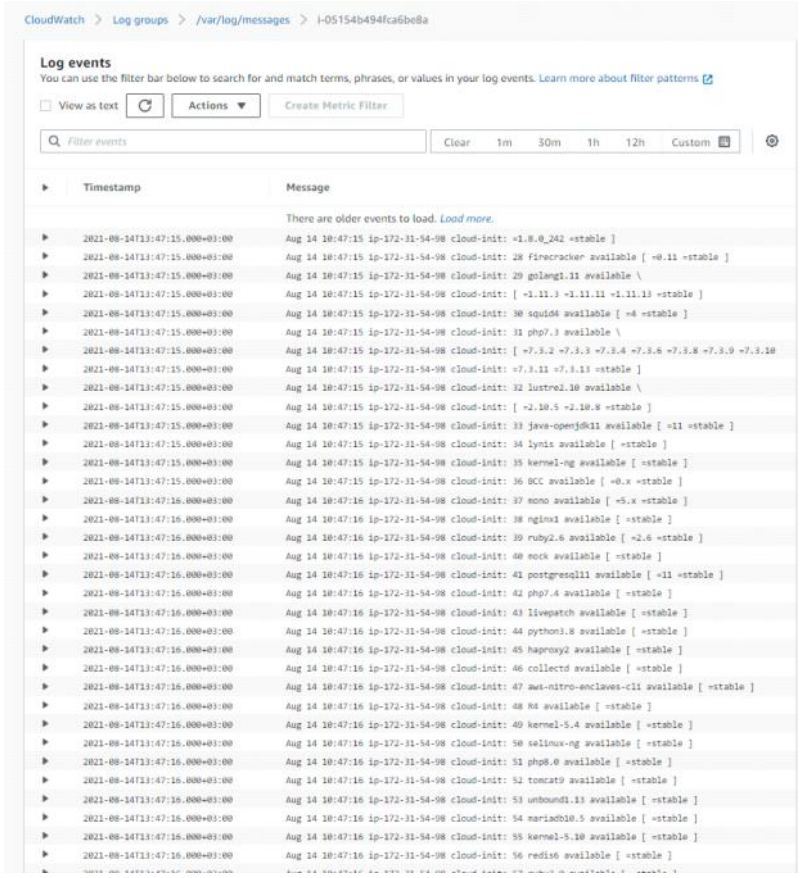
Yukaridaki komutlari sirayla girelim

Cloudwatch ekraninda instance olusturmadan once en alttaki yoktu





Instancenin ilk acilistaki log kayitlari



Su an sistem loglarini gonderiyor ayarlama sonrasinda ngnix log kayitlarini da gonderecegiz

```
[ec2-user@ip-172-31-54-98 ~]$ cd /etc/awslogs/
[ec2-user@ip-172-31-54-98 awslogs]$ ls
awscli.conf  awslogs.conf  config  proxy.conf
[ec2-user@ip-172-31-54-98 awslogs]$ sudo nano awslogs.conf
```

```
su: user nano does not exist
[ec2-user@ip-172-31-54-98 awslogs]$ sudo nano awslogs.conf
```

Dosya icerigi

```
GNU nano 2.9.8                          awslogs.conf

# ------------------------------------------------------------------------------------------$
#  %S          Second as a zero-padded decimal numbers.                                     $
# ------------------------------------------------------------------------------------------$
#  %f          Microsecond as a decimal number, zero-padded on the left.                    $
# ------------------------------------------------------------------------------------------$
#  %z          UTC offset in the form +HHMM or -HHMM (empty string if the the object is naive).$
# ------------------------------------------------------------------------------------------$
#  %j          Day of the year as a zero-padded decimal number.                             $
# ------------------------------------------------------------------------------------------$
#  %U          Week number of the year (Sunday as the first day of the week) as a zero padded$
#              decimal number. All days in a new year preceding the first Sunday are considered
#              to be in week 0.
# ------------------------------------------------------------------------------------------$
#  %W          Week number of the year (Monday as the first day of the week) as a decimal number.$
#              All days in a new year preceding the first Monday are considered to be in week 0.
# ------------------------------------------------------------------------------------------$
#  %c          Locale's appropriate date and time representation.                           $
# ------------------------------------------------------------------------------------------$


[/var/log/messages]
datetime_format = %b %d %H:%M:%S
file = /var/log/messages
buffer_duration = 5000
log_stream_name = {instance_id}
initial_position = start_of_file
log_group_name = /var/log/messages
```

```
[/var/log/nginx/access.log]
datetime_format = %b %d %H:%M:%S
file = /var/log/nginx/access.log
buffer_duration = 5000
log_stream_name = {instance_id}
initial_position = start_of_file
log_group_name = AccessLog

[/var/log/nginx/error.log]
datetime_format = %b %d %H:%M:%S
file = /var/log/nginx/error.log
buffer_duration = 5000
log_stream_name = {instance_id}
initial_position = start_of_file
log_group_name = ErrorLog
```

```
GNU nano 2.9.8

log_stream_name = {instance_id}
initial_position = start_of_file
log_group_name = /var/log/messages

[/var/log/nginx/access.log]
datetime_format = %b %d %H:%M:%S
file = /var/log/nginx/access.log
buffer_duration = 5000
log_stream_name = {instance_id}
initial_position = start_of_file
log_group_name = AccessLog

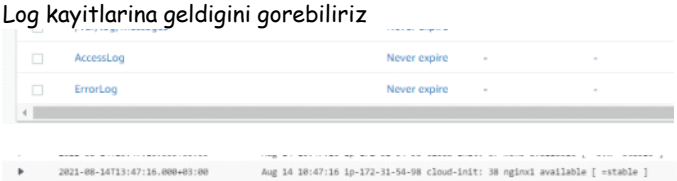
[/var/log/nginx/error.log]
datetime_format = %b %d %H:%M:%S
file = /var/log/nginx/error.log
buffer_duration = 5000
log_stream_name = {instance_id}
initial_position = start_of_file
log_group_name = ErrorLog
```

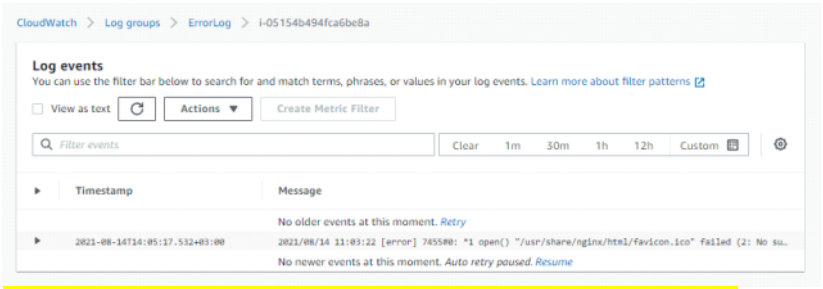
Kaydedip cikalim asagidaki komutlari girelim
Once calisan sistemi durdurmak gerekir stop komutu ile

```
sudo systemctl stop awslogsd
sudo systemctl start awslogsd
```

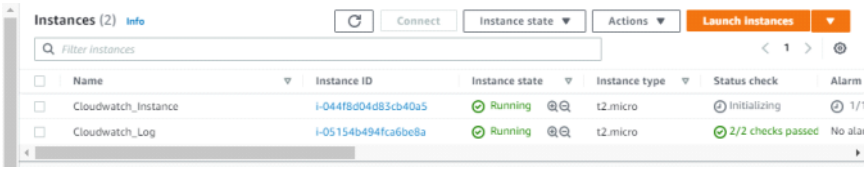
```
[ec2-user@ip-172-31-54-98 awslogs]$ sudo systemctl start awslogsd
[ec2-user@ip-172-31-54-98 awslogs]$ sudo systemctl enable awslogsd.service
```

Log kayitlarina geldigini gorebiliriz





Bu arada kapali instancenin tekrar calistigini gorebiliriz ayarladigimiz RULES sayesinde



Silinecekler

- Lambda function
- Role durabilir
- Ec2 lari silebiliriz
- Rules lari silebiliriz
- Log gruplarini silebiliriz
- Bugun olusturdugumuz alarmi silebiliriz
- Dashboardin icerisini silebiliriz (3 e kadar ucretsiz)