A quick intro to serverless for web developers

James Beswick, AWS Serverless

November 16, 2020

aws

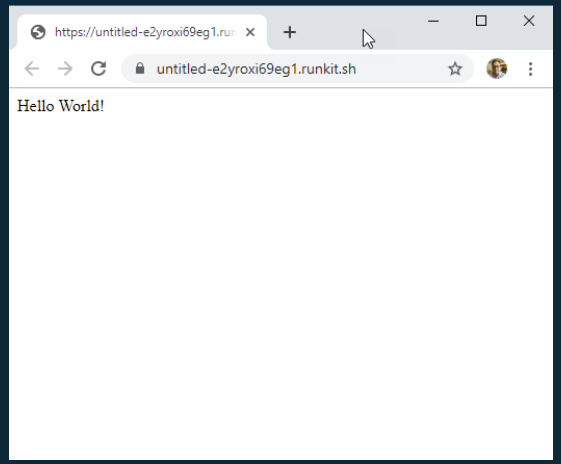
About me

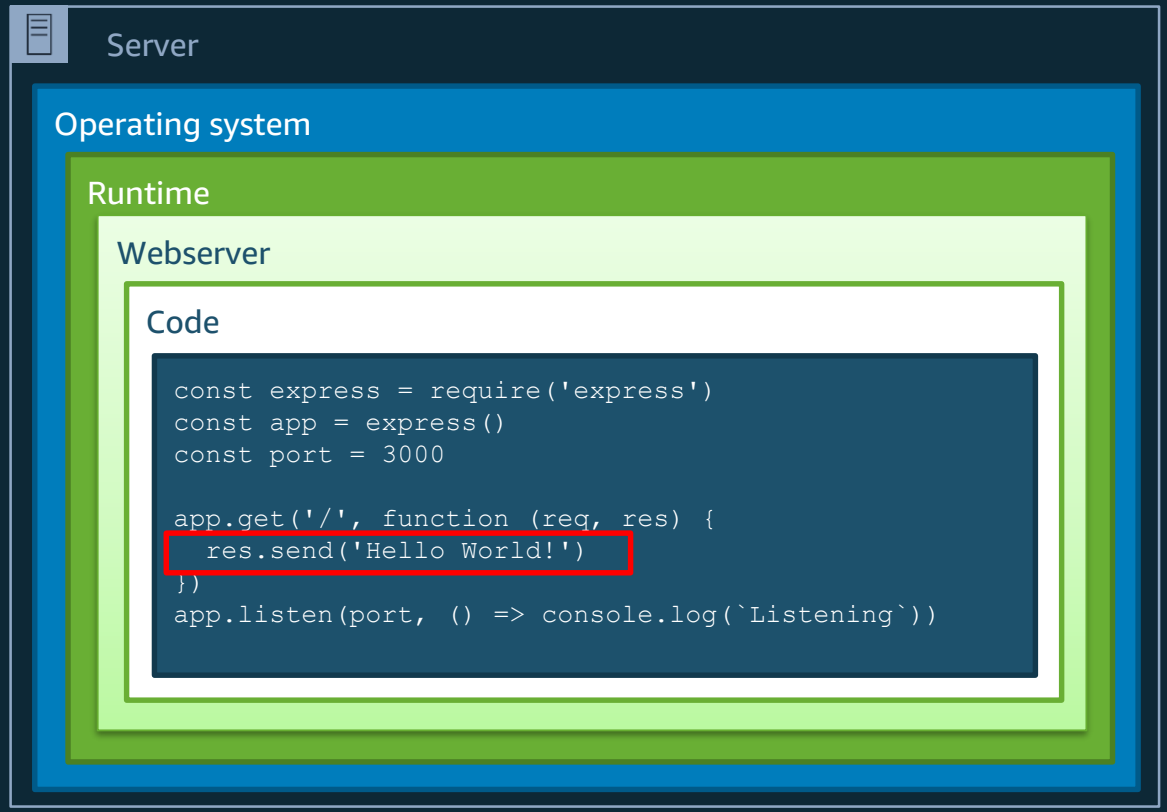
- James Beswick
  - Email: jbeswick@amazon.com
  - Twitter: @jbesw
- Developer Advocate – AWS Serverless
- Self-confessed serverless geek
- Previously:
  - Software Developer and Product Manager
  - Multiple start-up tech guy
  - Rackspace, USAA, Morgan Stanley, J P Morgan
  - Enjoys comedy, travel, coffee and theme parks…

aws

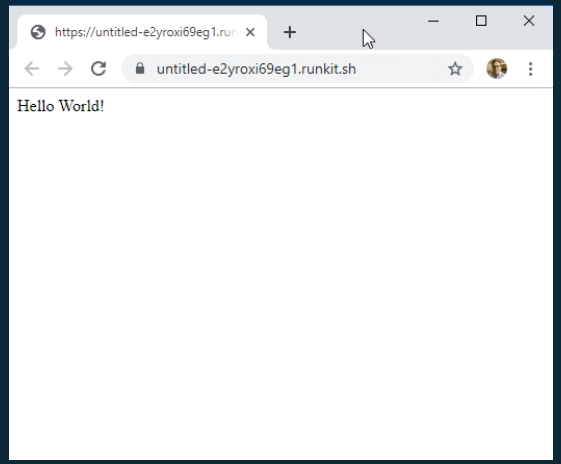
Hello World

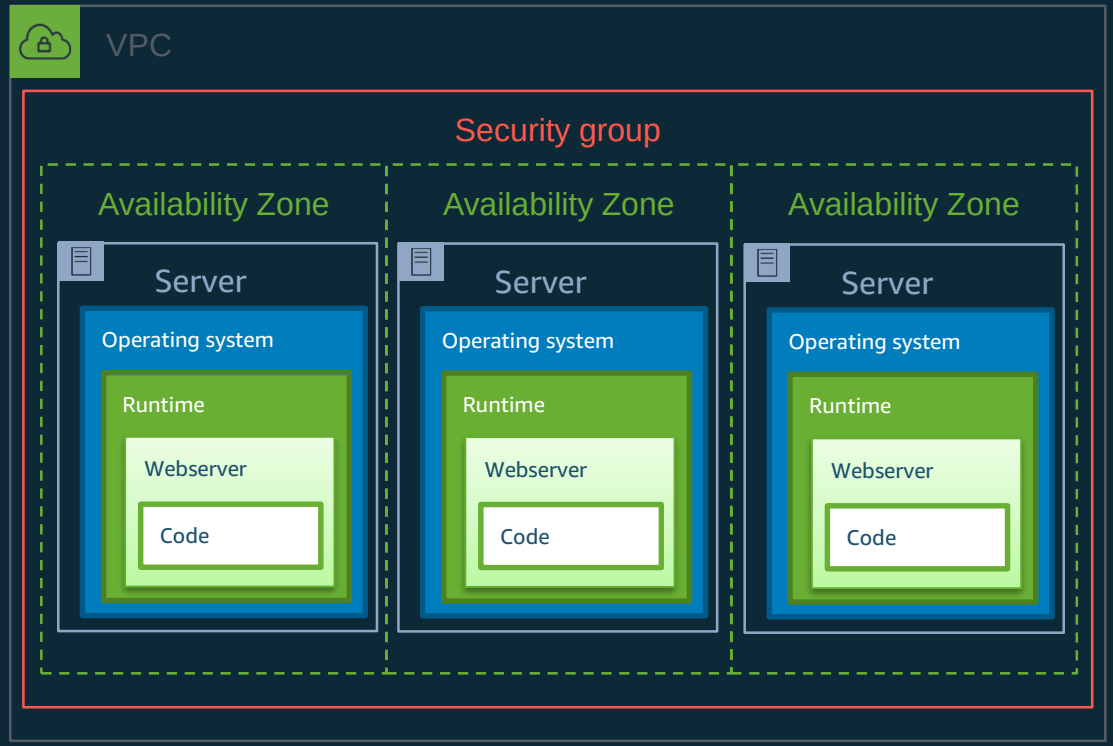
A “Hello World!” application





A “Hello World!” application - at scale…



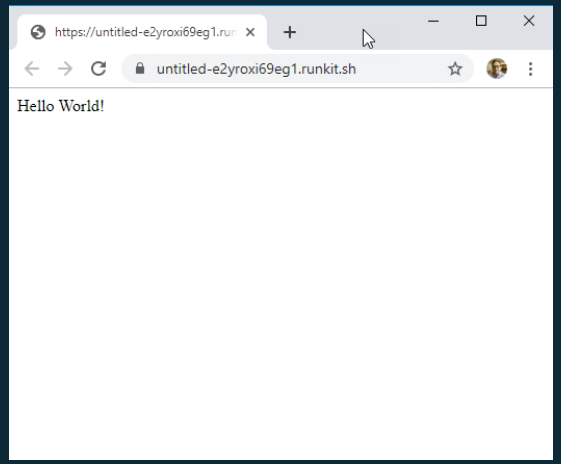


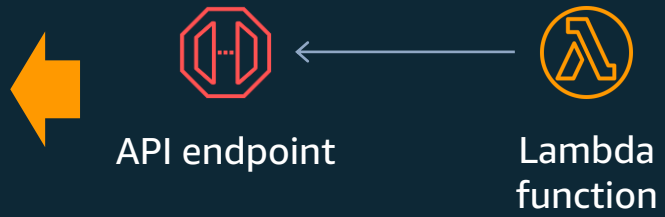
But I just want to run…

```
res.send('Hello World!')
```


aws

“Hello World!” – The serverless way





Code

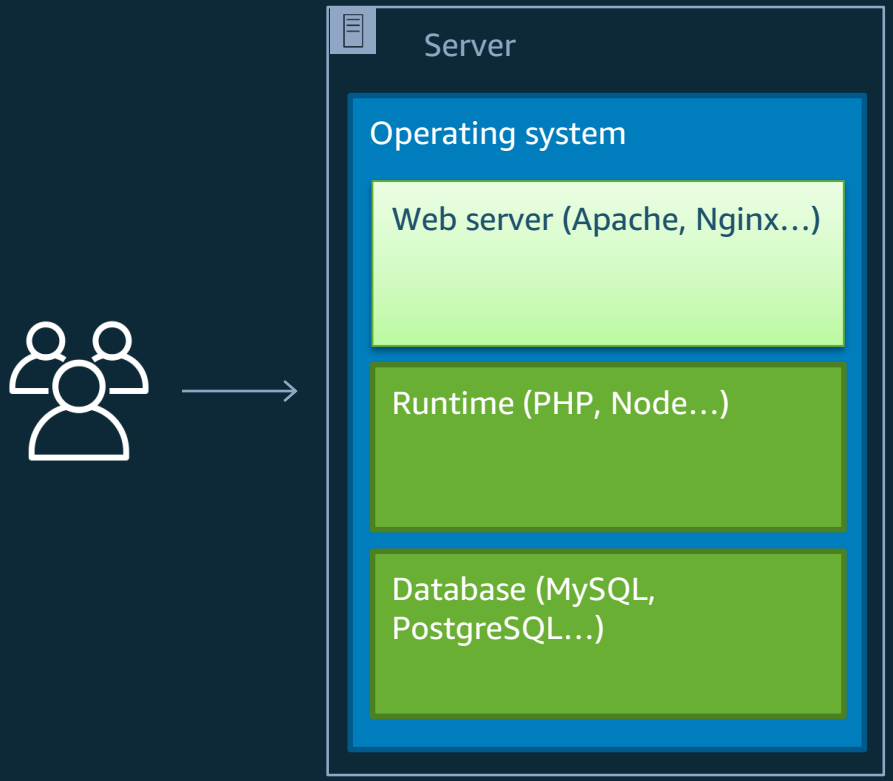
```
exports.handler = async (event) => {
  return "Hello World!“
}
```


aws

Focus on your business logic, not the infrastructure.

It’s about simplicity.

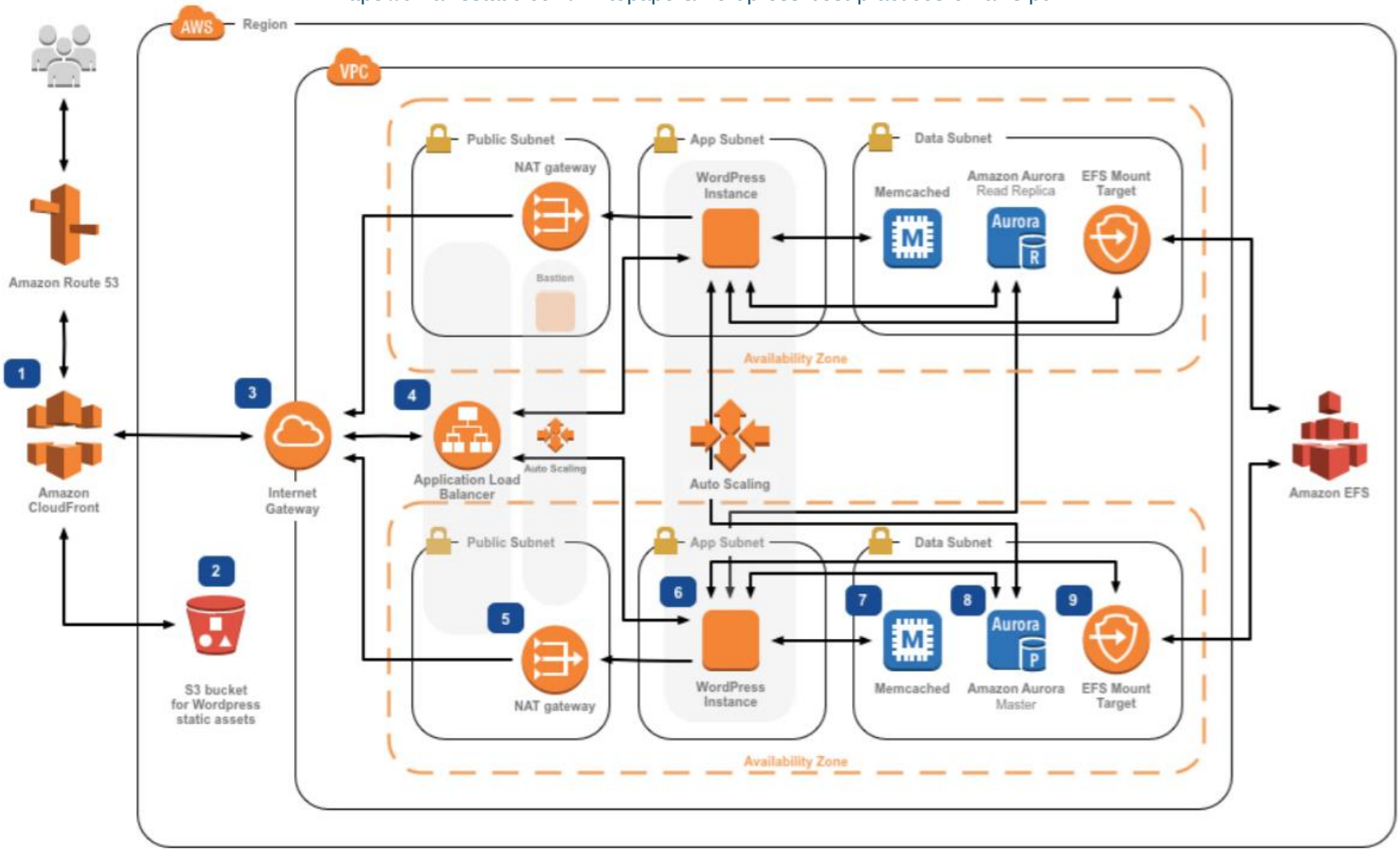
A classic web app stack



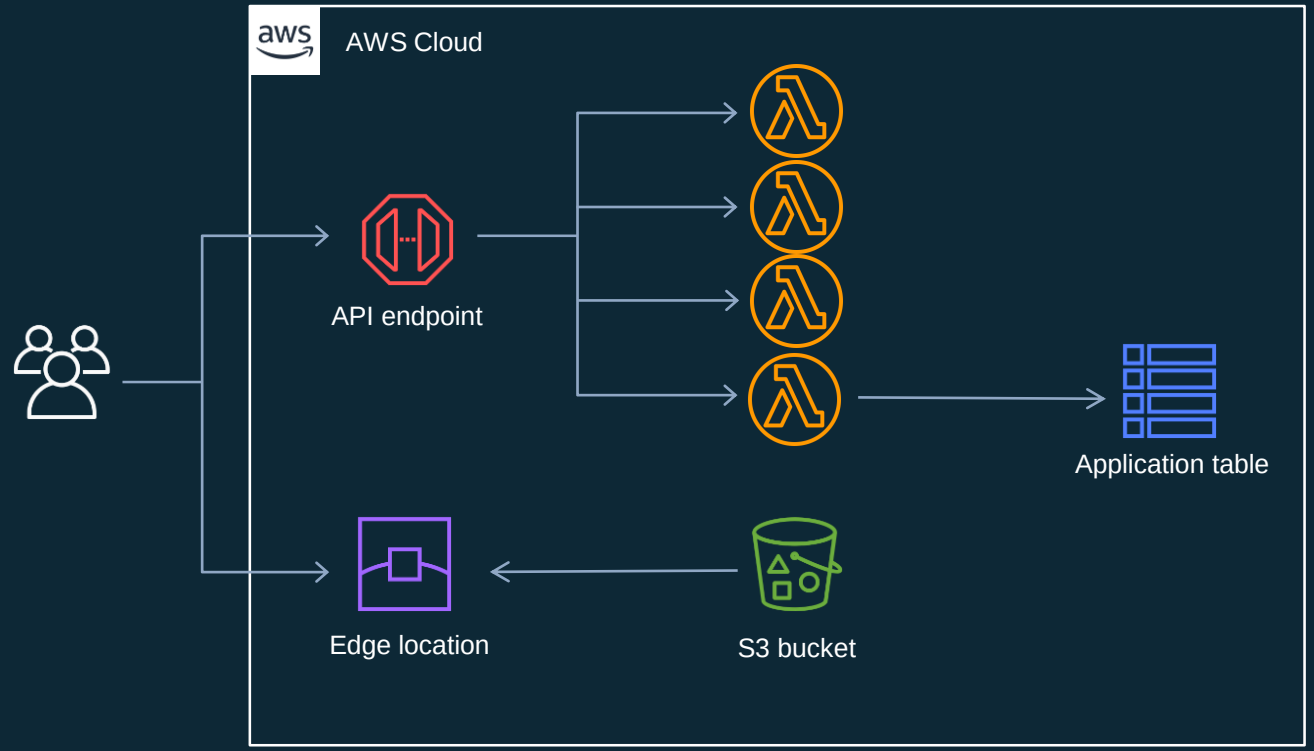
Things to consider:

- Vertical scaling
- Horizontal scaling
- Load balancing
- Availability
- Security
- Maintenance
- Monitoring
- And… actual features

https://d1.awsstatic.com/whitepapers/wordpress-best-practices-on-aws.pdf



Serverless web app architecture



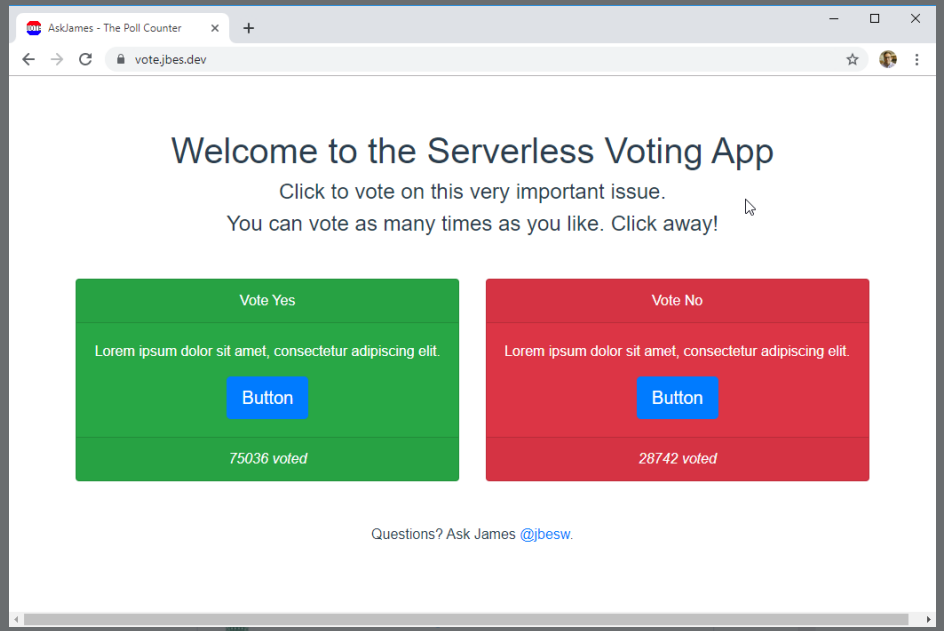
The voting website

Build a voting website that doesn't crash under load (in under an hour)

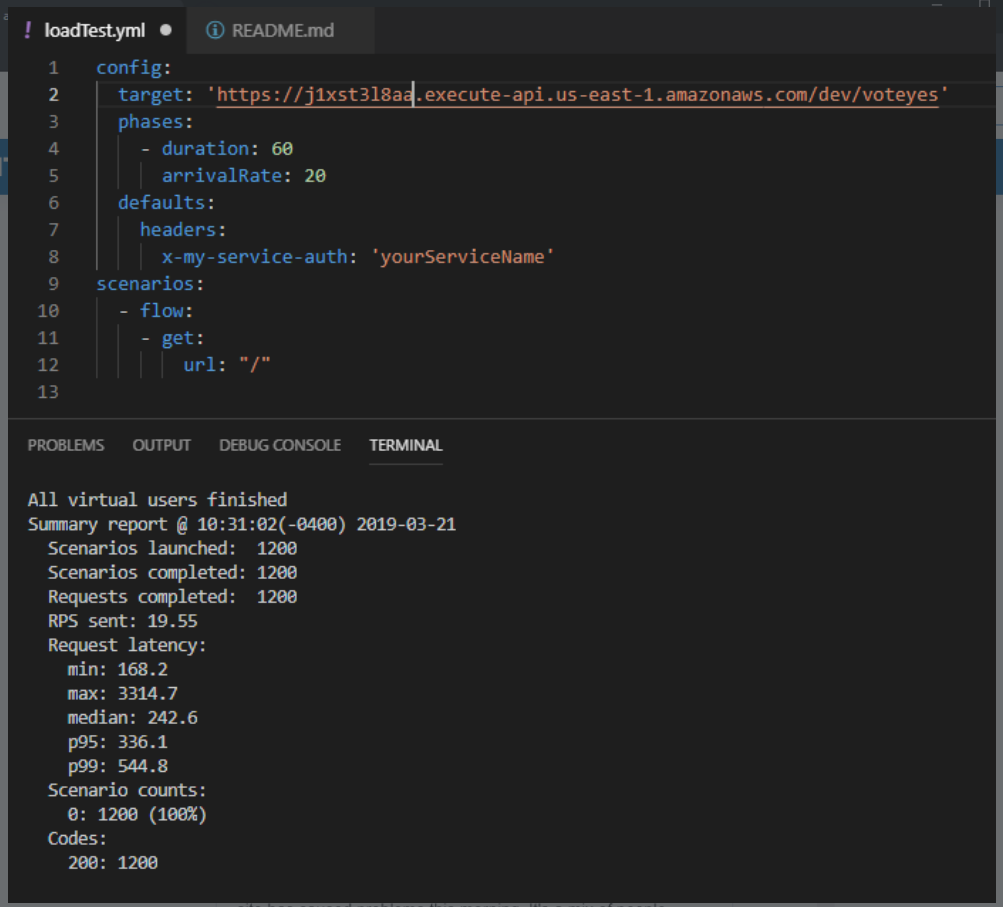
James Beswick Follow
Mar 21 · 4 min read

Spiky traffic? Unpredictable load? Sounds like a job for serverless.

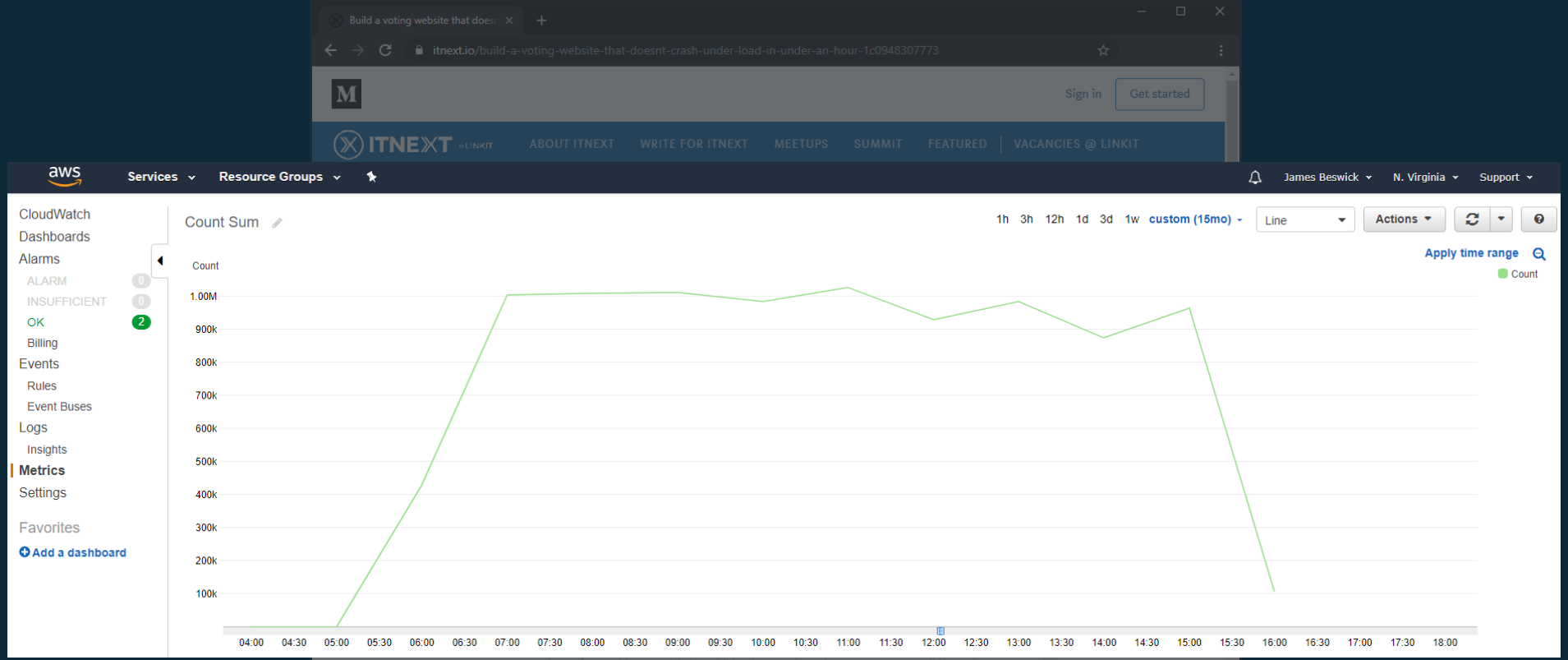
Petition websites can experience the extremes of demand — when a popular motion is set before the public, hundreds of thousands of people can appear from nowhere. Worse yet, many stay after voting, F5-ing their browser to see the new tallies, adding even more load onto the already-smoking servers:



https://vote.jbes.dev



https://vote.jbes.dev



https://vote.jbes.dev

Things I didn’t have to do…

Managing load balancers. Managing SSH keys. Monitoring hardware errors. Migrating instances. Migrating data centers. Dynamic volume provisioning. Orchestrating containers. Rotating root admin keys. Configuring subnets and VPCs. Handling datacenter republication. Renewing certificates. Monitoring disk usage. Resizing clusters. Auto-scaling instances. Restarting servers. sudo yum update. Defining network policies. Scheduling operating system updates. Troubleshooting networking. Configuring IP tables. Volume snapshots. Running server healthchecks.

aws

Agility

The time when you know what to build…

Beginning of project?

End of project?

Along the way?

Non-stop changes?

Why does serverless help with agility?

Less code, focused on business logic

Microservices are more agile

Services for common tasks

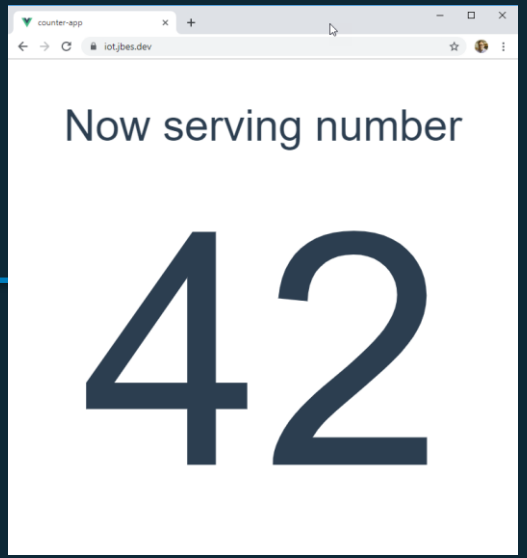
CI/CD, versioning and automation

aws

What a serverless application looks like

What a serverless application looks like

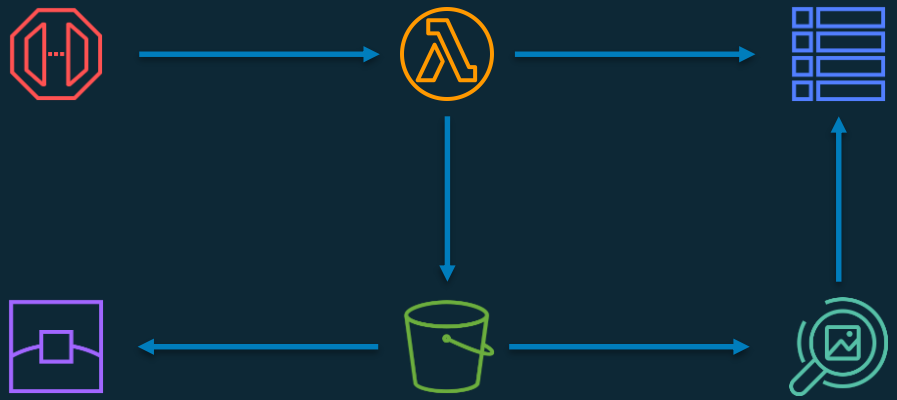
```
exports.handler = async (event) => {
  const payload = event.clickType
  const params = {
    topic: 'turnstyle-counter',
    payload,
    qos: 0
  }
  try {
    const result = await iot.publish
                        (params).promise()
    return { statusCode: 200 }

  } catch (err) {
    console.error('Error! ', err)
    return { statusCode: 501 }
  }
}
```



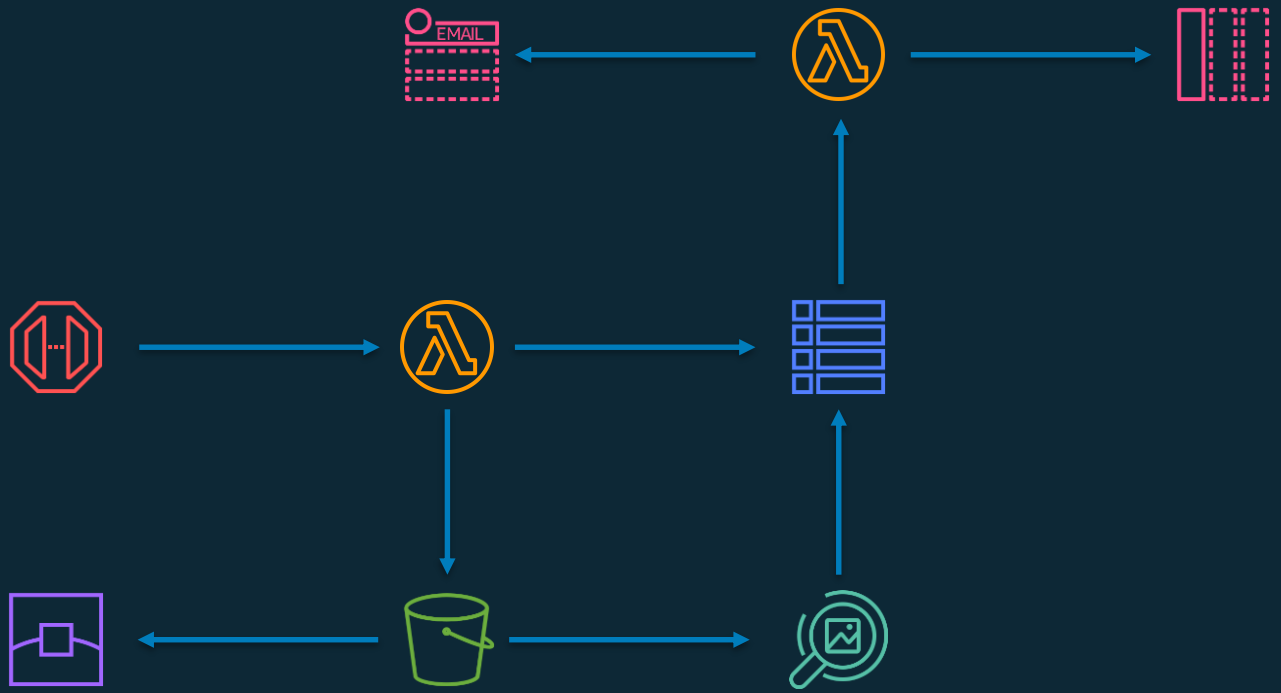
What a serverless application looks like

Serverless applications

Serverless applications



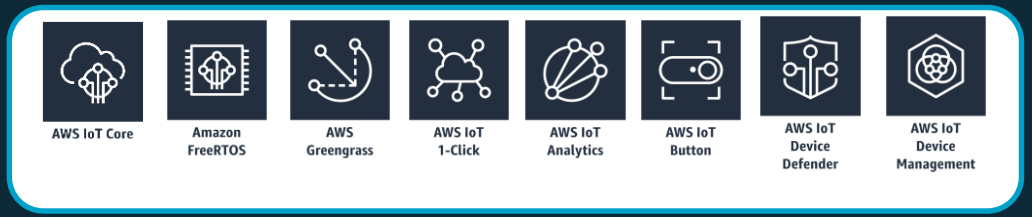
Serverless applications



Application services

Machine Learning

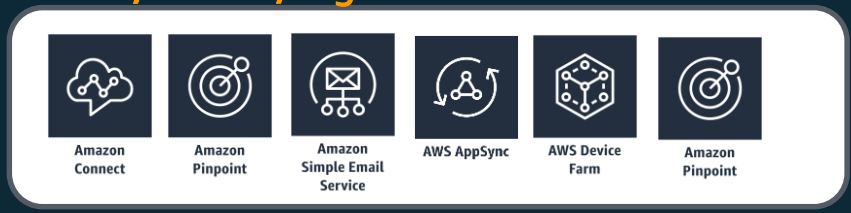
Internet of Things



Analytics

Media

Web/Mobile/Digital



aws

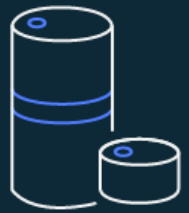
Common serverless application types

Web applications

Backends

Data processing

Chatbots



Amazon Alexa

IT Automation

aws

Fundamental changes to development

Agility

Fundamental changes to development

Agility

Scaling

Fundamental changes to development

Agility

Scaling

Cost

aws

Fundamental changes to development

Agility

Scaling

Cost

Environment

aws

Thank you!

James Beswick, AWS Serverless
@jbesw