PAGER DUTY, BETTER UP TIME USED FOR MONITORING

Logging, monitoring, alerts and status pages

Logging

Logging refers to the practice of recording events, messages, or data points generated by software applications and systems. These logs can include various types of information, such as:

- **Error Messages**: Details about errors and exceptions that occur.
- **Access Logs**: Records of who accessed what resources and when.
- **Audit Logs**: Records for compliance and security purposes.

Monitoring

Monitoring involves the continuous observation of a system to ensure it is operating correctly and efficiently. It includes tracking various metrics and performance indicators, such as:

- **CPU Usage**: Monitoring how much processing power is being used.
- **Memory Usage**: Tracking the amount of memory being utilized.
- **Disk I/O**: Observing read/write operations on storage devices.
- **Space**: Total amount of space available on the machine

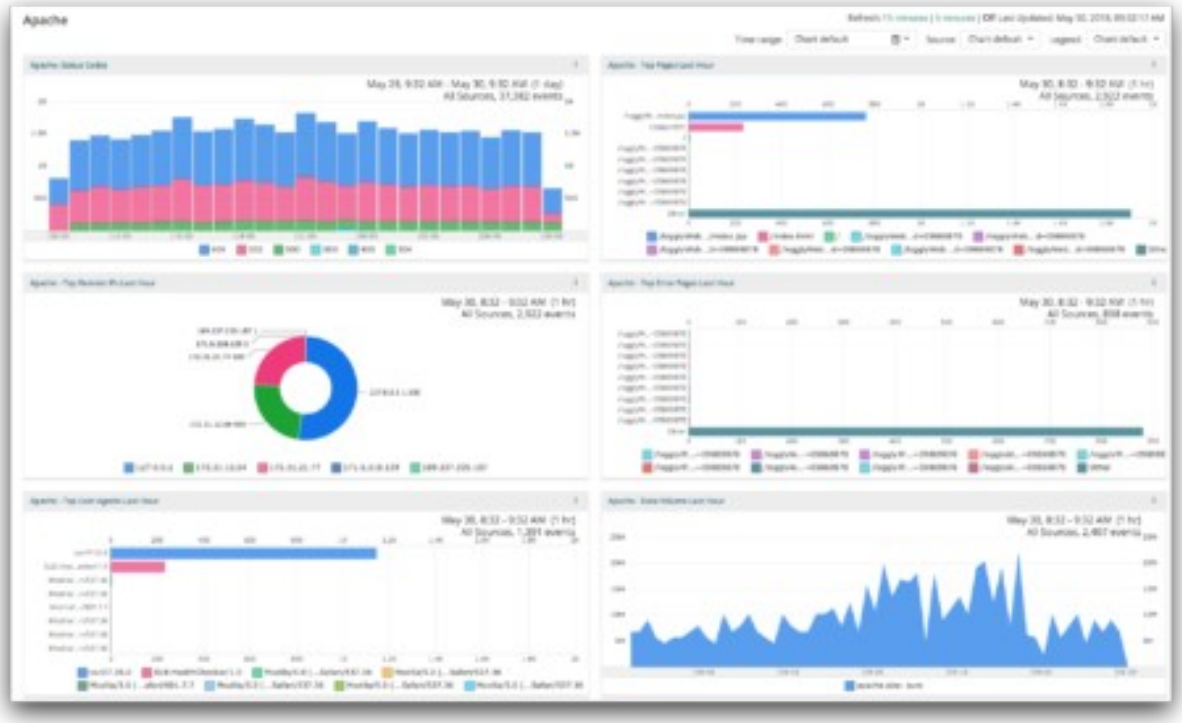
- **Network Traffic:** Measuring data transfer rates and network activity.
- **Application Performance:** Monitoring response times, throughput, and error rates.

Alerts

When there are logging/monitoring systems in place, you can put up alerts to be called/messaged/slacked/paged when

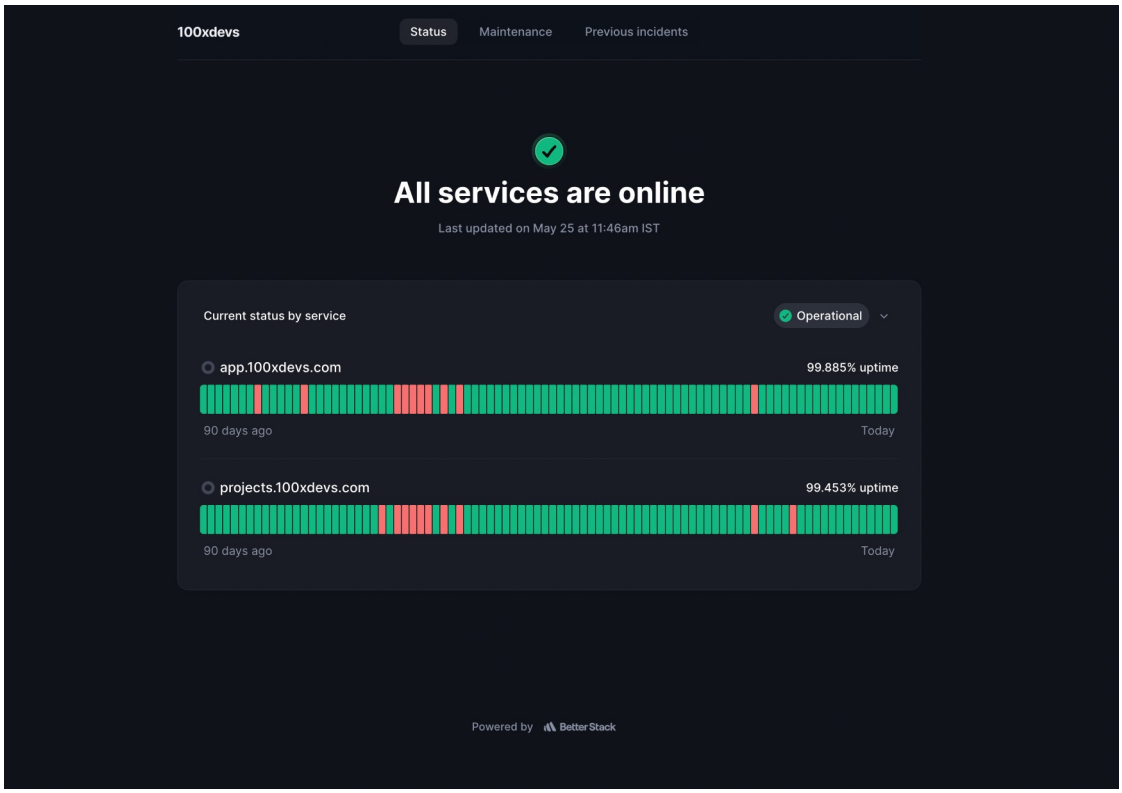
1. A system goes down
2. CPU usage goes above a certain point
3. Error count goes up

Monitoring dashboards

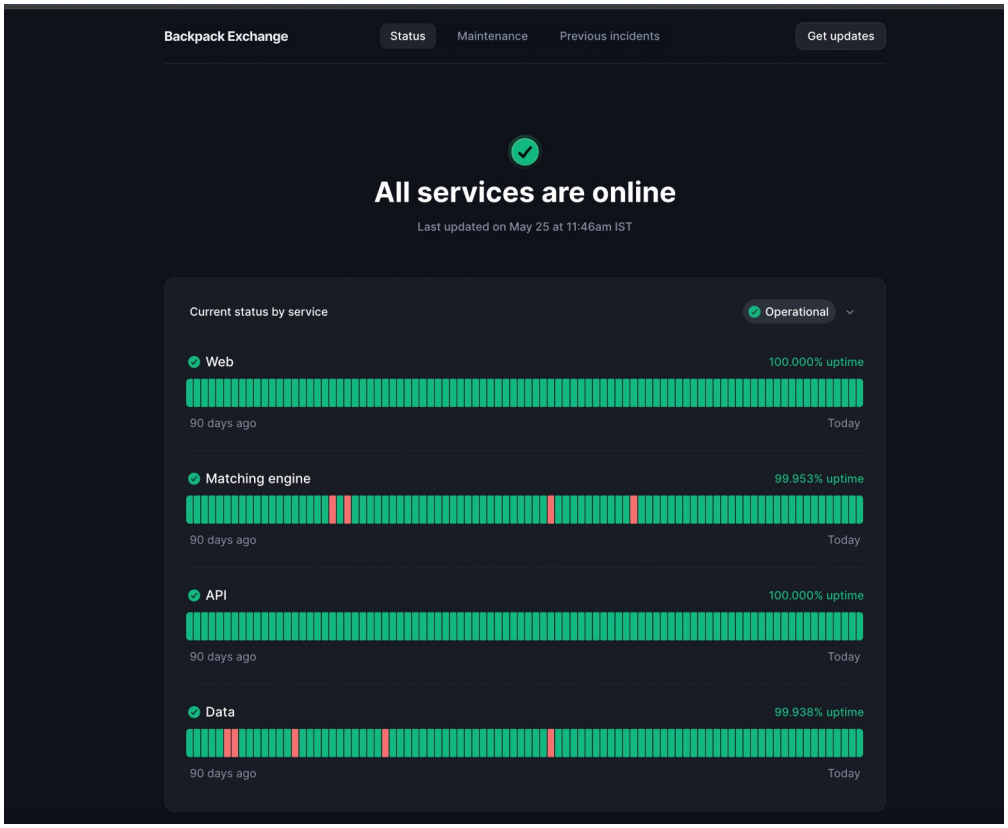


Status pages

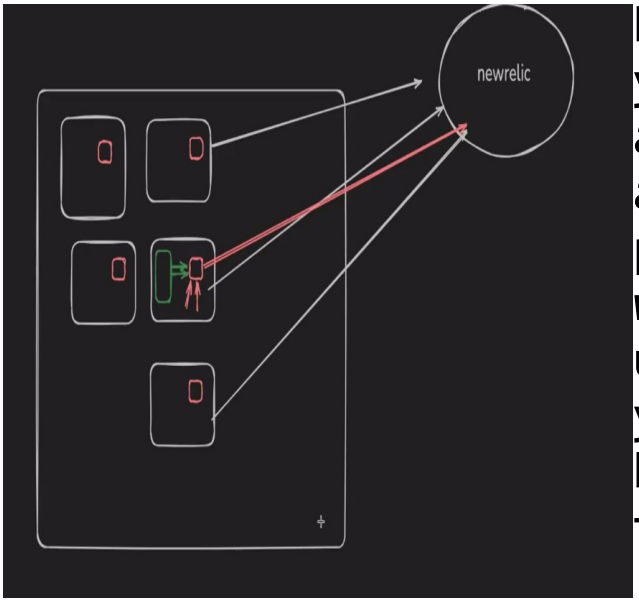
https://status.100xdevs.com/



https://status.backpack.exchange/



sign in to newrilic
select node js
linux



how the monitoring is done like you have to first install the agent of newrelic in the systems and then time to time it will post all the system metrics and will send to website for the usage and to be used and seen
you can also setup this offline by selecting mac in the os in the starting

create a instance and then get the access in the machine
sudo -s for the root access
then install the newrelic agent
taking the command from the newrelic website but there is a error

```
ubuntu@ip-172-31-32-120:~$ sudo -s
root@ip-172-31-32-120:/home/ubuntu# curl -Ls https://download.newrelic.com/install/newrelic-cli/scripts
/install.sh | bash && sudo NEW_RELIC_API_KEY=NRAK-X8XOYTJJU0IJDR8NR5OIORHE2IT NEW_RELIC_ACCOUNT_ID=8062
798 /usr/local/bin/newrelic install
Installing New Relic CLI v0.112.6
Installing to /usr/local/bin

 _   _                 ____      _ _
| \ | | _____      __ |  _ \ ___| (_) ___
|  \| |/ _ \ \ /\ / / | |_) / _ | | |/ __|
| |\  |  __/\ V  V /  |  _|  __| | | (__
|_| \_|\___| \_/\_/   |_| \_\___|_|_|\___|

Welcome to New Relic. Let's set up full stack observability for your environment.
Our Data Privacy Notice: https://newrelic.com/termsandconditions/services-notices

✓Connecting to the New Relic platform
  Connected


Installing New Relic

==> Installing Infrastructure Agent
there is no newrelic infrastructure agent available for the distribution 'resolute'.
  --------------------
  What's next?
  ⊘  Infrastructure Agent  (unsupported)

  The installation is incomplete. Follow the instructions at the URL below to complete the installation
 process.
  →  https://onenr.io/0Bj3rKZkGRX

  --------------------
```

we not able to install noooo
so install it using docker if it does not work
docker does not work on the mac machine bcoz it has a option for the build host which is not present in the mac
choose docker then docker file, then default host
so first install docker in the machine
now follow the website commands
this installs a agent

check if running or not using sudo docker ps
now click next and test the connection
all the monitoring things are visible in the entities part
its like now the agent is installed it can now see everything and access everything
if you create like sudo docker nginx and you created a container it can show that how much usage does that container is doing and how much usage its taking

now to test we are creating a inf node js application
for that installing node js
use option 3 to install the node js
after installing do
npm init -y
vi index.js
we are running

```
while(1){
   Math.random();
}
```

it will just occupy the 1 cpu single threaded
now again ssh into the machine and then run htop to check the usage
this is used like at night you are sleeping and in the morning in the newrelic you can see the system was all night high and you can then check why this happened and then fix that properly

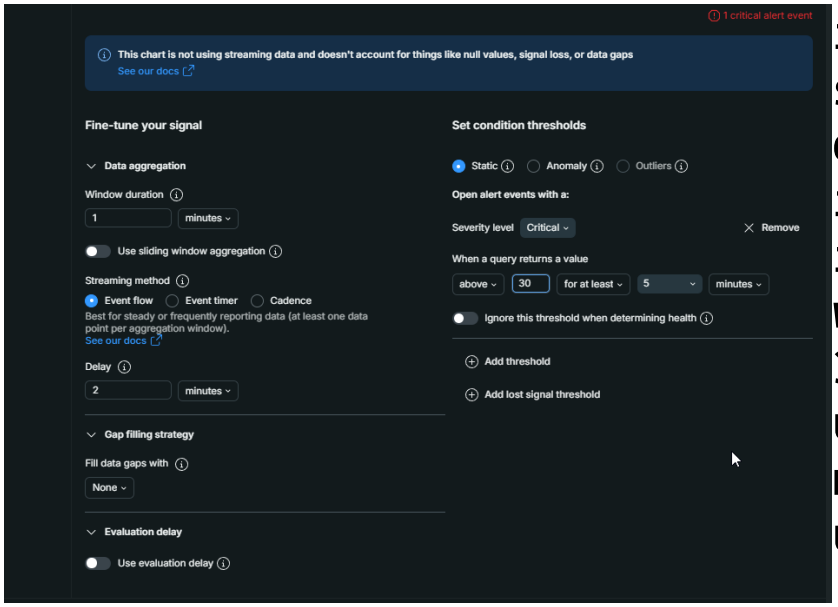
you can also create your own dashboard go to dashboard and then try to create the dashboard of your own or use the template
NRQL == newrelic query language
every graph in the default page is a sql query only you can use that in any page just go to that page and do view query to view that page

the table queries are called time series queries
we need to reload every time we use the page
so we can add our own dashboard items

FACET means device ka basis par lines banao

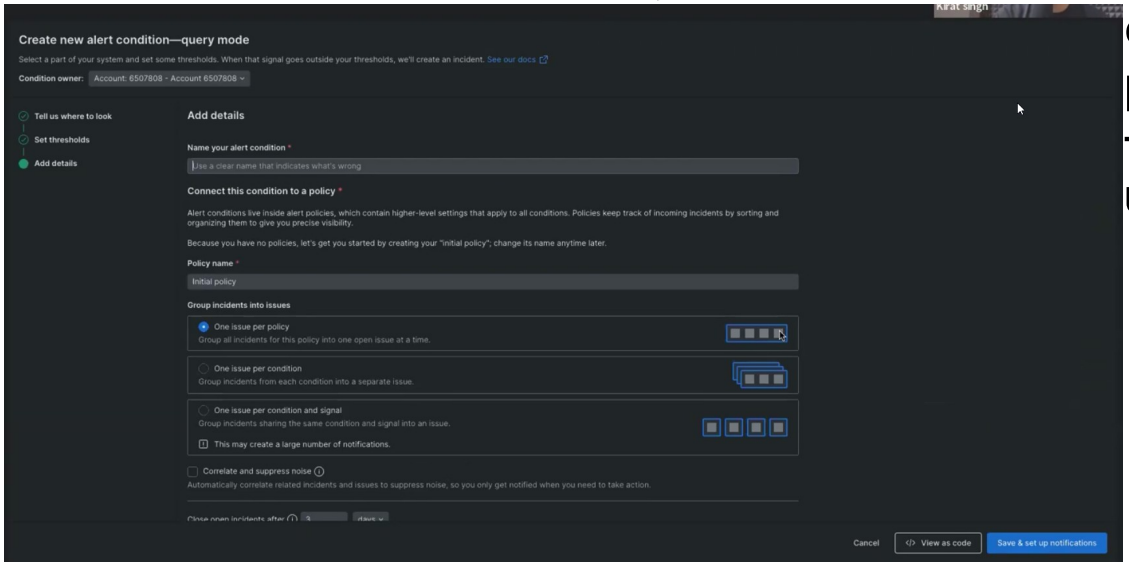
just its like you want your custom dashboard just make your own

FOR ALERT
just on the dashboard click on the three dots and create alert condition



in this you can add the settings, duration, delay and the imp part is the query after which it will return the value which is like if we set 30 so after 30 % of usage for continuously 5 minutes it will notify us

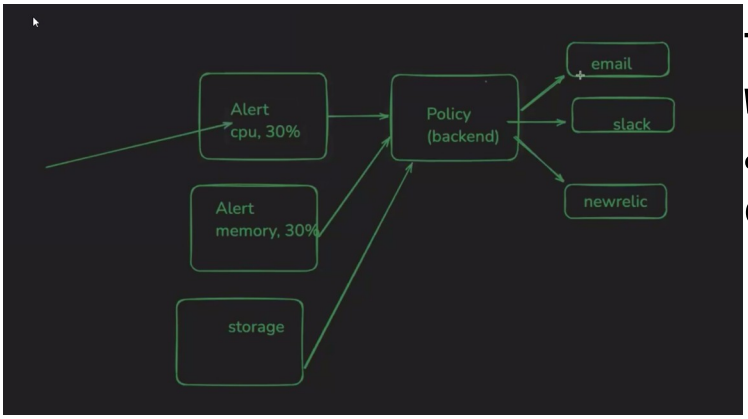
then crate a new policy



create this new policy and then further setup using email

add the new email and the name of the team and then the issue part also and then activate the workflow

then this will be added in the alert section also



this is what will happen when the cpu usage increases and the policy hits then the email, msg is sent

load test is a module which helps us to send multiple request from a system to a specific website
do this install in root
sudo -s
npm install -g loadtest
loadtest -c 10 -rps 20000 https://google.com

we did the above thing to see the double graph data how to make this dashboard read the query buddy

```
SELECT average(transmitBytesPerSecond) AS `Transmit bytes per second`, average(receiveBytesPerSecond) AS `Receive bytes per second` FROM NetworkSample WHERE (entityGuid = 'ODA2Mjc5OHxJTkZSQXxOQXwxMjYzNzk1Mzc4MzgzOTIyMjE') TIMESERIES AUTO SINCE 30 minutes ago UNTIL now
```

its like when you want in one graph just take the query and sepearate them with a commas

WHAT IS FACET?
Imagine you have 2 devices and you want the avg of the cpu % of both of them in the graph we use the facet

if no average then only one line and when we add facet it create different lines for both the entity

without facet it would have shown the avg of all the machines on each interval of time

its like if you had 5 machines and you want to see the cpu usage average of all the machines it will show that

NRQL

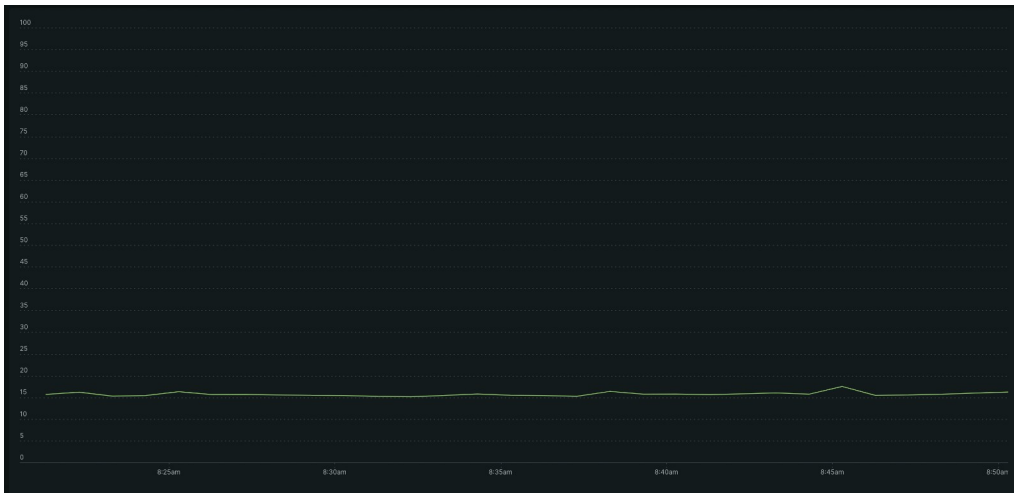
NRQL (New Relic Query Language) is a SQL-like query language used to query data within New Relic, a monitoring and observability platform. NRQL allows users to perform complex data analysis and create custom dashboards and alerts by querying their application and infrastructure performance data stored in New Relic.

It supports most of the graphs but some of them it also not supports then you have to read the stackoverflow and all to get the code for that

CPU Usage

- Fetch all CPU usage

```
SELECT average(cpuPercent) AS `CPU used %` FROM SystemSample WHERE
(entityGuid = 'xyz') TIMESERIES AUTO
```

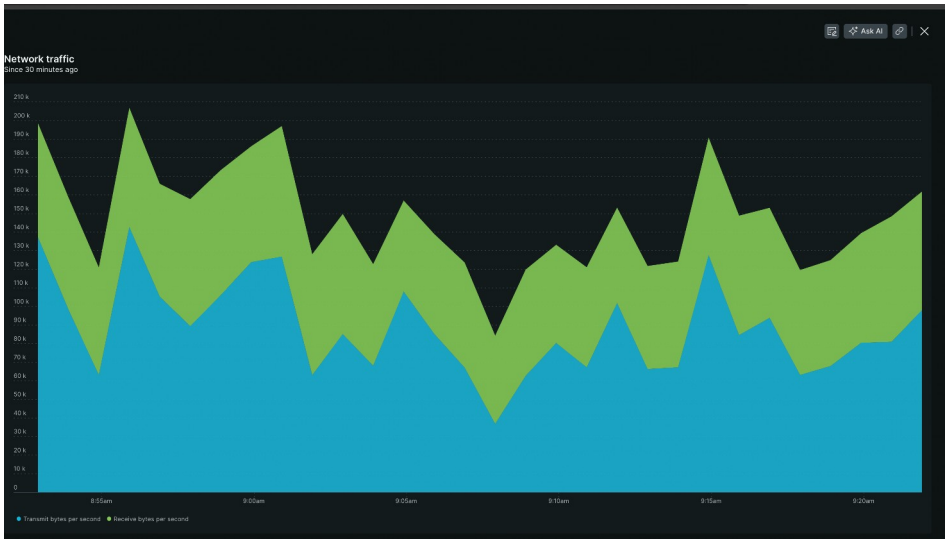


- Filter by High CPU Usage

```
SELECT average(cpuPercent) AS `CPU used %`
FROM SystemSample
WHERE (entityGuid = 'NDQ2MDY2NXxJTkZSQXxOQXwzOTczMjM4ODc0NzI4NDk0NzYz') AND
cpuPercent > 2
TIMESERIES AUTO
```

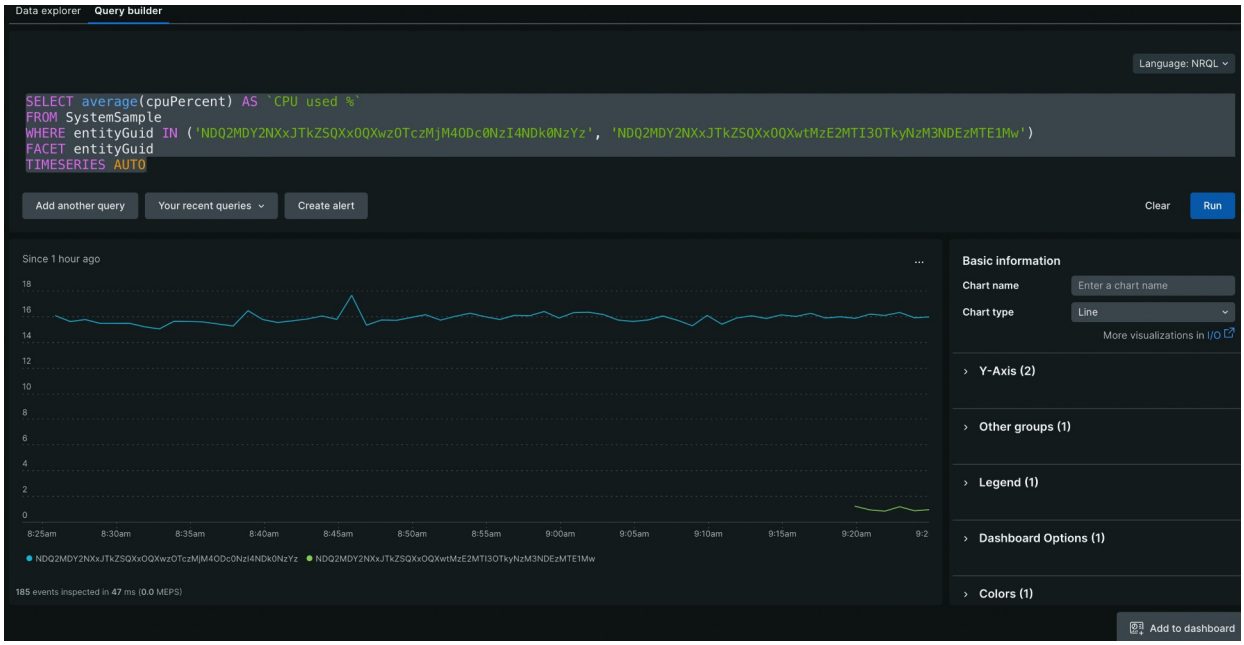
- Multiple graphs in the same timeline

```
SELECT average(transmitBytesPerSecond) AS `Transmit bytes per second`,
average(receiveBytesPerSecond) AS `Receive bytes per second` FROM
NetworkSample WHERE (entityGuid =
'NDQ2MDY2NXxJTkZSQXxOQXwtMzE2MTI3OTkyNzM3NDEzMTE1Mw') TIMESERIES AUTO
```



- Facets

```
SELECT average(cpuPercent) AS `CPU used %`
FROM SystemSample
WHERE entityGuid IN ('xyz', 'abc')
FACET entityGuid
TIMESERIES AUTO
```



now we just not want to track the host machines but we want to track all the node js , rust processes also so to do that we use this
APM(application perf monitoring)AND SERVICES
setup it using the node js first then on the host machine option

now just create a folder in the machine
do npm init -y
then do npm install express
vi index.js
then add this code

```
require("newrelic");
const express = require("express");
const app = express();

app.get("/", (req, res) ⇒ {
    console.log("route hit");
    res.json({message: "hi there"})
})

app.listen(3000, () ⇒ {
    console.log("listening on port 3000");
});
```

for this code the newrelic import is required and then use the command they gave to run the node process not the node index.js to do this do the below thing

use shift v to select the whole line and then press any alphabet
now do vi package.json
in that delete the script and then use this script

```
“start”:”NEW_RELIC_APP_NAME=simple-node-app NEW_RELIC_LICENSE_KEY=1cc012c4df04af7837fe5c92191674c60febNRAL node -r newrelic index.js”
```

npm install newrelic
then do npm run start
TO CHECK IF WORKING JUST DO THE IP:3000

ALSO OPEN THE 3000 PORT IF IT DOES NOT RUNS ON THE PORT

NOW IF ITS RUNNING GO BACK TO THE ALERT PART IF IT REFRESHES ADD THE CODE AND THEN KEEP THE SAME NAME WHICH WAS

simple-node-app
just after that test connection and then see your data

now when you hit the website many times you can see the graph

now with this you can see all the usage of that code and how many times it was hitted blah blah blah

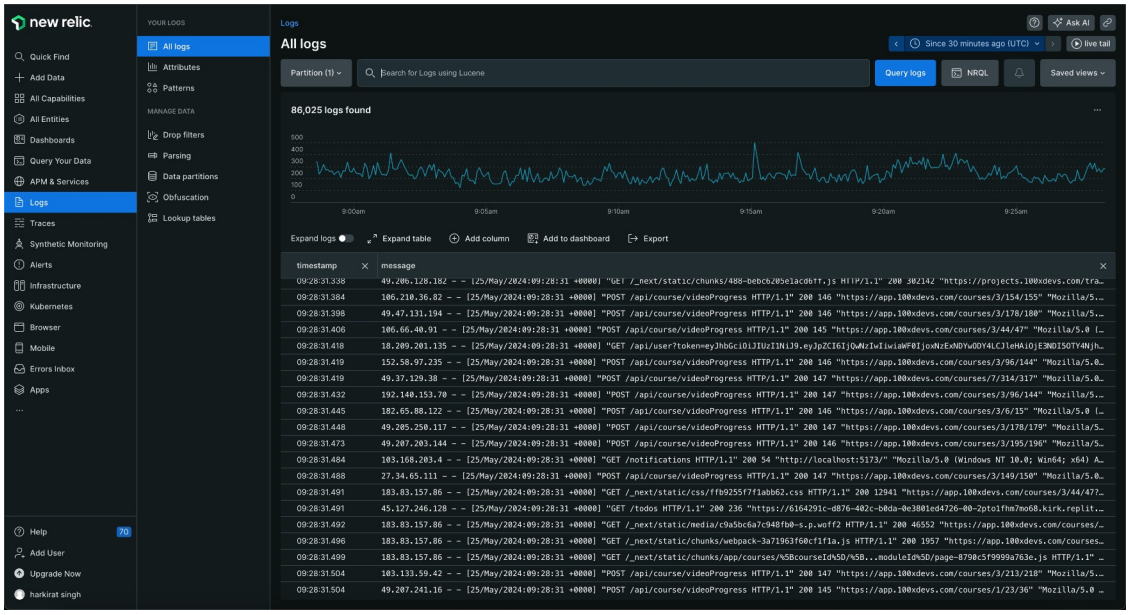
APM

Newrelic also lets you see all the logs and stats from your process, not just the host machine

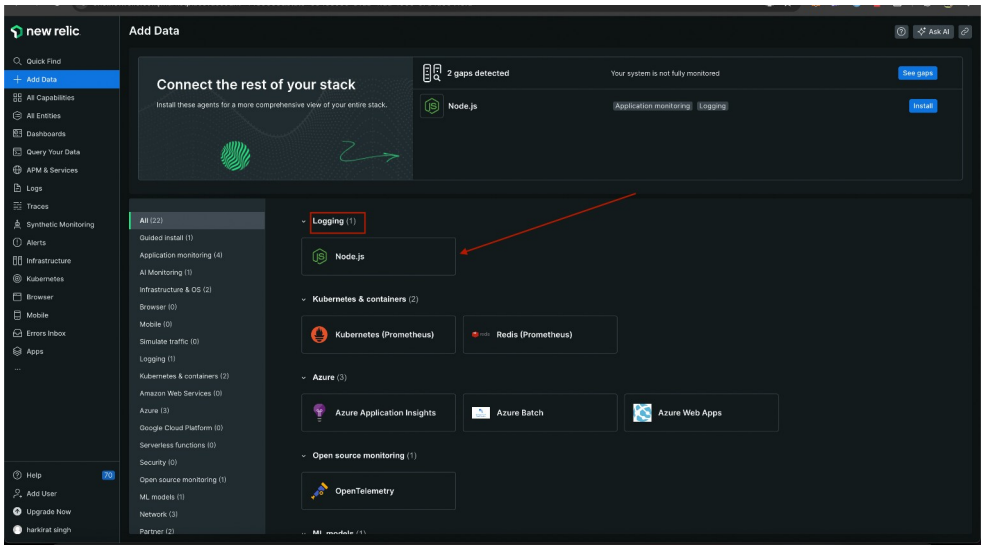
Default logs

By default, you will see it is tailing global nginx logs, but not the application logs

/var/log/nginx/access.log



Setting up APM for a Node.js process



- Create a new data source with Node.js
- Use one of 5 ways to set it up (we'll go with on the host)
- Install dependencies

```
npm install newrelic express
```

- Update package.json

```
  "scripts": {
    "start": "NEW_RELIC_APP_NAME=test
NEW_RELIC_LICENSE_KEY=license_key node -r newrelic
index.js"
  },
```

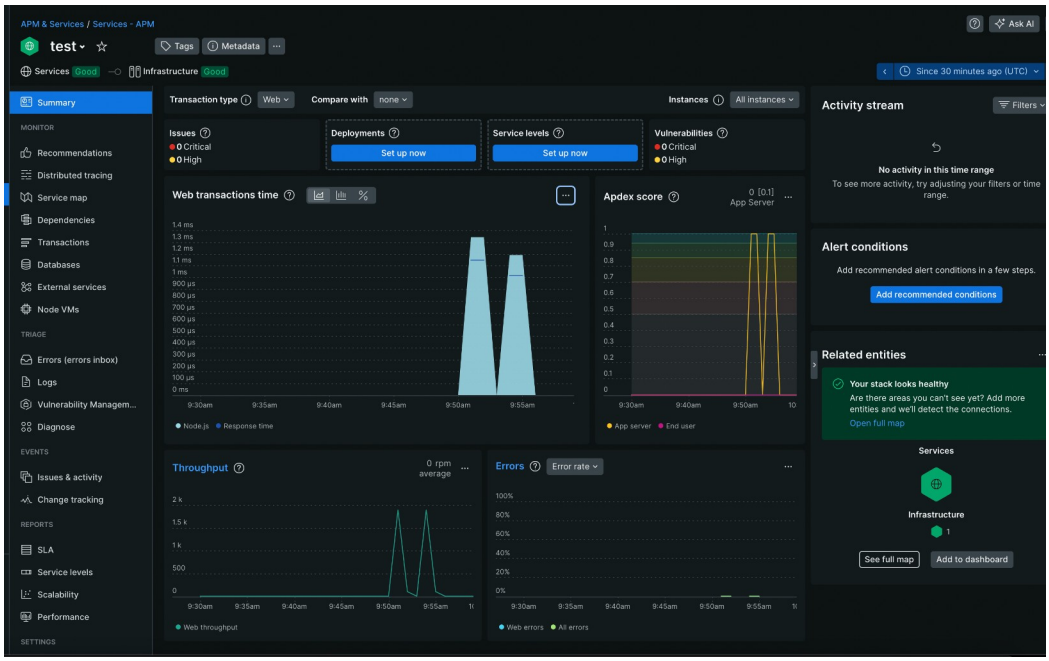
- Create a simple express app

```
require("newrelic");

const express = require("express");

const app = express();

app.get("/", (req, res) => {
        console.log("route hit");
        res.json({message: "hi there"})
})

app.listen(3000, () => {
        console.log("listening on port 3000");
});
```

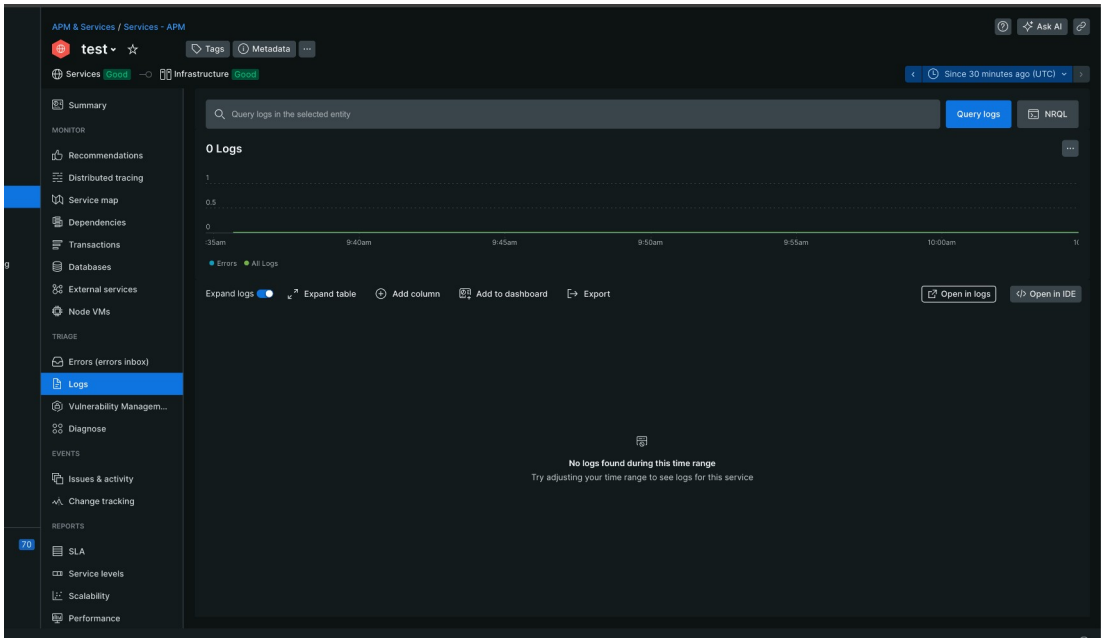
- Open another terminal and loadtest the app

```
npm i -g loadtest
loadtest -c 10 --rps 200 http://localhost:3000/
```

- Check the APM dashboard



- You will notice logs are still empty



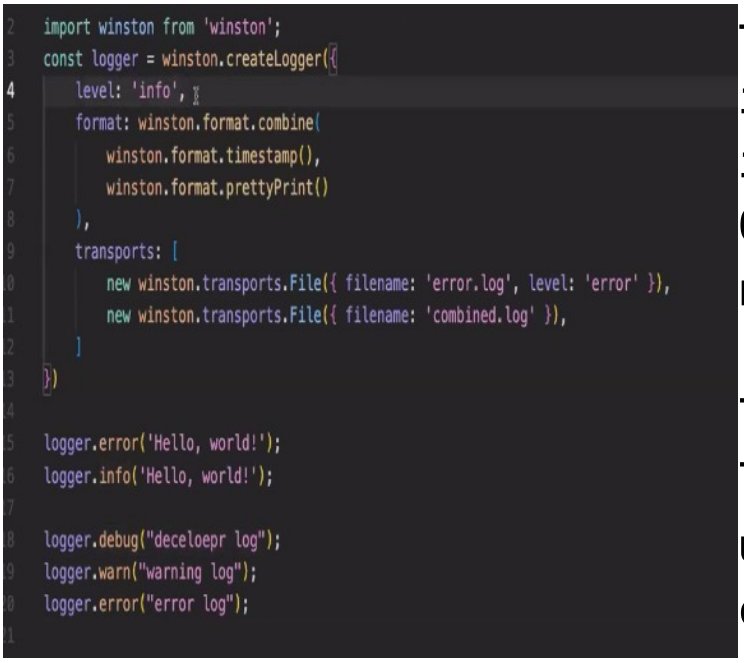
LEARNING ABOUT A LOGGING winston module

this is used to shove the data in any format in which we want

there are various levels of logging

Logging levels in winston conform to the severity ordering specified by RFC5424: severity of all levels is assumed to be numerically ascending from most important to least important.

```
const levels = {
  error: 0,
  warn: 1,
  info: 2,
  http: 3,
  verbose: 4,
  debug: 5,
  silly: 6
};
```



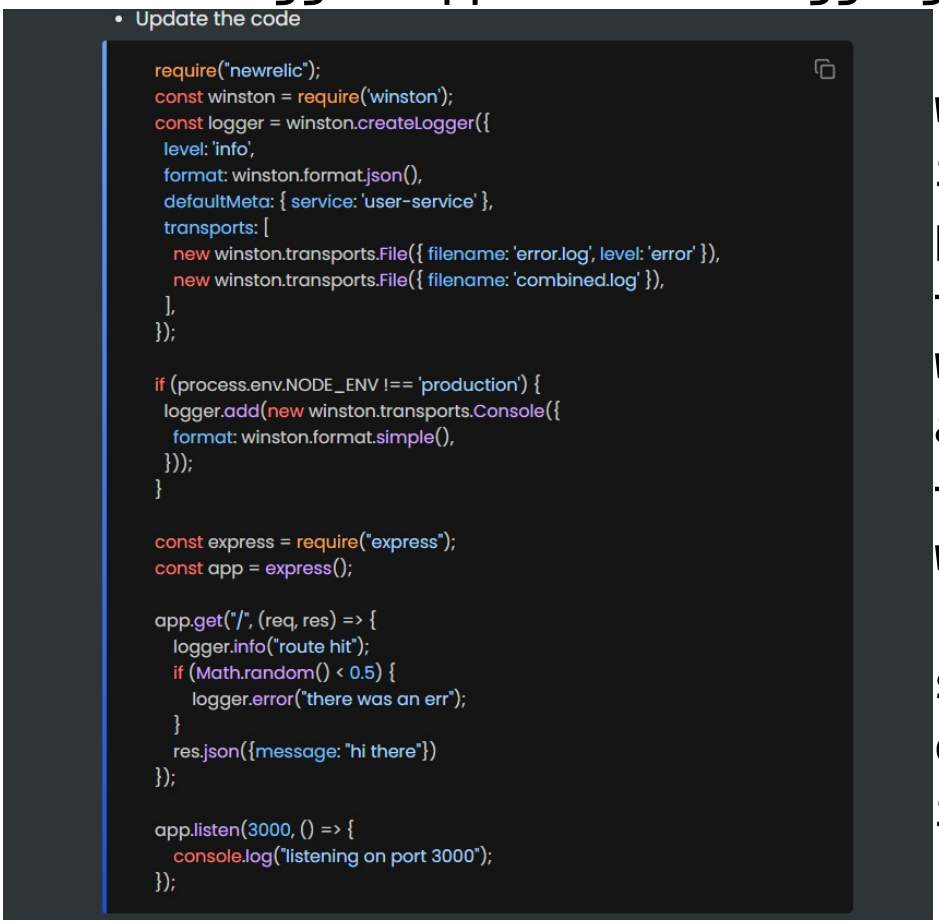
this is like various levels in which like we chose info it means above this level 0,1,2 only will be shown not any other

like below there are many logging messages it helps us in using certain types of logs for certain machines also

format: winston.format.simple()

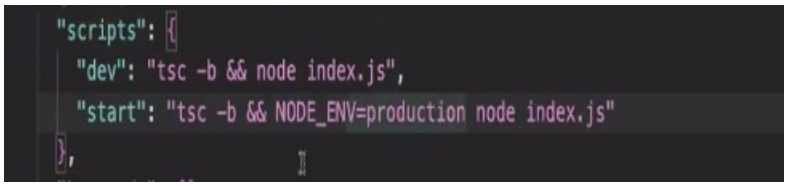
main benefit is in the transport thingy
it creates file for different error like for errors
it will create a error.log and will also create a combined file

new.winston.transports.console()
in this also it will show the logs in the console also
format can be many such as
format: winston.format.json();
format: winston.format.combine(
    winston.format.timestamp(),
    winston.format.json(),
    winston.format.prettyPrint()
)
this shows them in a good format and better view
for a bigger application logging is required



we can add thing like if we want the console print just for us not for the production what we can do is just add in the code
first do npm install winston

see that prod line only show console logs if not in production



if we pass this in package.json then it will not show the console logs as it is set to production env

bcoz I will run is npm run dev and in prod we will run npm run start

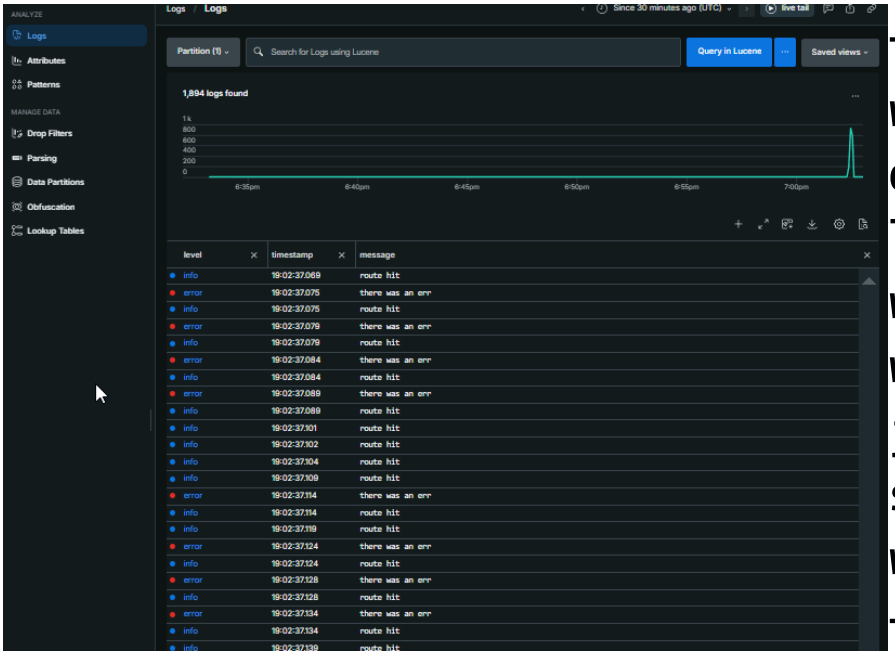
before trying first do
rm index.js
then do vi index.js and paste the above code

do npm install winston
then also need to add this in the package.json

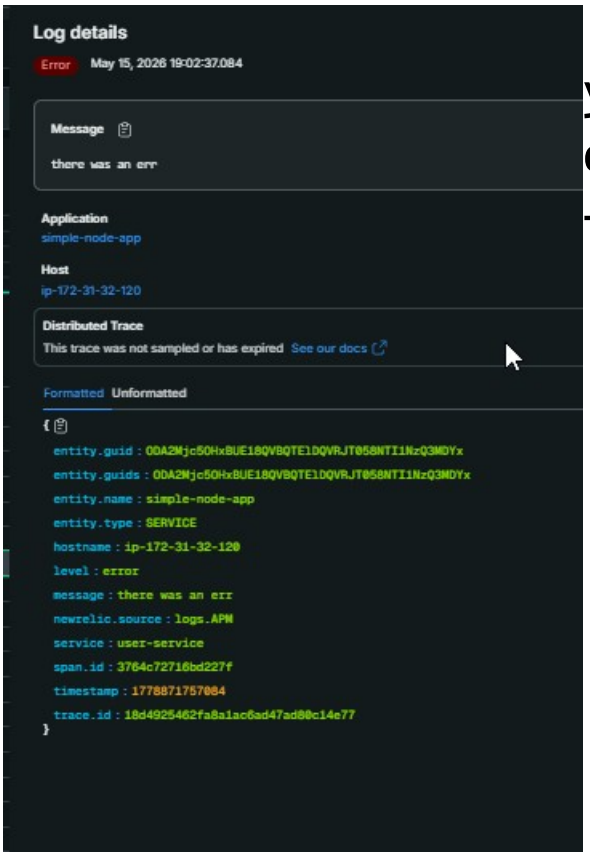
```
  "scripts": {
    "start": "NEW_RELIC_APPLICATION_LOGGING_FORWARDING_ENABLED=true NEW_RELIC_APP_NAME=test NEW_RELIC_LICENSE_KEY=your_key node -r newrelic index.js"
  },
```

this is done to forward the logs to newrelic
then do npm run start
if it crashes check for
sudo lsof -i :3000
then do kill -9 PID
then send the request
loadtest -c 10 --rps 200 http://localhost:3000/

now check in the logs of the newrelic



this is usefull like whenever someone is doing something fishy like in the samay website someone told we can hack it he was just hitting the signin endpoint and was trying to brute force the otp for which the spike was visible here
you can check for backend errors if happens and can check where is a actuall error is



you can right click and add to query to search based on that filter

Winston logger

Transports

Winston lets you stash logs in various places (files, console, postgres tables etc)

```
const logger = winston.createLogger({
  transports: [
    new winston.transports.Console(),
    new winston.transports.File({ filename:
'combined.log' })
  ]
});
```

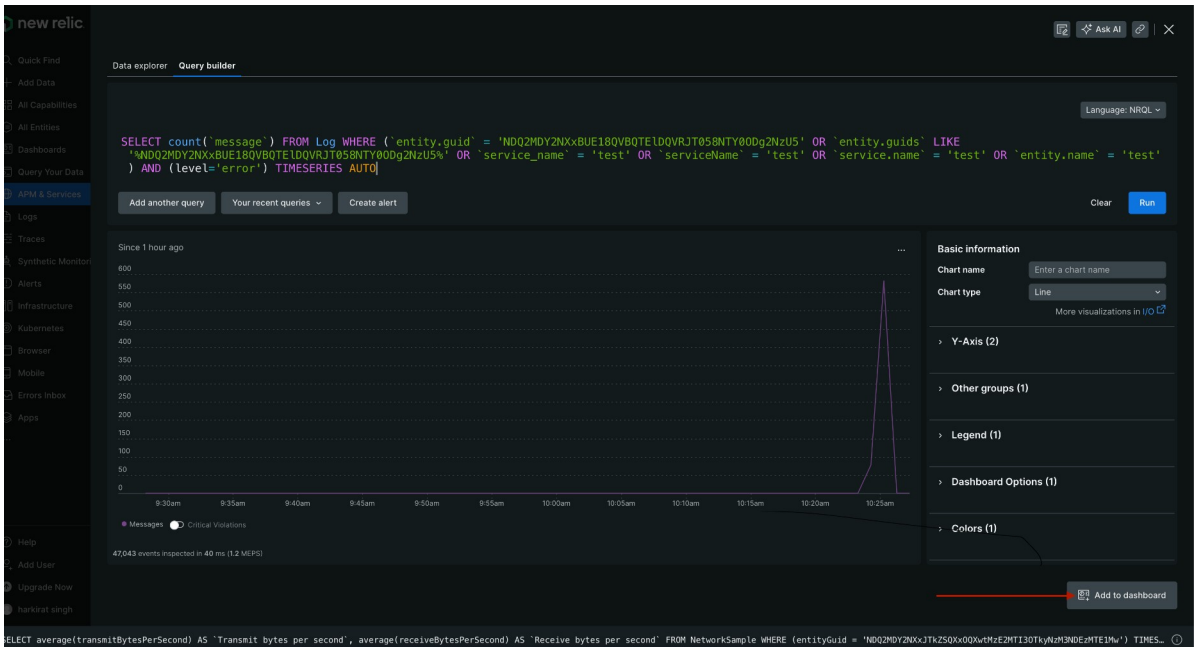
💡

Good list of transports - https://github.com/winstonjs/winston/blob/master/docs/transports.md#postgresql-transport

Metrics on logs

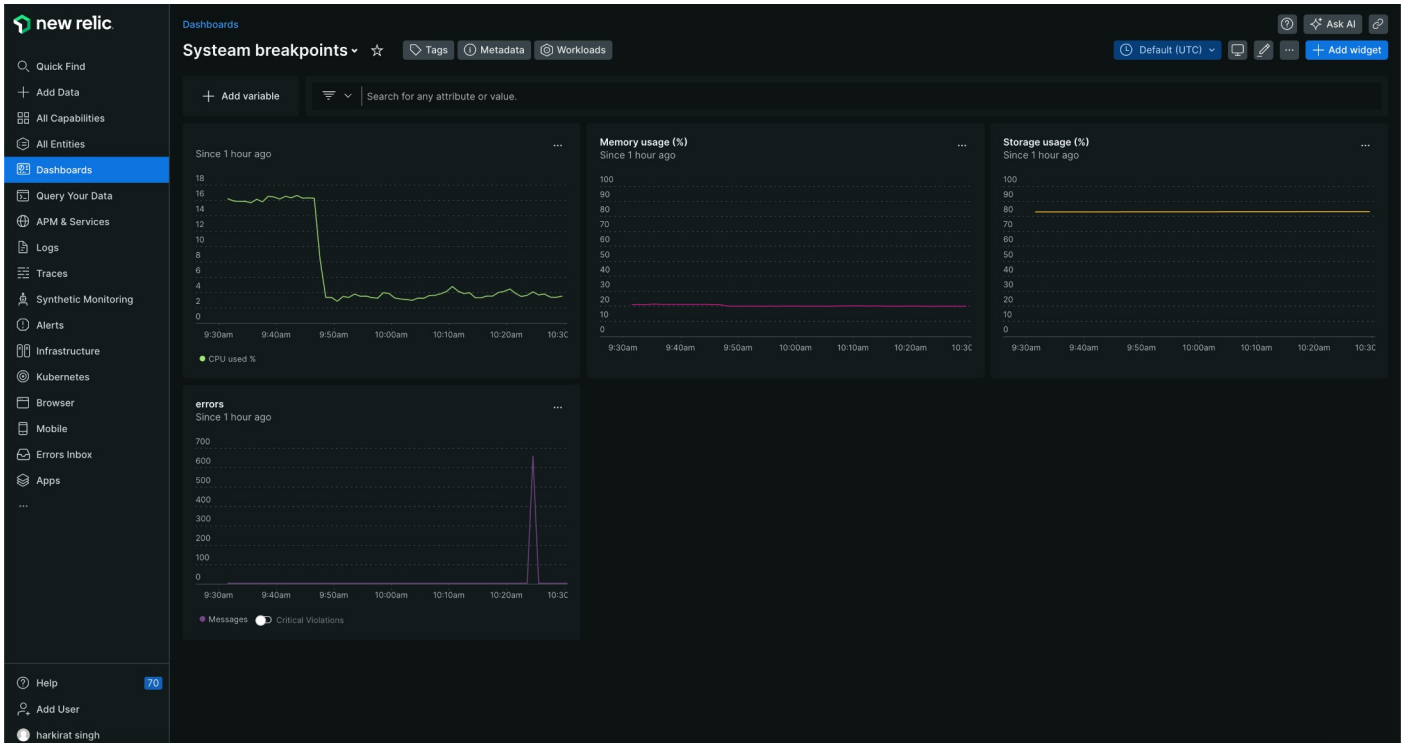
You can add metrics on top of log counts (esp for errors) to catch if a certain error is being thrown too often/there is a spike

```
SELECT count(`message`) FROM Log WHERE (`entity.guid` =
'NDQ2MDY2NXxBUE18QVBQTElDQVRJT058NTY0ODg2NzU5' OR `entity.guids`
LIKE '%NDQ2MDY2NXxBUE18QVBQTElDQVRJT058NTY0ODg2NzU5%' OR
`service_name` = 'test' OR `serviceName` = 'test' OR `service.name`
= 'test' OR `entity.name` = 'test') AND (level='error') TIMESERIES
AUTO
```



You can also add individual set of metrics on individual errors

```
AND message LIKE '%there was an error%'
```



MONITORING QUESTION INTERVIEW

Suppose you have a system which has like 1lakh users in india who are getting like 1 ms of res time but there are like some users in nigeria which are like getting 200s (4-5 users)of res time why that happens? What are your thoughts?

p95, p99

When you are measuring things like response times , cpu usage etc, which of the following should you use?

1. Mean
2. Median
3. Something else?

Example

Let's say in a 20 second interval there were 20 requests that came and the response times were -

[1, 2, 3, 4, 4, 4, 5, 6, 7, 9, 10, 11, 11, 11, 12, 13, 13, 13, 50, 100]

Average - 14.45
Median - 9.5

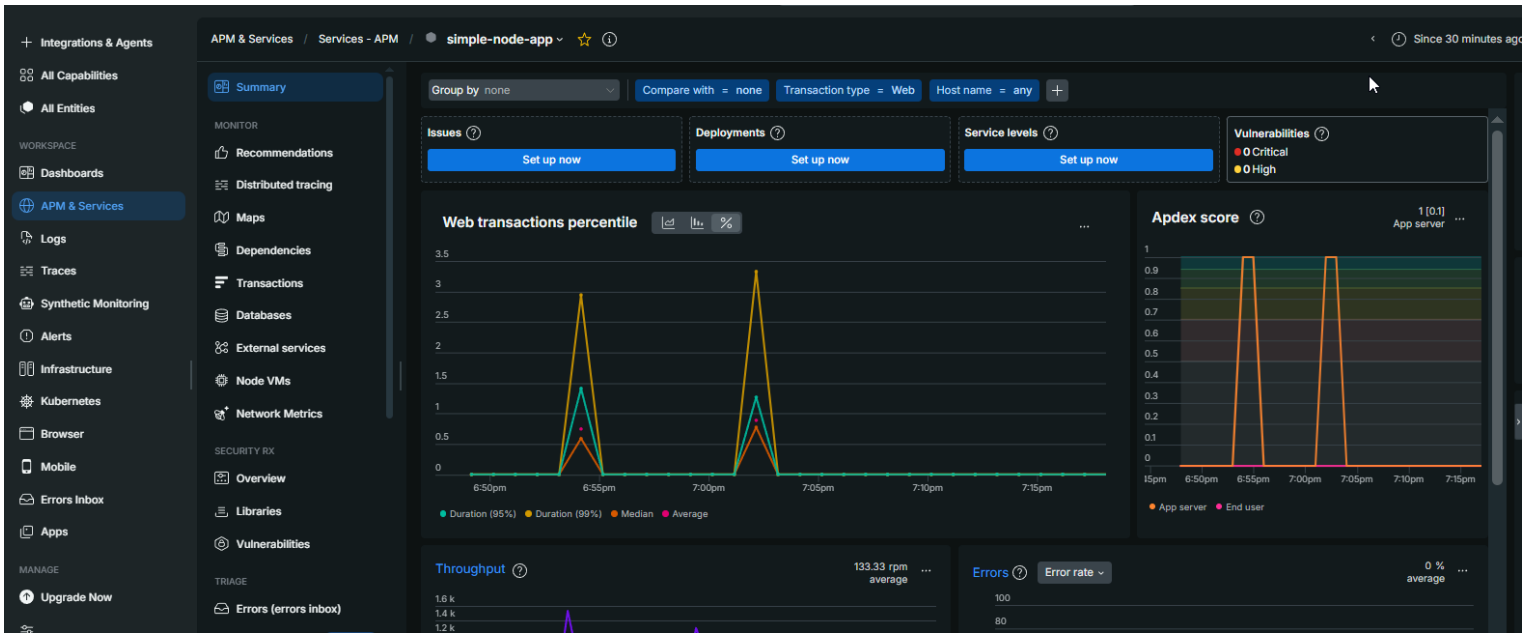
Better metrics

As you can see, the average is skewed a little to the right because of two anomolies (50, 100)
This means the website is performing good for 90% of the users, but 10% of the users are having slower response times.

- P90 here is 13, since 90% of the users are equal to or below 13ms

Can you guess what P95 would be?
P95 is 50, since 95% of the users are equal to or below

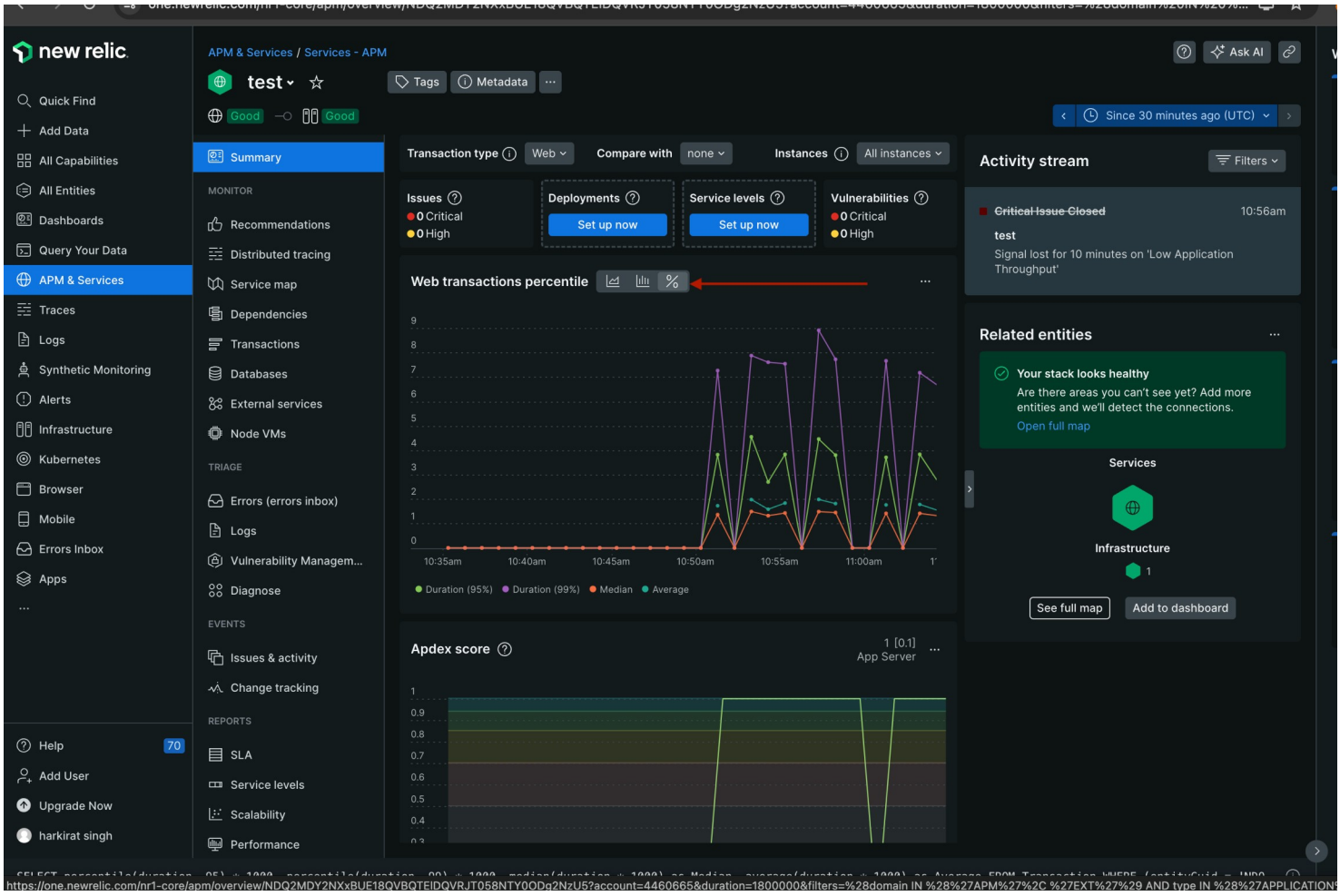
SO WHAT WE HAVE TO THINK IS LIKE FOR P99 WE CARE ABOUT THE 99 PERCENT OF THE USER NOT THAT 1 PERCENT AND IN THE P95 WE CARE ABOUT THE 95 OF THE USERS NOT THE 5 % USERS
WE OPTMISIZE THE SYSTEM MOSTLY FOR THE P99 AND P95 ONLY
SO FOR THE ABOVE DATA P95 WILL BE 12
P99 WILL BE 50



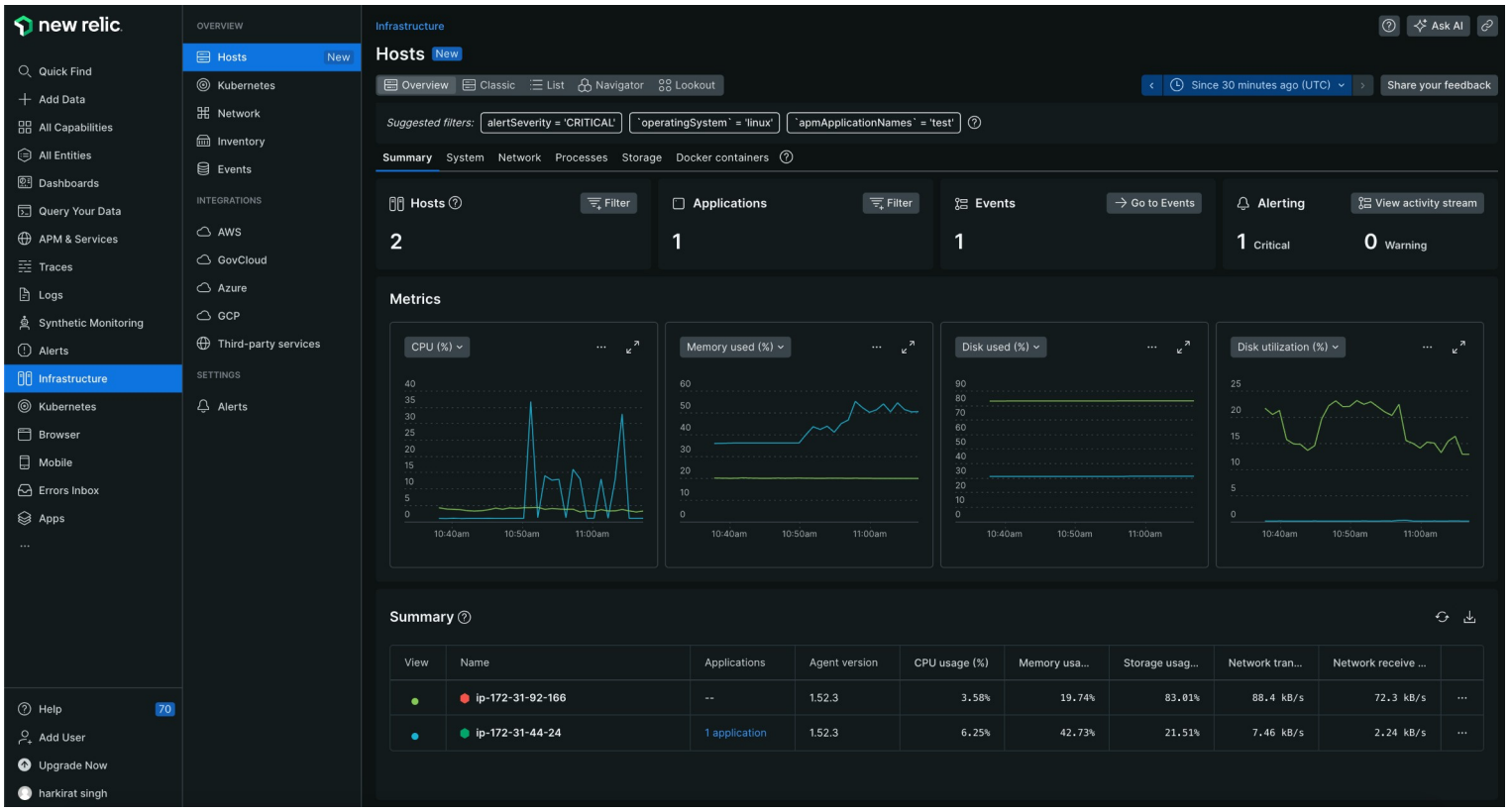
See in this you can choose the metric the sum, average, or the percentile above the graph see the three boxes

Percentile based metrics in newrelic

If you open the APM dashboard, you'll see the response time percentiles tab



Infrastructure tab



Status pages/Notifiers

Pagerduty and Better stack are two popular services used by teams for on call management, raising a call to the manager incase the on call is asleep

https://betterstack.com/uptime

bettstack we can use for checking the working of the website

Problem?

All services we've used (newrelic, uptime, datadog) can get very expensive very quickly.

You also become highly dependant on them as time goes by so very hard to get the costs down.

Solution

Use open source tools that let you do the same thing and self host.
Most common stack for monitoring, observability and logging - Grafanna, Prometheus and Loki

1. https://github.com/grafana/grafana
2. https://prometheus.io/
3. https://github.com/grafana/loki