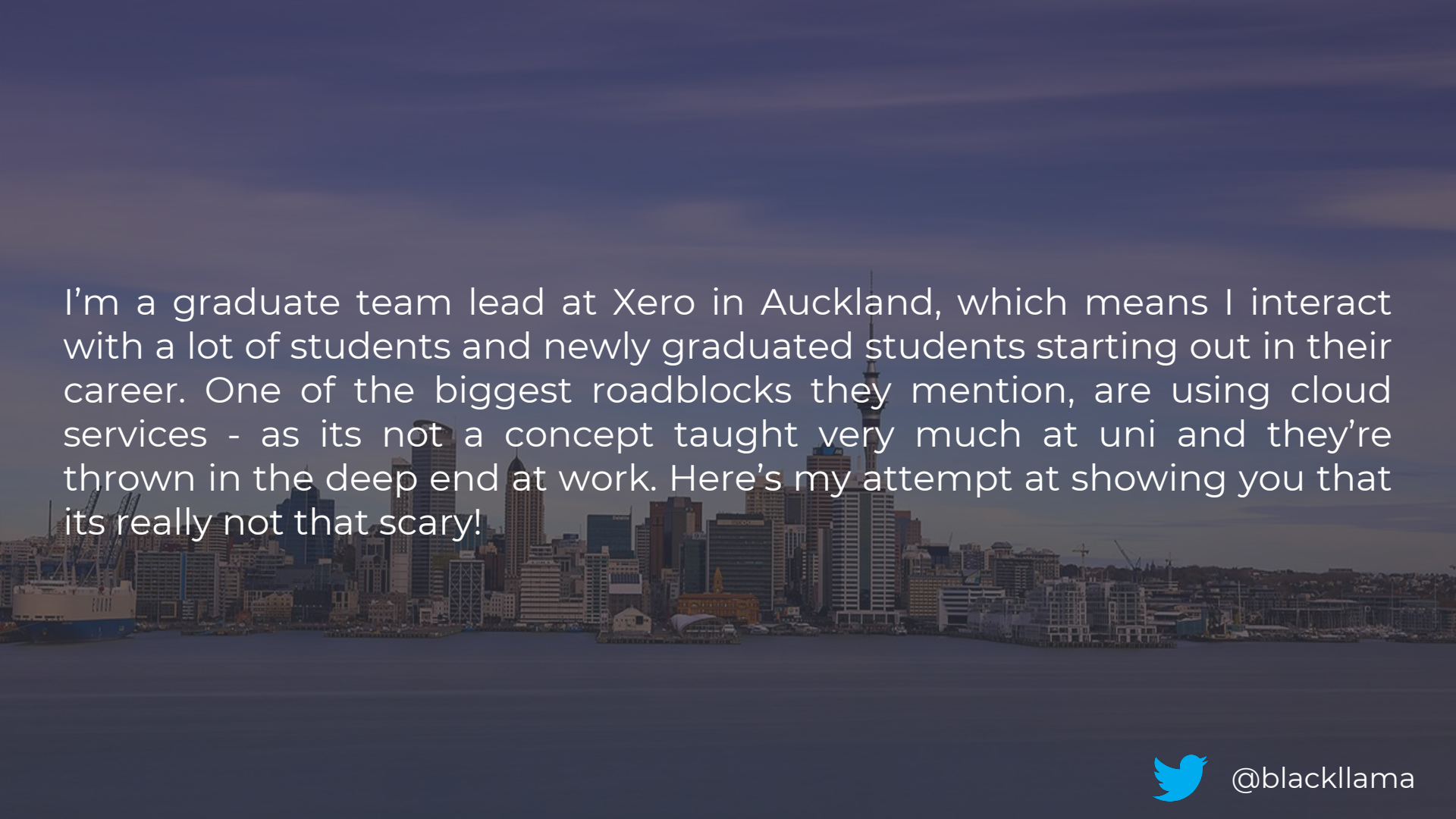
Zero to Serverless

Hallo, it me, Suki!

@blackllama



I’m a graduate team lead at Xero in Auckland, which means I interact with a lot of students and newly graduated students starting out in their career. One of the biggest roadblocks they mention, are using cloud services - as its not a concept taught very much at uni and they’re thrown in the deep end at work. Here’s my attempt at showing you that its really not that scary!

@blackllama

I use slack a lot, and I love making custom slack emojis

BUT, the requirement that the images are 128x128 means I repeatedly spend time in paint resizing pictures

Let’s enable my procrastination

Create an S3 bucket, and skip through the steps

Add a bucket policy

(This gives it rights to get and put objects)

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"AddPerm",
      "Effect":"Allow",
      "Principal": "*",
      "Action":["s3:GetObject", "s3:PutObject"],
      "Resource":["arn:aws:s3:::BUCKET NAME HERE/*"]
    }
  ]
}
```

NOTE: Don’t do this for productionised apps!

This creates a publically accessible bucket. You would not usually do this - and instead put it behind IAM (Identity and Access Management), but it serves our purposes for now.

Add static website hosting to the bucket and note the url

Create a Lambda function with a custom role

(This gives the function rights to put objects in the S3 bucket)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::BUCKET NAME HERE/*"
    }
  ]
}
```

Add an API Gateway trigger

API Name = hello-levels-api

Deployment stage = prod

Security = open

Note the host

Lambda Code

The code can be found at https://github.com/blackllama/LevelsConf2018

Upload the zip file

Add environment variables

BUCKET = Bucket name

URL = s3 hosting site that you noted earlier

S3 bucket redirection

```
<RoutingRules>
    <RoutingRule>
        <Condition>
            <KeyPrefixEquals/>
            <HttpErrorCodeReturnedEquals>404</HttpErrorCodeReturnedEquals>
        </Condition>
        <Redirect>
            <Protocol>https</Protocol>
            <HostName>ENTER API GATEWAY URL HERE</HostName>
            <ReplaceKeyPrefixWith>prod/resize?key=</ReplaceKeyPrefixWith>
            <HttpRedirectCode>307</HttpRedirectCode>
        </Redirect>
    </RoutingRule>
</RoutingRules>
```

Let’s see if it worked!

Testing!

- Upload an image to S3
- Navigate to url (BucketStaticHost/Filename)
  `http://hellolevels.s3-website-ap-southeast-2.amazonaws.com/happyllama.jpg`
- Its a big image, not slack friendly!
- Configure test event on lambda
  - API Gateway proxy event type
  - Modify the query parameters to be `key: filename`
- Test, and navigate to url returned in the response
- Tada! Slackified image! (and you can retrieve the file from s3)

Thanks!

@blackllama