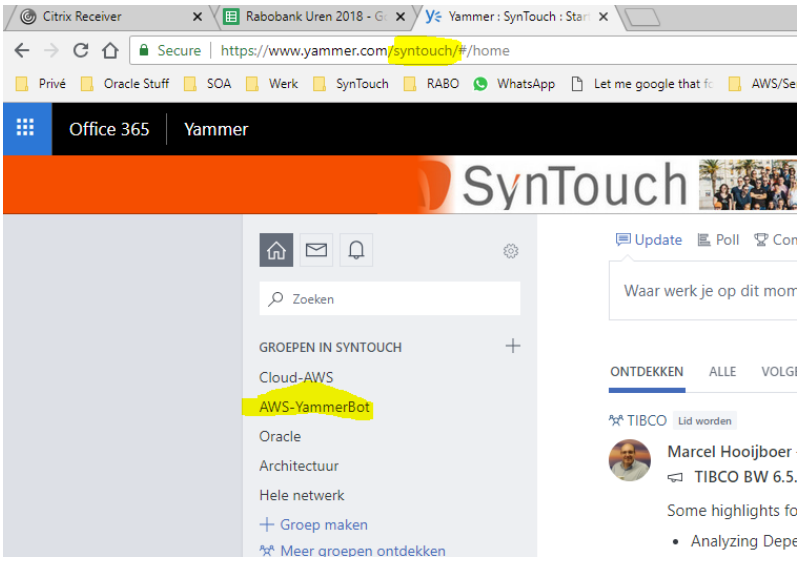
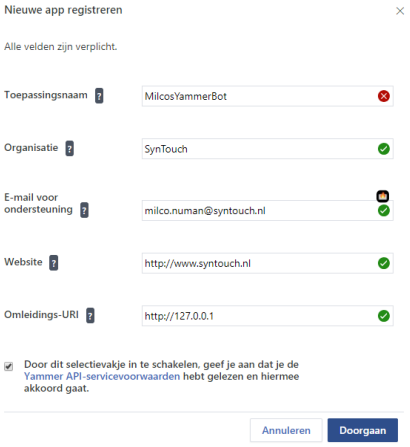
Guys don’t want to commit!

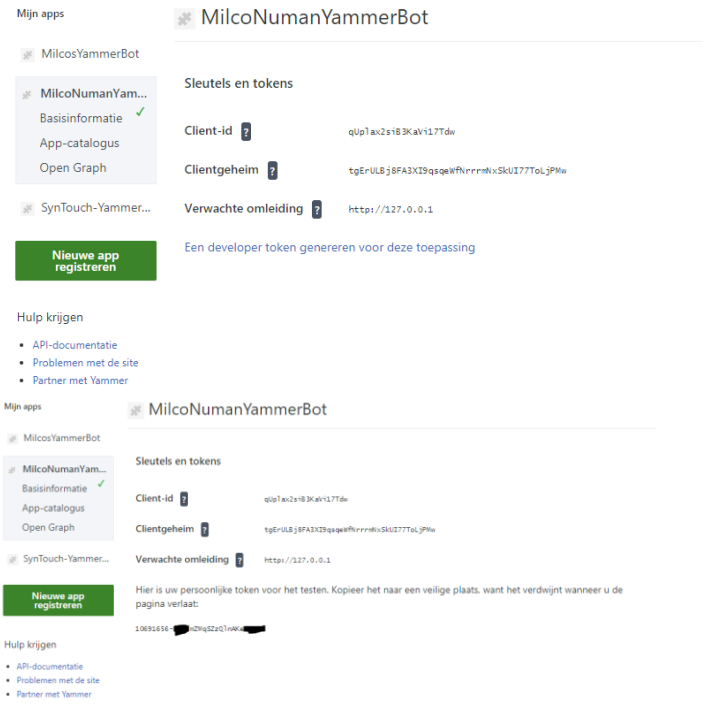
1) Use your current Yammer-account to log into our **SynTouch** Yammer-network; if you have multiple networks on your Yammer account, please make sure to switch to the SynTouch-network:



2) Open a new tab on your browser to https://www.yammer.com/client_applications
3) Use the button to register a new application, prefix this with your name:

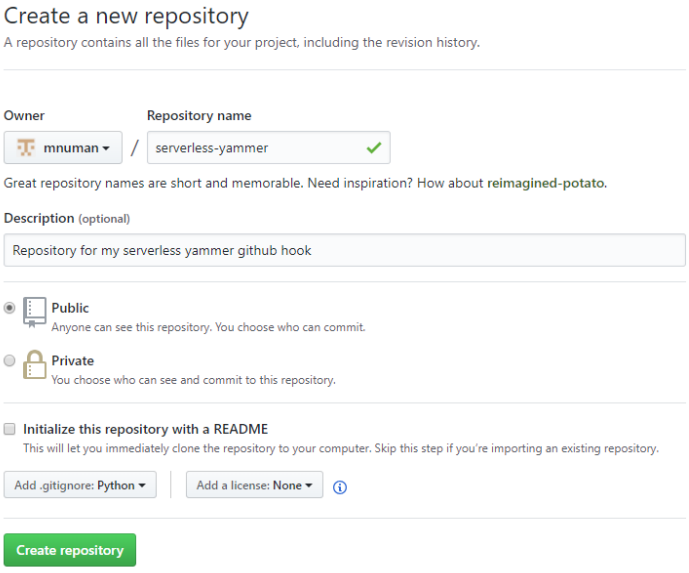


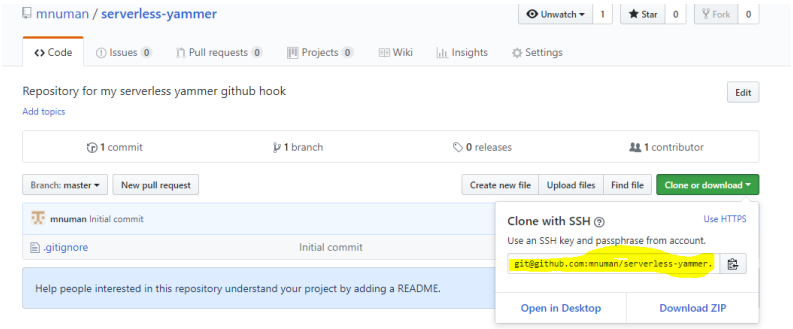
4) On the confirmation page, use the link to generate a developer token and capture the value:



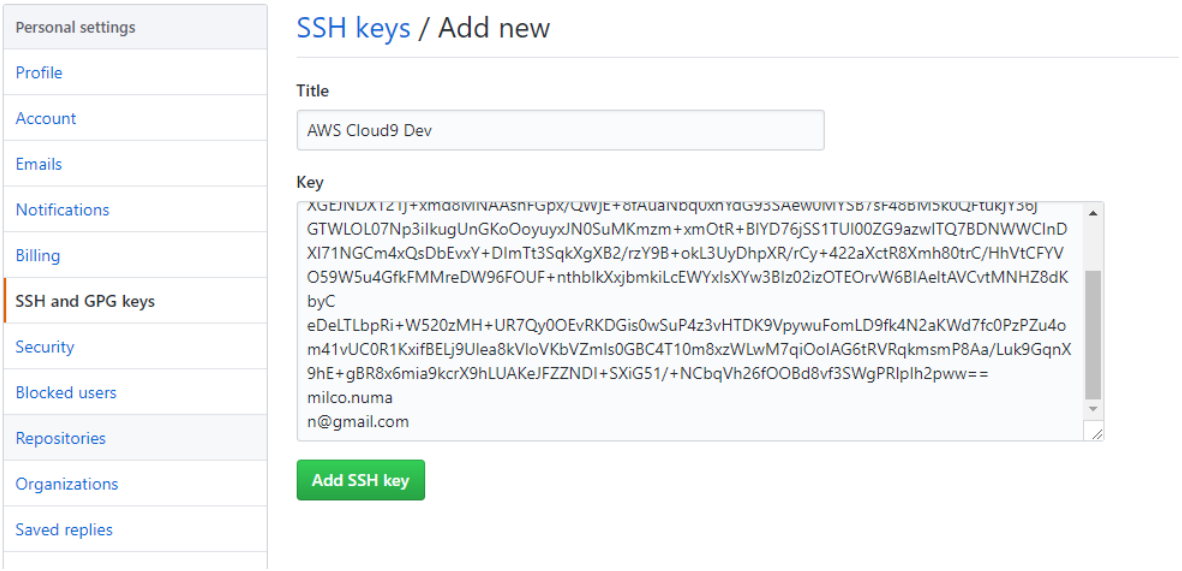
5) Open a new tab in your browser and lookup the groups in your network on URL https://www.yammer.com/api/v1/groups.json.
Find the AWS-YammerBot group and note its id

6) Create a new empty repository in Github (note its SSH address), use the Python gitignore option:



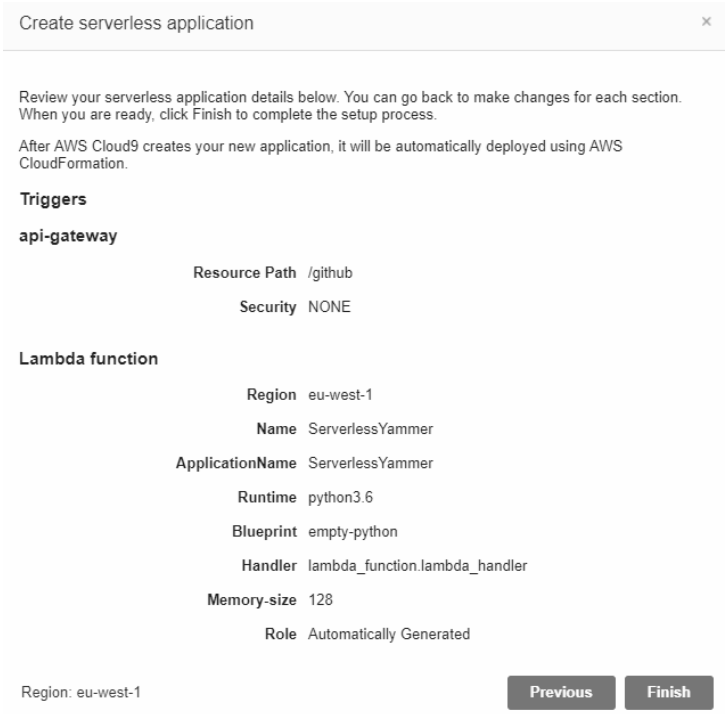


7) Generate a new keypair from your Cloud9 terminal window:
   ssh-keygen -t rsa -b 4096 -C "**your-email-for-github**"
8) Configure your git client by setting your identity for git (Cloud9, terminal window):
   git config --global user.name "User name to show in github"
   git config – global user.email "your-github-email-here"
9) Now ad d the public key (~/.ssh/id_rsa.pub) you generated in step 6 to your Github account:

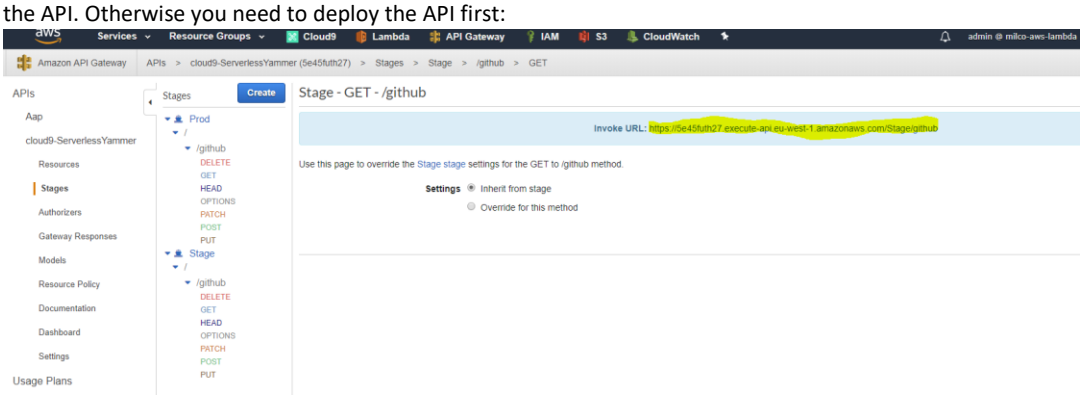


10) In Cloud9's terminal window, move to your home directory and initialize a new git repository there:
    git init .
11) Connect the new local repository to your GitHub repository (text in italics should be changed into YOUR repo address!):
    git remote add origin *git@github.com:mnuman/serverless-yammer.git*
12) Synchronize by pulling in all remote changes and merging them into the master branch:
    git pull origin master

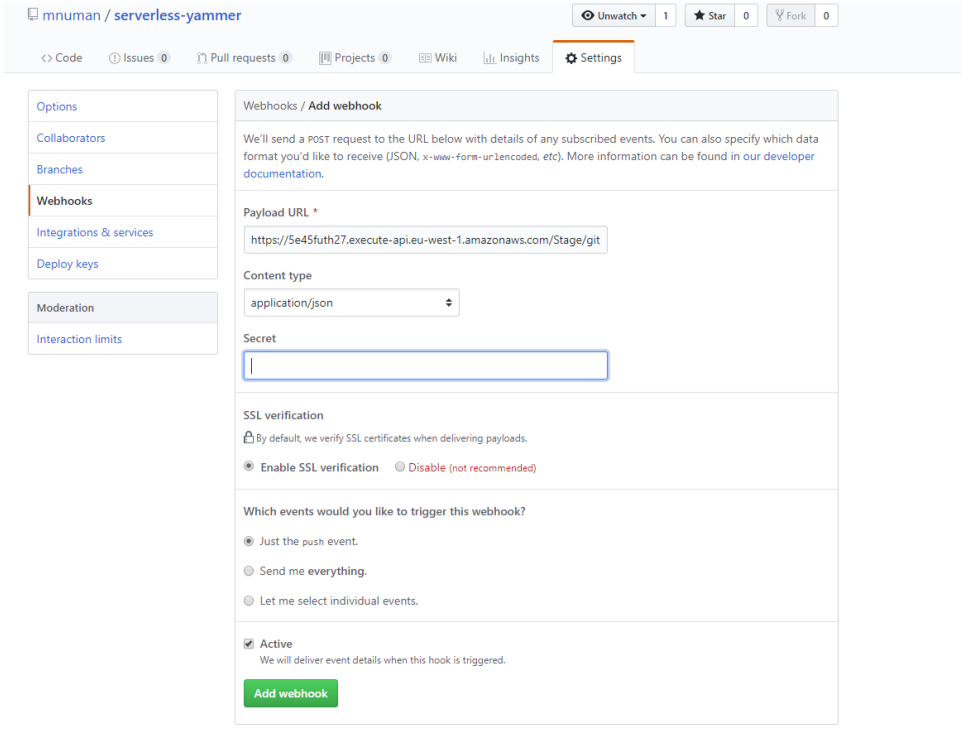
13) Create a new serverless application + function in Cloud9, type empty python 3.6, triggered by API Gateway on resource path /github, no security. Defaults apply:



14) Verify if the API has been deployed to a stage (API > Stages). If it has been deployed, pick one of the endpoints for the API. Otherwise you need to deploy the API first:

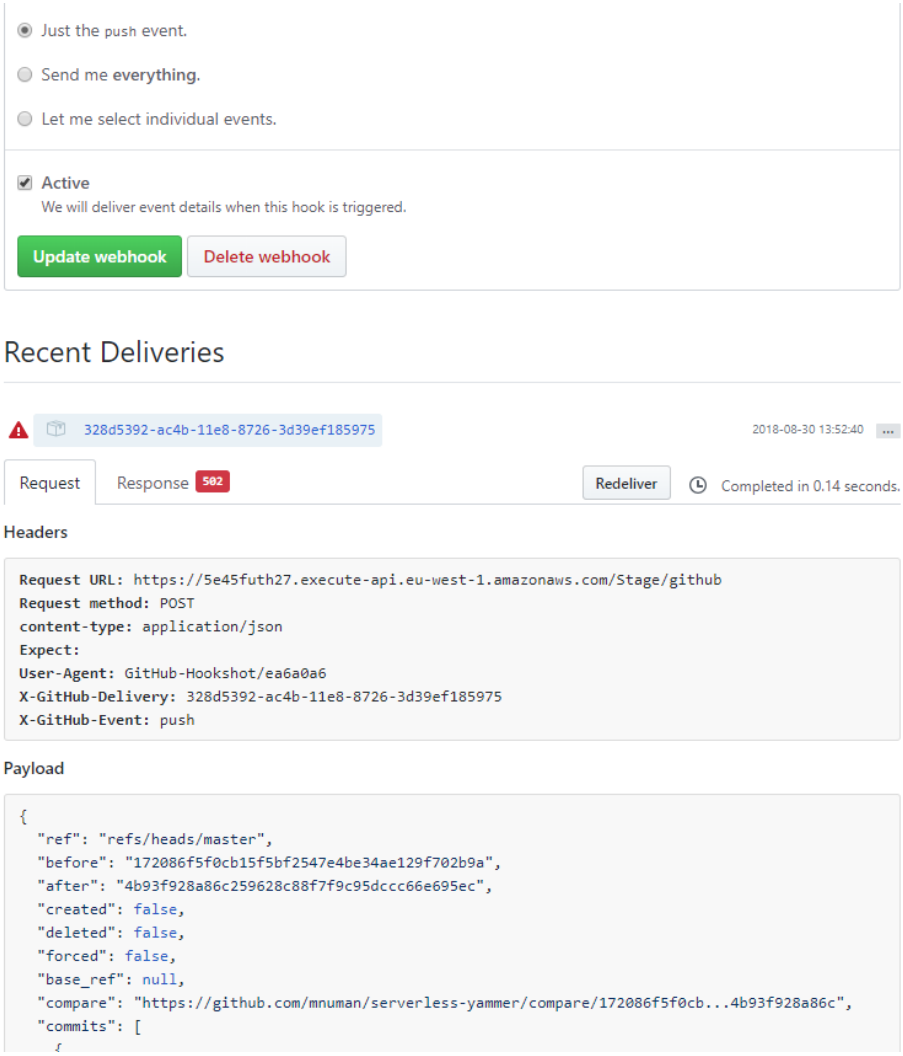


Navigate to Github, add a new webhook from the repository settings in the repository you created for this purpose:

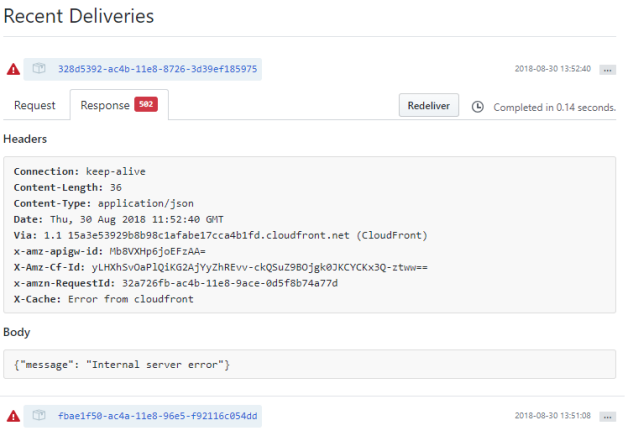


15) Now commit your code from Cloud9's terminal window and push this to Github. This should trigger the actual lambda function:
    (From my home directory)
    git add ServerlessYammer # add all resources in my serverless application directory
    git commit -m "Initial commit, empty shell"
    git push --set-upstream origin master
16) Verify in CloudWatch that your lambda function has been triggered.

17) From Github you can also inspect the payload that has been sent and the response that Github has received:



Currently, API Gateway still returns an error message (statuscode 5xx) as we did not put in any effort to format the answer properly – for now that is okay:

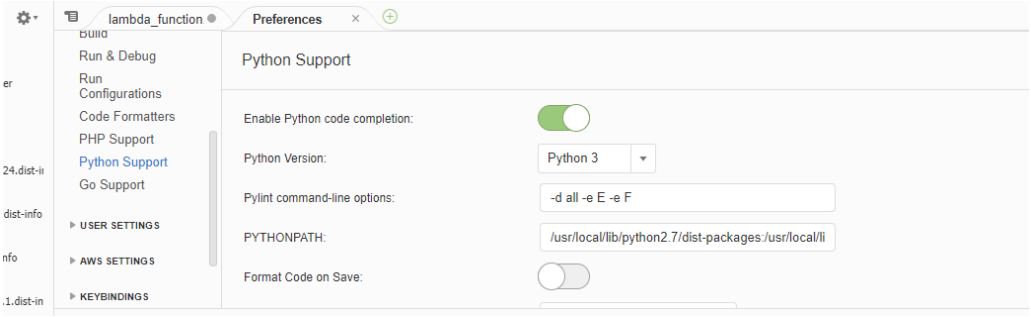


18) Change into your **Application's** directory using the Cloud9 terminal, enable the virtual environment by source the activate script from venv/bin/activate and install the requests package locally:

```
admin:~/environment (master) $ cd ServerlessYammer/
admin:~/environment/ServerlessYammer (master) $ source venv/bin/activate
(venv) admin:~/environment/ServerlessYammer (master) $
```

```
pip install requests -t .
```

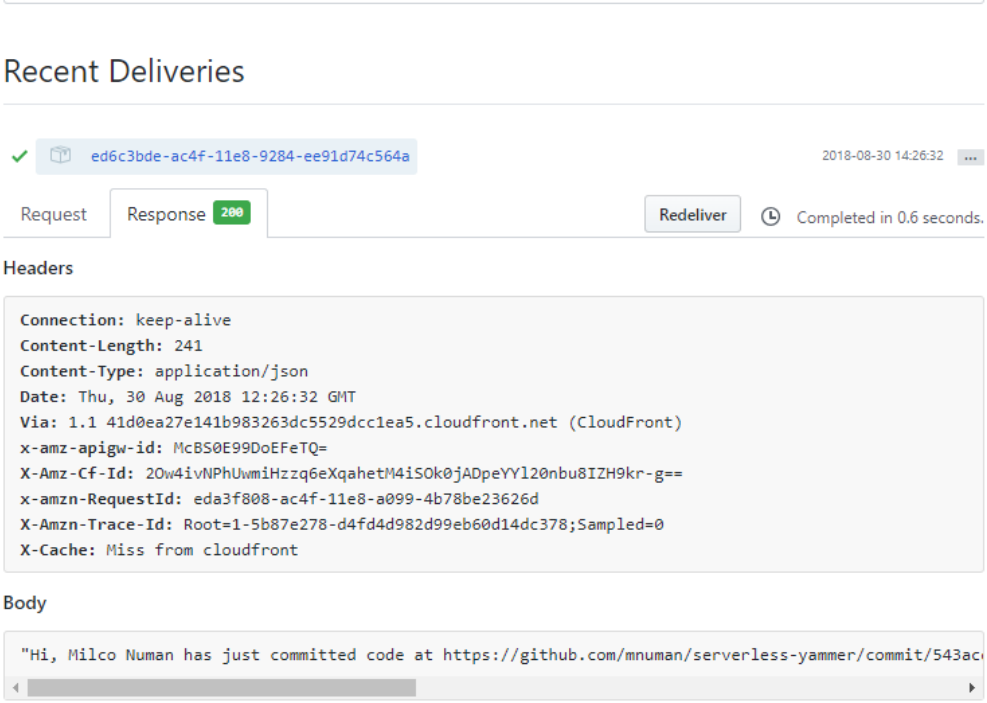
19) Make sure that the python support in Cloud9's preferences is set to Python3:



20) Define some implementation code to compose a message (to be sent out later) and prepare a response for GitHub:

```
import json
import requests
def lambda_handler(event, context):
    body = json.loads(event['body'])
    myMessage = f"Hi, {body['head_commit']['author']['name']} has just
committed code at {body['head_commit']['url']} with message
{body['head_commit']['message']}"

    return {
        "isBase64Encoded" : "false",
        "statusCode" : 200,
        "headers" : {},
        "body" : json.dumps(myMessage)
    }
```

21) Deploy your code to AWS
22) Again, commit your changes locally and commit your change to Github. Now verify that Github receives a proper response:



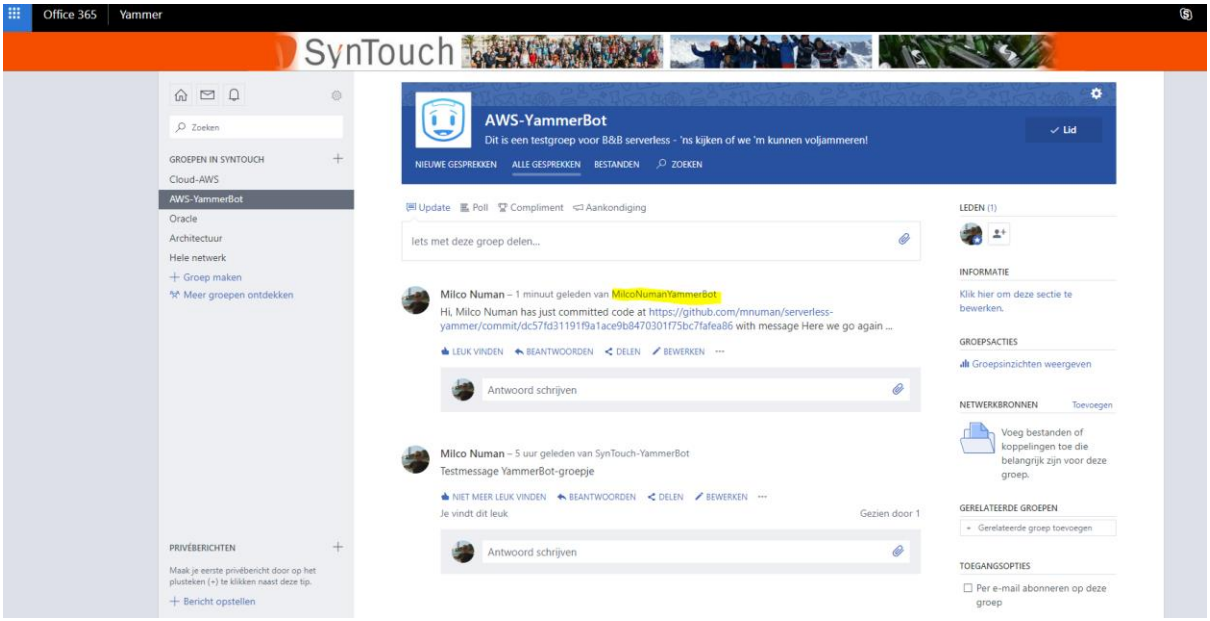
23) Now we're ready to push the message to the world …
24) In order to do this, you will need the Yammer developer token you jotted down earlier; we're leveraging the Yammer REST API (that's why you installed the requests package). This API is documented here. To authenticate your call, you need to pass in a dictionary in the headers parameters with the HTTP headers; the

header's name must be "Authorization", it's value is your developer token prefixed by "Bearer " (that is Bearer followed by a single space (OAuth).
The address for the Yammer API is https://www.yammer.com/api/v1/messages.json. The payload is entered as a dictionary in the data parameter, this dictionary must have a body key holding the actual message and a group_id holding the id of the AWS-Yammer group:

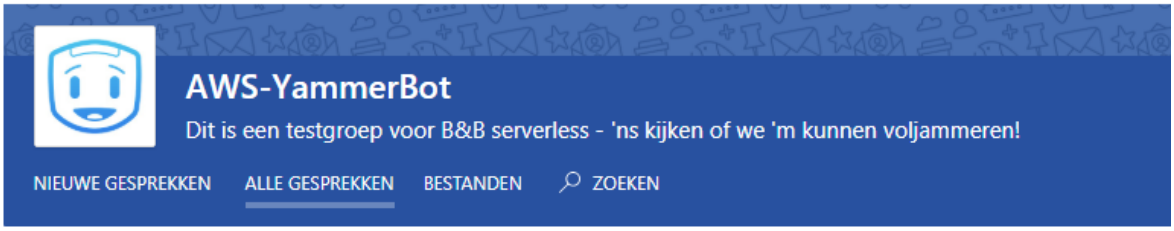
```
import json
import requests
def lambda_handler(event, context):
    print(json.dumps(event))

    body = json.loads(event['body'])
    myMessage = f"Hi, {body['head_commit']['author']['name']} has just
committed code at {body['head_commit']['url']} with message
{body['head_commit']['message']}"
    auth = { "Authorization" : "Bearer 10691656-iLf4mZMqSZzQlnAKeQq4aA"}
    payload = { "body" : myMessage, "group_id" : 15767042}
    r = requests.post("https://www.yammer.com/api/v1/messages.json",
headers=auth, data=payload)

    print("Status code Yammer call:" + str(r.status_code))

    return {
        "isBase64Encoded" : "false",
        "statusCode" : 200,
        "headers" : {},
        "body" : json.dumps(myMessage)
    }
```

Should you mess up (e.g. commit and push the code and forget to redeploy it first …), you can also redeliver message from GitHub:

And then, finally …



As an extra exercise, retrieve the authentication token from the function's environment variables (or from the secrets manager) – e.g. us os.environ to retrieve values.



Update Poll Compliment Aankondiging

lets met deze groep delen...

Milco Numan – 50 seconden geleden van MilcoNumanYammerBot

Hi, Milco Numan has just committed code at https://github.com/mnuman/serverless-yammer/commit/9d5234ab9d8e23c01fd7bbd06214a1e608ff4992 with message Invalidated my developer token and generated a new one; pushed the token into a environment variable set MANUALLY on the Lambda function after deployment

LEUK VINDEN BEANTWOORDEN DELEN BEWERKEN ...

Antwoord schrijven

Milco Numan – 14 minuten geleden van MilcoNumanYammerBot

Hi, Milco Numan has just committed code at https://github.com/mnuman/serverless-yammer/commit/dc57fd31191f9a1ace9b8470301f75bc7fafea86 with message Here we go again ...

LEUK VINDEN BEANTWOORDEN DELEN BEWERKEN ...

Antwoord schrijven