#SERVERLESS

How to Monitor Stack Overflow Activity in Slack



by Nicolas Grenié

Powered by a complete serverless stack

Developer Advocates or Evangelists are often asked how they measure success.

One could argue that GitHub ⭐'s on repositories are great. And Ash Hathaway shares her thoughts here.

Others could say that Stack Overflow questions give a good sense of an API popularity. I would agree with this statement.

If you don't have a forum for your product, developers may end up asking questions directly on Stack Overflow.

I would not encourage companies to redirect all their support questions there or pushing for their brand tags, this is just bad. You are hijacking a community for your own needs instead of contributing to it.

But :

What happens when someone is posting a question there?

How do you get notified?

How can you react quickly?

In my experience, you can only receive notifications when there are new questions on a particular tag. This might not cover all the questions posted about your API

questions posted about your API.

You can also use tools like Mention. It covers more than Stack Overflow, but it's not instantaneous.

To be more reactive, I hacked together something for our team at 3scale a long time ago. I recently rewrote it using serverless technologies. It monitors questions about **3scale** and posts them on our **#support** Slack channel. The support team can jump on it and answer quickly.

I want to share this project, so you too can monitor Stack Overflow directly in Slack.

The tools?

To fit in 2017 trend, we will only use serverless technologies, which will make this tool free to use:

- AWS Lambda to host the logic of our app

- FaunaDB to store databases

- Serverless Framework to simplify deployments

- Slack to be notified when a new question is asked



The logic⚙️

Stack Exchange API has an endpoint to search for questions. You can look for a term in tags, in the title, in the body of the question or all at the same time. In the example, we will search in all attributes.

You should create a key to access Stack Exchange API here.

Our function will regularly call this endpoint to check if there are new questions posted. We will use the *Schedule Events* feature offered natively by Lambda, it's a similar behavior as a cron job.

We will store the questions in a database to keep track of the ones we already sent to Slack and avoid duplicates.

If a Stack Overflow question is not in our database, we add it to the database and send details to Slack.

FaunaDB

I discovered FaunaDB a few months ago at Gluecon. They present themselves as the first **serverless** database engine. Everything is hosted on their end. FaunaDB is a globally distributed database that doesn't require any provisioning. Capacity is metered and available on demand, so you only pay for what you use.

If you are familiar with Firebase, you will recognize a similar data structure and routes to access resources. But it comes with more features, which makes it easier for example to query the database.

```
client.query(
  q.Create(
    q.Class("posts"),
    { data: { title: "What I had for breakfast .." } }));


=> {
  "ref": { "@ref": "classes/posts/104979509693618791" },
  "class": { "@ref": "classes/posts" },
  "ts": 1436375112432352,
  "data": { "title": "What I had for breakfast .." }
}
```

Example: How to create an entry in FaunaDB

For this app, you will need a database, with a **questions** class. We will also add an index **questions_by_id** on terms **data** and **question_id.** This will let us query the questions class by the Stack Overflow id.

If you are concerned about your database usage, you can add TTL to the **questions** class. This will automatically delete instances older than the TTL value.

Finally, you will need to create a server key for the questions class.

This key will be used to authenticate our function to FaunaDB servers.

```
client.query(q.Get(Ref("classes/posts/104979509694379214")));


=> {
  "ref": { "@ref": "classes/posts/104979509694379214" },
  "class": { "@ref": "classes/posts" },
  "ts": 1436375112490514,
  "data": { "title": "My cat and other marvels" }
}
```

Example: How to retrieve an entry in FaunaDB

Slack

To post to Slack, we would just need a simple incoming web hook.

Create one here.

Configure the project

Make sure you have installed Serverless framework and configured

the AWS CLI tool.

You can now clone this project locally.

```
git clone git@github.com:picsoung/stackOverflo
```

In `serverless.yml` you will modify the environment variables to your own values.

`FAUNADB_SECRET` is the secret we created earlier to access FaunaDB

`STACK_EXCHANGE_KEY` is the API key to access Stack Exchange API

`SLACK_WEBHOOK_URL` is the URL of the Slack incoming webhook you've created

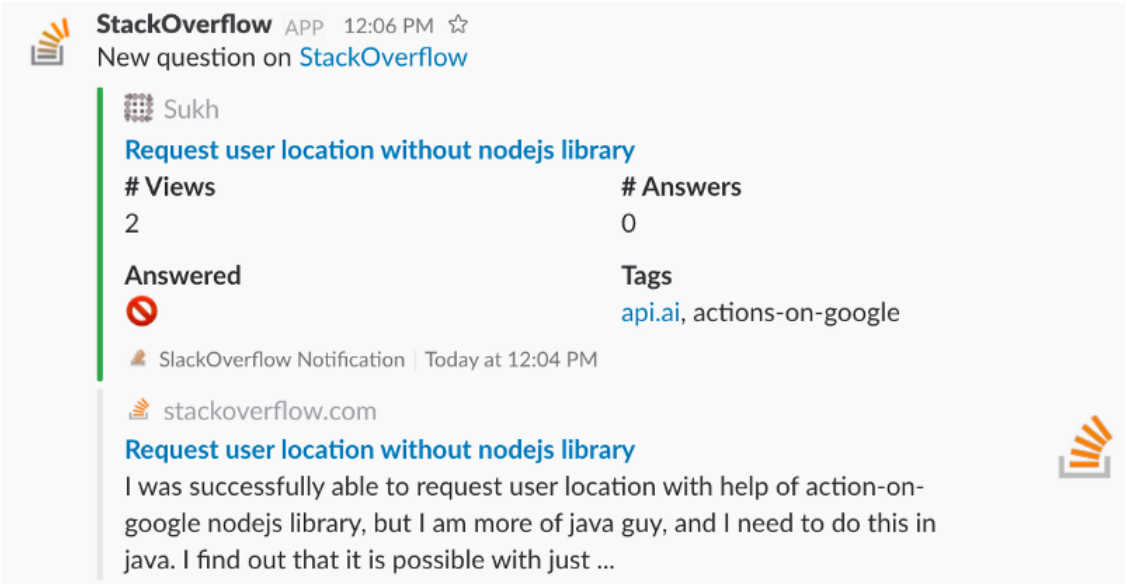
`SLACK_CHANNEL` should be an existing channel name in your Slack team such as #support or #stackoverflow

`SEARCH_KEYWORD` is the keyword you are interested in monitoring such as Node.js or Angular2

Once you have changed all the variables to your own values, we can test if everything works. We invoke the function locally with the following command:

```
serverless invoke local — function getStackOve
```

As it's the first time you are launching the function, it should post a message to your Slack channel. It should look like this:



How the notification should look like in your Slack channel

If you are happy with the result you can now deploy the function with the following command:

```
serverless deploy
```

By default, the function is called every 20 minutes. You can customize it by changing the schedule property in `serverless.yml` file.

Extend the project

For now, we are only monitoring one term. You can launch multiple instances of this function to watch more terms or tags on Stack

Overflow.

If you are interested in near to real-time solution, I encourage you to check Streamdata.io. Their tools turn pulling API into streaming API.

If you want some nice dashboards that show how active your community is on Stack Overflow, I recommend Keen.io. You can send all your Stack Overflow data there. Keen offers a variety of libraries to build beautiful dashboards.

We can also add more features in Slack, like buttons or menus. So people can claim a question or get assigned a question to answer.

Conclusion

This was a small project that led me to discover how to use FaunaDB. Using AWS Lambda it's way more efficient that the Heroku instance that I had in the past.

I hope you've found this serverless example useful. The code is openon GitHub so feel free to contribute and add new features.

If you are working for a company that is selling to developers I am sure you already heard the "Be where the developers are."

In the online world you have a good chance to find developers on websites like Hackers News, Stack Overflow, or GitHub. It's important to measure what people say about your product or technologies on those sites.

This was a small project that led me to discover how to use FaunaDB. Using AWS Lambda is way more efficient than the Heroku instance that I had in the past.

I hope you've found this serverless example useful. The code is openon GitHub so feel free to contribute and add new features.