Tampere University

Lecture 9; wrapping up cloud native
Kari Systä, 02.11.2021

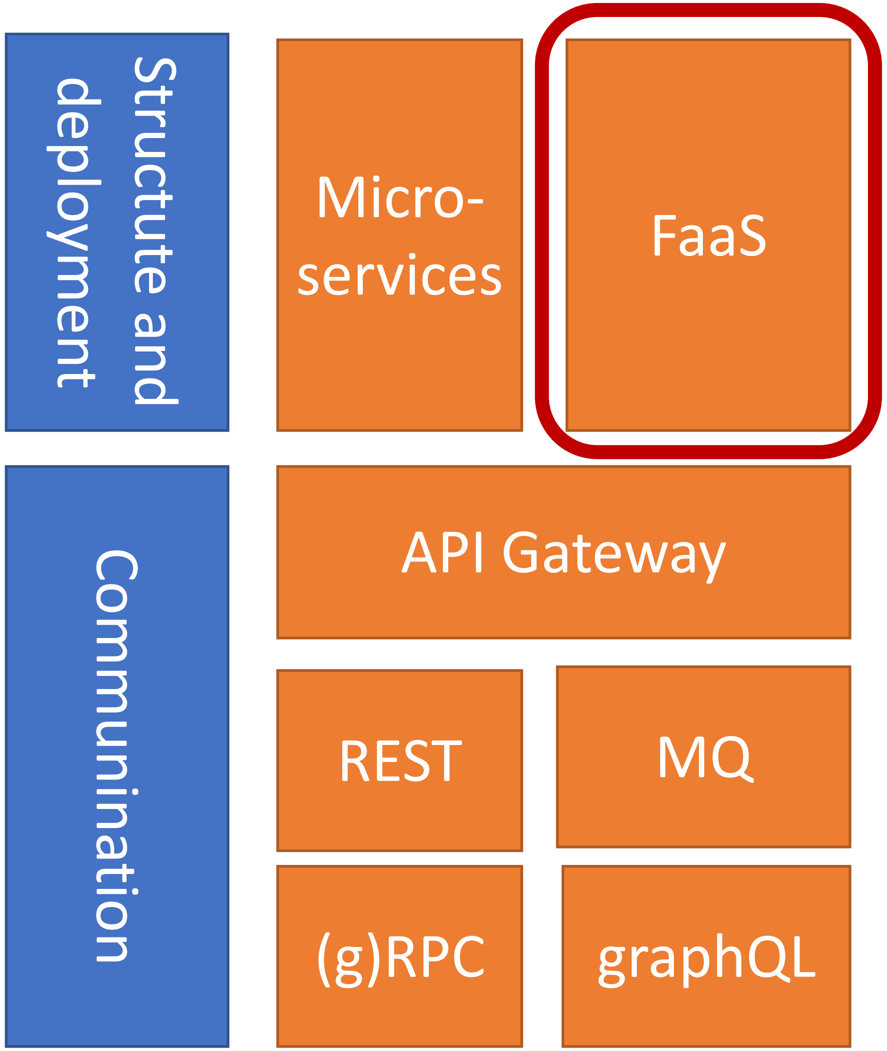
Course matters

- Our plan is to publish the project next week
  - Long time window for returning
- I’m still waiting opinions about the exam
- About Ansible exercise
  - Use of ”uptime” was a bad choice, but I do not want to change

    Create an Ansible playbook that has two tasks (plays)

    - Ensure (not just “check”) that the image has the latest version of git version management system
    - Queries the uptime (linux command uptime) of target host

    “Check” is accepted.
- Last lecture traditional?


Structute and deployment
Micro-services
FaaS
Communination
API Gateway
REST
MQ
(g)RPC
graphQL

Tampereen yliopisto
Tampere University

Function as a service/ serverless computing

Do you really want to keep
your containers running all the time
if you need to pay for it?

Do you really want to operate
and maintain your containers –
your developers could also
do something else.

Serverless computing

Baldini et all: Serverless Computing:

Current Trends and Open Problems, Research Advanges in Cloud Computing, Springer, 2017.

A cloud-native platform

for

- short-running, stateless computation
- event driven applications

which

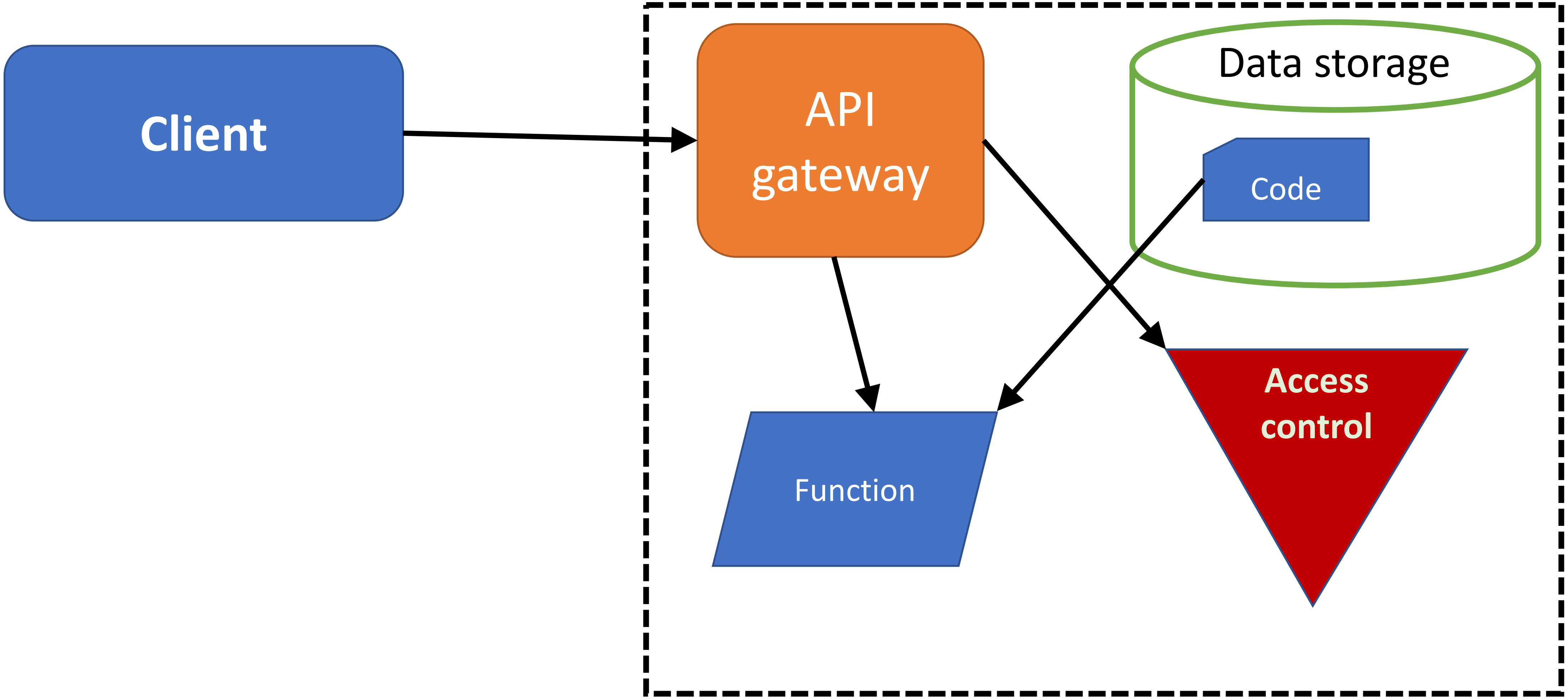
- scale up and down instantly and automatically

and

- charge for actual usage and high granulatity https://medium.com/@BoweiHan/an-introduction-to-serverless-and-faas-functions-as-a-service-fb5cec0417b2

”… you can simply upload modular chunks of functionality into the cloud that are executed independently.
Imagine the possibilities! Instead of scaling a monolithic REST server to handle potential load, you can now split the server into a bunch of functions which can be scaled automatically and independently.”

Function as a service?



A simple example from https://www.scalyr.com/blog/simple-detailed-introduction-google-cloud-functions/

- Package.json

```
{ "name": "my-first-function", "version": "0.0.1" }
```

- Code

```
exports.helloWorld = (req, res) => {
    let message = req.query.message ||
        req.body.message || 'Hello World!’;
    res.status(200).send(message);
};
```

- Deploy with

```
gcloud functions deploy my-first-function --trigger-http \\
--runtime nodejs8 --entry-point=helloWorld
```

- Use as

```
http://<location>/my-first-function?message=BAM
```

Tampereen yliopisto
Tampere University

A simple example from https://www.scalyr.com/blog/simple-detailed-introduction-google-cloud-functions/

- Package.json

```
{ "name": "my-first-function", "version": "0.0.1" }
```

- Code

```
exports.helloWorld = (req, res) => {
    let message = req.query.message ||
        req.body.message || 'Hello World!';
    res.status(200).send(message);
};
```

- Deploy with

```
gcloud functions deploy my-first-function --trigger-http \\
--runtime nodejs8 --entry-point=helloWorld
```

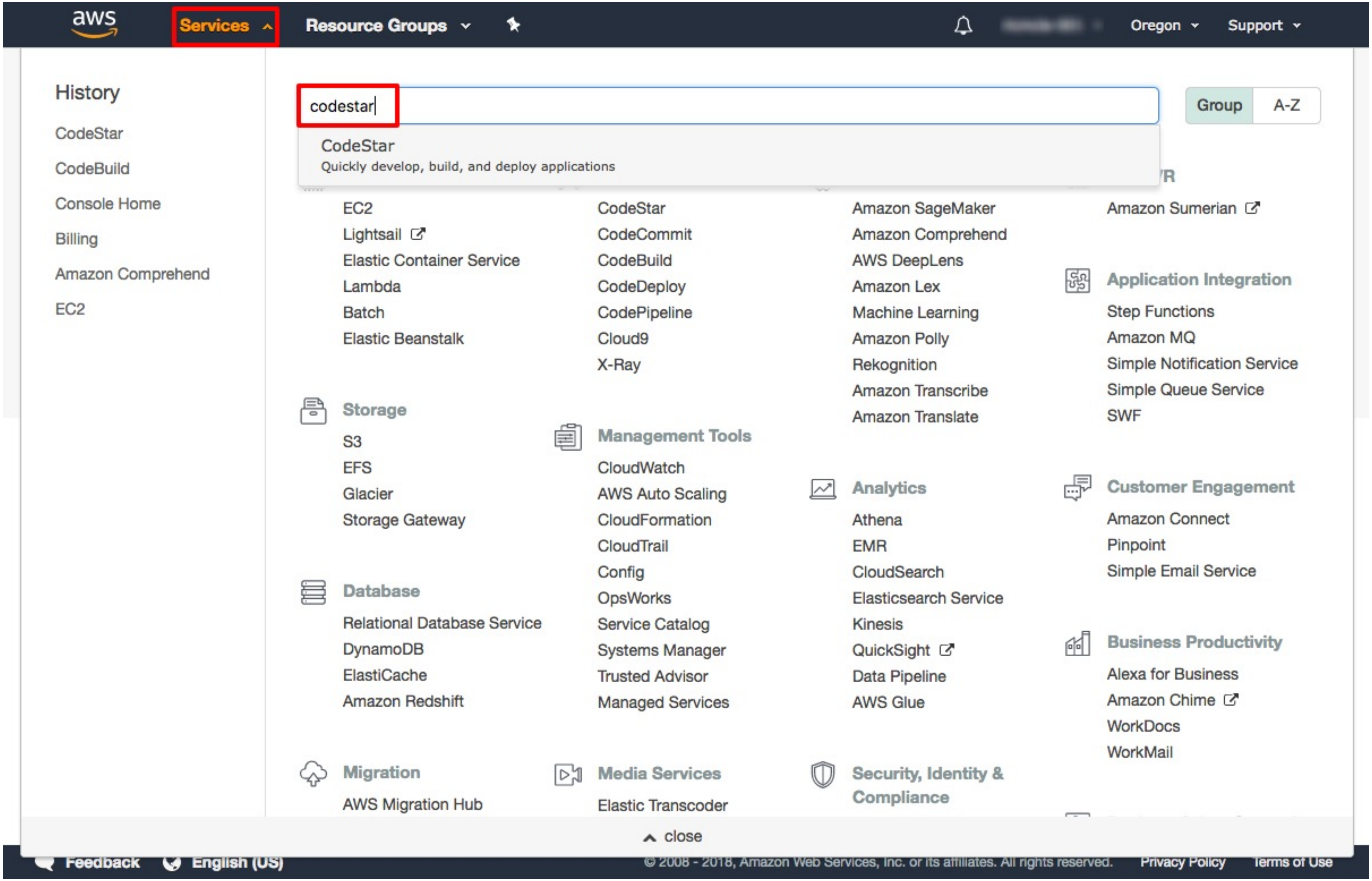
- Use as

```
http://<location>/my-first-function?message=BAM
```

Something to do with functional programming?

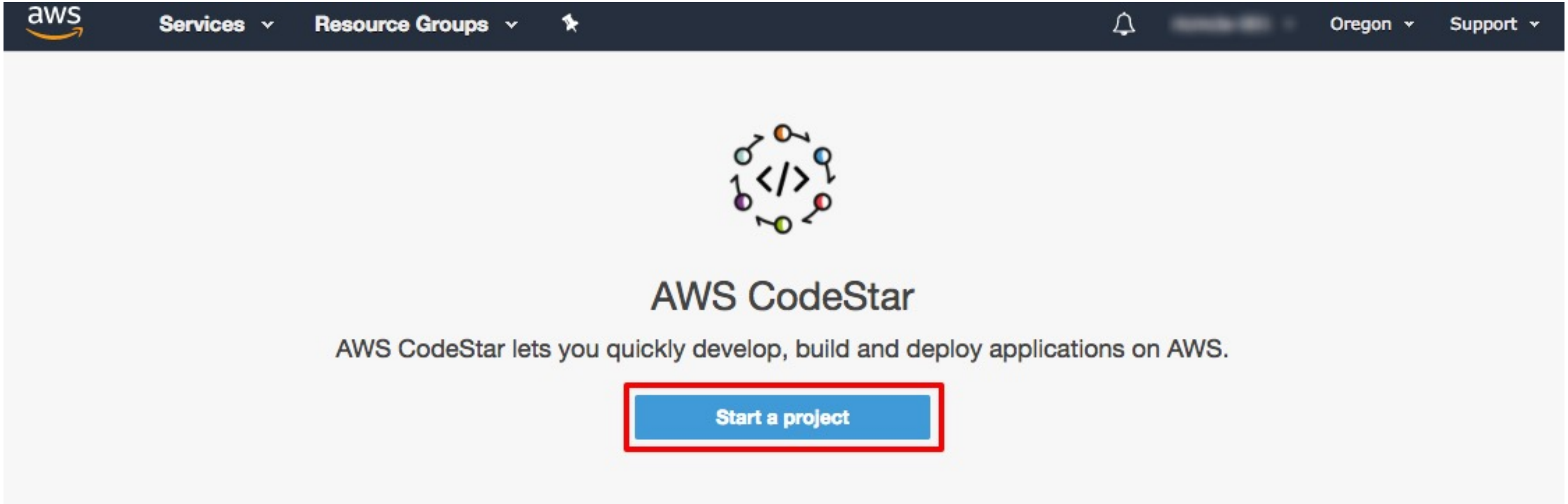
The actions with AWS Lambda

https://aws.amazon.com/getting-started/tutorials/build-serverless-app-codestar-cloud9



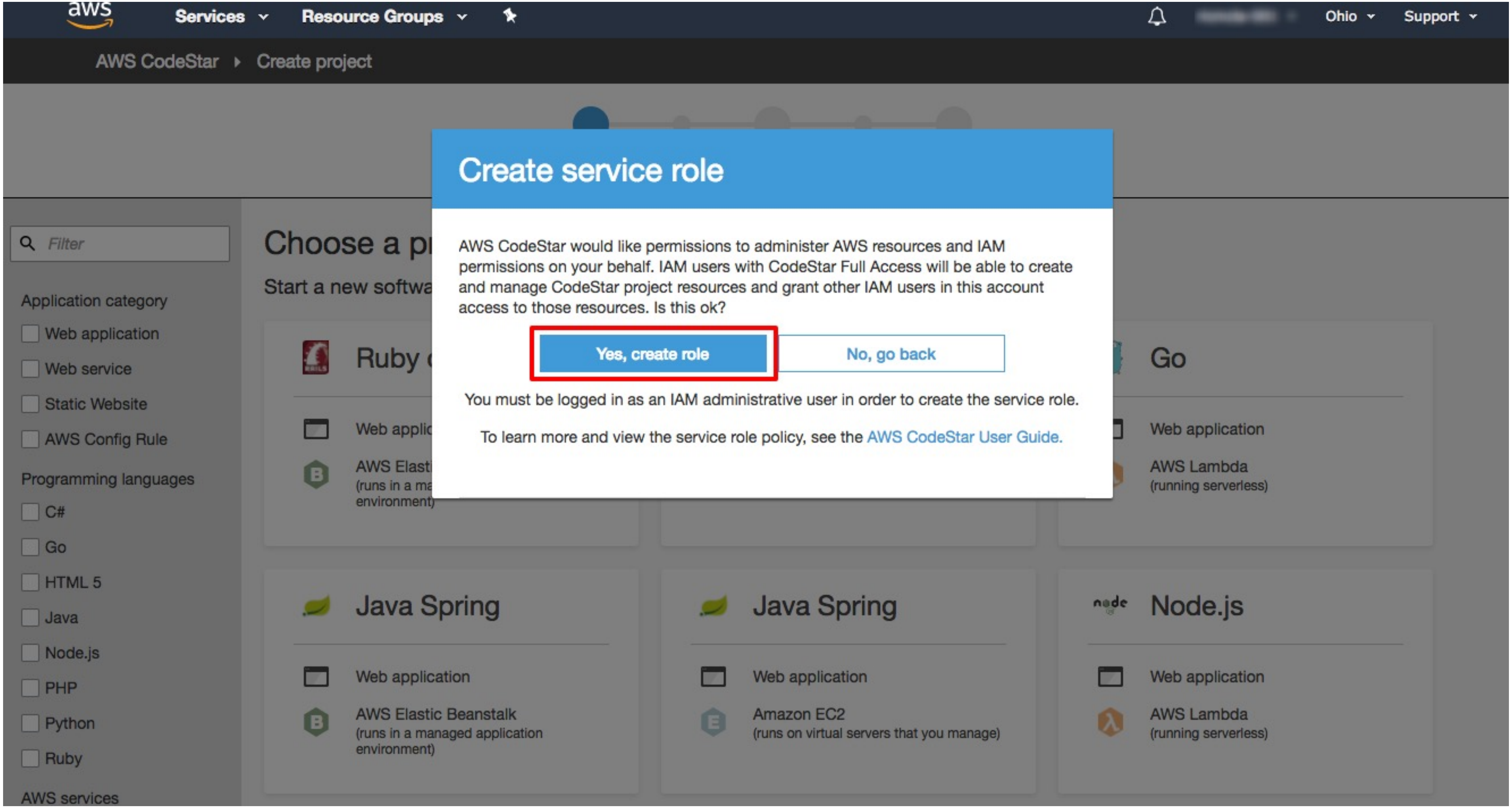
codestar

CodeStar
Quickly develop, build, and deploy applications



AWS CodeStar

AWS CodeStar lets you quickly develop, build and deploy applications on AWS.

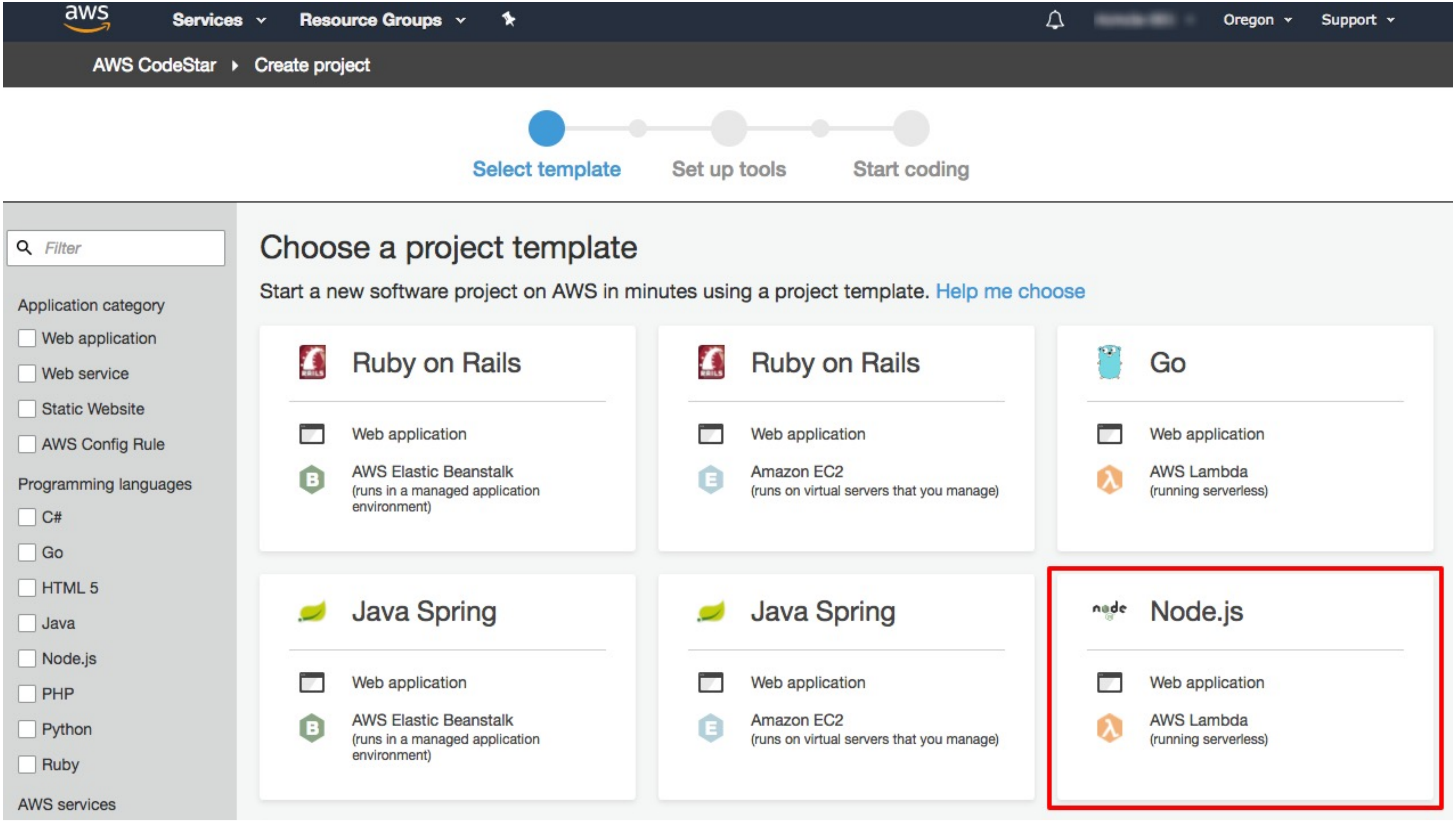


Create service role

AWS CodeStar would like permissions to administer AWS resources and IAM permissions on your behalf. IAM users with CodeStar Full Access will be able to create and manage CodeStar project resources and grant other IAM users in this account access to those resources. Is this ok?

You must be logged in as an IAM administrative user in order to create the service role.

To learn more and view the service role policy, see the AWS CodeStar User Guide.



Application category

- □ Web application
- □ Web service
- □ Static Website
- □ AWS Config Rule

Programming languages

- □ C#
- □ Go
- □ HTML 5
- □ Java
- □ Node.js
- □ PHP
- □ Python
- □ Ruby

AWS services

Choose a project template

Start a new software project on AWS in minutes using a project template. Help me choose

Ruby on Rails

Web application

AWS Elastic Beanstalk
(runs in a managed application environment)

Ruby on Rails

Web application

Amazon EC2
(runs on virtual servers that you manage)

Go

Web application

AWS Lambda
(running serverless)

Java Spring

Web application

AWS Elastic Beanstalk
(runs in a managed application environment)

Java Spring

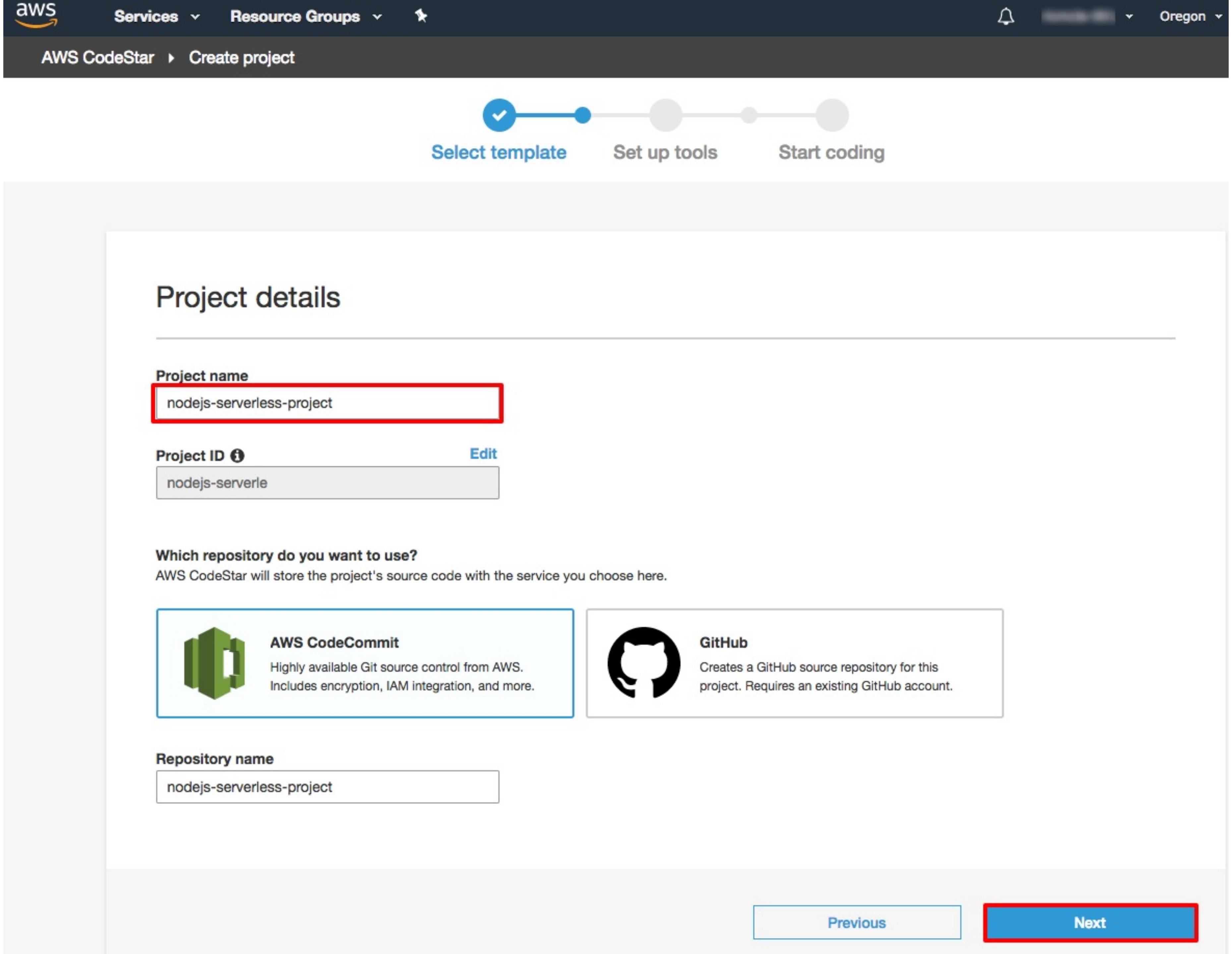
Web application

Amazon EC2
(runs on virtual servers that you manage)

Node.js

Web application

AWS Lambda
(running serverless)



Project details

Project name

nodejs-serverless-project

Project ID

nodejs-serverle

Which repository do you want to use?

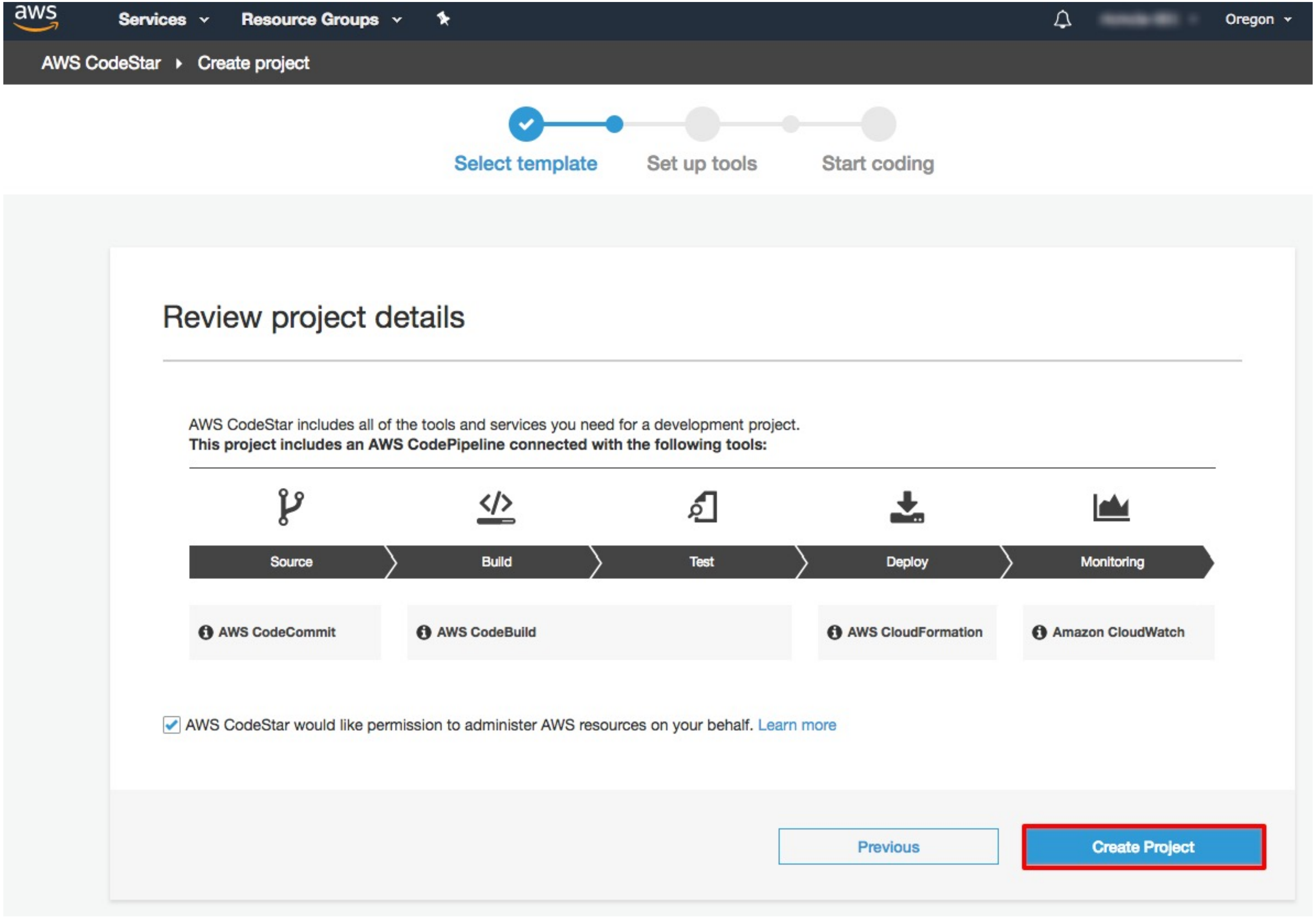
AWS CodeStar will store the project's source code with the service you choose here.

AWS CodeCommit
Highly available Git source control from AWS. Includes encryption, IAM integration, and more.

GitHub
Creates a GitHub source repository for this project. Requires an existing GitHub account.

Repository name

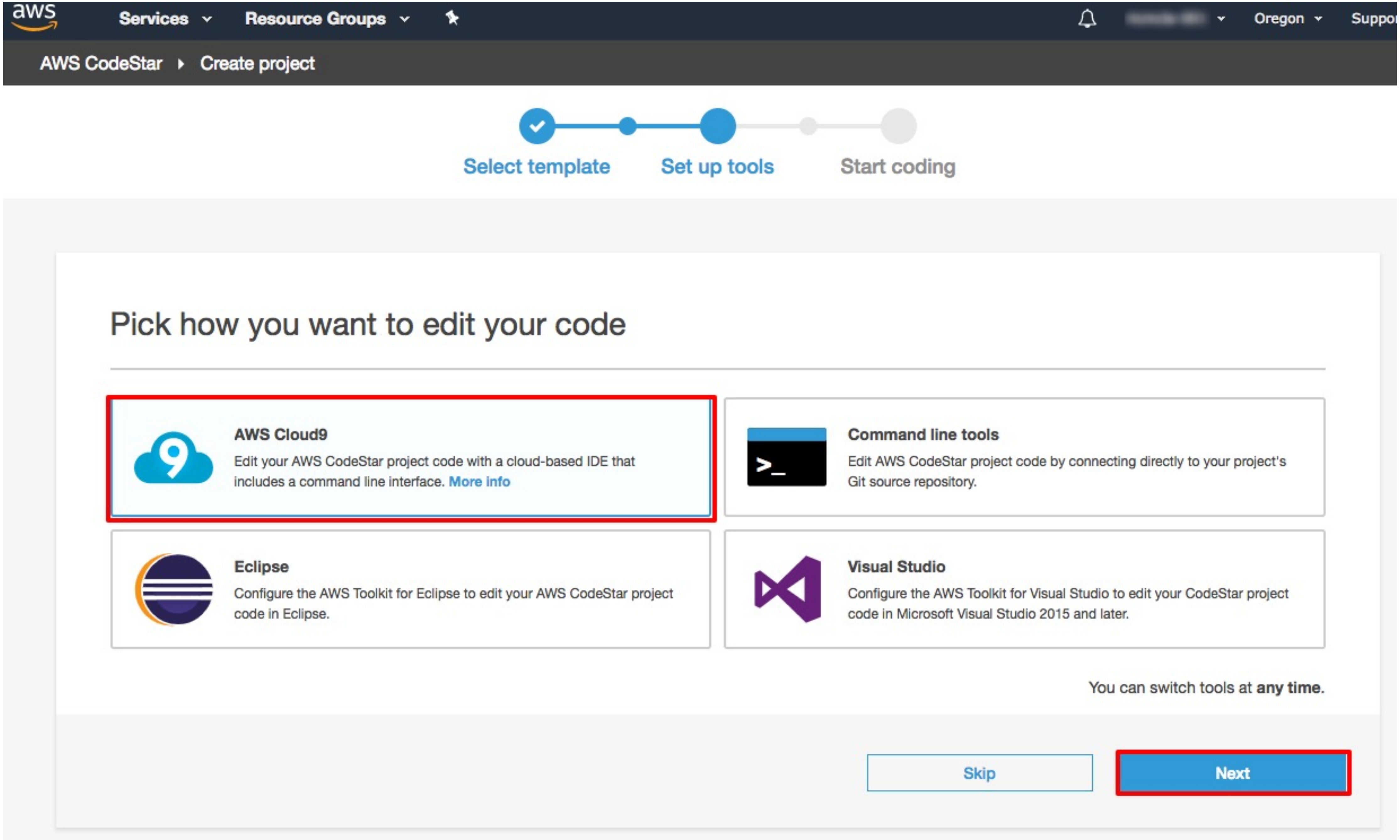
nodejs-serverless-project



Review project details

AWS CodeStar includes all of the tools and services you need for a development project.
This project includes an AWS CodePipeline connected with the following tools:

☑ AWS CodeStar would like permission to administer AWS resources on your behalf. Learn more



Pick how you want to edit your code

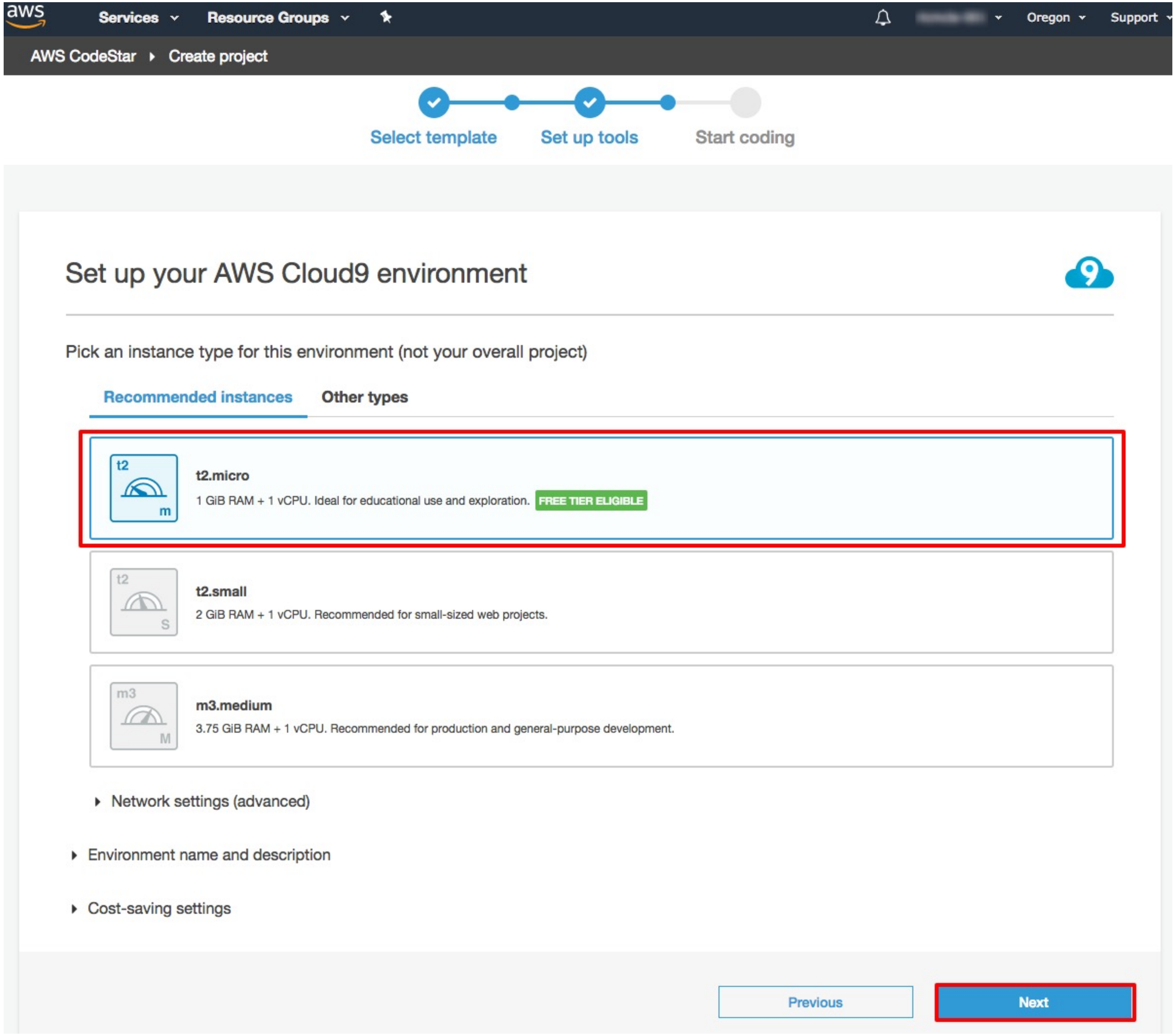
AWS Cloud9
Edit your AWS CodeStar project code with a cloud-based IDE that includes a command line interface. More info

Command line tools
Edit AWS CodeStar project code by connecting directly to your project's Git source repository.

Eclipse
Configure the AWS Toolkit for Eclipse to edit your AWS CodeStar project code in Eclipse.

Visual Studio
Configure the AWS Toolkit for Visual Studio to edit your CodeStar project code in Microsoft Visual Studio 2015 and later.

You can switch tools at **any time**.



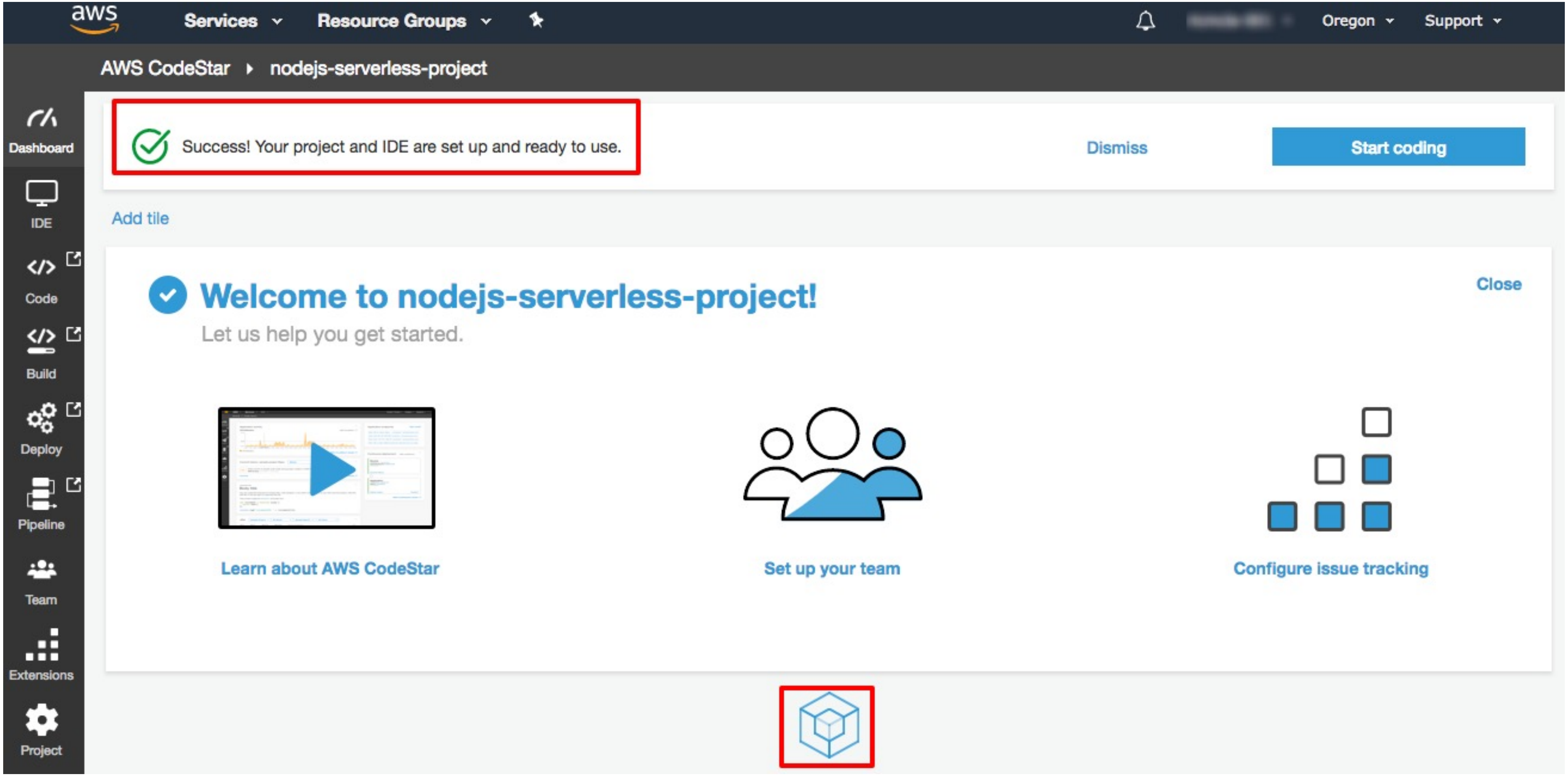
Set up your AWS Cloud9 environment

Pick an instance type for this environment (not your overall project)

t2.micro
1 GiB RAM + 1 vCPU. Ideal for educational use and exploration. FREE TIER ELIGIBLE

t2.small
2 GiB RAM + 1 vCPU. Recommended for small-sized web projects.

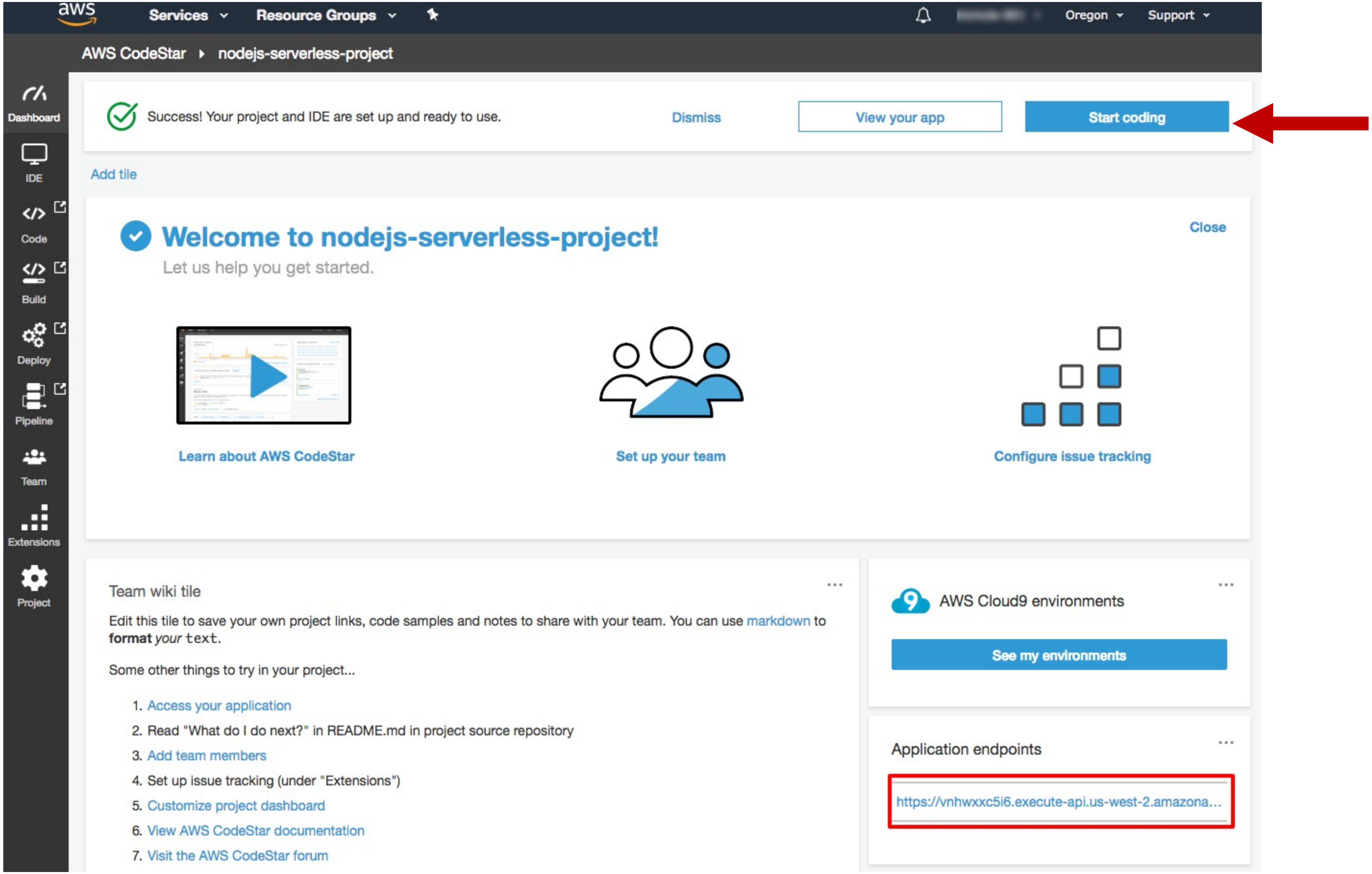
m3.medium
3.75 GiB RAM + 1 vCPU. Recommended for production and general-purpose development.



Success! Your project and IDE are set up and ready to use.

Welcome to nodejs-serverless-project!

Let us help you get started.



Success! Your project and IDE are set up and ready to use.

Welcome to nodejs-serverless-project!

Let us help you get started.

Team wiki tile

Edit this tile to save your own project links, code samples and notes to share with your team. You can use markdown to **format** *your* `text`.

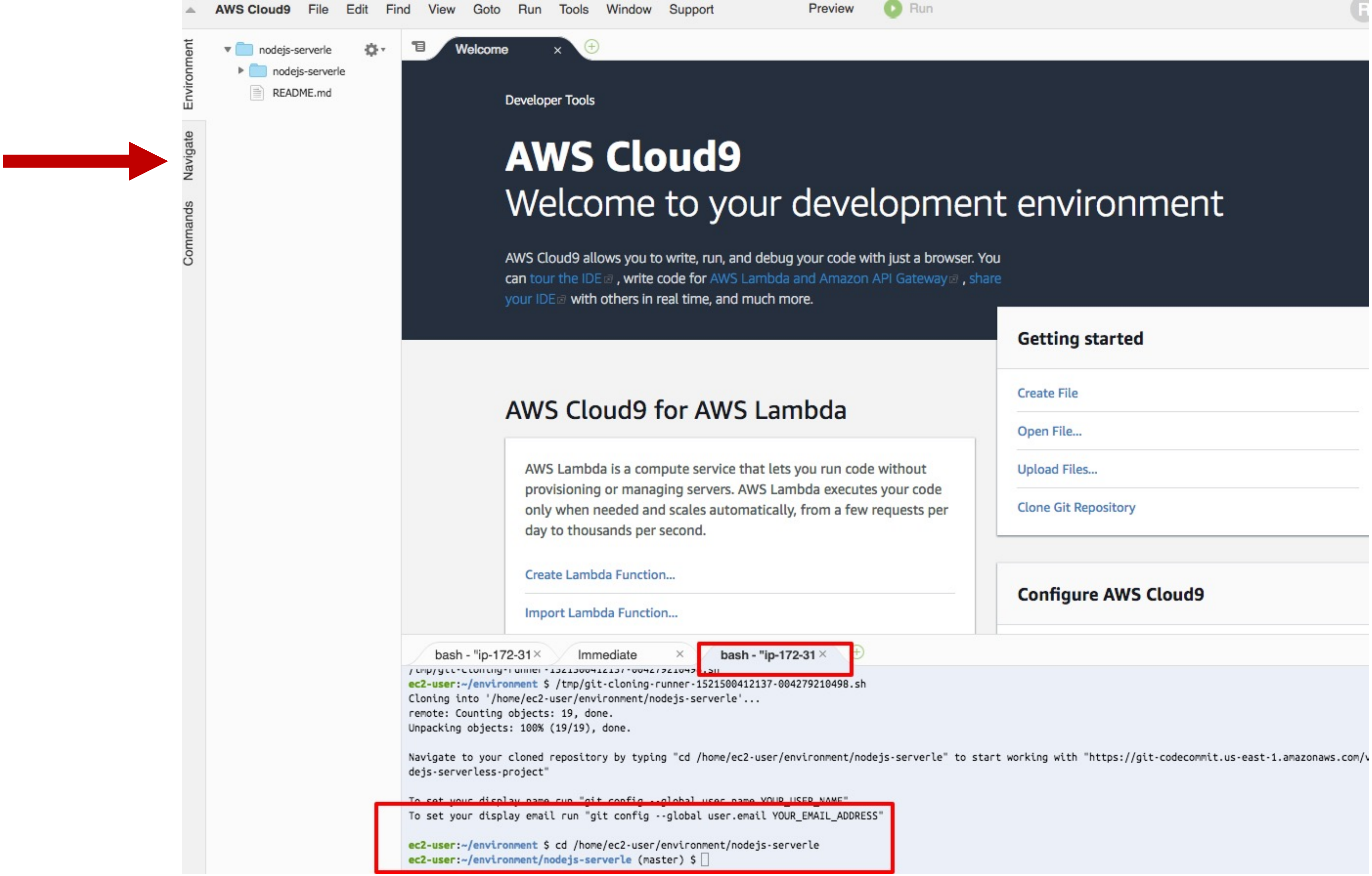
Some other things to try in your project...

1. Access your application
2. Read "What do I do next?" in README.md in project source repository
3. Add team members
4. Set up issue tracking (under "Extensions")
5. Customize project dashboard
6. View AWS CodeStar documentation
7. Visit the AWS CodeStar forum

AWS Cloud9 environments

Application endpoints

https://vnhwxxc5i6.execute-api.us-west-2.amazona...



Developer Tools

AWS Cloud9

Welcome to your development environment

AWS Cloud9 allows you to write, run, and debug your code with just a browser. You can tour the IDE, write code for AWS Lambda and Amazon API Gateway, share your IDE with others in real time, and much more.

AWS Cloud9 for AWS Lambda

AWS Lambda is a compute service that lets you run code without provisioning or managing servers. AWS Lambda executes your code only when needed and scales automatically, from a few requests per day to thousands per second.

Create Lambda Function...

Import Lambda Function...

Getting started

Create File

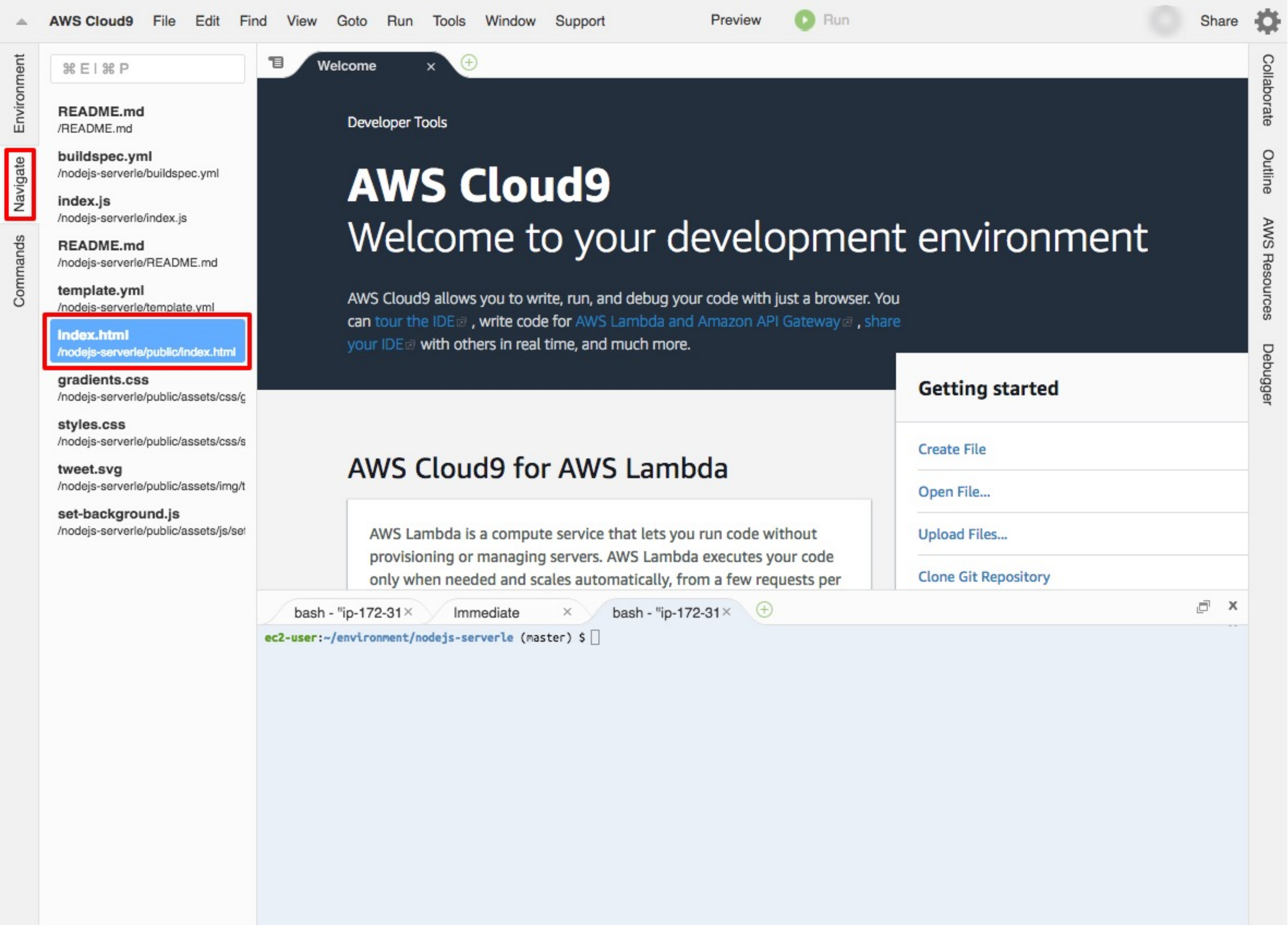
Open File...

Upload Files...

Clone Git Repository

Configure AWS Cloud9

```
/tmp/git-cloning-runner-1521500412137-004279210498.sh
ec2-user:~/environment $ /tmp/git-cloning-runner-1521500412137-004279210498.sh
Cloning into '/home/ec2-user/environment/nodejs-serverle'...
remote: Counting objects: 19, done.
Unpacking objects: 100% (19/19), done.

Navigate to your cloned repository by typing "cd /home/ec2-user/environment/nodejs-serverle" to start working with "https://git-codecommit.us-east-1.amazonaws.com/v
dejs-serverless-project"

To set your display name run "git config --global user.name YOUR_USER_NAME"
To set your display email run "git config --global user.email YOUR_EMAIL_ADDRESS"

ec2-user:~/environment $ cd /home/ec2-user/environment/nodejs-serverle
ec2-user:~/environment/nodejs-serverle (master) $ 
```



Developer Tools

AWS Cloud9

Welcome to your development environment

AWS Cloud9 allows you to write, run, and debug your code with just a browser. You can tour the IDE, write code for AWS Lambda and Amazon API Gateway, share your IDE with others in real time, and much more.

Getting started

Create File

Open File...

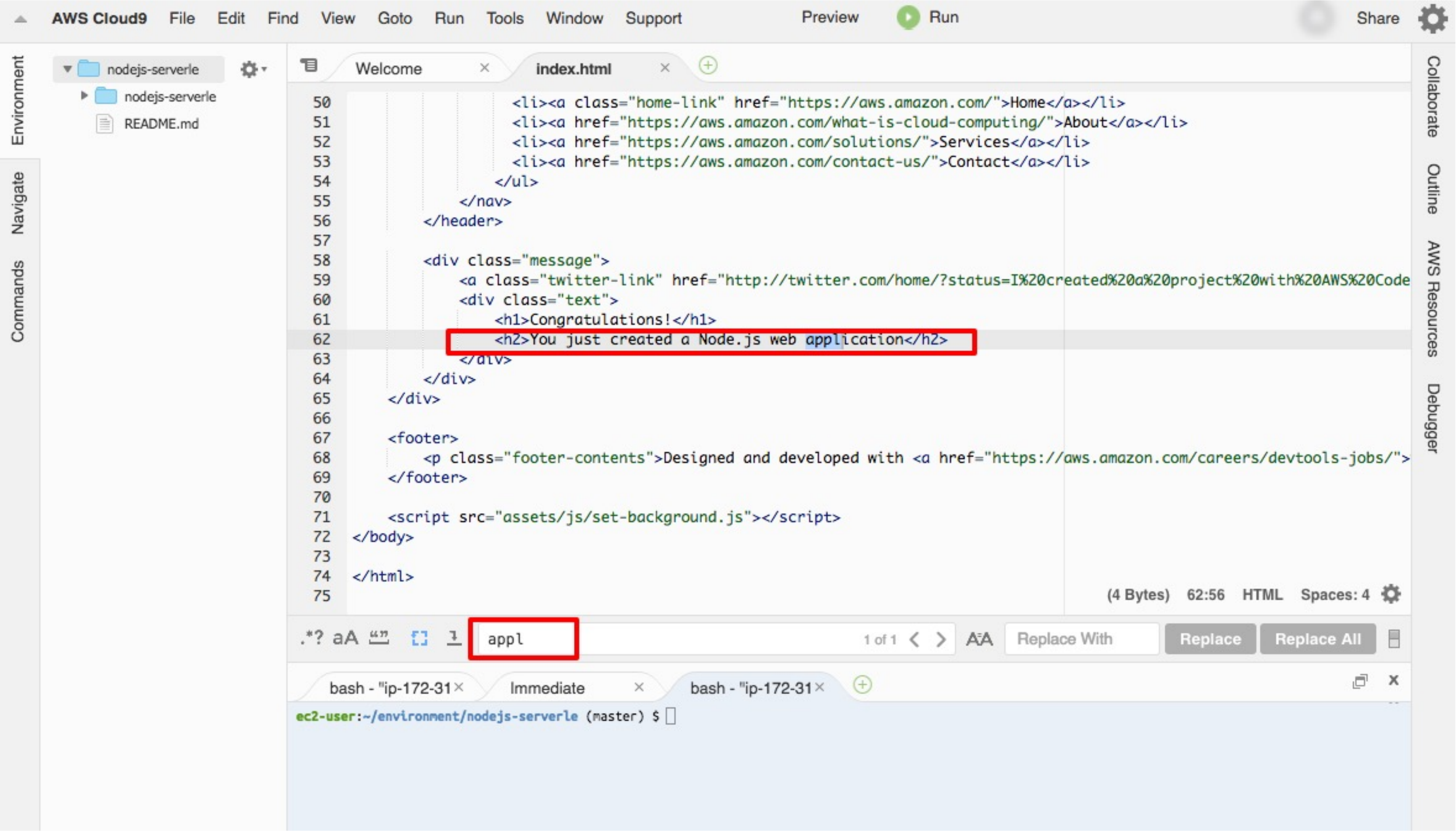
Upload Files...

Clone Git Repository

AWS Cloud9 for AWS Lambda

AWS Lambda is a compute service that lets you run code without provisioning or managing servers. AWS Lambda executes your code only when needed and scales automatically, from a few requests per

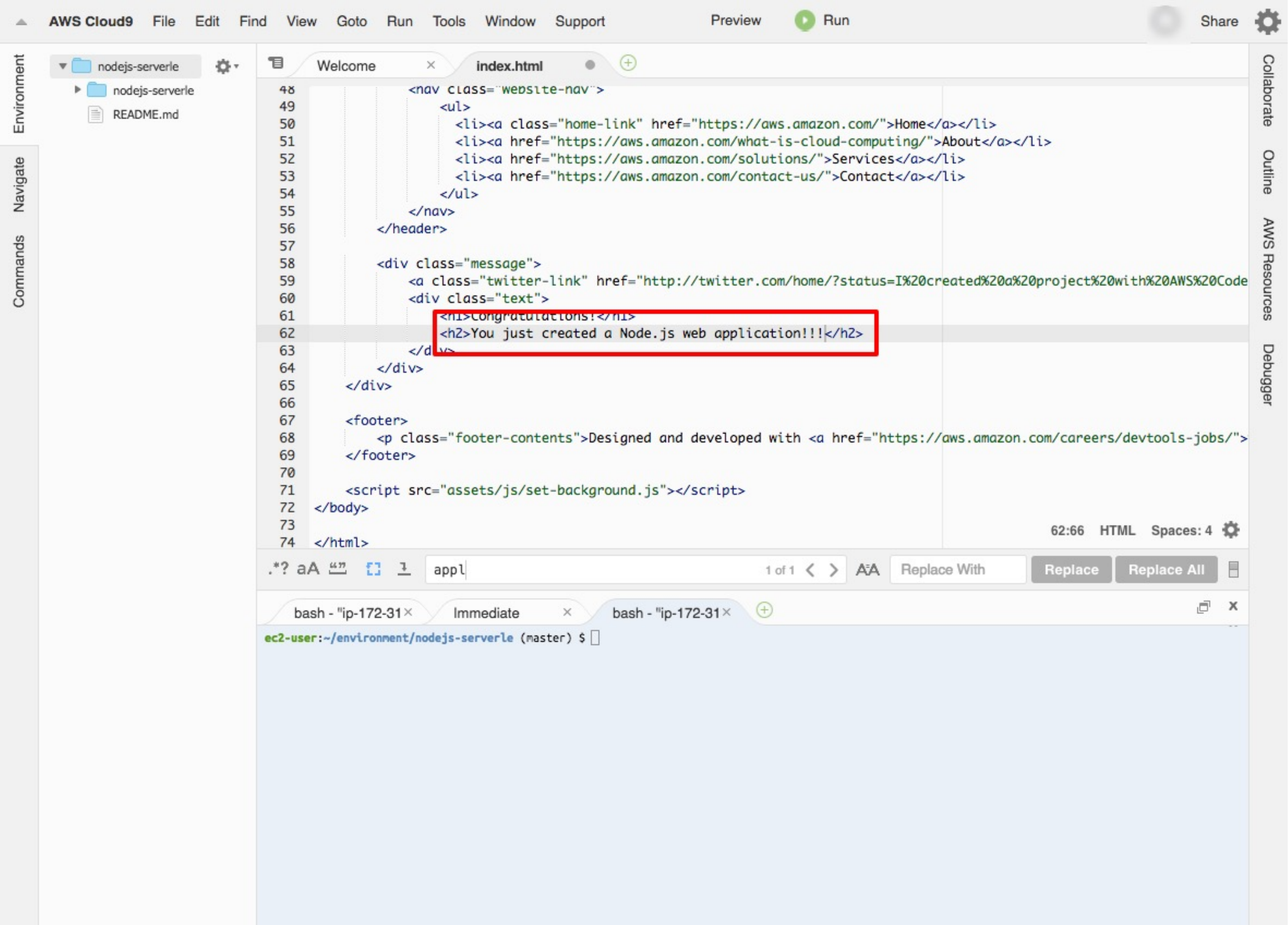
```
ec2-user:~/environment/nodejs-serverle (master) $
```



```
                    <li><a class="home-link" href="https://aws.amazon.com/">Home</a></li>
                    <li><a href="https://aws.amazon.com/what-is-cloud-computing/">About</a></li>
                    <li><a href="https://aws.amazon.com/solutions/">Services</a></li>
                    <li><a href="https://aws.amazon.com/contact-us/">Contact</a></li>
                </ul>
            </nav>
        </header>

        <div class="message">
            <a class="twitter-link" href="http://twitter.com/home/?status=I%20created%20a%20project%20with%20AWS%20Code
            <div class="text">
                <h1>Congratulations!</h1>
                <h2>You just created a Node.js web application</h2>
            </div>
        </div>
    </div>

    <footer>
        <p class="footer-contents">Designed and developed with <a href="https://aws.amazon.com/careers/devtools-jobs/">
    </footer>

    <script src="assets/js/set-background.js"></script>
</body>

</html>
```

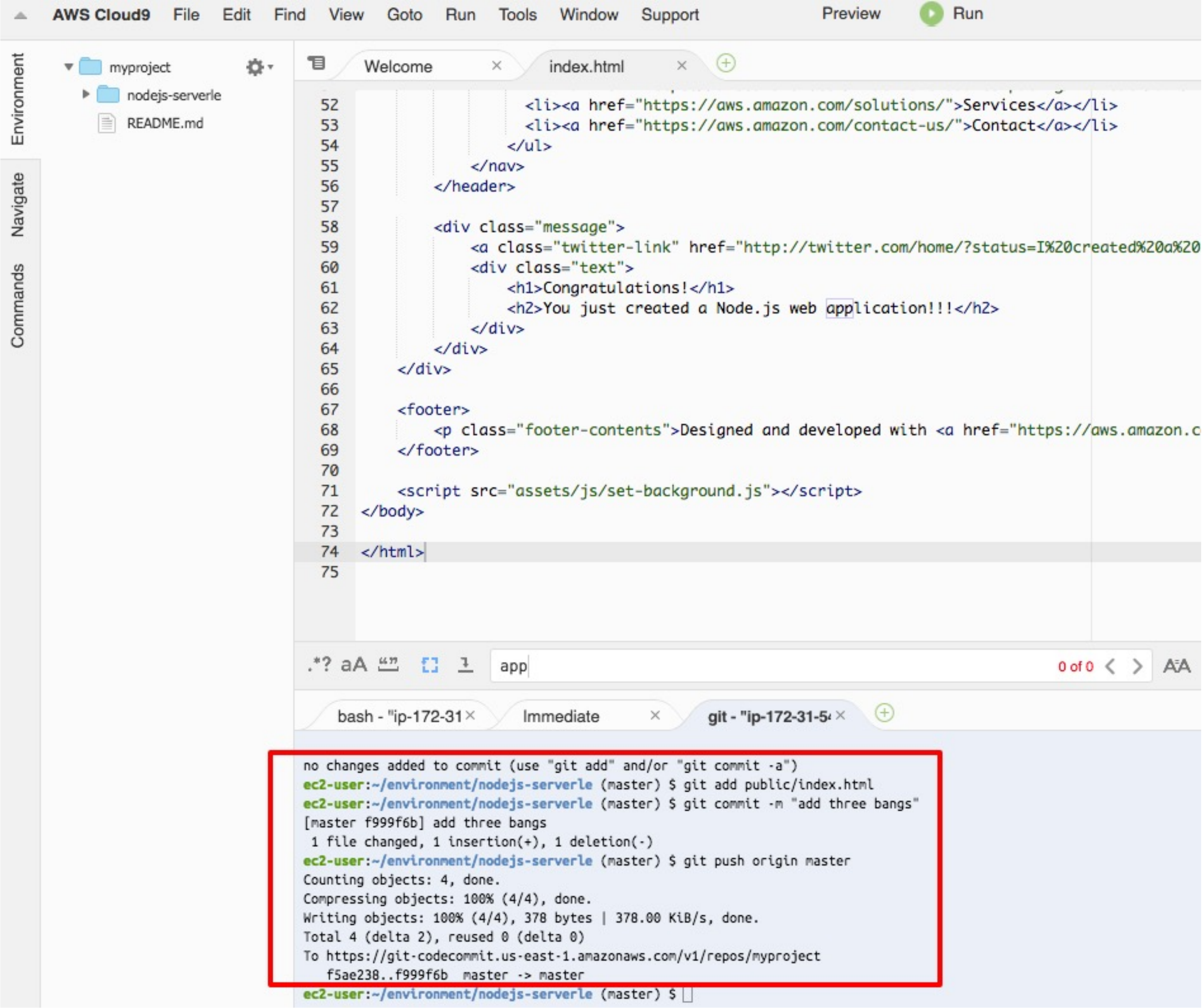
```
appl
```

```
ec2-user:~/environment/nodejs-serverle (master) $
```



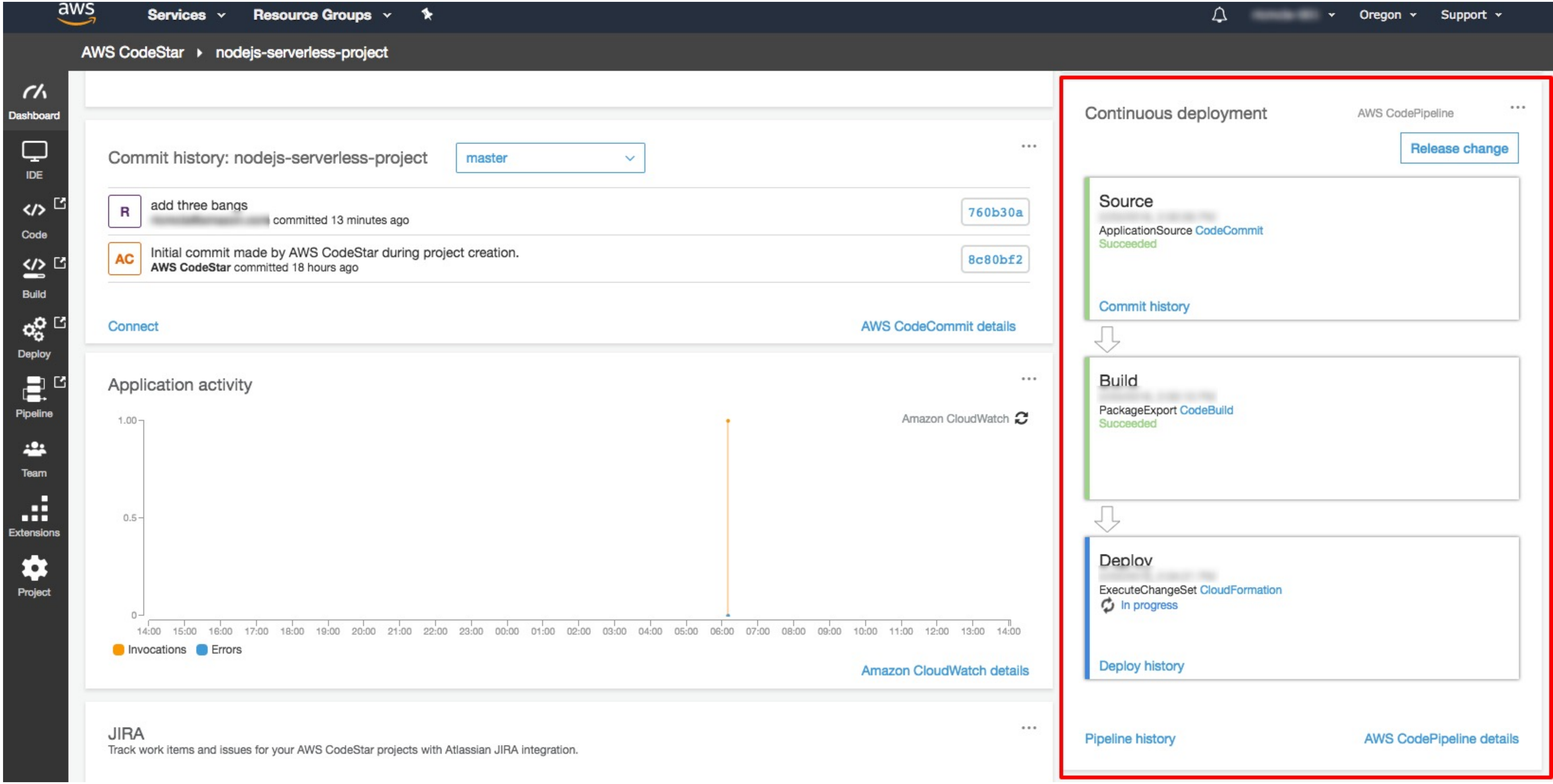
```
            <nav class="website-nav">
                <ul>
                  <li><a class="home-link" href="https://aws.amazon.com/">Home</a></li>
                  <li><a href="https://aws.amazon.com/what-is-cloud-computing/">About</a></li>
                  <li><a href="https://aws.amazon.com/solutions/">Services</a></li>
                  <li><a href="https://aws.amazon.com/contact-us/">Contact</a></li>
                </ul>
            </nav>
        </header>

        <div class="message">
            <a class="twitter-link" href="http://twitter.com/home/?status=I%20created%20a%20project%20with%20AWS%20Code
            <div class="text">
                <h1>Congratulations!</h1>
                <h2>You just created a Node.js web application!!!</h2>
            </div>
        </div>
    </div>

    <footer>
        <p class="footer-contents">Designed and developed with <a href="https://aws.amazon.com/careers/devtools-jobs/">
    </footer>

    <script src="assets/js/set-background.js"></script>
</body>

</html>
```

```
ec2-user:~/environment/nodejs-serverle (master) $
```



```
                  <li><a href="https://aws.amazon.com/solutions/">Services</a></li>
                  <li><a href="https://aws.amazon.com/contact-us/">Contact</a></li>
                </ul>
            </nav>
        </header>

        <div class="message">
            <a class="twitter-link" href="http://twitter.com/home/?status=I%20created%20a%20
            <div class="text">
                <h1>Congratulations!</h1>
                <h2>You just created a Node.js web application!!!</h2>
            </div>
        </div>
    </div>

    <footer>
        <p class="footer-contents">Designed and developed with <a href="https://aws.amazon.c
    </footer>

    <script src="assets/js/set-background.js"></script>
</body>

</html>

```

```
no changes added to commit (use "git add" and/or "git commit -a")
ec2-user:~/environment/nodejs-serverle (master) $ git add public/index.html
ec2-user:~/environment/nodejs-serverle (master) $ git commit -m "add three bangs"
[master f999f6b] add three bangs
 1 file changed, 1 insertion(+), 1 deletion(-)
ec2-user:~/environment/nodejs-serverle (master) $ git push origin master
Counting objects: 4, done.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 378 bytes | 378.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0)
To https://git-codecommit.us-east-1.amazonaws.com/v1/repos/myproject
   f5ae238..f999f6b  master -> master
ec2-user:~/environment/nodejs-serverle (master) $ 
```



Commit history: nodejs-serverless-project

master

R	add three bangs [redacted] committed 13 minutes ago	760b30a
AC	Initial commit made by AWS CodeStar during project creation. **AWS CodeStar** committed 18 hours ago	8c80bf2

Application activity

JIRA

Track work items and issues for your AWS CodeStar projects with Atlassian JIRA integration.

Continuous deployment

Source

[redacted]

ApplicationSource CodeCommit

Succeeded

Build

[redacted]

PackageExport CodeBuild

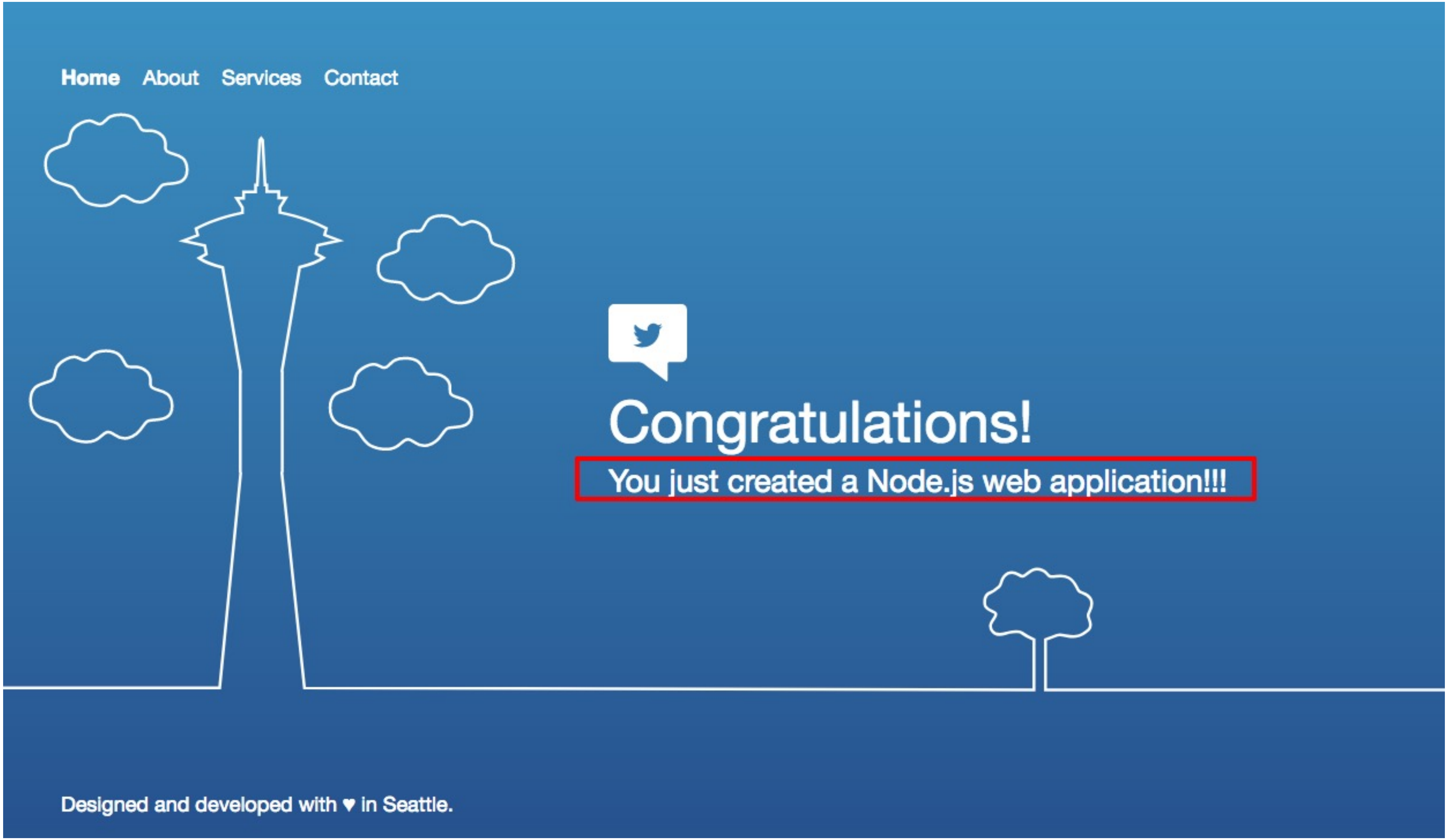
Succeeded

Deploy

[redacted]

ExecuteChangeSet CloudFormation

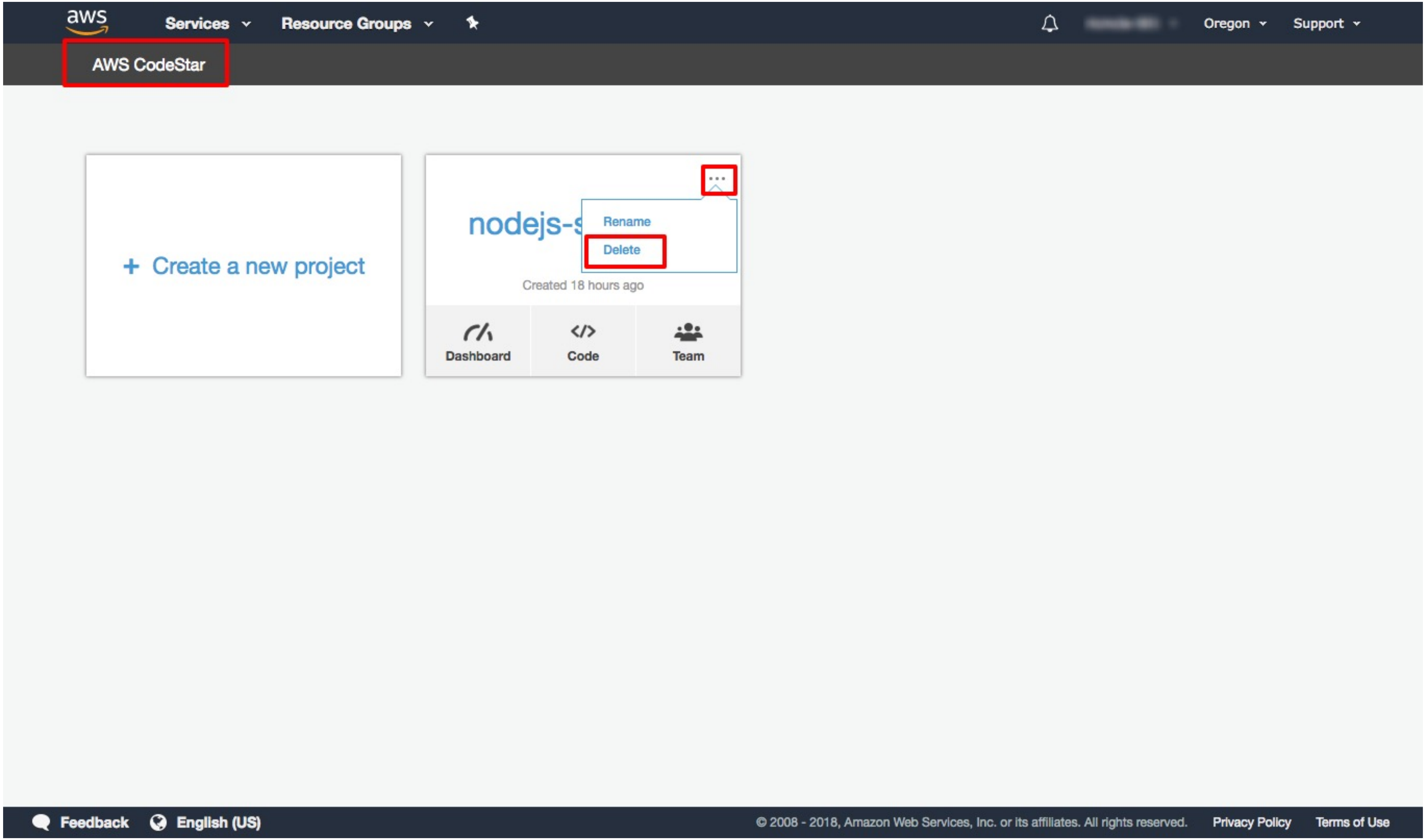
In progress



Congratulations!

You just created a Node.js web application!!!

Designed and developed with ♥ in Seattle.


aws
Services
Resource Groups
[redacted]
Oregon
Support
AWS CodeStar
+ Create a new project
nodejs-s
...
Rename
Delete
Created 18 hours ago
Dashboard
Code
Team
Feedback
English (US)
© 2008 - 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Claimed FaaS advantages

- Smaller for developer since infrastructure is handled by somebody else
  => more time for writing application code
- Inherently scalable
- No need to pay for idle resources
  (temptation to miss-use)
- Available and fault tolerant
- No explicit multi-tenancy
- Forces modular business logic

Claimed FaaS disadvantages

- Decreased transparency
- Maybe challenging to debug
- Autoscaling of functions may lead to autoscaling of cost
- Keeping track of huge numbers of functions is tough
- Chaching of requests?

Summary of cloud native

What it means to be Cloud Native approach — the CNCF way

https://medium.com/developingnodes/what-it-means-to-be-cloud-native-approach-the-cncf-way-9e8ab99d4923

1. Containerization

- **Docker container** image is a lightweight, standalone, executable package of software that includes everything needed to run an application.

2. CI/CD
3. Orchestration

- **Kubernetes** is the market-leading orchestration solution.

4. Observability & Analysis

- Monitoring, logging, and tracing

5. Service MESH

6. Networking and Policy

- Flexibility with authorization, admission control and data filtering

7. Distributed Database

- When you need more resiliency and scalability than you can get from a single database

8. Messaging

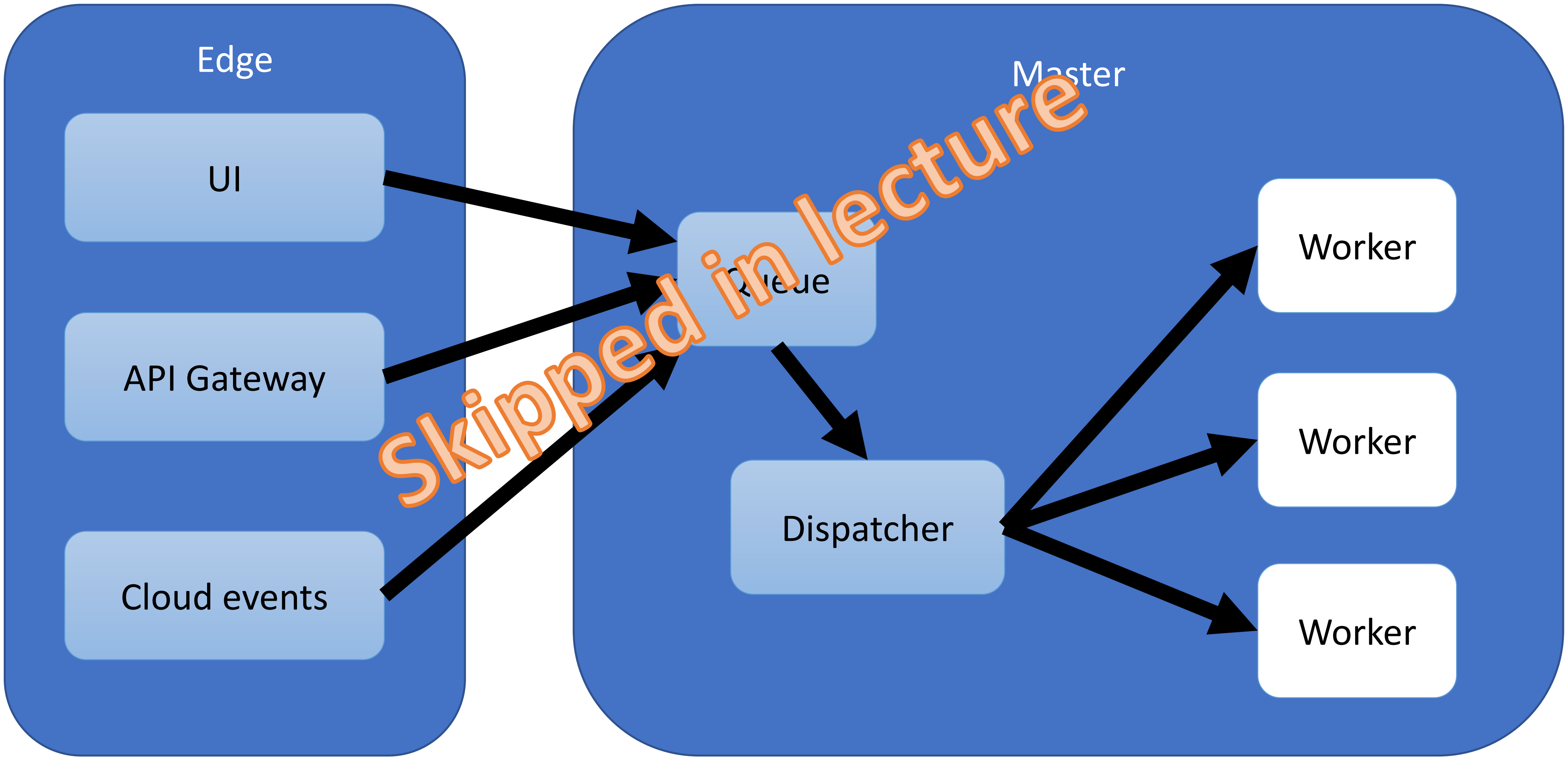
9. Container registry and runtimes

10. Sofware distribution

Tampereen yliopisto
Tampere University

Serverless

Baldini et al: Serverless Computing: Current Trends and Open Problems



Skipped in lecture

Microservices vs. Serverless/FaaS

(They are different – do not call serveless microservices)

- Microservice
  - Small services running in their own process and communicating with lightweight services
  - Can be stateful

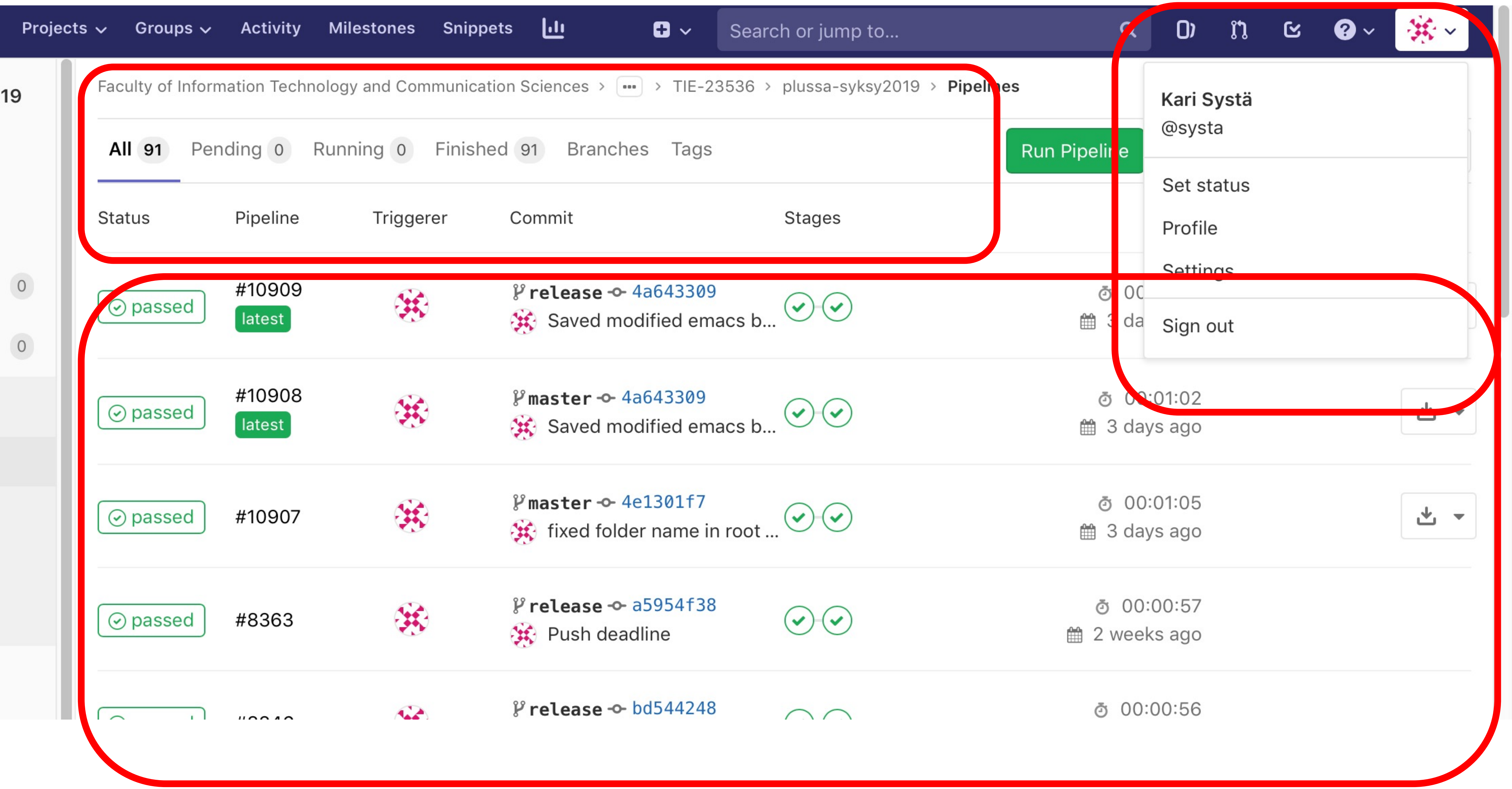
- Serverless / FaaS
  - Short term execution triggered by a request, then closes down
  - For stateless computing

Some comparison

	Microservice	Serverless / FaaS
Bug hunting	Easier (but not easy)	Difficult
Infrastructure code	May be complex	Minimal or even non-existent
Scaling	Need to be implemented	Automatic
Performance	Good	Possible cold-start issues
Running cost	May include cost of idle time	Pay only per use

Tampereen yliopisto
Tampere University

Microfront-ends



Faculty of Information Technology and Communication Sciences › ... › TIE-23536 › plussa-syksy2019 › Pipelines

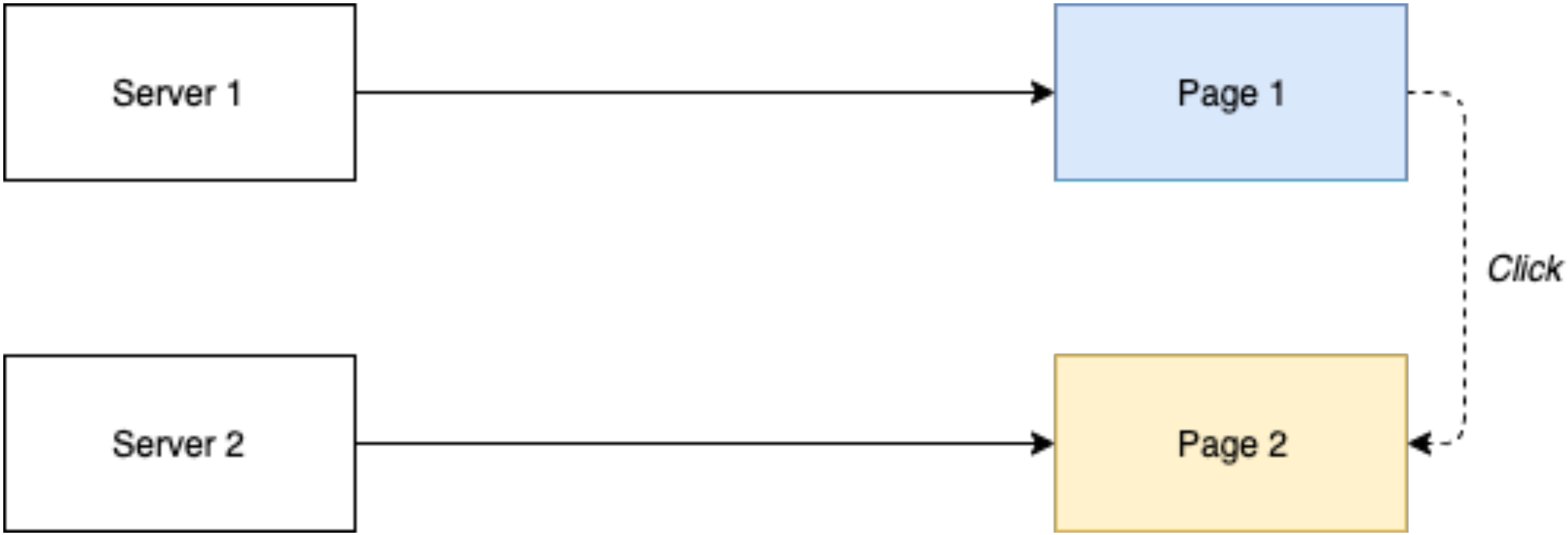
All 91 | Pending 0 | Running 0 | Finished 91 | Branches | Tags

Status	Pipeline	Triggerer	Commit	Stages	
passed	#10909 latest		release 4a643309 Saved modified emacs b...	✓ ✓	00 3 da
passed	#10908 latest		master 4a643309 Saved modified emacs b...	✓ ✓	00:01:02 3 days ago
passed	#10907		master 4e1301f7 fixed folder name in root ...	✓ ✓	00:01:05 3 days ago
passed	#8363		release a5954f38 Push deadline	✓ ✓	00:00:57 2 weeks ago
			release bd544248		00:00:56

Alternative architectures

(from https://morioh.com/p/ee1b48c9de16)

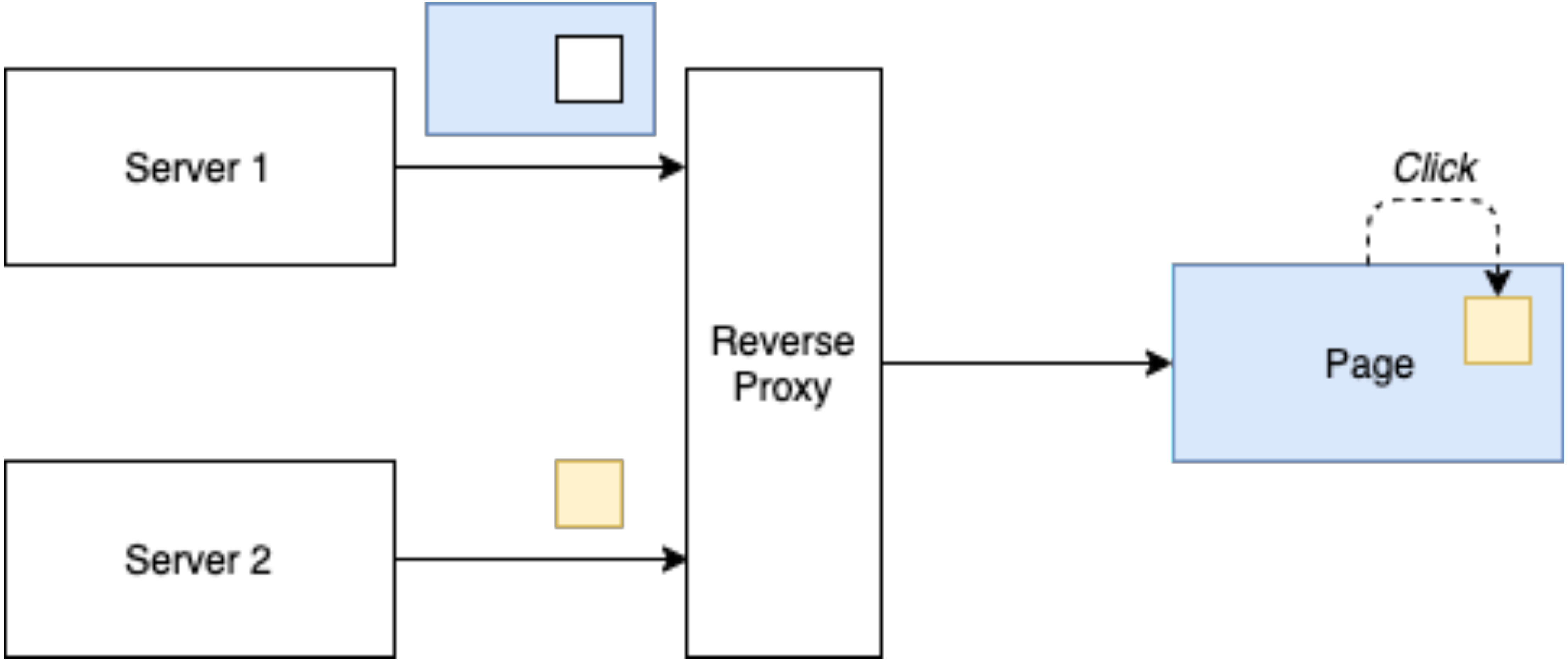
1. Web Approach



Alternative architectures

(from https://morioh.com/p/ee1b48c9de16)

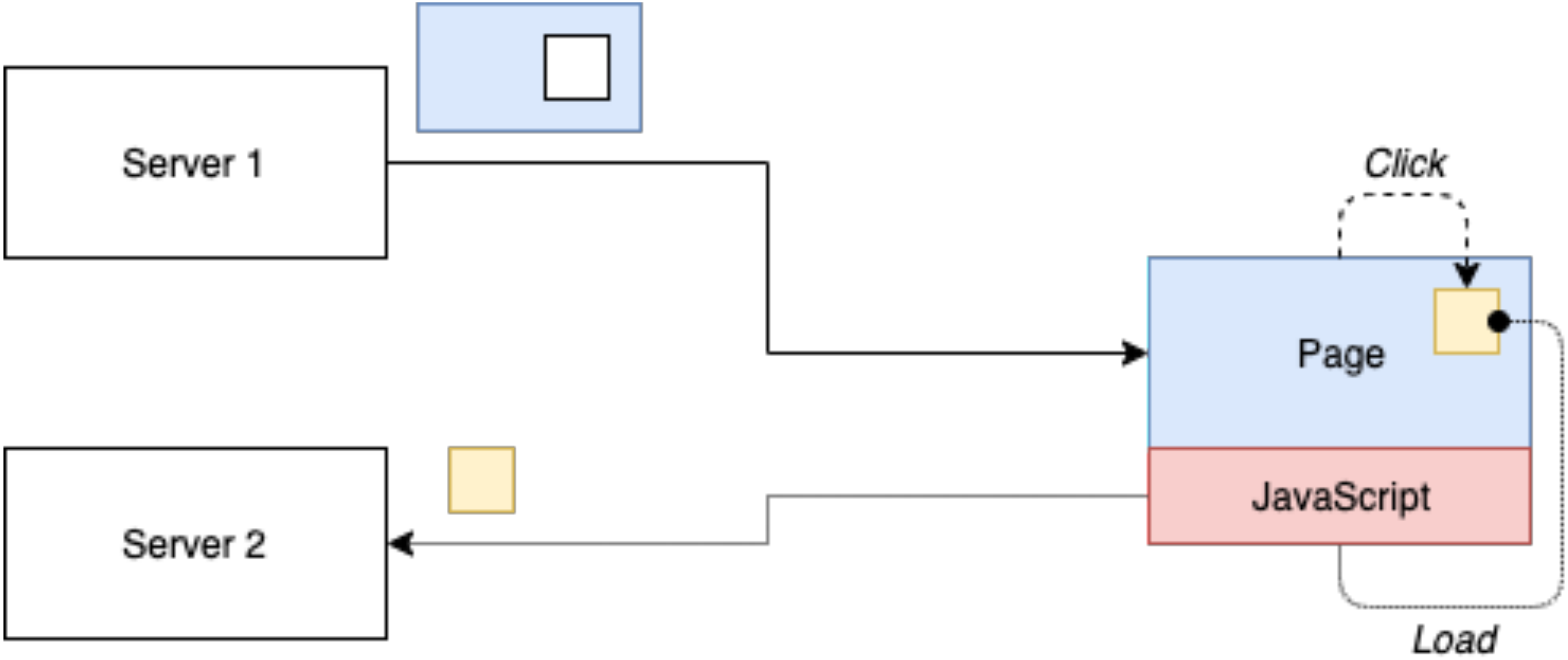
2. Server-side composition



Alternative architectures

(from https://morioh.com/p/ee1b48c9de16)

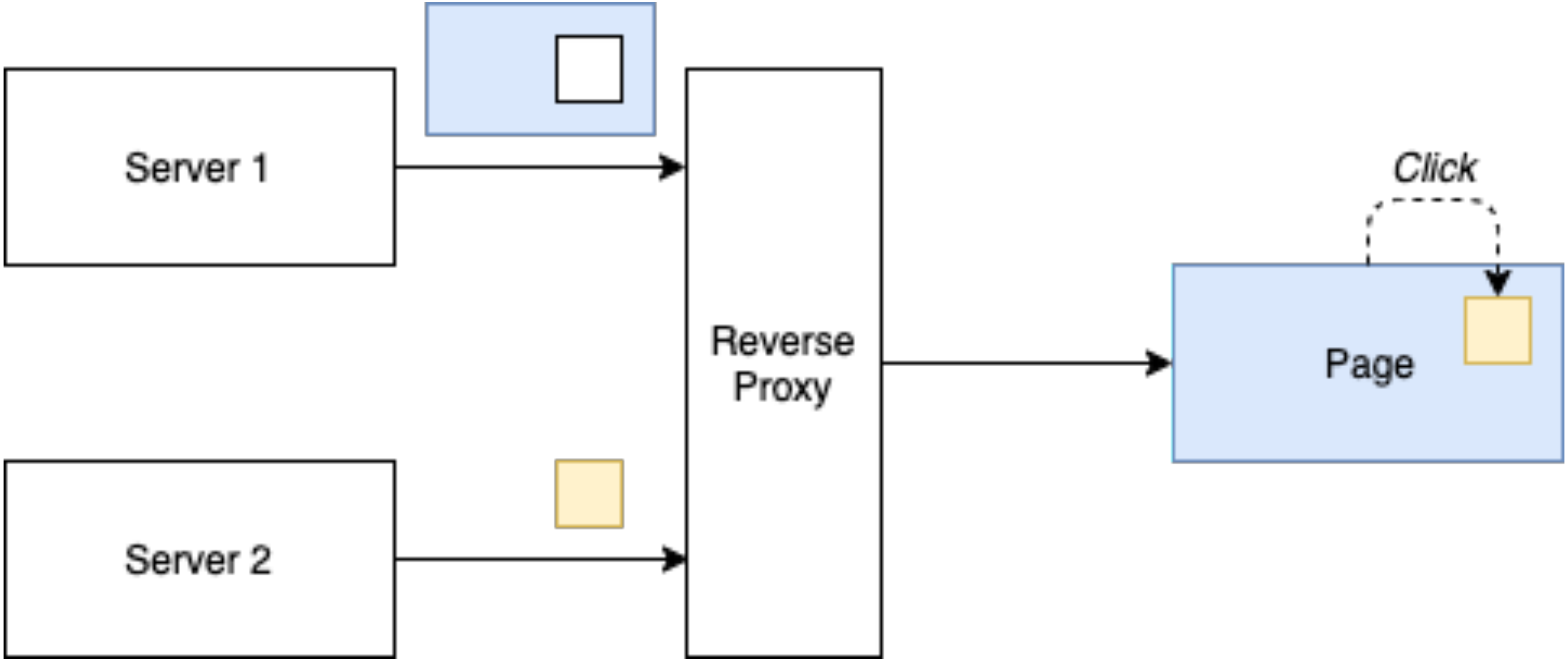
3. Client-side composition



Alternative architectures

(from https://morioh.com/p/ee1b48c9de16)

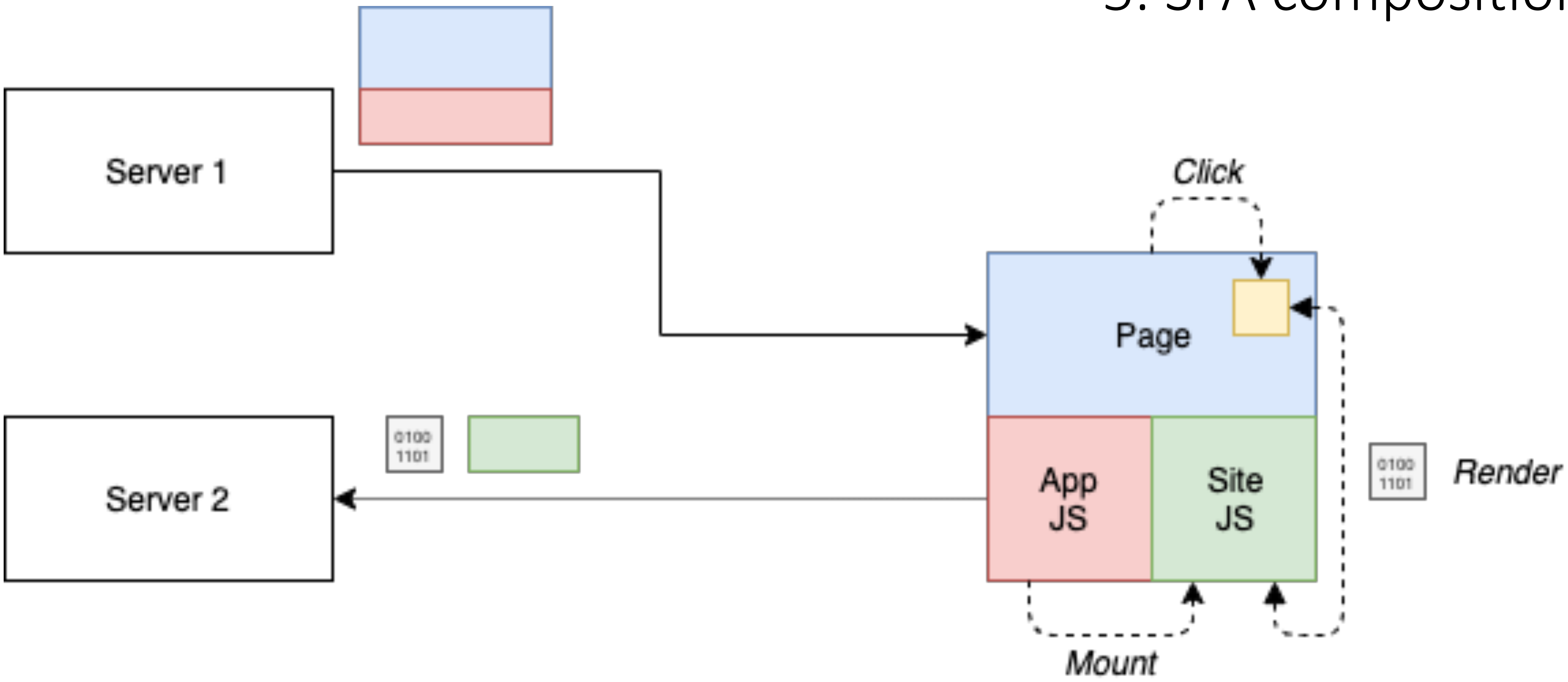
4. Client-side rendering



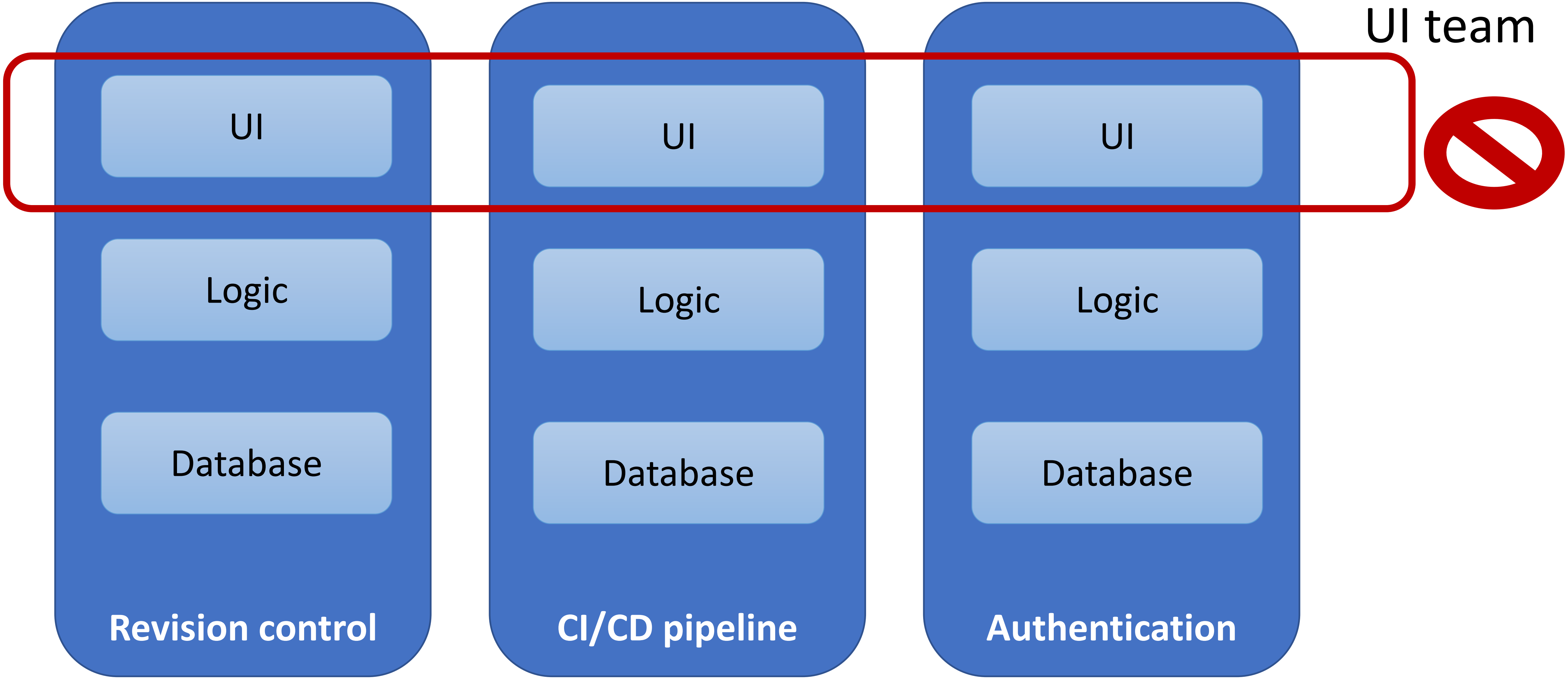
Alternative architectures

(from https://morioh.com/p/ee1b48c9de16)

5. SPA composition



Organization and process issues



UI team

UI

Logic

Database

Revision control

UI

Logic

Database

CI/CD pipeline

UI

Logic

Database

Authentication

Tampereen yliopisto
Tampere University

Stateful vs stateless computation

- If a service has an internal state it is difficult to
  - Scale it
  - Move it to other server or other hosting system

  => Stateless Services are subject to cloud-specific optimizations
- The internal state my be
  - volatile or
  - non-volatile
  - ... in memory, file local to container,
- Serverless / FaaS

7R’s of cloud Micration

Replace with imilar or improved but SaaS

Reuse in the new SaaS version

Refactor towards cloud-native architecture

Replatform by using cloud services

Rehost to a VM

Retain

Retire

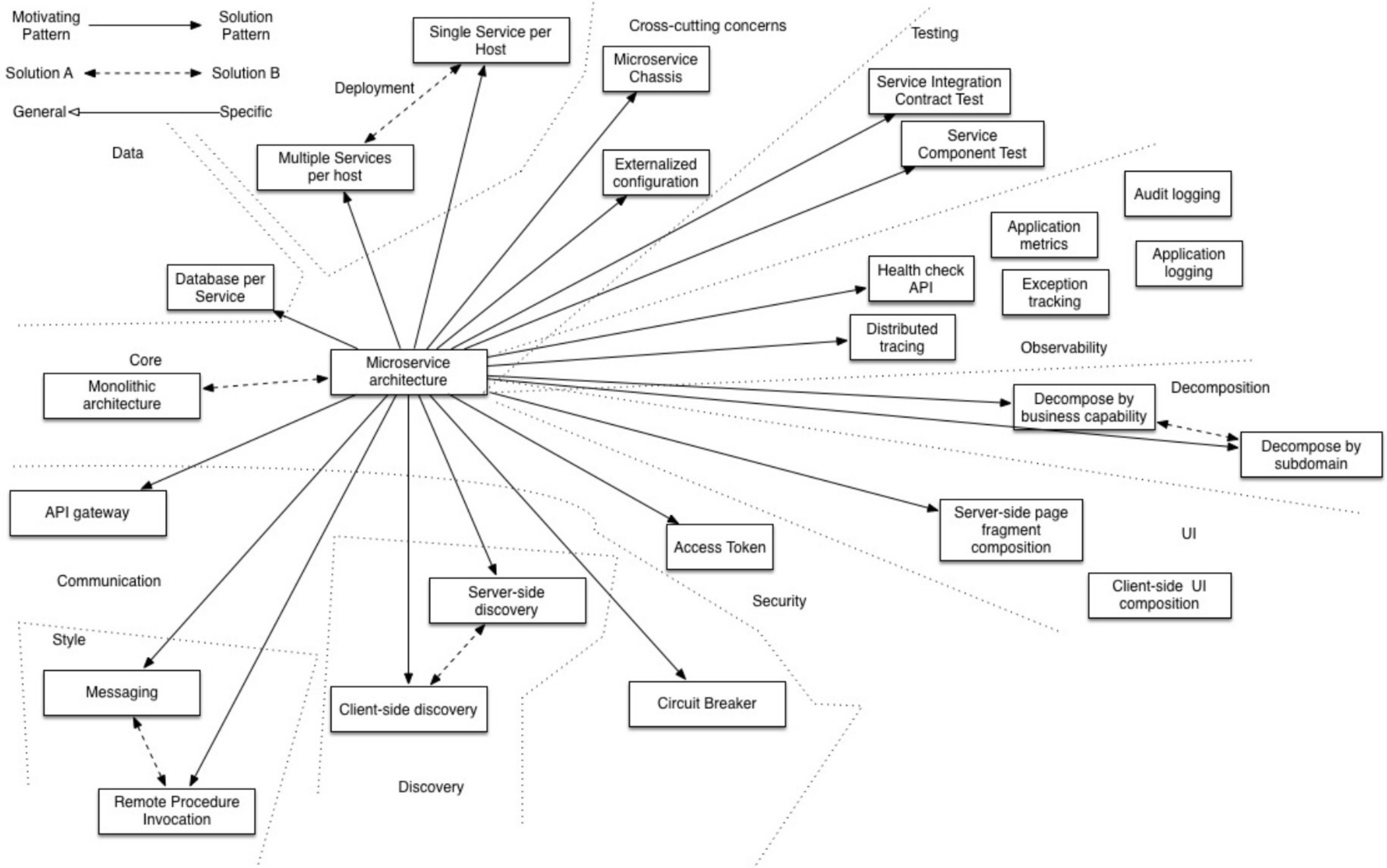
Reminder

https://thenewstack.io/10-key-attributes-of-cloud-native-applications/

- **Packaged as lightweight containers**
- **Developed with best-of-breed languages and frameworks**
- **Designed as loosely coupled microservices**
- **Centered around APIs for interaction and collaboration**
- **Architected with a clean separation of stateless and stateful services**

- **Isolated from server and operating system dependencies**
- **Deployed on self-service, elastic, cloud infrastructure**
- **Managed through agile DevOps processes**
- **Automated capabilities**
- **Defined, policy-driven resource allocation**

https://microservices.io/patterns/microservices.html



Nice video about microservices

- Netflix story (Mastering Chaos - A Netflix Guide to Microservices) <https://www.youtube.com/watch?v=CZ3wIuvmHeM>