

# AWS CodeStar

## Step 1 Creating an AWS CodeStar Project

### Introduction

AWS CodeStar is a complete package for developing and releasing your applications. AWS CodeStar leverages lower level AWS Code Services such as AWS CodeCommit, and AWS CodeDeploy, but frees you from setting everything up. The applications can be deployed on Amazon Elastic Compute Cloud (EC2) instances, AWS Elastic Beanstalk, or AWS Lambda. The template you choose to base your project upon determines how the project is deployed. AWS CodeStar supports a variety of programming languages including JavaScript, Java, PHP, Python, and Ruby. In this Lab, you will work with a Node.js (JavaScript) project. The language won't be a focus, so you could easily get started with a project in a different language after completing the lab.

In this Lab Step, you will create a Node.js project from a template. You will also inspect everything that comes along with starting a project in AWS CodeStar.

### Instructions

1. In the AWS Management Console, navigate to **Services > Developer Tools > CodeStar**:



Developer Tools

CodeStar

CodeCommit

CodeBuild

CodeDeploy

CodePipeline

X-Ray

2. Click **Start a project** on the welcome page:



## AWS CodeStar

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

[Start a project](#)

Take a moment to see all of the different templates available in AWS CodeStar.

3. Check the following boxes on the left filter bar to narrow down the listed templates:

- **Application category:** *Web application*
- **Programming languages:** *Node.js*
- **AWS services:** *EC2*


The choice of **Application category** and **Programming language** will be driven by the requirements of your project and skills available to you. The choice of **AWS services** may not be as easy. Some guidelines for choosing between the alternatives are:

- **AWS Elastic Beanstalk:** A good choice for a fully managed application environment running on EC2 servers. This option allows you to stay focused on your code.
- **Amazon EC2:** Preferable when you want to host the application on servers that you manage yourself, including on-premise servers.
- **AWS Lambda:** Choose this option if you want to run a serverless application.

4. Select the **Express.js** project template:


## Choose a project template


Start a new software project on AWS in minutes using a project template. [Help me choose](#)




### Node.js

---


 Web application


 Amazon EC2  
(runs on virtual servers that you manage)



### Express.js

---

 Web application

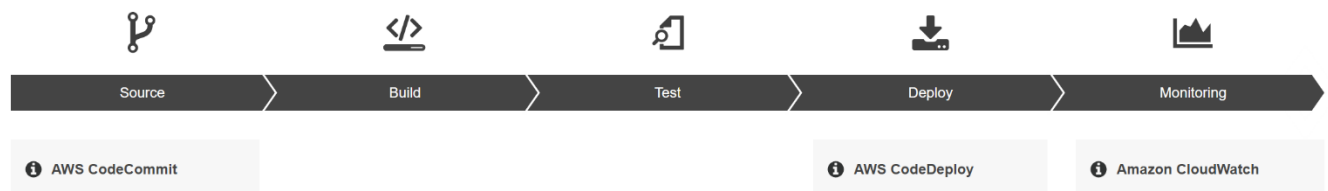
 Amazon EC2  
(runs on virtual servers that you manage)

Express.js is a popular Node.js web application framework.

5. In the next step of the **Create project** wizard, enter the following:

- **Project name:** *ca-app-`<Unique_String>`* (Replace `<Unique_String>` with a 6 characters. The name must be unique for the region because of AWS CodeCommit repository name restrictions)
- **Project ID:** Accept the default value
- **Edit Amazon EC2 Configuration:**
  - **Instance type:** *t2.micro* (default value)
  - **VPC:** Select the non-default VPC (The VPC without "(Default)"), or the VPC with only two subnets if there is no (Default) label
  - **Subnet:** Select the *us-west-2a* subnet
- **AWS CodeStar would like permission to administer AWS resources on your behalf:** *checked*

The instructions in this Lab use *ca-app* for the project name, but you should use a different name or the project creation may fail if it is already in use. Notice the pipeline that will be created for you when the project is created:

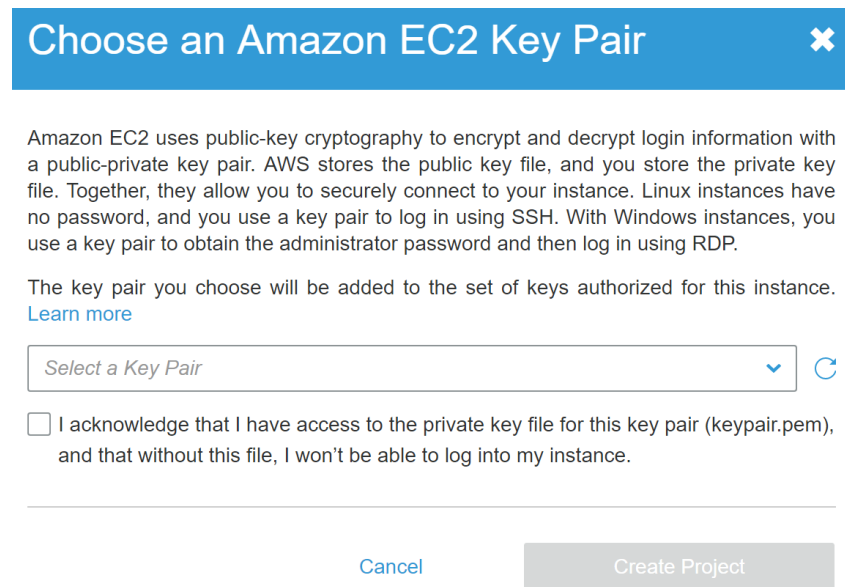


The pipeline is built using AWS CodePipeline and includes a source repository in AWS CodeCommit, and a deployment group in AWS CodeDeploy. You can configure the pipeline later if needed, including adding build and test stages using AWS CodeBuild. But the template

comes with everything needed to initially release your application. The project is automatically monitored by Amazon CloudWatch.

6. Click **Next**.

7. In the **Choose an Amazon EC2 Key Pair** pop-up dialog, select the Key Pair available in the drop-down menu:



Choose an Amazon EC2 Key Pair

Amazon EC2 uses public-key cryptography to encrypt and decrypt login information with a public-private key pair. AWS stores the public key file, and you store the private key file. Together, they allow you to securely connect to your instance. Linux instances have no password, and you use a key pair to log in using SSH. With Windows instances, you use a key pair to obtain the administrator password and then log in using RDP.

The key pair you choose will be added to the set of keys authorized for this instance. [Learn more](#)

Select a Key Pair

☐ I acknowledge that I have access to the private key file for this key pair (keypair.pem), and that without this file, I won't be able to log into my instance.

Cancel Create Project

The Key Pair is set as a Key Pair for the EC2 instances where the application will be deployed using AWS CodeDeploy. This allows you to connect to the instances using SSH, if necessary.

8. Click **Create Project**.

AWS CodeStar will begin provisioning the resources needed for your application. It takes about five minutes to complete and there is a progress bar to let you know how things are going.

9. Enter your **AWS CodeStar user settings**:

- **Display Name:** *student* (or whatever you prefer)
- **Email Address:** Any email address (this Lab won't use the email address provided)

10. Click **Next**.

11. In the **Set up tools** step, Click **Skip**.

You will complete this after the project is created.

12. In the **User Profile Information** pop-up, re-enter the **Display Name** and **Email Address** from before.

You will return to AWS CodeStar in a couple Lab Steps. While the provisioning progresses, you can connect to the *dev-instance* EC2 instance in the next Lab Step. It is running Amazon Linux (log in with user name *ec2-user*). You will use dev-instance to commit code changes later.

## Summary

In this Lab Step, you created an AWS CodeStar project using a Node.js web application template. You configured your project to use Amazon EC2 instances for the application. The project includes an AWS CodeCommit repository, AWS CodeDeploy deployment group wrapped in a continuous deployment pipeline by AWS CodePipeline.

## Step 2 Creating and connecting to the Virtual Machine using SSH

### Introduction

#### Create New Instance: EC2 > Launch instance (ami-922914f7)

##### Step 1: Choose an Amazon Machine Image (AMI)

[Cancel and Exit](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.



Quick Start

My AMIs

AWS Marketplace

Community AMIs

Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-922914f7

Amazon Linux Free tier eligible

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root device type: ebs Virtualization type: hvm

Select

64-bit

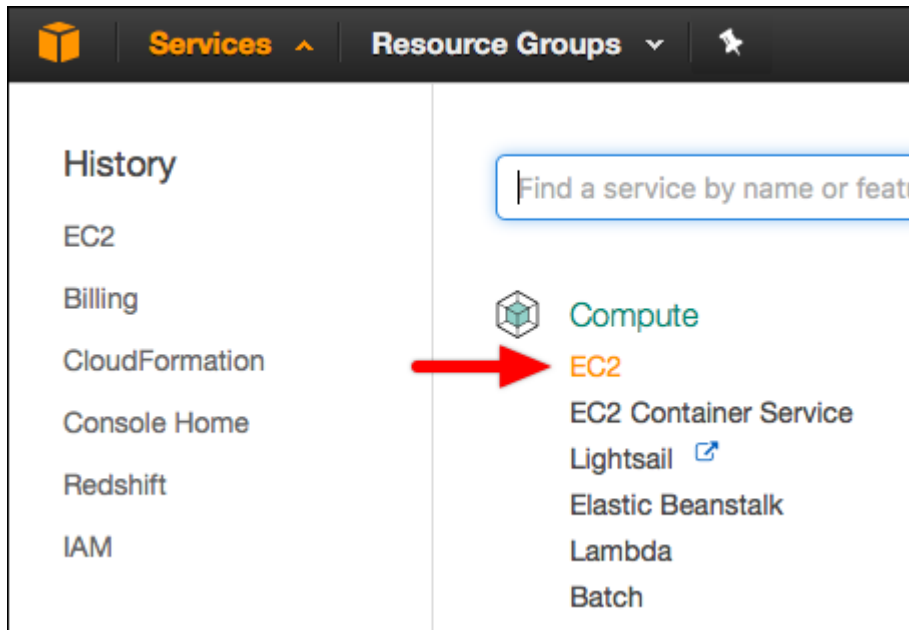
**Note!!!** Complete configuration providing relevant params defining **TCP port 3000** to custom security group and finish providing **RH-124** key to access the instance.

In this Lab Step, you will employ a Secure Shell (SSH) client to connect to a remote Linux server. SSH is a cryptographic network protocol for securing data communication. SSH establishes a secure channel over an unsecured network. Common applications include remote login and remote command execution.

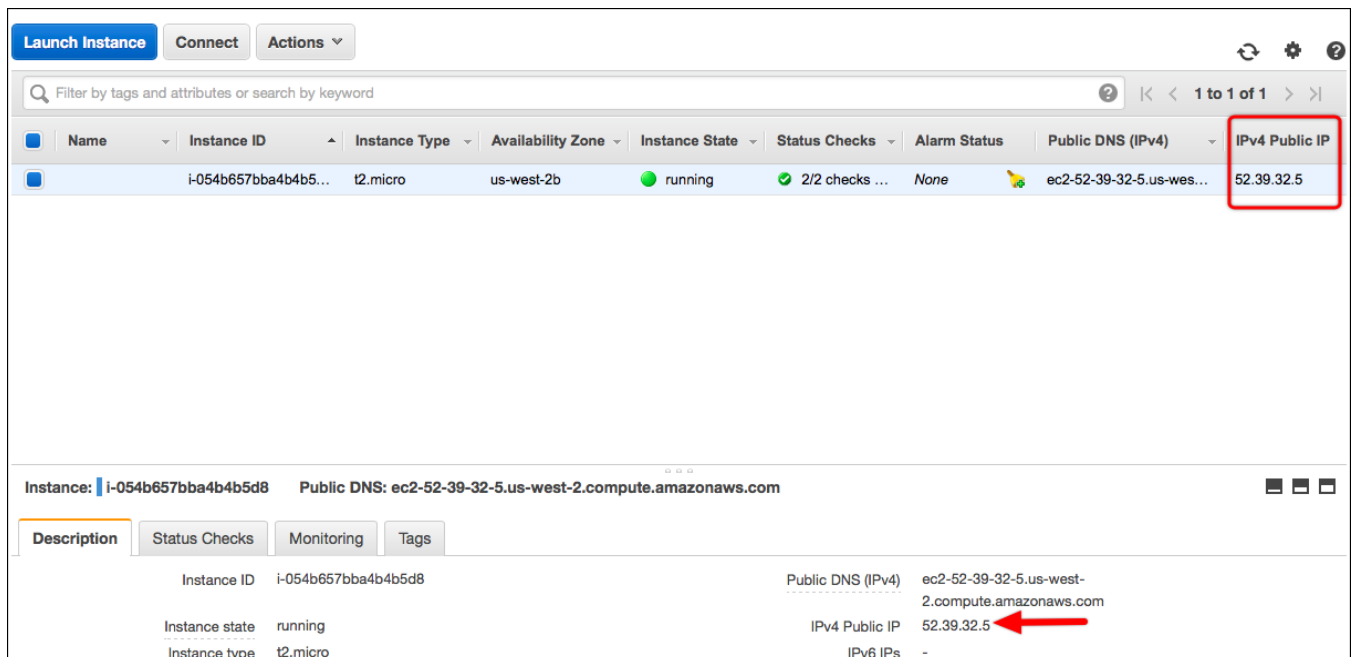
### Instructions

#### *Locating the virtual machine IP address*

1. In the AWS Management Console, navigate to **Services > Compute > EC2**:



2. Click on **Running Instances** and select the target virtual machine and locate the **IPv4 Public IP** address. An example IPv4 address number is 52.39.32.175. Copy your virtual machine IP address for later use:



3. Proceed to the **Connecting using Linux / macOS** or **Connecting using Windows** instructions depending on your local operating system.

## Connecting using Linux / macOS

Linux distributions and macOS include an SSH client that accepts standard PEM keys. Complete the following steps to connect using the included terminal applications:

a. Open your terminal application. If you need assistance finding the terminal application, search for *terminal* using your operating system's application finder or search commands.

b. Enter the following command and press *Enter*:

```
ssh -i /Path/To/Your/KeyPair.pem AMIUserName@YourIPv4Address
```

where the command details are:

`ssh` initiates the SSH connection.

`-i` specifies the identity file.

`/Path/To/Your/KeyPair.pem` specifies the location and name of your key pair. An example location might be `/Home/YourUserName/Downloads/KeyPair.pem`.

`AMIUserName` specifies the SSH user:

For the Amazon Linux Amazon Machine Image (AMI), a standard SSH user is `ec2-user`.

For Ubuntu images, a standard SSH user is `ubuntu`.

For CentOS images, a standard SSH user is `centos`.

For Debian images, a standard SSH user is `admin`.

For Red Hat 6.4 and later images, a standard SSH user is `ec2-user`.

`YourIPv4Address` is the IPv4 address noted earlier in the instructions.

*Note:* Your SSH client may refuse to start the connection due to key permissions. If you receive a warning that the key pair file is unprotected, you must change the permissions. Enter the following command and try the connection command again:

```
chmod 600 /Path/To/Your/KeyPair.pem
```

c. After successfully connecting to the virtual machine, you should reach a terminal prompt similar to the one shown in the image below.

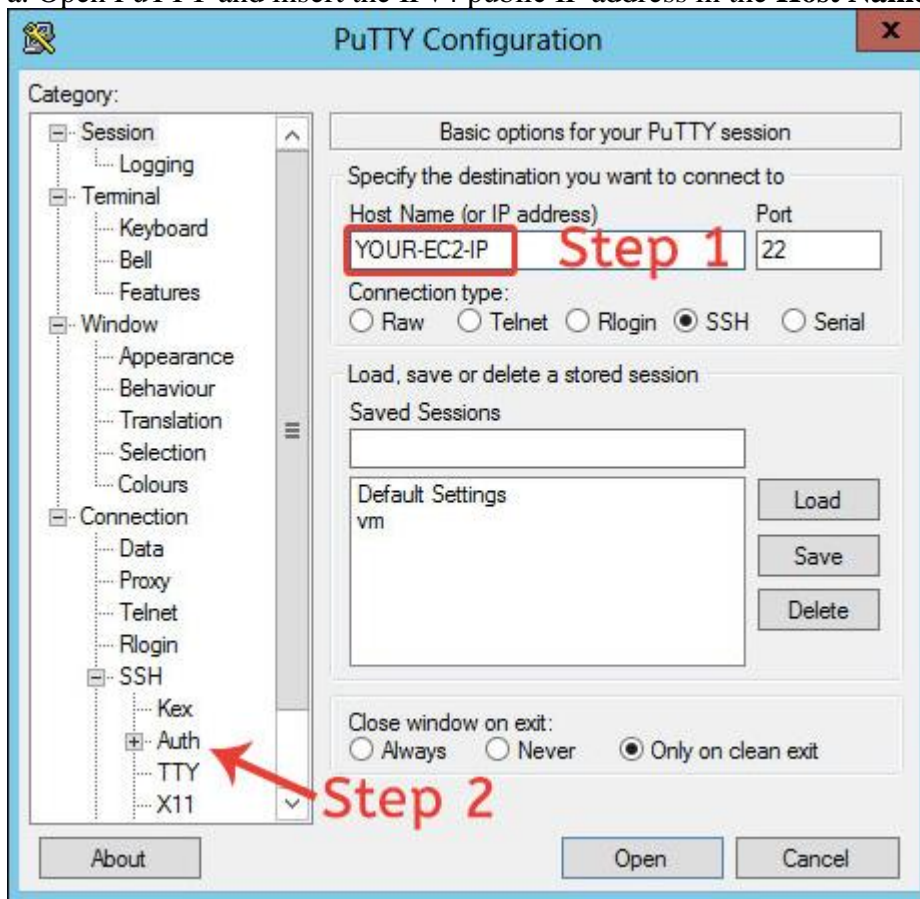
*Note:* If you receive a warning that the host is unknown, enter `y` or `yes` to add the host and complete the connection.

```
 _ | _ | )  
 _ | ( /  Amazon Linux AMI  
 _ | \ | _ |  
  
https://aws.amazon.com/amazon-linux-ami/2017.03-release-notes/  
6 package(s) needed for security, out of 6 available  
Run "sudo yum update" to apply all updates.  
[ec2-user@ip-172-31-22-167 ~]$
```

### Connecting using Windows

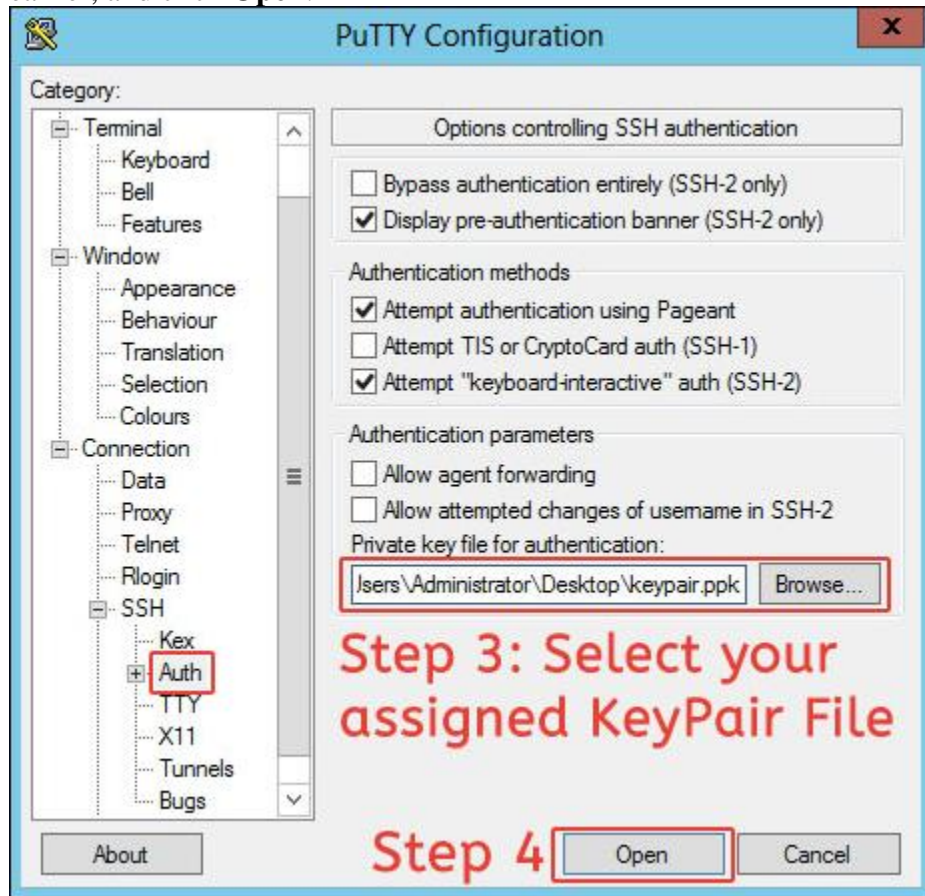
Windows does not include an SSH client. You must download an application that includes one. A free and useful utility is called PuTTY. PuTTY supports SSH connections as well as key generation and conversion. Download PuTTY at <http://www.putty.org>. Complete the following steps to use PuTTY to create an SSH connection.

- a. Open PuTTY and insert the IPv4 public IP address in the **Host Name (or IP address)** field.





b. Navigate to the **Connection > SSH > Auth** section. Select the PPK key pair you downloaded earlier, and click **Open**.



c. Wait several seconds for the authentication prompt. Enter the SSH username for the virtual machine operating system, such as ec2-user for Amazon Linux, and press *Enter*.

Additional example SSH usernames include:

For Amazon Linux, a standard SSH user is ec2-user.

For Ubuntu images, a standard SSH user is ubuntu.

For CentOS images, a standard SSH user is centos.

For Debian images, a standard SSH user is admin.

For Red Hat 6.4 and later images, a standard SSH user is ec2-user.

## Summary

In this Lab Step you connected to a virtual machine using an SSH client.

## Step 3 Touring the AWS CodeStar Project Website

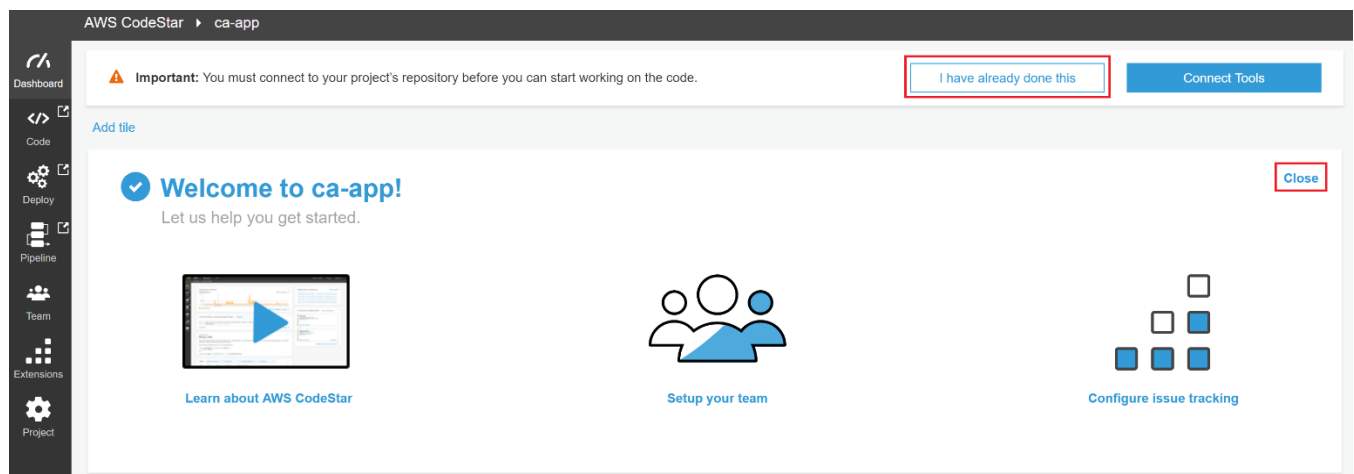
### Introduction

AWS CodeStar provides a website that integrates everything you need to manage your project. In this Lab Step, you will familiarize yourself with the different components of the website.

By now, your AWS CodeStar project should almost be finished provisioning. If it hasn't finished, wait until the progress bar reaches 100% and disappears so that all of the interface components are available.

### Instructions

1. In your AWS CodeStar project browser tab, click **I have already done this** and then **Close** on two uppermost tiles in your **Dashboard**:



These tiles only show up when beginning your project. This Lab will go over everything you need and you can safely close them.

2. Observe the tiles that are included in your **Dashboard**:

- **Team wiki:** This is an editable tile where you can leave relevant project notes for all members of your team to see:

## 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. Setup issue tracking (under "Extensions")
5. [Customize project dashboard](#)
6. [View AWS CodeStar documentation](#)
7. [Visit the AWS CodeStar Forum](#)

You can edit the tile by selecting **Edit** from the drop-down ellipsis (...) menu in the top right corner.

- **Commit history:** Displays the most recent code commits for the selected branch:

Commit history: ca-app master ▼ ...

---

 Initial commit of sample code made during project creation in AWS CodeStar  
**AWS CodeStar** committed 11 hours ago

[ec58332](#)

[Connect](#)

[AWS CodeCommit details](#)

Currently there is only a master branch and the initial commit to display. The committer is displayed in bold. **AWS CodeStar** made the initial commit during the project creation. Each commit also includes a button on the right to view the code changes in AWS CodeCommit. You will look at the code in a future Lab Step.

**Continuous deployment:** This tile shows a graphical representation of the release pipeline for your project:

[Release change](#)**Source**

6/16/2017, 2:05:38 AM

ApplicationSource [CodeCommit](#)**Succeeded**[Commit history](#)**Application**

6/16/2017, 2:07:18 AM

Deploy [CodeDeploy](#)**Succeeded**[Deploy history](#)[Endpoint\(1\)](#)[AWS CodePipeline details](#)

Any time you commit a code change to the master branch, the pipeline will automatically deploy your application. As your application grows and the requirements for your release pipeline change, you can modify the pipeline by clicking [AWS CodePipeline details](#). For example, you may want to add an automated test stage, invoke an AWS Lambda function, or modify the deployment group to deploy to an Auto Scaling group. The [Release change](#) button can be used to force a deployment of the latest commit. That can be useful if you modify the pipeline or something went wrong with the release. If something does go wrong with a pipeline stage, you will see the bar on the left turn red.

- **Application endpoints:** The endpoint for accessing your application is shown in this tile:

**Application endpoints**

...

---

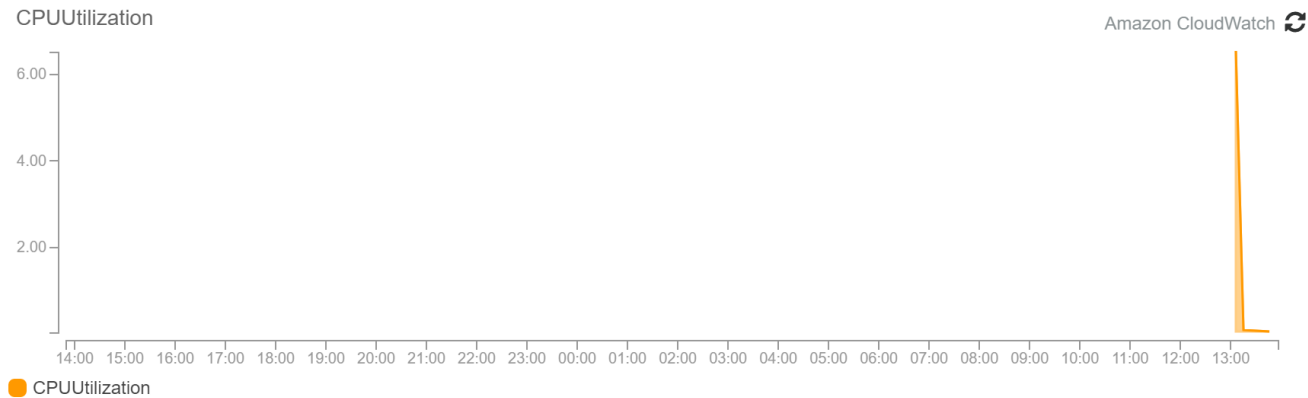
<http://ec2-35-163-40-229.us-west-2.compute.amazonaws.com>

---

The deployment created for your project deploys to a single EC2 instance. The endpoint is the public DNS name of the instance. This would be different for project templates using AWS Lambda or AWS Elastic Beanstalk.

- **Application activity:** This tile shows the **CPUUtilization** of the EC2 instance where your application is deployed.

Application activity



[Amazon CloudWatch details](#)

The lower-right corner hosts a link to Amazon CloudWatch where you can view other metrics for the instance. For a project based on an AWS Lambda template, invocation and error metrics are displayed.

- **JIRA:** If you have created a JIRA project for tracking your AWS CodeStar project, you can see issues in this tile:

JIRA

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

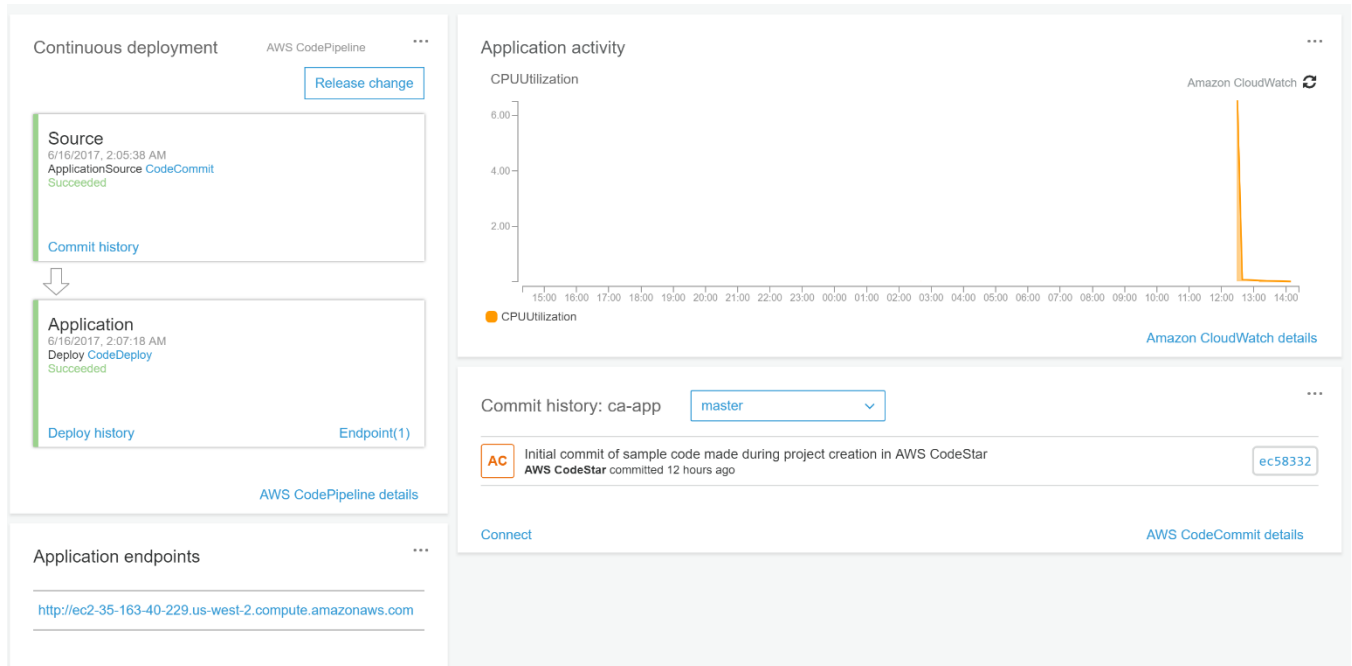
...

Connect

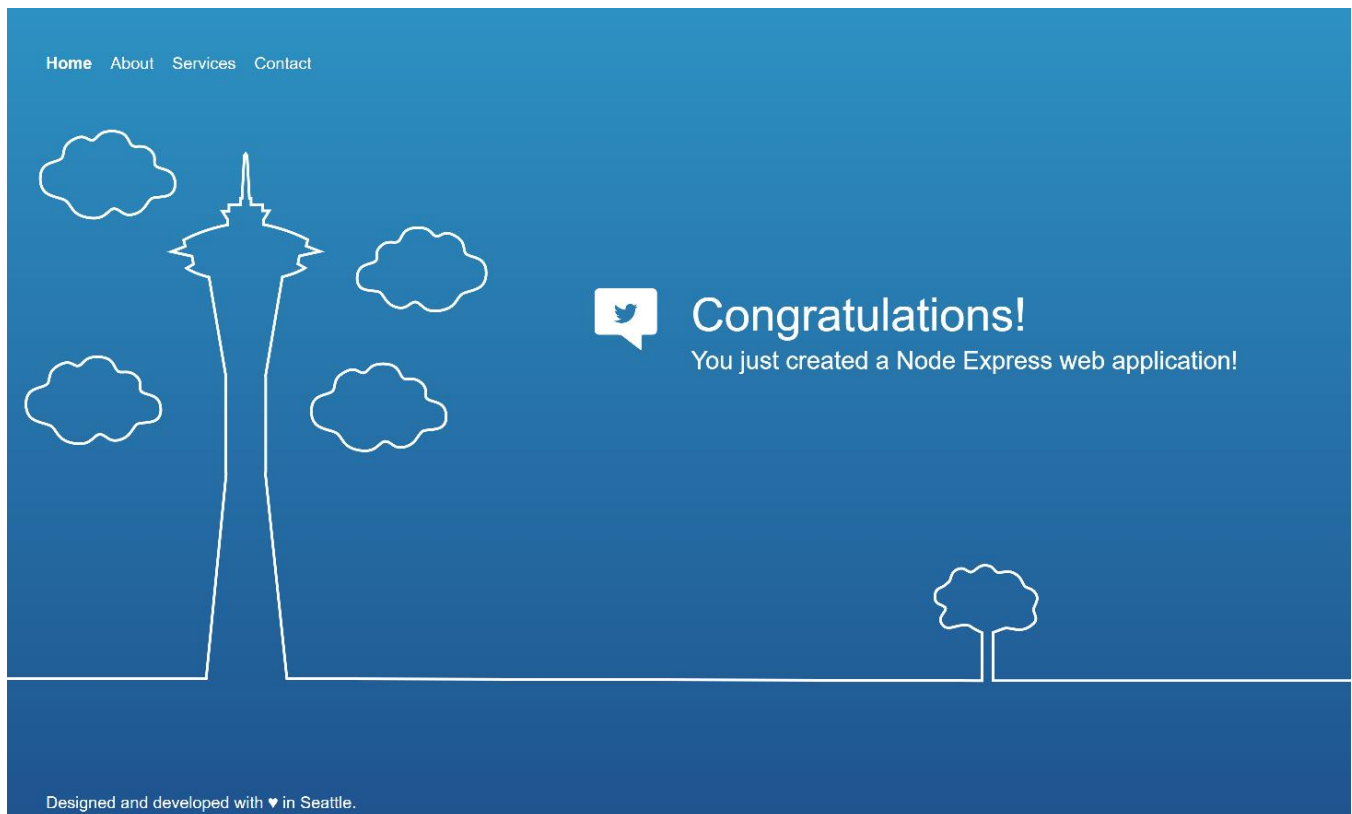
3. This Lab doesn't include a JIRA project, so click on the ellipsis in the upper-right corner of the tile and select **Remove from Dashboard**.

The **Add tile** button at the top of the Dashboard can be used to add tiles you have removed. Currently you are only allowed one of each type of tile in your Dashboard.

4. Click anywhere on a tile and drag it to a new location. Customize the layout to best suit your needs:



5. Click on the application endpoint in the **Application endpoints** tile to view the application included in the template:



Depending on your time of day, the background will change. You will commit a code change later to modify the appearance of the application.

6. Inspect the other navigation tabs on the left side of the project website:



The top three tabs open links to the AWS Code Services used by your project:

1. **Code** opens your repository in AWS CodeCommit,
2. **Deploy** opens your application in AWS CodeDeploy, and
3. **Pipeline** opens your release pipeline in AWS CodePipeline.

You won't go into the details of each of the code services in this Lab. Just know that if you need to diagnose issues or modify the project, the individual code services can be used.

The **Team** tab is where you manage your team. You will go into the details of teams in an upcoming Lab Step.

The **Extensions** tab contains all the extensions that integrate with AWS CodeStar. Only JIRA is available at the time of writing.

7. Click on the last navigation tab, **Project**.

The most interesting thing to see here is the list of all the **Project Resources** created by the project template:

#### Project Resources

Type	Name	ARN
AWS CloudFormation	stack/awscodestar-ca-app/6fdec530-51f4-11e7-97b...	arn:aws:cloudformation:us-west-2:638744092352:stack/awscodestar-ca-app/6fdec530-51f4-11e7-97b8-50a68a2012ba
AWS CodeCommit	ca-app	arn:aws:codecommit:us-west-2:638744092352:ca-app
AWS CodeDeploy	application:ca-app	arn:aws:codedeploy:us-west-2:638744092352:application:ca-app
AWS CodeDeploy	deploymentgroup:ca-app/ca-app-Env	arn:aws:codedeploy:us-west-2:638744092352:deploymentgroup:ca-app/ca-app-Env
AWS CodePipeline	ca-app-Pipeline	arn:aws:codepipeline:us-west-2:638744092352:ca-app-Pipeline
AWS IAM	role/CodeStarWorker-ca-app-WebApp	arn:aws:iam::638744092352:role/CodeStarWorker-ca-app-WebApp
AWS IAM	role/CodeStarWorker-ca-app-CodeDeploy	arn:aws:iam::638744092352:role/CodeStarWorker-ca-app-CodeDeploy
AWS IAM	role/CodeStarWorker-ca-app-CodePipeline	arn:aws:iam::638744092352:role/CodeStarWorker-ca-app-CodePipeline
Amazon EC2	instance/i-0274f21a6c45833b6	arn:aws:ec2:us-west-2:638744092352:instance/i-0274f21a6c45833b6
Amazon EC2	security-group/sg-770f1b0c	arn:aws:ec2:us-west-2:638744092352:security-group/sg-770f1b0c
Amazon S3	aws-codestar-us-west-2-638744092352-ca-app-pip...	arn:aws:s3::aws-codestar-us-west-2-638744092352-ca-app-pipeline

AWS CodeStar saved you a lot of time over manually configuring everything that is included! Notice that **AWS CloudFormation** includes a **stack** resource. That is how AWS CodeStar works behind the scenes. Each project template creates a stack in AWS CloudFormation. Of course, you don't need to know any of the details. AWS CodeStar does everything for you so you can focus on development.

If you ever need to delete an AWS CodeStar project, you can click the **Delete project** button. You will be given a choice of keeping the associated resources or also deleting the associated resources.

## Summary

In this Lab Step, you acquainted yourself with the various components of the AWS CodeStar project website. You learned what each tile in the Dashboard can be used for and how to arrange the Dashboard layout to suit your needs. You also saw where to find the list of all the resources associated with an AWS CodeStar project.

## Step 4 Developing Your AWS CodeStar Project

### Introduction

In this Lab Step, you will connect to the code repository for your AWS CodeStar project and commit a change. You will create the IAM credentials required to connect to the project's code repository. The EC2 instance you previously connected to will be used to commit the changes.



## Instructions

1. Navigate to **Services > Security, Identity & Compliance > IAM:**



Security, Identity & Comp.

IAM

Inspector

Certificate Manager

Directory Service

WAF & Shield

Compliance Reports

2. Click on **Users** in the left navigation panel.

3. In the **Users** table, click on **student**.



*Note:* You will see error messages. This is normal. You only have the permissions required to complete the Lab.

4. Click on the **Security credentials** tab.

5. Scroll down to the bottom and click on **Generate** under **HTTPS Git credentials for AWS CodeCommit:**

Permissions

Groups

Security credentials

Access Advisor

Sign-in credentials

Console password

N/A

Manage password

Console login link

N/A

Last login

2017-06-02 21:45 UTC+0800

Assigned MFA device

N/A

Signing certificates

N/A

Access keys

Use access keys to make secure REST or HTTP Query protocol requests to AWS service APIs. For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation. [Learn more](#)

Create access key

Access key ID	Created	Last used	Status	
No results				

SSH keys for AWS CodeCommit

Use SSH public keys to authenticate access to AWS CodeCommit repositories. [Learn more](#)

Upload SSH public key

SSH key ID	Uploaded	Status	
No results			

HTTPS Git credentials for AWS CodeCommit

Generate a user name and password you can use to authenticate HTTPS connections to AWS CodeCommit repositories. You can generate and store up to 2 sets of credentials. [Learn more](#)

Generate

This will show a pop-up dialog showing you your credentials.

6. Click **Show** to reveal the **Password** and copy the **User name** and **Password** to a file for reference:

## Git credentials generated



IAM has generated a user name and password for you to use when authenticating to AWS CodeCommit. You can use these credentials when connecting to AWS CodeCommit from your local computer and from tools that require a static user name and password. [Learn more](#)

**User name** student-at-123456789012

**Password** \*\*\*\*\* [Show](#)

This is the only time the password will be available to view, copy, or download. We recommend downloading these credentials and storing the file in a secure location. You can reset the password in IAM at any time.

[Download credentials](#)

[Close](#)

Now you have the credentials needed to connect to your AWS CodeStar code repository. All that is left is to get the URL of the repository.

7. Return to your AWS CodeStar project Dashboard and click **Connect** on the **Commit history** tile.

8. In the **Clone repository URL** section, click **Copy** and paste the repository into the file with your code repository credentials:

### Clone repository URL

HTTPS

`https://git-codecommit.us-west-2.amazonaws.com/v1/repos/ca-app`

[Copy](#)

9. Return to the SSH shell connected to the *dev-instance* EC2 instance and enter `cd` to ensure you are in your home directory of `/home/ec2-user`.

10. Enter the following command to tell Git to cache your credentials for a few hours after you enter them:

```
git config --global credential.helper 'cache --timeout=10800'
```

11. Tell Git your user name:

```
git config --global user.name "student"
```

This name will show up on the commits in your project dashboard.

12. Clone your AWS CodeStar project repository by entering:

```
git clone <YOUR_PROJECT_REPOSITORY_URL>
```

where `<YOUR_PROJECT_REPOSITORY_URL>` is the URL you copied a few instructions before. It will be similar to `https://git-codecommit.us-west-2.amazonaws.com/v1/repos/ca-app`.

13. When prompted, enter the **Username** and **Password** you saved in a text file earlier in this Lab Step.

*Tip:* The password generated by AWS is long and it is easy to make a typo when entering it. To avoid errors copy and paste the password.

14. Change the repository directory name to `ca-app`:

```
mv ca-app-<Unique_string> ca-app
```

*Note:* Change `ca-app-<Unique_string>` to the name of your repository.

This won't change the repository name. It will only simplify the instructions at the command-line by not having to enter your unique string following `ca-app` in this and later Lab Steps.

15. Change into the directory:

```
cd ca-app
```

16. Enter `ls` to get a quick overview of the project structure.

There are several files:

- **app.js:** JavaScript file that starts the server
- **appspec.yml:** Configuration file that instructs AWS CodeDeploy what steps to perform to deploy your application
- **package.json:** Metadata and dependencies related to your project
- **README.md:** Text file explaining the project template

There is no need to get into the details of the file contents at this time. However, it is good to know that the `appspec.yml` file specifies scripts that run during the deployment of your application. The scripts are contained in one of the two project directories:

- **public:** Static assets used for your application
- **scripts:** Scripts executed by AWS CodeDeploy during the deployment of your application

Now you can get the server running on your development machine.

17. Install npm package, project dependencies using Node package manager (npm) and start the Node.js server:

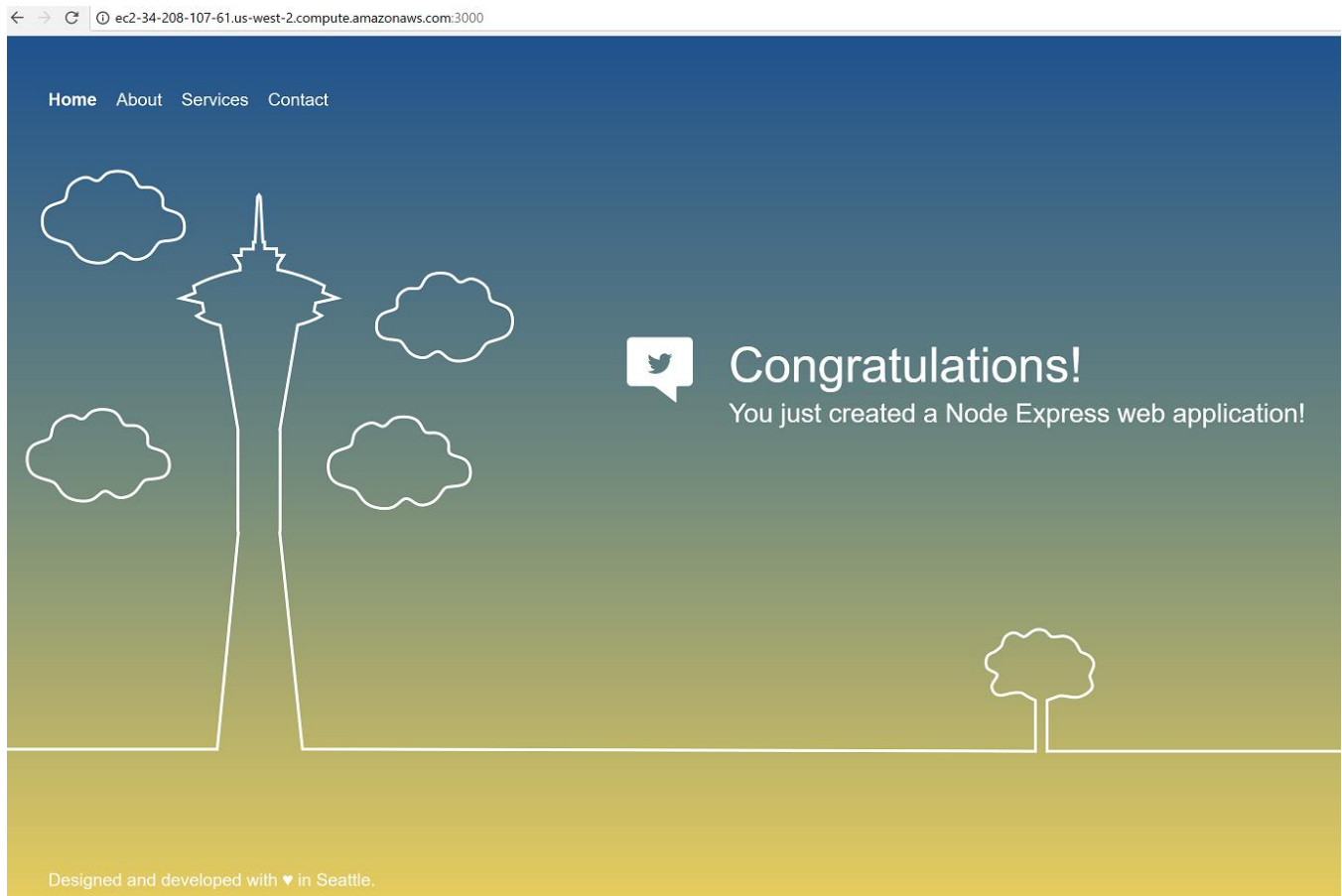
```
sudo yum install nodejs npm --enablerepo=epel
npm install
node app.js
```

While the server is running you won't be able to enter new commands. That won't be a problem. Now you can test that the development server is serving the application.

18. Navigate to **Services > Compute > EC2 > INSTANCES > Instances** and select the instance named **dev-instance**.

19. Copy the **Public DNS**.

20. Open a new browser tab and paste the public DNS and append `:3000` to the end and press enter:



Now that you verified the application works on the development machine, you can make some code changes.

21. Return to the SSH shell and press Ctrl+C to kill the running Node.js server.

22. Enter the following multiline command at the shell prompt to update a file in the project:

```
echo 'var idx = Math.floor(new Date().getHours());
var body = document.getElementsByTagName("body")[0];
var idxStep = 1;
var refreshRate = 1000;

function adjustIdx() {
  if (idx <= 0) {
    // Start increasing idx
    idxStep = 1;
  } else if (idx >= 23) {
    // Start decreasing idx
    idxStep = -1;
  }
  idx += idxStep;
  body.className = "heaven-" + idx;
}

body.className = "heaven-" + idx;
setInterval(adjustIdx, refreshRate);' > public/js/set-background.js
```

23. Test the changes by running the server again with `node app.js` and refresh the browser tab with your development application.

24. Stop the Node.js server with Ctrl+C.

25. View the local repository status:

```
git status
```

```
On branch master
Your branch is up-to-date with 'origin/master'.
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   public/js/set-background.js

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        node_modules/

no changes added to commit (use "git add" and/or "git commit -a")
```

This tells you that you are on the master branch and working from the initial code commit. The output also shows the `set-background.js` file was modified. You need to add the file to stage it before committing.

26. Add the modified file to the staged changes in the commit:

```
git add public/js/set-background.js
```

27. Commit the staged changes to the local repository and add a short message about the changes:

```
git commit -m "animation"
```

28. Push the changes in your local repository to the remote AWS CodeStar project repository so they are synchronized:

```
git push
```

Now that you have made a change to your code, you will see how the changes are deployed in the next Lab Step.

## Summary

In this Lab Step, you committed a code change to your AWS CodeStar project repository. You created the required credentials and tested the application on your development server.

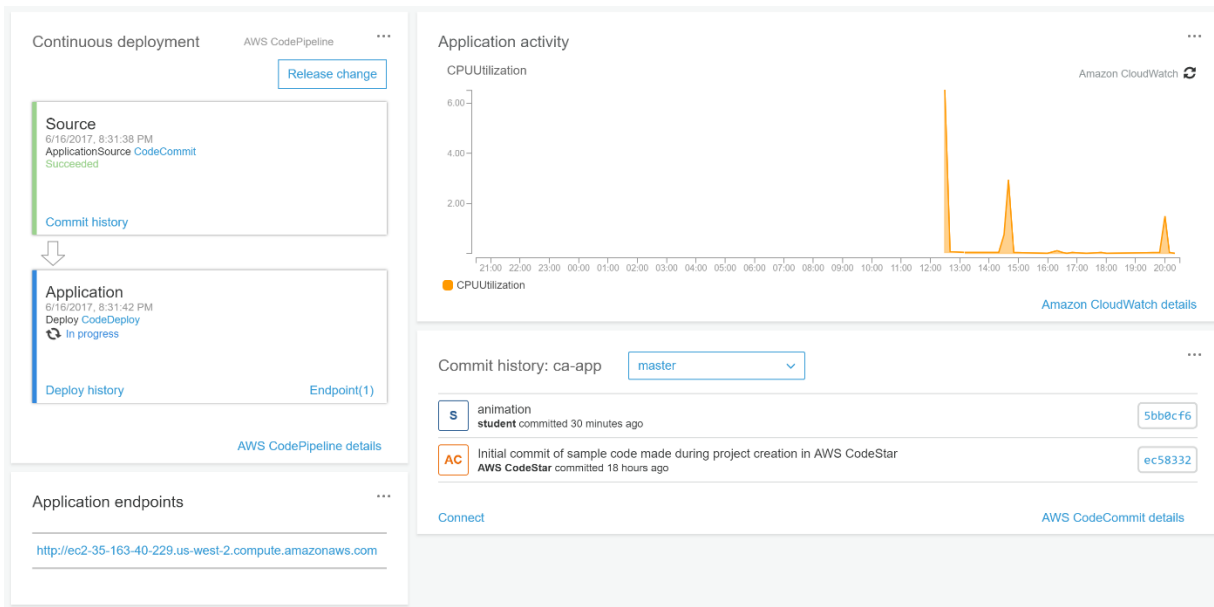
# Step 5 Deploying Your AWS CodeStar Project

## Introduction

In this Lab Step, you will follow your AWS CodeStar project's pipeline as the new version of the application you committed in the previous Lab Step is deployed. You will also take a quick look at the AWS CodeCommit repository and AWS CodeDeploy application. When there are problems with your AWS CodeStar project, it is useful to know how to debug the issues with the help of AWS CodeCommit and AWS CodeDeploy.

## Instructions

1. Return to your AWS CodeStar project Dashboard:



There are a few things to notice since you were here last:

1. Your **commit** is now visible in the **Commit history** tile
2. Your **Application activity** tile might show some spikes in **CPUUtilization** if your application has already been deployed
3. Your **Continuous deployment** pipeline may show one of the pipeline stages **In progress** or you may see a recent timestamp inside each stage box telling you the new version has been deployed.

If you missed the release flowing through the stages of the pipeline, go ahead and click **Release change** and **Continue** in the pop-up.

2. Inspect the code commit by clicking on the button containing the commit ID on the right side of the **animation** commit.
3. Look at the code changes:



Commit 5bb0cf623cbe1236260b7c57605dae7d572ecd85

student authored 34 minutes ago

Parent: [ec583320](#)

Commit message

animation

1 changed file from [ec583320](#)

[Hide whitespace changes](#)

[Unified](#)

[Split](#)

public/js/set-background.js

```
1 1 var idx = Math.floor(new Date().getHours());
2 2 var body = document.getElementsByTagName("body")[0];
3 + var idxStep = 1;
4 + var refreshRate = 1000;
5 +
6 + function adjustIdx() {
7 +   if (idx <= 0) {
8 +     // Start increasing idx
9 +     idxStep = 1;
10 +   } else if (idx >= 23) {
11 +     // Start decreasing idx
12 +     idxStep = -1;
13 +   }
14 +   idx += idxStep;
15 +   body.className = "heaven-" + idx;
16 + }
17 +
18 body.className = "heaven-" + idx;
19 + setInterval(adjustIdx, refreshRate);
```

Additions appear in green and removals would appear in red, if any were present. This is an easy way to keep track of what is happening to the code in your AWS CodeStar project. If your project has multiple branches, the commit history tile will allow you to choose between them.

4. Return to your AWS CodeStar Dashboard and click on **CodeDeploy** inside the **Application** pipeline stage in the **Continuous deployment** tile.

This opens your application in AWS CodeDeploy:

AWS CodeDeploy ▾ Applications > ca-app

### Application details: ca-app ?

Manage your application. Create, edit, or set up triggers for deployment groups. View application revision details.

#### Deployment groups

View, edit, and redeploy revisions to your deployment groups. View details and set up triggers to receive notifications about deployment group events. [Learn more](#)

[Create deployment group](#) [Actions ▾](#) [Refresh](#)

Filter by deployment group name Deployment groups per page 10 ▾ « < Viewing 1 to 1 of 1 deployment groups > »

	Name	Status	Last attempted revision	Last successful revision	Triggers
<input type="radio"/>	ca-app-Env	<span>✓ Succeeded</span>	June 16, 2017 12:32:27 PM UT0	June 16, 2017 12:32:27 PM UT0	

« < Viewing 1 to 1 of 1 deployment groups > »

▸ Revisions

You can see the **Deployment Groups** created for deploying your application. In this case there will be just one with a **Name** ending in **-Env**. The **Status** column will tell you if your last deployment **Succeeded** or failed. The time of the **Last attempted revision** and **Last successful**

**revision** are also recorded. A revision is how AWS CodeDeploy refers to different versions of the application.

5. Click the black triangle to the left of the deployment group name to expand and show the deployment group details.

Notice that by default **Rollback enabled** is **false**. That means if your deployment fails, AWS CodeDeploy will not attempt to deploy the last successful version. That is something you might consider changing when you use AWS CodeStar for one of your projects.

On the right half of the drop-down, you see how to create **Triggers** to publish messages about deployment using Amazon Simple Notification Service (SNS) and a link to **Add alarms** in AWS CloudWatch. It may be useful, for example, to create a trigger when deployment fails or to add an alarm when your deployment group instance is overloaded.

6. Scroll down the page and expand the **Revisions** section.

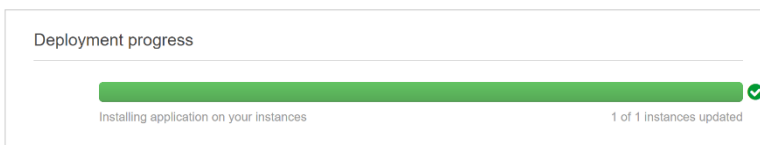
Each revision that was attempted to be deployed is recorded here along with a link to where the artifacts are located on Amazon S3.

7. Return to your AWS CodeStar Dashboard and click on **Deploy history** on the bottom of the **Application** pipeline stage.

This opens details of the most recent deployment:

Deployment: d-M06SFUYUM

✔ Deployment Succeeded



▸ Deployment details

▼ Instance activity

Filter <b>Status</b> ▲		Instances per page 10 ▼		< Viewing 1 to 1 of 1 instances >		
Instance ID	Start time	End time	Duration	Status	Most recent event	Events
i-0274f21a6c45833b6	June 16, 2017 12:31:50 PM UTC	June 16, 2017 12:32:23 PM UTC	33 secs	Succeeded	ValidateService	<a href="#">View events</a>

The **Deployment progress** bar at the top shows the state of the deployment operation. The **Deployment details** section shows information similar to what you saw on the AWS CodeDeploy application page. The **Instance activity** section tells you the start and end times as well as the **Duration** of the deployment. The **View events** link on the right is the most useful for investigating failed deployments.

8. Click **View events** to open deployment life cycle events the instance in the deployment group went through to deploy your application:

Event	Start time	End time	Duration	Status	Logs	Actions
ApplicationStop	June 16, 2017 12:31:50 PM UTC	June 16, 2017 12:31:50 PM UTC	less than one second	Succeeded		
DownloadBundle	June 16, 2017 12:31:51 PM UTC	June 16, 2017 12:31:51 PM UTC	less than one second	Succeeded		
BeforeInstall	June 16, 2017 12:31:53 PM UTC	June 16, 2017 12:31:53 PM UTC	less than one second	Succeeded		
Install	June 16, 2017 12:31:54 PM UTC	June 16, 2017 12:31:54 PM UTC	less than one second	Succeeded		
AfterInstall	June 16, 2017 12:31:56 PM UTC	June 16, 2017 12:32:20 PM UTC	24 secs	Succeeded		
ApplicationStart	June 16, 2017 12:32:21 PM UTC	June 16, 2017 12:32:21 PM UTC	less than one second	Succeeded		
ValidateService	June 16, 2017 12:32:23 PM UTC	June 16, 2017 12:32:23 PM UTC	less than one second	Succeeded		

In case of a failed deployment, one of the events will record the failure and provide a link under the **Logs** column to investigate the command and logs related to the failure. If you recall, the `appspec.yml` file in the code project was used to instruct AWS CodeDeploy how to deploy your application. Your project provides different scripts to run for some of the events listed in the table.

9. Finally, return to the AWS CodeStar Dashboard and click on the URL in the **Application endpoints** tile.

If everything went well, you will see the latest version of your application including the animation commit deployed and available to the world.

## Summary

In this Lab Step, you followed the release pipeline of your project. You learned how to track changes in the project code repository. You also saw the configuration of your project deployment and learned how to diagnose issues if a deployment fails.

# Step 6 Managing Your AWS CodeStar Project Team

## Introduction

As the owner of an AWS CodeStar project, you are able to add other team members to a project. The team members will need an IAM user in your account. AWS CodeStar provides three built-in roles for team members to be attached to. If you have permission to modify policies, you can customize the roles to best suit your needs. The default policies work well in general and are what you will use in this Lab Step. You will assign the two team members to different roles and see an example difference between the permissions each has.

## Instructions

1. Return to your AWS CodeStar project Dashboard and click on the **Team** tab then click **Add team member**.

The **Project Team** view is now visible:

**Project Team**  
Manage users and permissions in your project.

▲ Add team member

Select user ▼

Display Name

Type display name

Email Address

Type email address (Optional)

Project Role

Viewer ▼

Remote Access

☐

Add

Team member list (1)

	Name ▲	Email	Role	Remote Access	
S	student [You]	student@email.com	Owner	<a href="#">Add Public SSH key</a>	<a href="#">Remove</a>   <a href="#">Edit</a>

In the upper portion, you can add team members and the lower portion lists all of the team members including their **Role**.

2. Click on the **Select user** drop-down menu and click on **Logan**.

3. Set the team member values for Logan to:

- **Project Role:** *Contributor*
- **Remote Access:** *Checked* (This allows the team member to upload an SSH public key to connect to EC2 instances)

The difference between the default **Project Roles** is:

- **Viewer:** Access to the project dashboard and able to view a few project resources
- **Contributor:** Everything Viewer can access plus view, modify, and access all project resources
- **Owner:** Everything Contributor has plus adding and removing team members, and deleting the project

4. Click **Add**.

You will see **Logan** appear in the **Team member list**.

5. Click **Add team member** and select **Bessie** from the drop-down menu.

6. Enter the following values and click **Add**:

- **Project Role:** *Viewer*
- **Remote Access:** *Unchecked*

Now you can briefly experience the differences between the project roles.

7. At the top of this Lab page, click on the **OPEN AWS CONSOLE** button.

This will sign you out of the student user and allow you to sign in as a different user.

8. Log in to AWS using the team member in the viewer role:

- **User Name:** *Bessie*
- **Password:** *Lab-Viewer1*

9. Navigate to **Services > Developer Tools > CodeStar**.

10. Click on your project name.

11. In the **User Profile Information** pop-up, enter:

- **Display Name:** *Bessie*
- **Email:** *Bessie@email.com*

12. Click **Save**.

Observe that the viewer role has access to the same dashboard you created.

13. Click on a code commit ID and see that a viewer is allowed to view code changes.

14. Return to the project dashboard and click **Release change** in the **Continuous deployment** tile, then **Continue**.

You will receive an error message stating that you are not authorized to perform that action:

## Release change



### Error

User: arn:aws:iam::123456789012:user/Bessie is not authorized to perform: codepipeline:StartPipelineExecution on resource: arn:aws:codepipeline:us-west-2:123456789012:ca-app-Pipeline

Releasing a change will detect the most recent change in each location configured in your source action(s), and run that change through the pipeline. Do you want to continue?

Cancel

Continue

15. At the top of this Lab page, click on the **OPEN AWS CONSOLE** button and sign in again with the following credentials:

- **User Name:** *Logan*
- **Password:** *Lab-Contributor1*

The user Logan is in the contributor role, which has additional permissions than the viewer role.

16. Navigate to the project dashboard and in the **User Profile Information** pop-up, enter:

- **Display Name:** *Logan*
- **Email:** *Logan@email.com*

17. Click **Release change**, then **Continue**.

The contributor has permission to perform this action:

## Source

6/16/2017, 11:13:05 PM

ApplicationSource [CodeCommit](#)



In progress

[Commit history](#)

18. Click on the **Team** tab.

Notice that you can only remove yourself from the team and not other members. That is a distinction between the contributor and owner roles.

19. One last time, click on the **OPEN AWS CONSOLE** button and sign in with the student credentials given in the **YOUR LAB DATA** section of the Lab.

You need to be logged in as student to complete the next Lab Step.

## Summary

In this Lab Step, you added team members to your AWS CodeStar project. You also experienced the differences between the built-in project roles: Owner, Contributor, and Viewer.

## Step 7 Cleaning Up Your AWS CodeStar Project


### Introduction


The time has come to wind down your AWS CodeStar project. In this Lab Step you will clean up the project and all associated resources. This is a good habit to get into in practice to help minimize your cloud costs.

### Instructions

1. Return to your AWS CodeStar project website and click on the **Project** tab.
2. Scroll to the bottom and click **Delete project**.
3. Enter your project's name in the pop-up dialog:

## Delete project



 **You are about to delete an AWS CodeStar project**  
Are you sure you want to delete the project ca-app? This cannot be undone. [Learn more](#)

Type the following project ID to confirm:  
**ca-app**

*Enter the id of the project to be deleted*

☒ Keep associated AWS resources but delete project in AWS Codestar. [Learn more](#)

[Cancel](#)[Delete](#)

4. *Uncheck* the **Keep associated AWS resources but delete project in AWS Codestar** checkbox.

By default AWS CodeStar will leave the project resources after you delete the project. In this case, the project was just for demonstration purposes and there is nothing to persist.