

cfsummit.vegas

## Part 1 - Create a Local Development Environment

### Step 1 - Establish the Repository

- Log into BitBucket and create a new repository
  - Create a new project
  - Create a new repository
  - Assign a default branch name, such as production
  - Select ColdFusion as the language

## Create a new repository

[Import repository](#)

Workspace  Code Monkey

Project name\*  x

Repository name\*

Access level  Private repository

Uncheck to make this repository public. Public repositories typically contain open-source code and can be viewed by anyone.

Include a README?

Default branch name

Include .gitignore?

▼ Advanced settings

Description

Forking

Language  x v

Create repository [Cancel](#)

- Clone the repository into your local project folder.

## Step 2 - Local Repository Setup

- In the project directory, create an `app` directory.
- In the `app` directory, create an `index.cfm` file.
- In the `index.cfm` file, paste the following code:

```
<h2>cfsummit.vegas</h2>
<cfoutput>
```

```
<p>#server.coldfusion.productName# (#server.coldfusion.productLevel#)
version #server.coldfusion.productVersion#</p>
<p>#dateFormat( now(), 'long' )#</p>
</cfoutput>
```

## Step 3 - Use the CF Docker Image to create a local development environment

On an Intel / AMD64 / Windows environment

- In VS Code, create a file called `docker-compose.yml` in the root of the project folder with the following content:

```
version: "3.8"

services:
  cfsummit:
    image: adobecoldfusion/coldfusion:latest
    platform: linux/amd64
    container_name: cfsummit
    hostname: cfsummit
    environment:
      - acceptEULA=YES
      - password=cfsummit
      - enableSecureProfile=false
      - installModules=all
    ports:
      - 8500:8500
    volumes:
      - ./app:/app

networks:
  cfsummit_network:
```

On an ARM based (Apple Silicon) environment

Adobe has not yet created Docker images that work seamlessly on Apple Silicon. Due to this, I can't demonstrate a Docker environment using the ColdFusion Docker images on a Mac. Instead, I will use the Ortus Solutions CommandBox Docker images to spin up ColdFusion 2023.

- In Visual Studio Code, create a file called `docker-compose.yml` in the root of the project folder with the following content:

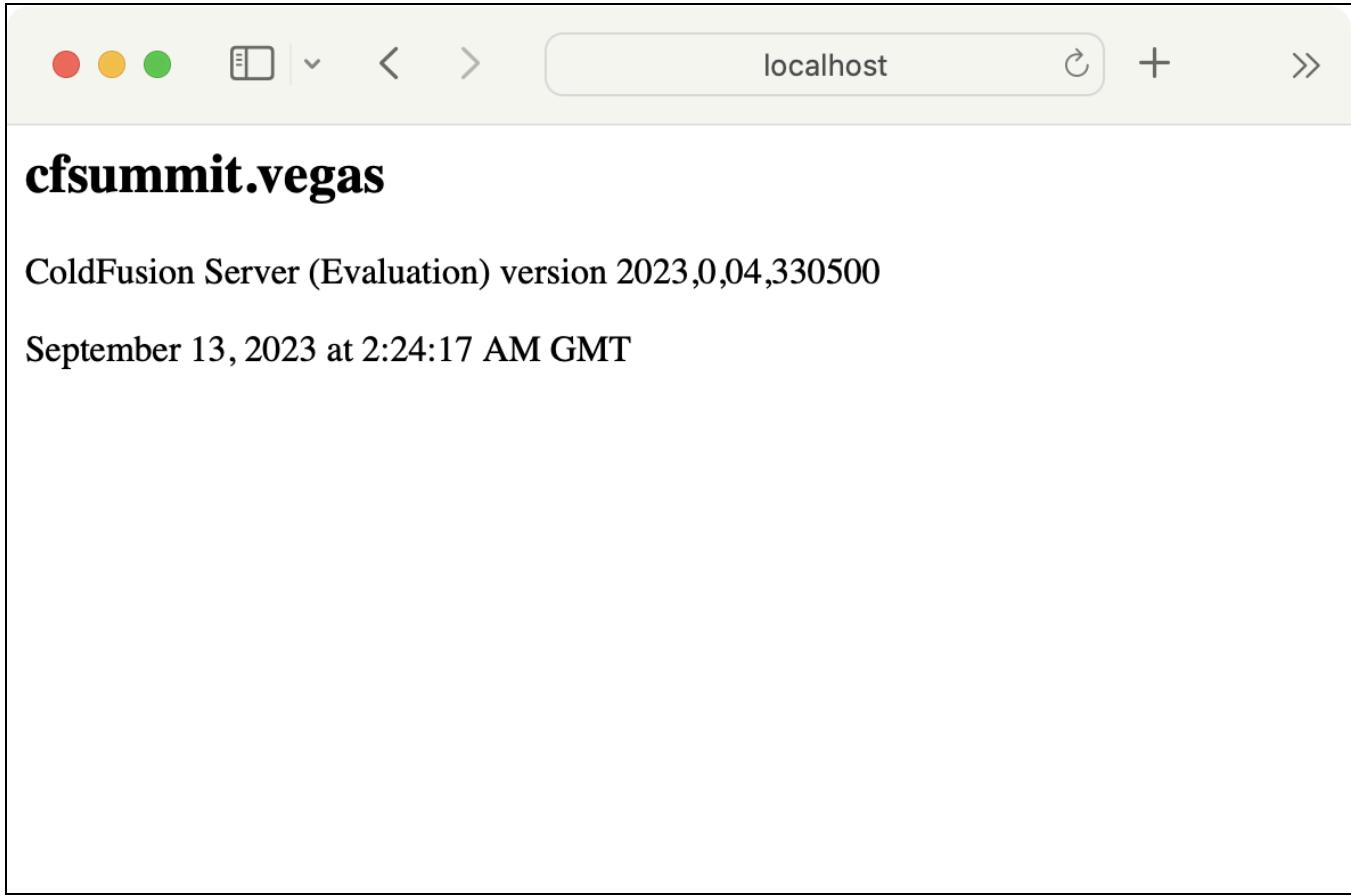
```
version: "3.8"

services:
  cfsummit:
    image: ortussolutions/commandbox:adobe2023
    container_name: cfsummit
    hostname: cfsummit
    environment:
      - BOX_INSTALL=true
      - cfconfig_adminPassword=cfsummit
      - TZ="US/Pacific"
    ports:
      - 80:8080
      - 8500:8500
    volumes:
      - ./app:/app

networks:
  cfsummit_network:
```

Once the `docker-compose.yml` file is created

- In a command line terminal, browse to the project folder and enter `docker compose up -d`.
- Check to see that the server is up and running from the Docker Dashboard.
- In a browser, browse to `localhost`. The rendered contents of our `index.cfm` file should resemble the screenshot below. Refreshing the page will update the timestamp.



Let's make that look better

- Replace the contents of the `index.cfm` file with the following:

```
<!doctype html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>cfsummit.vegas</title>

        <!-- Bootstrap 5.3.1 -->
        <link
            href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css"
            rel="stylesheet"
            integrity="sha384-4bw+/aepP/YC94hEpVNvgiZdgIC5+VKNBQNGCHeKRQN+PtmoHDEXuppvnDJzQIu9"
            crossorigin="anonymous">
            <script
                src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/js/bootstrap.bundle.min.js"
                integrity="sha384-HwwvtgBNo3bZJJLYd8oVXjrBZt8cqVSpeBNS5n7C8IVInixGAoxmn1MuBnhbgrkm"
                crossorigin="anonymous"></script>
```

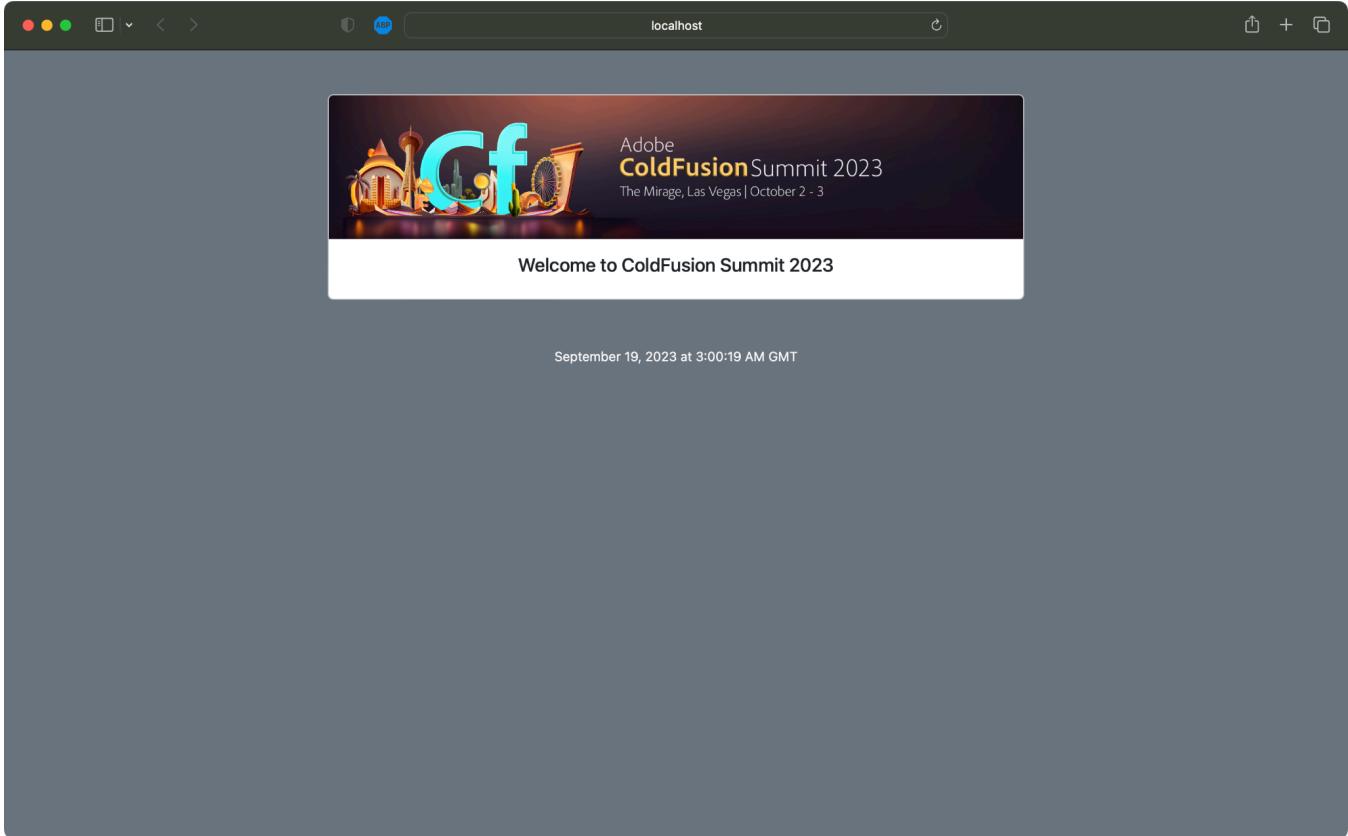
```

</head>

<body class="bg-dark py-5">
    <cfoutput>
        <div class="container">
            <div class="row">
                <div class="col-11 col-sm-10 col-md-9 col-lg-8 col-xxl-7
mx-auto">
                    <div class="card">
                        
                        <div class="card-body">
                            <h5 class="card-title text-center">Welcome
to ColdFusion Summit 2023</h5>
                        </div>
                    </div>
                    <p class="text-light text-center
my-5"><small>#dateTimeFormat( now(), 'long' )#</small></p>
                </div>
            </div>
        </cfoutput>
    </body>
</html>

```

- Refresh the browser. The rendered contents of our `index.cfm` file should resemble the screenshot below. Refreshing the page will update the timestamp.



## Part 2 - AWS Identity Access Management (IAM) Setup

During this part, we are going to set up IAM on AWS for the user that will be used to push code via BitBucket Pipelines and AWS CodeDeploy . We will create three elements in AWS IAM:

- A Role,
- A User,
- and A Group

### IAM Role

Let's start with the IAM Role, since the user will use this role to be granted permissions to push the code using AWS CodeDeploy.

- From the IAM Dashboard, select “Roles” in the left column.
- Click on the “Create Role” button.

The screenshot shows the AWS IAM Roles page. On the left, there's a navigation sidebar with options like Dashboard, Access management, Access reports, and Related consoles. The main area displays a table of existing roles, each with a Role name, Trusted entities, and Last activity. Below the table, there's a section titled 'Roles Anywhere' with three options: 'Access AWS from your non AWS workloads' (using X.509 Standard), 'X.509 Standard' (using your own PKI infrastructure or AWS Certificate Manager Private Certificate Authority), and 'Temporary credentials' (using temporary credentials for enhanced security). At the top of the main area, there's a 'Create role' button, which is highlighted with a red arrow.

## Step 1 - Select trusted Entity

- Select “AWS Service” as the Trusted Entity Type.
- Select “EC2” as the Use Case.
- Click the “Next” button at the bottom of the page.

The screenshot shows the 'Select trusted entity' step of creating a new AWS IAM role. The 'Trusted entity type' section is expanded, showing five options: 'AWS service' (selected), 'AWS account', 'Web identity', 'SAML 2.0 federation', and 'Custom trust policy'. Below this, the 'Use case' section is shown, with a dropdown menu containing 'EC2' (selected) and other options like 'EC2 Role for AWS Systems Manager' and 'EC2 Spot Fleet Role'. The bottom of the screen shows standard AWS navigation links and copyright information.

## Step 2 - Add Permissions

- Add the following permissions to the Role:
  - AmazonEC2FullAccess
  - AmazonS3FullAccess
  - AWSCodeDeployFullAccess
- Click the “Next” button at the bottom of the page.

The screenshot shows the AWS IAM 'Create role' wizard at Step 2: 'Add permissions'. The user has searched for 'AmazonEC2' and found 16 matches. The 'AmazonEC2FullAccess' policy is highlighted with a blue selection bar and a red arrow pointing to it.

Policy name	Type	Description
AmazonEC2ContainerRegistryFullAccess	AWS managed	Provides administrative access to Amazon EC2 Container Registry
AmazonEC2ContainerRegistryReadOnlyAccess	AWS managed	Provides full access to Amazon EC2 Container Registry
AmazonEC2ContainerRegistryReadAccess	AWS managed	Provides read-only access to Amazon EC2 Container Registry
AmazonEC2ContainerServiceAutoscalingTaskRole	AWS managed	Policy to enable Task Autoscaling for Amazon EC2 Container Service
AmazonEC2ContainerServiceEventsRole	AWS managed	Policy to enable CloudWatch Events for Amazon EC2 Container Service
AmazonEC2ContainerServiceforECSRole	AWS managed	Default policy for the Amazon EC2 Container Service
AmazonEC2ContainerServiceRole	AWS managed	Default policy for Amazon ECS service
<b>AmazonEC2FullAccess</b>	AWS managed	Provides full access to Amazon EC2 via the AWS Management Console
AmazonEC2ReadOnlyAccess	AWS managed	Provides read only access to Amazon EC2
AmazonEC2RoleforAWSCodeDeploy	AWS managed	Provides EC2 access to S3 bucket to do code deployment
AmazonEC2RoleforAWSCodeDeployDataPipeline	AWS managed	Provides EC2 limited access to S3 bucket for AWS Code Deploy Data Pipeline
Default policy for the Amazon EC2 Role		

## Step 3 - Name, Review, and Create

- Name the Role **AWSCodeDeployRole**.
- Click the Create Role button at the bottom of the page.

The screenshot shows the AWS IAM 'Create role' wizard. The current step is 'Step 1: Select trusted entity'. The 'Role name' field is highlighted with a red arrow, containing the value 'AWSCodeDeployRole'. The 'Description' field contains the text 'Allows EC2 instances to call AWS services on your behalf.' Below the fields, there is a code editor showing the JSON trust policy:

```
1 - {
2 -   "Version": "2012-10-17",
3 -   "Statement": [
4 -     {
5 -       "Effect": "Allow",
6 -       "Action": [
7 -         "sts:AssumeRole"
8 -       ],
9 -       "Principal": [
10 -         "Service": [

```

## Step 4 - Add Additional Principals to the Trust Policy

- From the IAM > Roles page, click on the newly created AWSCodeDeployRole role to edit it.
- Select the Trust Relationships tab.
- Click the Edit Trust Policy button.

The screenshot shows the AWS IAM Roles page. On the left, there's a sidebar with 'Identity and Access Management (IAM)' selected. The main area displays the 'AWSCodeDeployRole' details. The 'Summary' section includes fields like Creation date (September 18, 2023, 20:24 (UTC-07:00)), ARN (arn:aws:iam::894991230159:role/AWSCodeDeployRole), Last activity (-), and Maximum session duration (1 hour). Below the summary are tabs for 'Permissions', 'Trust relationships' (which is highlighted with a red arrow), 'Tags', 'Access Advisor', and 'Revoke sessions'. The 'Trust relationships' tab shows a JSON policy document:

```
1 {  
2     "Version": "2012-10-17",  
3     "Statement": [  
4         {  
5             "Effect": "Allow",  
6             "Principal": {  
7                 "Service": "ec2.amazonaws.com"  
8             },  
9             "Action": "sts:AssumeRole"  
10        }  
11    ]  
12}
```

At the bottom right of this section is a 'Edit trust policy' button, which is also highlighted with a red arrow.

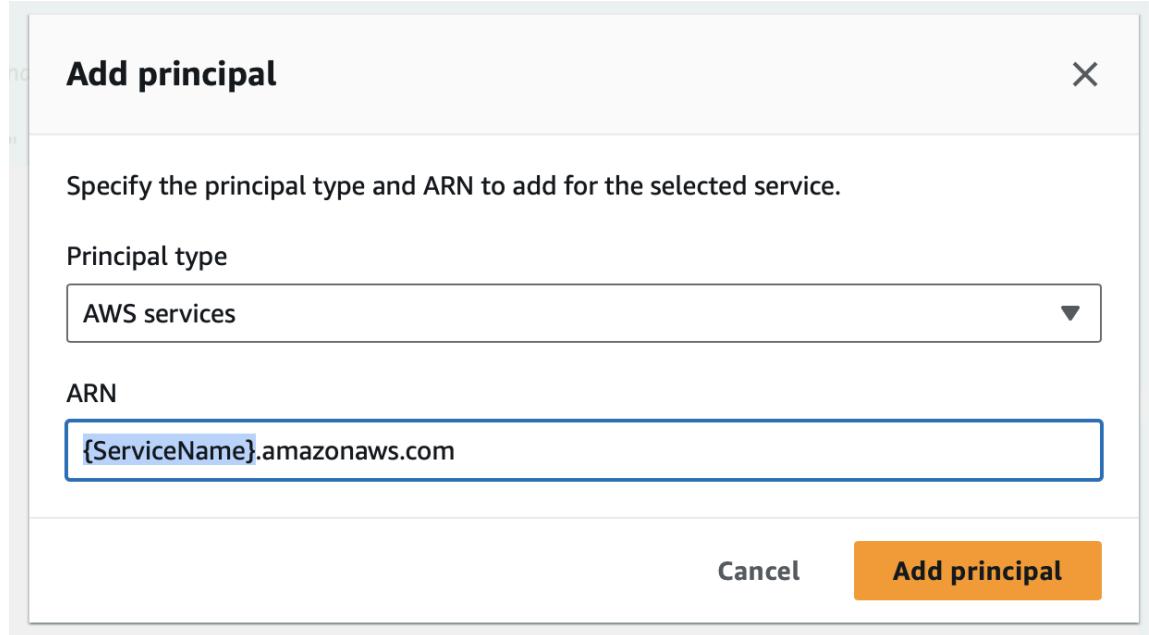
Click the Add button next to Add a Principal

The screenshot shows the 'Edit trust policy' page for the 'AWSCodeDeployRole'. The left pane displays the current policy JSON:

```
1 {  
2     "Version": "2012-10-17",  
3     "Statement": [  
4         {  
5             "Effect": "Allow",  
6             "Principal": {  
7                 "Service": "ec2.amazonaws.com"  
8             },  
9             "Action": "sts:AssumeRole"  
10        }  
11    ]  
12}
```

The right pane contains sections for 'Edit statement' (with a 'Remove' button), 'Add actions for STS' (with a 'Filter actions' input field), and two lists of actions: 'Access level - read' and 'Access level - read or write'. Under 'Access level - read or write', the 'AssumeRole' action is checked. At the bottom right of the right pane is a 'Add a principal' button, which is highlighted with a red arrow.

- Under Principal Type select AWS Services.
- Under ARN, replace {ServiceName} with “codedeploy.YOUR\_EC2\_INSTANCE\_REGION”.  
For example, if you intend to spin up your EC2 instances in the AWS region us-east-2, you would replace {ServiceName} with codedeploy.us-east-2.
- Click the Add principal button at the bottom of the modal window.



Once you add the CodeDeploy principal to the role, your trust policy should look like this:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "ec2.amazonaws.com",
          "codedeploy.us-east-2.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

As a shortcut, you can copy and paste this JSON snippet directly into the Edit Trust Policy page.

- Finally, click the “Update Policy” button at the bottom of the page.

The AWS IAM Role is now set up to be used by the IAM User, which we will create next.

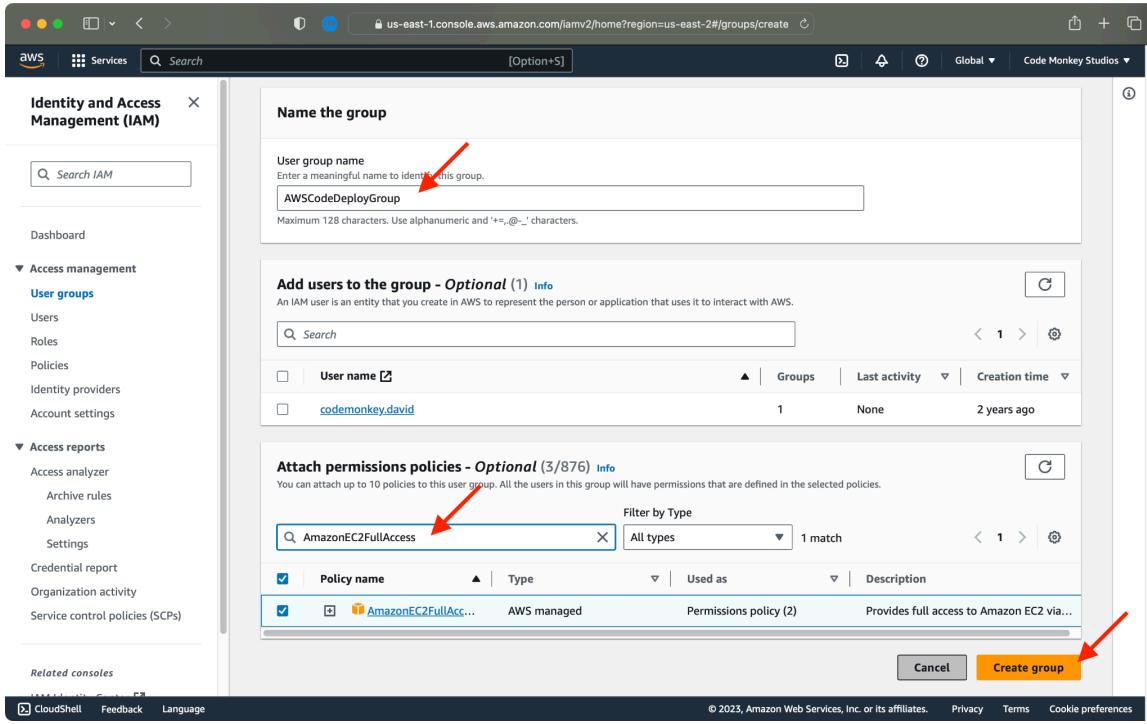
## IAM Group

Next, we will create an IAM Group for the User to join. This group sets the permissions for the user to deploy code.

- From the IAM Dashboard, select “User Groups” in the left column.
- Click on the “Create Group” button in the upper right side of the user groups page.

The screenshot shows the AWS IAM User Groups page. On the left, there's a navigation sidebar with 'Identity and Access Management (IAM)' selected. Under 'Access management', 'User groups' is also selected. The main area displays a table of user groups. One group, 'Administrators', is listed with the following details: Group name (Administrators), Users (1), Permissions (Defined), and Creation time (2 years ago). At the top right of the table, there are 'Delete' and 'Create group' buttons. A red arrow points to the 'Create group' button. The browser address bar shows 'us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-2#/groups'.

- Name the group AWSCodeDeployGroup.
- Add the following permissions to the Role:
  - AmazonEC2FullAccess
  - AmazonS3FullAccess
  - AWSCodeDeployFullAccess
- Click the “Create Group” button at the bottom of the page.



## IAM User

Finally, we will create an IAM User that will utilize the IAM Role and IAM Group we've created to deploy code using AWS CodeDeploy.

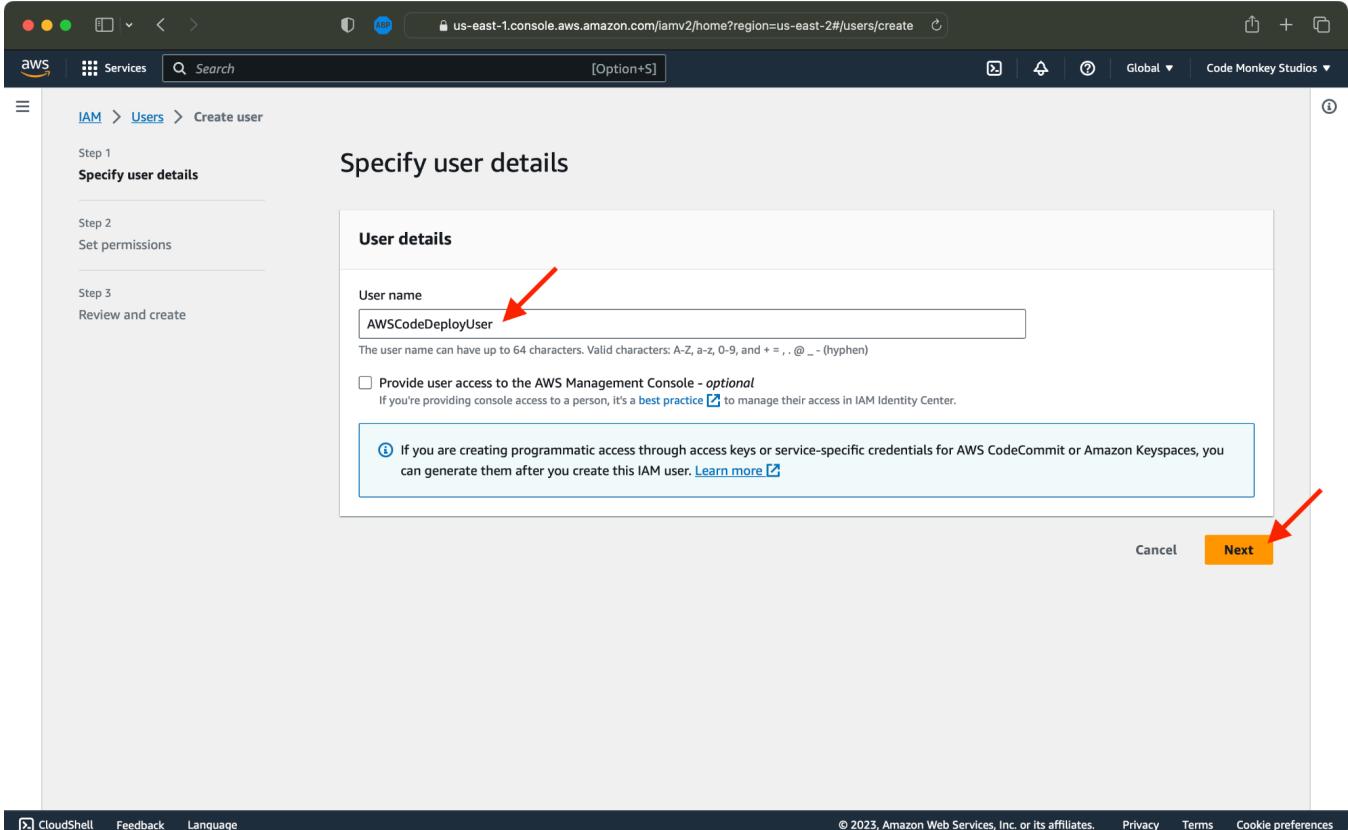
- From the left column of the AWS IAM Service, click on “Users” in the left column.
- Click on the “Create User” button on the right side of your screen.

The screenshot shows the AWS IAM (Identity and Access Management) service in a web browser. The URL is [us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-2#/users](https://us-east-1.console.aws.amazon.com/iamv2/home?region=us-east-2#/users). The left sidebar has a 'Users' link under 'Access management' highlighted with a red arrow. The main content area shows a table of users with one entry: 'codemonkey.david'. A red arrow points to the 'Create user' button in the top right of the table header. The table includes columns for User name, Path, Group, Last activity, MFA, Password age, and Consol.

User name	Path	Group	Last activity	MFA	Password age
codemonkey.david	/	1	769 days ago	-	898 days

## Step 1 - Specify User Details

- Enter the username AWSCodeDeployUser.
- Click on the “Next” button at the bottom of the page.



## Step 2 - Set Permissions

- Under Permissions options, select Add user to group.
- Under User groups, select the AwsCodeDeployGroup we created in the previous step.
- Click the Next button at the bottom of the page.

The screenshot shows the 'Set permissions' step of the 'Create user' wizard. In the 'Permissions options' section, the radio button for 'Add user to group' is selected, highlighted by a red arrow. Below it, a table lists three groups: 'Administrators', 'AWSCodeDeployGroup' (which is checked), and 'AWSCodeDeployGroup\_Ad...'. A second red arrow points to the 'AWSCodeDeployGroup' row. At the bottom right of the 'User groups' section is a 'Next' button, also highlighted with a red arrow.

## Step 3 - Review and Create

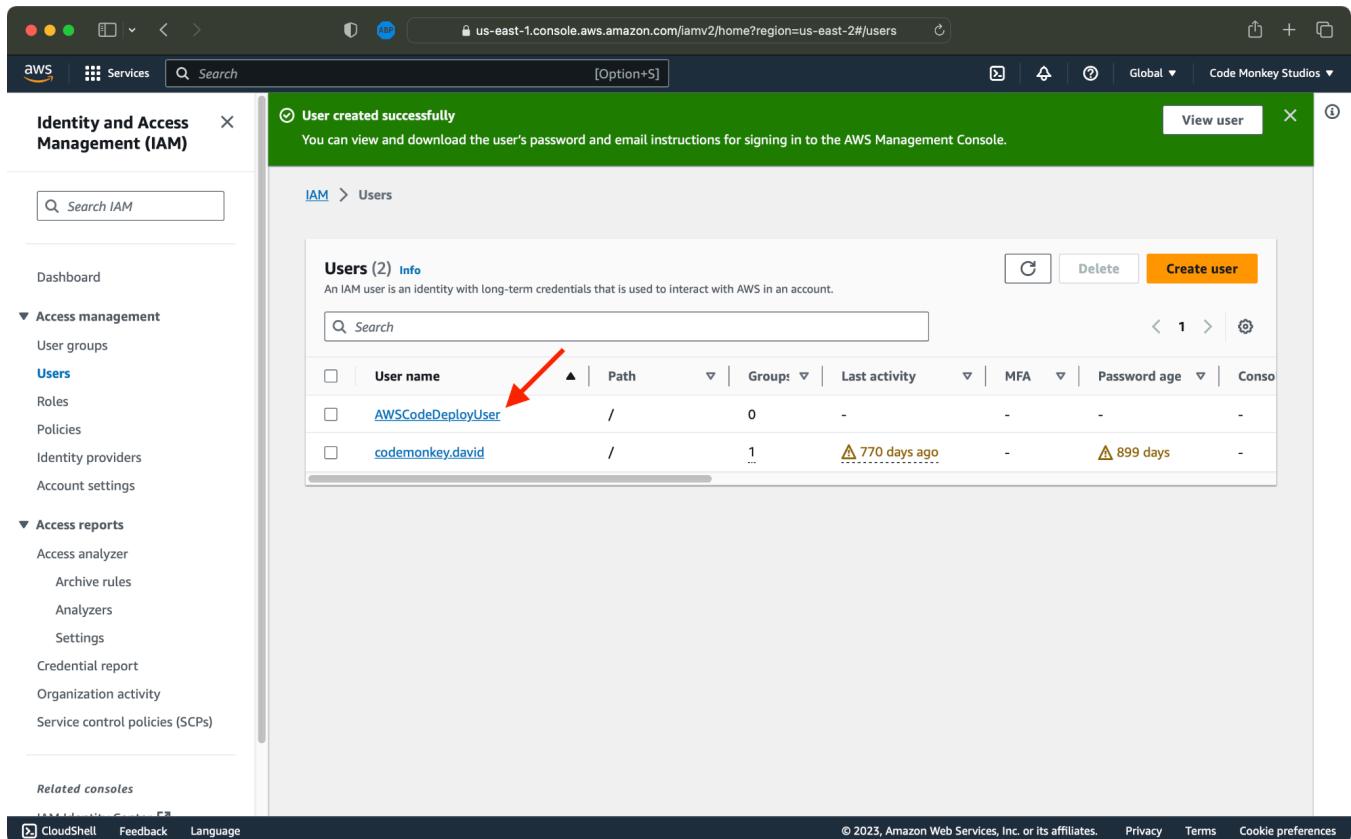
- On the Review and Create page, click the Create User button at the bottom of the page.

The screenshot shows the 'Review and create' step of the 'Create user' wizard. It displays 'User details' (User name: AWSCodeDeployUser, Console password type: None, Require password reset: No), a 'Permissions summary' table (Name: AWSCodeDeployGroup, Type: Group, Used as: Permissions group), and a 'Tags' section (Tags: None). At the bottom right is a 'Create user' button, highlighted with a red arrow.

## Step 4 - Set Programmatic Access for the User

AWS takes security seriously, and they really don't want you creating long-lived access keys that live in perpetuity, so they're going to try and discourage you from doing what I'm suggesting. Like I stated at the top of my presentation, we are throwing security out the window in order to complete this task in under an hour. Use common sense and your own security best practices when setting up a system like this on your own.

- Once the user has been created, select the AWSCodeDeployUser to edit their details.



The screenshot shows the AWS IAM service in the AWS Management Console. The left sidebar shows navigation options like Dashboard, Access management, and Access reports. The main content area is titled 'Users (2) Info' and displays a table of users. The table columns include User name, Path, Group, Last activity, MFA, and Password age. Two users are listed: 'AWSCodeDeployUser' and 'codemonkey.david'. A red arrow points to the 'User name' column for 'AWSCodeDeployUser'. The 'AWSCodeDeployUser' row is highlighted in blue.

User name	Path	Group	Last activity	MFA	Password age
AWSCodeDeployUser	/	0	-	-	-
codemonkey.david	/	1	770 days ago	-	899 days

- Click on the Security Tab in the user details page.

Screenshot of the AWS IAM User Details page for 'AWSCodeDeployUser'. The 'Security credentials' tab is selected. A red arrow points to the 'Create access key' button.

**AWSCodeDeployUser**

**Summary**

ARN arn:aws:iam::894991230159:user/AWSCodeDeployUser	Console access Disabled	Access key 1 <a href="#">Create access key</a>
Created September 19, 2023, 19:13 (UTC-07:00)	Last console sign-in -	

**Permissions** | Groups (1) | Tags | **Security credentials** → | Access Advisor

**Permissions policies (3)**

Permissions are defined by policies attached to the user directly or through groups.

Policy name	Type	Attached via
<a href="#">AmazonEC2FullAccess</a>	AWS managed	Group <a href="#">AWSCodeDeployGroup</a>
<a href="#">AmazonS3FullAccess</a>	AWS managed	Group <a href="#">AWSCodeDeployGroup</a>
<a href="#">AWSCodeDeployFullAccess</a>	AWS managed	Group <a href="#">AWSCodeDeployGroup</a>

About half way down the security tab, click the Create Access Key button.

Screenshot of the AWS IAM User Details page for 'AWSCodeDeployUser'. The 'Access keys' section is shown, and a red arrow points to the 'Create access key' button.

**Multi-factor authentication (MFA) (0)**

Use MFA to increase the security of your AWS environment. Signing in with MFA requires an authentication code from an MFA device. Each user can have a maximum of 8 MFA devices assigned. [Learn more](#)

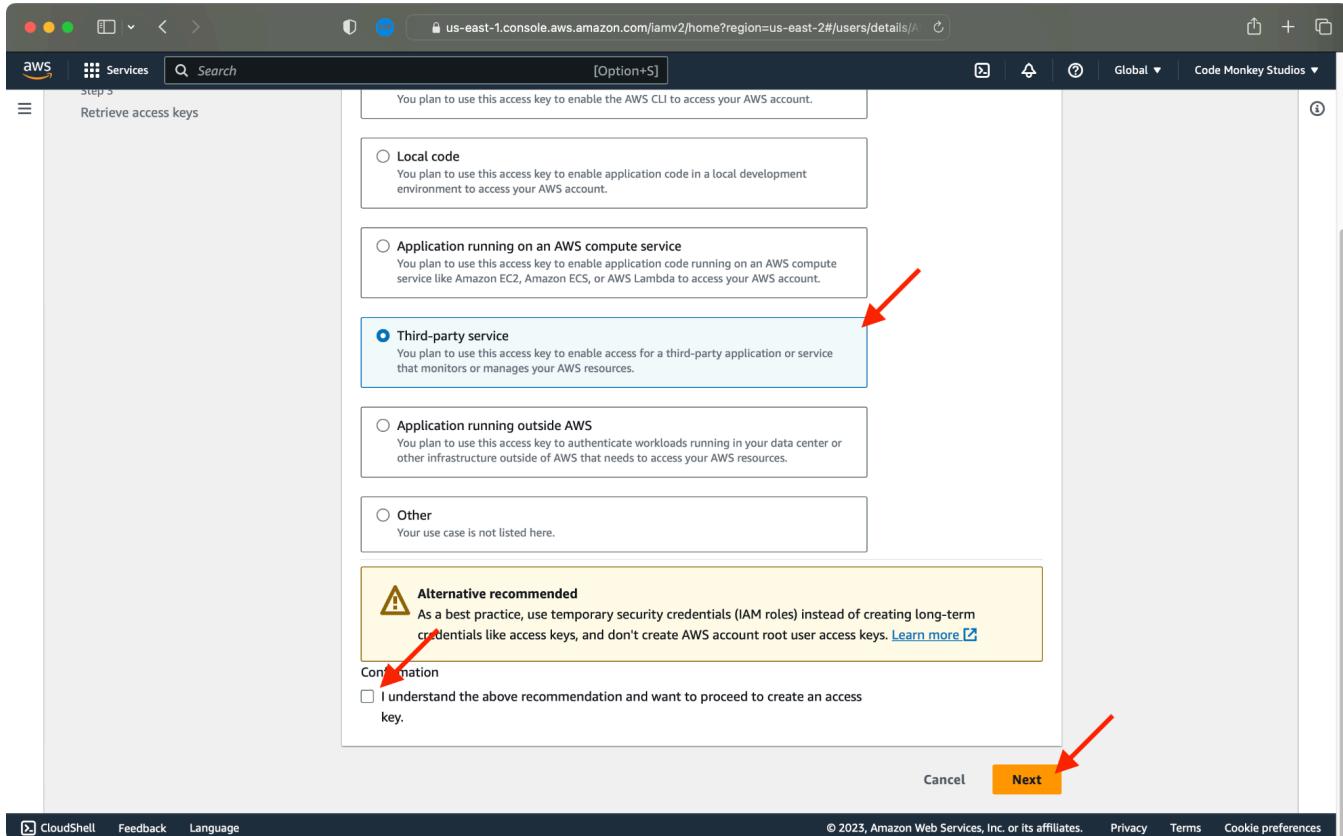
**Access keys (0)**

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

**SSH public keys for AWS CodeCommit (0)**

User SSH public keys to authenticate access to AWS CodeCommit repositories. You can have a maximum of five SSH public keys (active or inactive) at a time. [Learn more](#)

- On the Access Key Best Practices & Alternatives page, select the Third-Party Service radio button.
- Check the Confirmation checkbox.
- Click the Next button at the bottom of the page.



- On the next screen, set a description tag if you'd like, then click the Create Access Key button at the bottom of the page.

The screenshot shows the AWS IAM 'Create access key' process. The user is on Step 2 - optional, specifically setting a description tag. The 'Description tag value' field is empty. Below it, a note states: 'Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: \_ . : / = + - @'. At the bottom right of the form are 'Cancel', 'Previous', and 'Create access key' buttons. The 'Create access key' button is highlighted with a red arrow.

- From the Retrieve Access Keys page, make sure you have a copy of the Access Key and the Secret Access Key. Click on the Download .csv button to download a file containing your access keys. Do not lose this. If you lose your secret access key, your only option is to revoke the existing key and generate a new one.
- Once you have a copy of your access keys, click the Done button.

The screenshot shows the AWS IAM 'Create access key' page. At the top, a green banner says 'Access key created' with the note: 'This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.' Below this, the navigation path is IAM > Users > AWSCodeDeployUser > Create access key. On the left, there are three steps: Step 1 (Access key best practices & alternatives), Step 2 - optional (Set description tag), and Step 3 (Retrieve access keys). The main content area is titled 'Retrieve access keys' and contains an 'Access key' section. It shows two keys: 'Access key' (AKIA5AYNCMTHZG23JMAR) and 'Secret access key' (\*\*\*\*\*). A 'Show' link is next to the secret key. Below this is an 'Access key best practices' section with a bulleted list: 'Never store your access key in plain text, in a code repository, or in code.', 'Disable or delete access key when no longer needed.', 'Enable least-privilege permissions.', and 'Rotate access keys regularly.' A link to 'best practices for managing AWS access keys' is provided. At the bottom right of the main content area, there are two buttons: 'Download .csv file' and a large orange 'Done' button. Red arrows point from the text to both the 'Download .csv file' button and the 'Done' button.

We now have an AWS IAM User, Group, and Role that will be used in BitBucket to perform our Code Deployments.

## Part 3 - AWS Simple Storage Service (S3) Setup

The naming of the bucket here is critical. By default, BitBucket uploads your code to an AWS S3 bucket named {your AWS application name}-codedeploy-deployment. We will create the application later, but for now, I know that the application I am going to create will be called cfsummit-vegas-production.

- Go to the AWS S3 service.
- Click the Create Bucket button.
- On the Create Bucket screen, name the bucket **cfsummit-vegas-production-codedeploy-deployment**.
- Choose an AWS Region to create the bucket. This should be the same region you intend to create your AWS EC2 instances in. For this demonstration, I'm going to choose us-east-2 (Ohio).

The screenshot shows the AWS S3 'Create bucket' interface. In the 'General configuration' section, there are two fields highlighted with red arrows: 'Bucket name' containing 'deployments.cfsummit.vegas' and 'AWS Region' set to 'US East (Ohio) us-east-2'. The 'Object Ownership' section below shows 'ACLs disabled (recommended)' selected.

**General configuration**

Bucket name: deployments.cfsummit.vegas

AWS Region: US East (Ohio) us-east-2

**Object Ownership**

ACLs disabled (recommended)  
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

ACLs enabled  
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

- On the same page, make sure ACL's Disabled is selected.
- Make sure Block All Public Access is checked.
- Click the Create Bucket button at the bottom of the page.

The screenshot shows the AWS S3 Bucket Creation wizard. In the 'Object Ownership' section, the 'ACLs disabled (recommended)' option is selected, indicated by a red arrow. In the 'Block Public Access settings for this bucket' section, the 'Block all public access' checkbox is checked, also indicated by a red arrow.

**Object Ownership**

**ACLs disabled (recommended)**  
All objects in this bucket are owned by this account.  
Access to this bucket and its objects is specified using only policies.

**ACLs enabled**  
Objects in this bucket can be owned by other AWS accounts.  
Access to this bucket and its objects can be specified using ACLs.

**Block Public Access settings for this bucket**

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

**Block all public access**  
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- Block public access to buckets and objects granted through new access control lists (ACLs)**  
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- Block public access to buckets and objects granted through any access control lists (ACLs)**  
S3 will ignore all ACLs that grant public access to buckets and objects.
- Block public access to buckets and objects granted through new public bucket or access point policies**  
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- Block public and cross-account access to buckets and objects through any public bucket or access point policies**

CloudShell   Feedback   © 2023, Amazon Web Services, Inc. or its affiliates.   Privacy   Terms   Cookie preferences

## Part 4 - Create the AWS EC2 Instance

### Step 1 - Create and Configure a Security Group

- Go to the AWS EC2 Service.
- From the left column, click the link for Security Groups

The screenshot shows the AWS EC2 Home page. On the left, there's a navigation sidebar with links like EC2 Dashboard, EC2 Global View, Events, Instances, Images, Elastic Block Store, Network & Security (with a red arrow pointing to the 'Security Groups' link), and others. The main content area has sections for Resources (listing running instances, Auto Scaling Groups, Dedicated Hosts, etc.), Launch instance (with a prominent 'Launch instance' button), Service health (showing AWS Health Dashboard and region information), and Explore AWS (with various promotional cards). The top right shows account attributes like Default VPC and Settings.

- On the next page, click the Create Security Group button.
- In the Basic Details section:
  - Name the Security Group **cfsummit.vegas**.
  - Set the Description as **SSH, HTTP/S, and ColdFusion Administrator**.

This screenshot shows the 'Basic details' step of a security group creation wizard. It includes fields for the security group name (set to 'cfsummit.vegas'), a description ('SSH, HTTP/S, and ColdFusion Administrator'), and a VPC selection dropdown containing 'vpc-f8d7ea90'.

- In the Inbound Rules section, we are going to create four separate inbound traffic rules:
  - Type: SSH, Source: My IP
  - Type: HTTP, Source: Anywhere-IPv4
  - Type: HTTPS, Source: Anywhere-IPv4
  - Type: Custom TCP, Port Range: 8500, Source: My IP, Description: ColdFusion Administrator

Inbound rules [Info](#)

Type <a href="#">Info</a>	Protocol <a href="#">Info</a>	Port range <a href="#">Info</a>	Source <a href="#">Info</a>	Description - optional <a href="#">Info</a>
SSH	TCP	22	Any... ▾	<input type="text" value="0.0.0.0/0"/> X <a href="#">Delete</a>
HTTP	TCP	80	Any... ▾	<input type="text" value="0.0.0.0/0"/> X <a href="#">Delete</a>
HTTPS	TCP	443	Any... ▾	<input type="text" value="0.0.0.0/0"/> X <a href="#">Delete</a>
Custom TCP	TCP	8500	Any... ▾	<input type="text" value="ColdFusion Administrator"/> <a href="#">Delete</a>

[Add rule](#)

Remember... security? Out the window. You would never want to leave these security rules like this.

- Click the Create Security Group button at the bottom of the window.

## Step 2 - Configure and Launch the EC2 Instance

- Go to the AWS EC2 Service.
- Click the Launch Instance button.

The screenshot shows the AWS EC2 Home page. On the left, there's a sidebar with navigation links like EC2 Dashboard, EC2 Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and CloudShell. The main content area has sections for Resources (listing 1 Instance running, 0 Auto Scaling Groups, 0 Dedicated Hosts, 1 Elastic IP, 1 Instance, 2 Key pairs, 0 Load balancers, 0 Placement groups, 3 Security groups, 0 Snapshots, and 1 Volumes), Launch instance (with a red arrow pointing to the 'Launch instance' button), Service health (AWS Health Dashboard, Region US East (Ohio)), and Explore AWS (AWS Compute Cost Optimization Training, Up to 40% better performance; 20% lower cost, Amazon GuardDuty Malware Protection). The bottom right corner includes copyright information and links for Privacy, Terms, and Cookie preferences.

- On the Launch an Instance page, name the instance `cfsummit.vegas`.

The screenshot shows the 'Launch an instance' page. It has a header 'Launch an instance' with an 'Info' link. Below it, a sub-header says 'Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.' The main form is titled 'Name and tags' with an 'Info' link. It has a 'Name' field containing 'cfsummit.vegas' (with a red arrow pointing to it) and a 'Add additional tags' link. The background shows other sections of the page like 'Configure instance type and storage' and 'Review and launch'.

- In the Application and OS Images section:

- Select Ubuntu
- Select Ubuntu 22.04

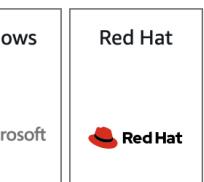
## ▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

Recents

Quick Start



Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type

ami-024e6efaf93d85776 (64-bit (x86)) / ami-08fdd91d87f63bb09 (64-bit (Arm))  
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

Description

Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2023-05-16

Architecture

64-bit (x86)

AMI ID

ami-024e6efaf93d85776

Verified provider

- In the Instance Type section, select m5a.large

## ▼ Instance type [Info](#)

Instance type

m5a.large

Family: m5a 2 vCPU 8 GiB Memory Current generation: true  
On-Demand SUSE base pricing: 0.142 USD per Hour  
On-Demand Linux base pricing: 0.086 USD per Hour  
On-Demand Windows base pricing: 0.178 USD per Hour  
On-Demand RHEL base pricing: 0.146 USD per Hour

All generations

[Compare instance types](#)

[Additional costs apply for AMIs with pre-installed software](#)

- In the Key Pair section, click the “Create a New Key Pair” link.

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

Select ▾

[Create new key pair](#)

- In the Create Key Pair modal:
  - Name the key pair cfsummit.us-east-2
  - Leave the key pair type as RSA and the private key file format as .pem.
  - Click the Create Key Pair button at the bottom of the modal window. This will download the .pem file to your local computer.

### Create key pair

Key pair name  
Key pairs allow you to connect to your instance securely.

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

RSA  
RSA encrypted private and public key pair

ED25519  
ED25519 encrypted private and public key pair

Private key file format

.pem  
For use with OpenSSH

.ppk  
For use with PuTTY

**⚠️** When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more ↗](#)

Cancel **Create key pair**

- In the Network Settings section:
  - Select the Select Existing Security Group radio button.
  - Click the dropdown to select the cfsummit.vegas security group we created in Step 1.

▼ Network settings [Info](#)

**Network** [Info](#)  
vpc-f8d7ea90

**Subnet** [Info](#)  
No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)  
Enable

**Firewall (security groups)** [Info](#)  
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group  Select existing security group

**Common security groups** [Info](#)  
Select security groups ▾

cfsummit.vegas sg-03aecf907e620632d X  
VPC: vpc-f8d7ea90

Compare security group rules

Security groups that you add or remove here will be added to or removed from all your network interfaces.

- In the Storage (Volumes) section:
  - Increase the storage size to 20 GB.

▼ Storage (volumes) [Info](#)

Simple

---

EBS Volumes [Hide details](#)

---

▼ Volume 1 (AMI Root) (Custom)

Storage type <a href="#">Info</a> EBS	Device name - required <a href="#">Info</a> <code>/dev/sda1</code>	Snapshot <a href="#">Info</a> <code>snap-07592d65a6f259c5b</code>
Size (GiB) <a href="#">Info</a> <input type="text" value="20"/> <span style="border: 1px solid #ccc; padding: 2px;">▲ ▼</span>	Volume type <a href="#">Info</a> <input type="text" value="gp2"/> <span style="border: 1px solid #ccc; padding: 2px;">▼</span>	IOPS <a href="#">Info</a> <code>100 / 3000</code>
Delete on termination <a href="#">Info</a> <input type="text" value="Yes"/> <span style="border: 1px solid #ccc; padding: 2px;">▼</span>	Encrypted <a href="#">Info</a> <input type="text" value="Not encrypted"/> <span style="border: 1px solid #ccc; padding: 2px;">▼</span>	KMS key <a href="#">Info</a> <input type="text" value="Select"/> <span style="border: 1px solid #ccc; padding: 2px;">▼</span> <small>KMS keys are only applicable when encryption is set on this volume.</small>

[Add new volume](#)

---

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance

---

**File systems** [Show details](#)

- Expand the Advanced Details section.
- From the IAM Instance Profile dropdown, select the AwSCodeDeployRole role we created earlier.

**Advanced details** [Info](#)

Purchasing option [Info](#)  
 Request Spot Instances

Domain join directory [Info](#)  
 Select [Create new directory](#)

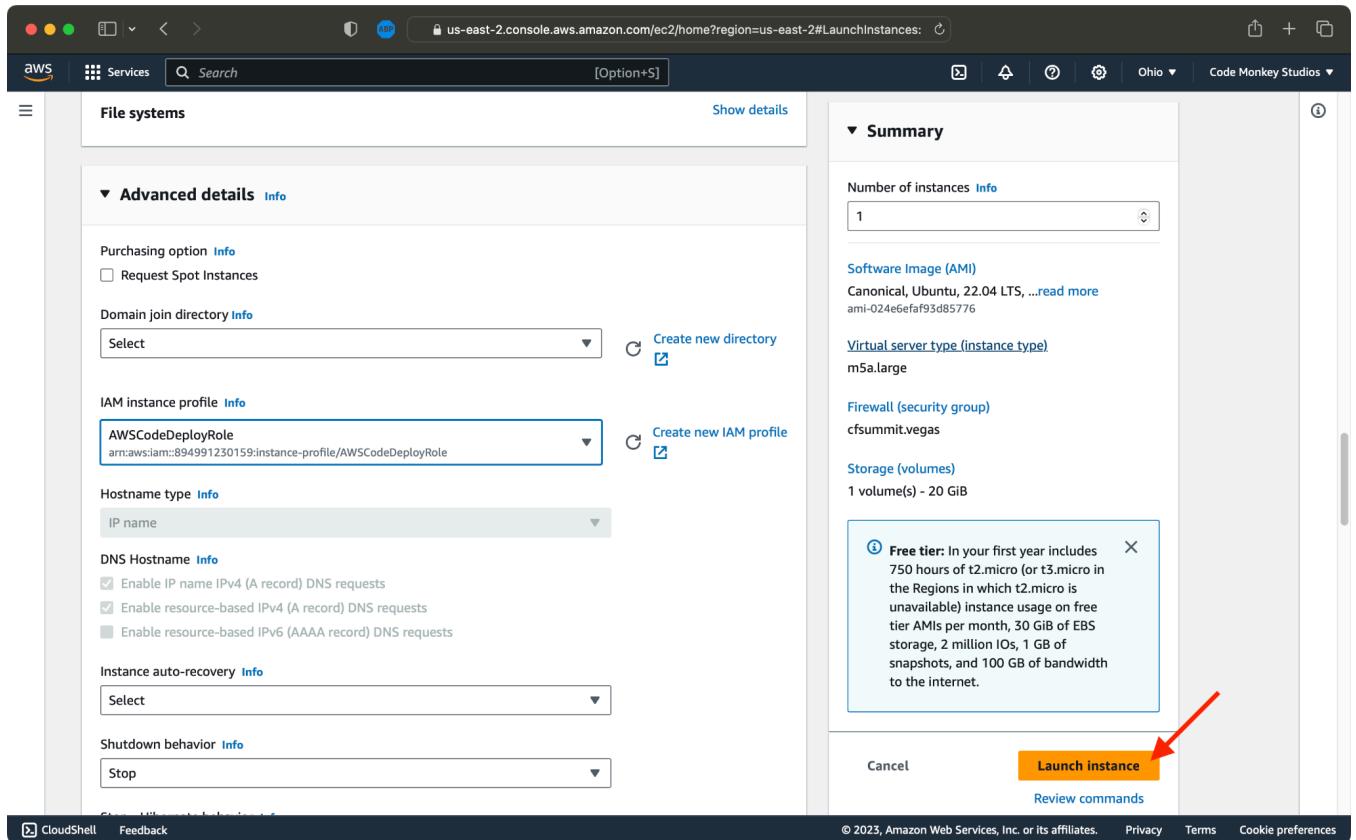
IAM instance profile [Info](#)  
 AWSCodeDeployRole [arn:aws:iam::894991230159:instance-profile/AWSCodeDeployRole](#) [Create new IAM profile](#)

Hostname type [Info](#)  
 IP name

DNS Hostname [Info](#)  
 Enable IP name IPv4 (A record) DNS requests  
 Enable resource-based IPv4 (A record) DNS requests  
 Enable resource-based IPv6 (AAAA record) DNS requests

Instance auto-recovery [Info](#)  
 Select

- Finally, click the Launch Instance button in the right column, near the bottom of the page.



The screenshot shows the AWS EC2 console interface for launching a new instance. The left side contains the configuration form from the previous image, including fields for Purchasing option, Domain join directory, IAM instance profile (set to AWSCodeDeployRole), Hostname type (IP name), and DNS Hostname settings. The right side displays the 'Summary' section with the following details:

- Number of instances: 1
- Software Image (AMI): Canonical, Ubuntu, 22.04 LTS, ...read more (ami-024e6efaf93d85776)
- Virtual server type (instance type): m5a.large
- Firewall (security group): cfsummit.vegas
- Storage (volumes): 1 volume(s) - 20 GiB

A callout box highlights a note about the Free tier, stating: "Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet."

At the bottom right, there are three buttons: "Cancel", "Launch instance" (which is highlighted with a red arrow), and "Review commands".

## Step 3 - Assign an Elastic IP Address to the EC2 Instance

- Go to the AWS EC2 Service.
- In the left column, select the link labeled Elastic IPs.

The screenshot shows the AWS EC2 Home page. On the left sidebar, under the 'Network & Security' section, the 'Elastic IPs' link is highlighted with a red arrow. The main content area displays various EC2 resources: Instances (running) 1, Auto Scaling Groups 0, Dedicated Hosts 0, Elastic IPs 1, Instances 1, Key pairs 3, Load balancers 0, Placement groups 0, Security groups 3, Snapshots 0, and Volumes 1. To the right, there are sections for Account attributes (Default VPC vpc-f8d7ea90), Service health (AWS Health Dashboard), and Explore AWS (AWS Compute Cost Optimization Training, Up to 40% better performance; 20% lower cost, Amazon GuardDuty Malware Protection). The URL in the browser is us-east-2.console.aws.amazon.com/ec2/home?region=us-east-2#Home:.

- Click the Allocate Elastic IP Address button.
- On the next page, leave everything default, and click the Allocate button at the bottom of the page.
- Copy the newly created IP Address.
- Return to the Elastic IP Addresses page.
- Check the checkbox next to the newly created IP Address.
- From the Actions dropdown menu at the top of the page, click the Associate Elastic IP Address button.

The screenshot shows the AWS EC2 console under the 'Elastic IP addresses' section. There are two entries in the table:

Name	Allocated IPv4 address	Type	Allocation ID	Reverse DNS record
-	18.220.142.180	Public IP	eipalloc-09314ce2df8b25a7a	-
<input checked="" type="checkbox"/>	3.136.240.153	Public IP	eipalloc-09314ce2df8b25a7a	-

In the Actions menu for the selected IP, the 'Associate Elastic IP address' button is highlighted with a red arrow. In the summary card for the selected IP, the 'Associate' link is also highlighted with a red arrow.

- In the Associate Elastic IP address page:
  - Expand the Instance dropdown.
  - Select the cfsummit.vegas instance.
  - Click the Associate button at the bottom of the page.

The screenshot shows the 'Associate Elastic IP address' step in the AWS EC2 console. The top navigation bar includes 'Services', 'Search', and 'Code Monkey Studios'. The main content area is titled 'Associate Elastic IP address' with an 'Info' link. It says 'Choose the instance or network interface to associate to this Elastic IP address (3.136.240.153)'. A section titled 'Elastic IP address: 3.136.240.153' is shown. Under 'Resource type', 'Instance' is selected. A note states: 'If you associate an Elastic IP address with an instance that already has an Elastic IP address associated, the previously associated Elastic IP address will be disassociated, but the address will still be allocated to your account.' Below this, it says 'If no private IP address is specified, the Elastic IP address will be associated with the primary private IP address.' The 'Instance' section contains a search bar with 'choose an instance' placeholder, a dropdown menu with two items: 'i-063048342f57d7b08 (codemonkey.studio) - running' and 'i-00ab2443525dc2129 (cfsummit.vegas) - running', and a 'Choose a private IP address' dropdown. The 'Reassociation' section has a checkbox 'Allow this Elastic IP address to be reassociated'. At the bottom are 'Cancel' and 'Associate' buttons, with 'Associate' highlighted by a red arrow. The footer includes links for CloudShell, Feedback, © 2023, Amazon Web Services, Inc. or its affiliates., Privacy, Terms, and Cookie preferences.

The Elastic IP Address is now associated with the EC2 instance and traffic to that IP will be directed at that server.

## Step 4 - Update the DNS

- Go to the AWS Route 53 Service.
- Select the link labeled Hosted Zones

The screenshot shows the AWS Route 53 Dashboard. On the left, a sidebar menu includes options like Hosted zones, Health checks, IP-based routing, Traffic flow, Domains, Resolver, DNS Firewall, and Application Recovery Controller. The main content area displays the 'Route 53 Dashboard' with several sections: 'DNS management' (3 Hosted zones), 'Traffic management' (2 Domains), 'Availability monitoring' (Create health check), and 'Domain registration' (2 Domains). A red arrow points to the 'Hosted zones' link under the DNS management section.

- Select the cfsummit.vegas hosted zone.

The screenshot shows the 'Hosted zones' page within the AWS Route 53 service. The sidebar menu is identical to the dashboard. The main content area lists 'Hosted zones (3)'. A red arrow points to the 'cfsummit.vegas' entry in the table, which is highlighted in blue. The table columns include Hosted zone name, Type, Created by, Record count, Description, and Hosted... (with ellipsis).

Hosted zone name	Type	Created by	Record count	Description	Hosted...
cf.expert	Public	Route 53	8	-	Z0726647...
clubsite.pro	Public	Route 53	4	HostedZone c...	Z0522956...
cfsummit.vegas	Public	Route 53	2	HostedZone c...	Z0164139...

- Select the Create Record button.

The screenshot shows the AWS Route 53 Hosted Zone Details page for the domain `cfsummit.vegas`. The left sidebar shows the navigation path: Route 53 > Hosted zones > `cfsummit.vegas`. The main content area displays the `Hosted zone details` for `cfsummit.vegas`, with tabs for `Records (2)`, `DNSSEC signing`, and `Hosted zone tags (0)`. The `Records (2)` tab is selected. Below it, there's a table showing two existing records: one NS record and one SOA record. At the top of the records table, there are buttons for `Delete record`, `Import zone file`, and `Create record`. A red arrow points to the `Create record` button. The right side of the screen shows a sidebar with the message `0 records selected` and `Select a record to see its details`.

- From the Create Record page:
  - Paste the IP Address we created in the previous step into the Value field.
  - Set the TTL to one minute.
  - Click the Create Records button at the bottom of the page.

The screenshot shows the 'Create record' page in the AWS Route 53 console. A red arrow points to the 'Value' field, which contains the IP address '3.136.240.153'. Another red arrow points to the 'TTL (seconds)' field, which is set to '60'. A third red arrow points to the 'Create records' button at the bottom right.

Route 53 > Hosted zones > cfsummit.vegas > Create record

Create record [Info](#)

Quick create record [Switch to wizard](#)

Record 1

Record name [Info](#) cfsummit.vegas

Record type [Info](#) A – Routes traffic to an IPv4 address and some AWS resources

Value [Info](#) 3.136.240.153

Alias

TTL (seconds) [Info](#) 60

+1m 1h 1d

Routing policy [Info](#) Simple routing

Add another record

Create records

Traffic addressed to cfsummit.vegas will now be routed to the Elastic IP Address.

## Step 5 - Connect to the EC2 Instance

- From the local computer, open the Terminal application.
- First we will change the permissions on the downloaded cfsummit.vegas.us-east-2.pem file by running the following command:

```
chmod 400 ~/Downloads/cfsummit.us-east-2.pem
```

- Copy the file to the local computer's SSH folder, which is `~/.ssh` on a Mac, by running the following command.

```
cp ~/Downloads/cfsummit.us-east-2.pem ~/.ssh
```

- Finally, connect to the EC2 Instance by running the following command:

```
ssh -i ~/.ssh/cfsummit.us-east-2.pem ubuntu@cfsummit.vegas
```

- Once connected, it's helpful to log in as a root account so we do not have to continuously run `sudo` at the beginning of every command.

```
sudo -s
```

We are now logged into the AWS EC2 instance with superuser privileges.

## Step 6 - Install Server Software

### Create a Maintenance Update Script

- Run the following commands:

```
cd /home/ubuntu
touch update-server
chmod +x update-server
nano update-server
```

- Paste the following:

```
#!/bin/bash
apt update && apt upgrade -y
```

- Press Ctrl-O, then Enter to save the file.
- Press Ctrl-X to exit nano.
- Run the following commands:

```
chmod +x update-server
./update-server
reboot now
```

### Apache 2

- Run the following command:

```
apt install -y apache2
```

- Once Apache is installed, run the following commands:

```
a2dissite 000-default  
rm -rf /var/www/html  
mkdir /var/www/cfsummit.vegas  
cd /etc/apache2/sites-available  
touch cfsummit.vegas.conf  
nano cfsummit.vegas.conf
```

- Paste the following:

```
<VirtualHost *:80>  
  
    ServerName cfsummit.vegas  
    ServerAdmin david@codemonkey.studio  
    DocumentRoot /var/www/cfsummit.vegas/  
    DirectoryIndex index.cfm  
  
    <Directory /var/www/cfsummit.vegas>  
        Options Indexes FollowSymLinks  
        AllowOverride All  
        Order allow,deny  
        Allow from all  
    </Directory>  
  
    ErrorLog ${APACHE_LOG_DIR}/cfsummit.vegas.error.log  
    CustomLog ${APACHE_LOG_DIR}/cfsummit.vegas.access.log combined  
  
</VirtualHost>
```

- Press Ctrl-O, then Enter to save the file.
- Press Ctrl-X to exit nano.
- Run the following commands:

```
a2ensite cfsummit.vegas  
systemctl reload apache2  
cd /var/www/cfsummit.vegas  
touch index.cfm  
nano index.cfm
```

- Paste the same basic index HTML we created in our local development environment.

```
<h2>cfsummit.vegas</h2>
```

```
<cfoutput>
    <p>#server.coldfusion.productName# (#server.coldfusion.productLevel#)
version #server.coldfusion.productVersion#</p>
    <p>#dateFormat( now(), 'long' )#</p>
</cfoutput>
```

- Press Ctrl-O, then Enter to save the file.
- Press Ctrl-X to exit nano.

At this point, we should be able to browse to cfsummit.vegas and see some kind of result.

## CertBot for SSL Certificates

- To install CertBot, Run the following command:

```
cd /home/ubuntu
snap install core; snap refresh core; snap install --classic certbot; ln -s
/snap/bin/certbot /usr/bin/certbot
```

- To install an SSL certificate for the website, run the following command:

```
certbot --apache
```

- Once certbot loads, follow the prompts and establish an SSL cert for the site.

## AWS Code Deploy

- To install the AWS CodeDeploy Agent on the EC2 instance, run the following commands:

```
cd /home/ubuntu
snap refresh; snap install ruby --classic --channel=2.7/stable
wget https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
chmod +x ./install; ./install auto
```

Note: Change the source of the installer to match the AWS region your EC2 instance was created in. For example, if your AWS EC2 instance is in the us-west-1 AWS region, you'd download the installer from

<https://aws-codedeploy-us-west-1.s3.us-west-1.amazonaws.com/latest/install>

- Verify the installation was successful by running the following command:

```
service codedeploy-agent status
```

## Java

Note: Installing Java is optional, especially if you update the JRE when installing and configuring ColdFusion. However, if you update the ColdFusion Server using the java-jar method, this becomes mandatory.

- Run the following command:

```
apt install -y default-jre
```

- To confirm, run the following command:

```
java --version
```

## Zip

Since the ColdFusion installer is zipped, we will need unzip software on the server.

- Run the following command:

```
apt install -y zip
```

## ColdFusion 2023

- Run the following commands:

```
cd /home/ubuntu
wget https://s3.us-east-2.amazonaws.com/acf.cfsummit.vegas/CF2023_linux.zip
unzip CF2023_linux.zip
unzip ColdFusion_WWEJ_linux64.zip -d /opt/
mv /opt/ColdFusion/ /opt/coldfusion2023
cd /opt/coldfusion2023/cfusion/bin
./cfinstall.sh
```

- Follow the command prompts to configure ColdFusion Server.
  - Accept the license agreement
  - Select 30-Day Trial.

- Establish an Administrator password.
  - Establish the built in web server port number.
  - Establish an RDS password.
  - Select the default user.
  - Specify Development as the type of deployment.
  - Proceed.
- Once ColdFusion has been configured, clean up the installer files by running the following commands:

```
cd /home/ubuntu
rm -rf CF2023_linux.zip
rm -rf ColdFusion_WWEJ_linux64.zip
rm -rf META-INF
rm -rf install
rm -rf install-aws-code-deploy
```

- Finally, start the ColdFusion server by running the following command:

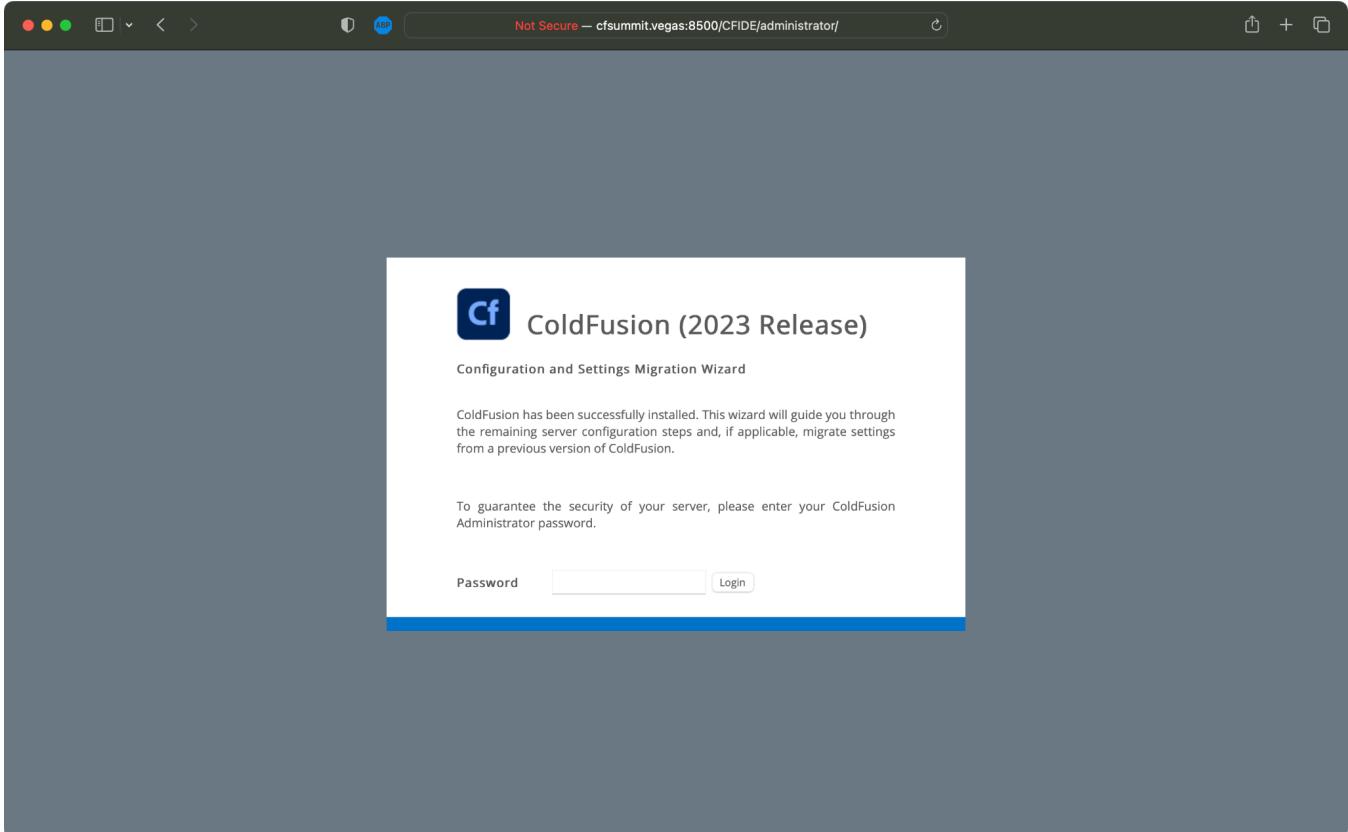
```
cd /opt/coldfusion2023/cfusion/bin
./coldfusion start
```

- Reboot the server by running the following command:

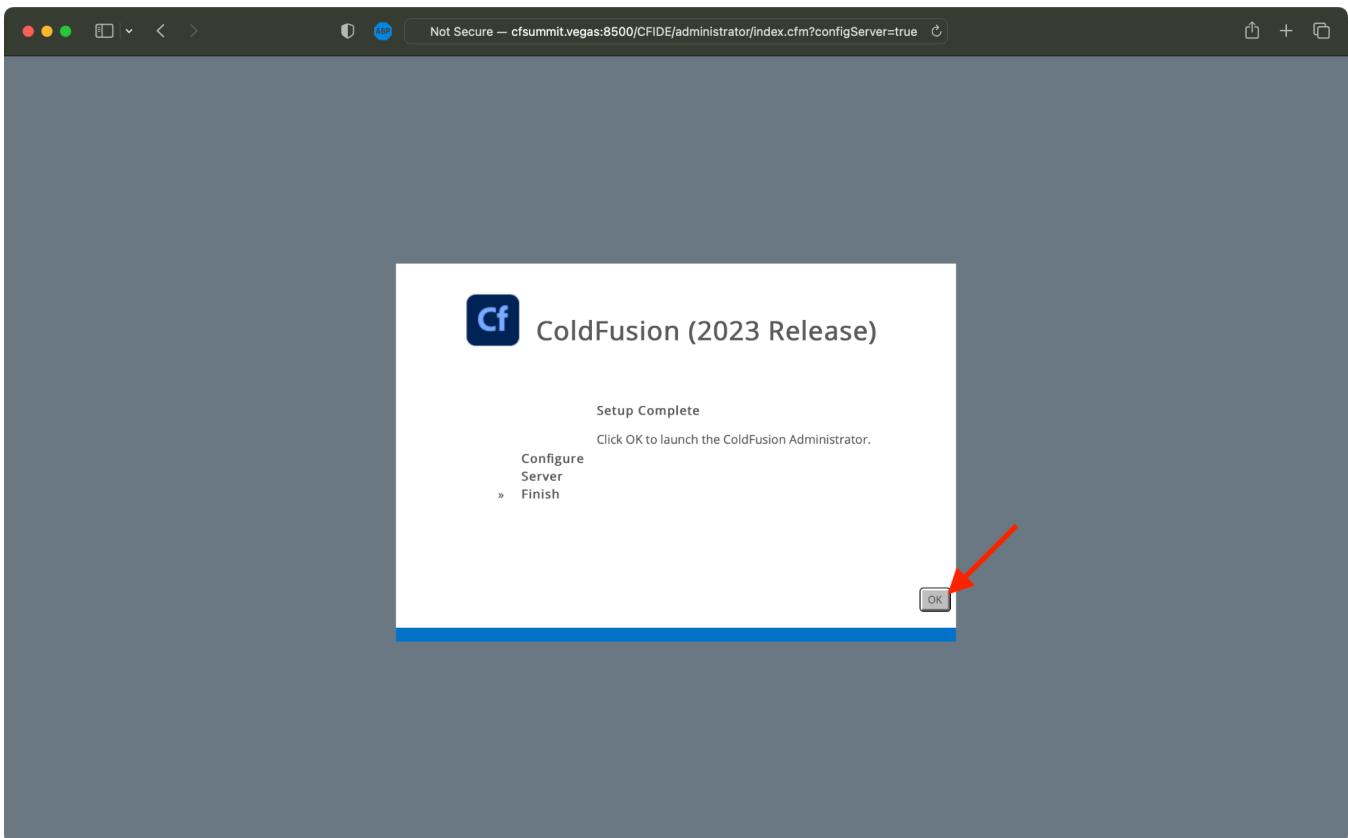
```
reboot now
```

## Log into the ColdFusion Administrator

- Browse to the ColdFusion Administrator at  
**<http://cfsummit.vegas:8500/CFIDE/administrator>**.
- The Configuration and Settings Migration Wizard should appear.



- Enter the ColdFusion Administrator Password created during setup.
- Once the server configuration setup is complete, click the OK button.



## Install ColdFusion Updates via the ColdFusion Administrator

- Once logged in, browse to the Package Manager.

The screenshot shows the ColdFusion (2023 Release) administrator interface. The top navigation bar includes links for ColdFusion Community, search, and logout. The main menu on the left has 'Server Settings' selected. The 'Settings Summary' tab is active in the top right. A sidebar on the left contains various icons, with a red arrow pointing to the 'Download and Install' icon. The main content area displays system information, package details, and JVM details. A note at the top says: "When clicked, generates a PDF with the Server Settings in a new Window." A 'Save as PDF' button is located in the top right corner of this note.

Report generated on Sep 26, 2023 02:57 AM

This report shows the status of all ColdFusion configuration settings. To display the area of the ColdFusion Administrator where you can edit the group settings, click any of the groups in the report.

**System Information**

Server Product	ColdFusion 2023
Version	2023.0.0.330468
Edition	Enterprise Trial
Operating System	UNIX
OS Version	6.2.0-1012-aws
Adobe Driver Version	5.1.4 (Build 0001)
Tomcat Version	9.0.72.0
Device ID	13c4b65a5ecc1df4d89a351077fe1b91925469e142ea53337ce21d6c8498e364

**Package Details**

Installed package(s)	adminapi[2023.0.0.330468] administrator[2023.0.0.330468]
----------------------	--

**JVM Details**

Java Version	17.0.6
--------------	--------

- Expand the Core Server section.
- Click Download and Install to install the latest ColdFusion 2023 hotfix.

The screenshot shows the ColdFusion (2023 Release) interface. In the top navigation bar, it says "Not Secure — cfsummit.vegas:8500/CFIDE/administrator/index.cfm". The main content area is titled "Package Manager" with a sub-menu "Packages". On the left, there's a vertical sidebar with various icons. The central panel displays a package named "Core Server" with the following details:

- Name: Core Server
- Update Level: 04
- Update Type: General
- Build Number: 330500

The "Update Description" section contains a note about ColdFusion (2023 release) Update 4, mentioning a serial filter for Java classes and packages. It also links to a technote for connector re-configuration.

The "Technote Link" is <https://helpx.adobe.com/coldfusion/kb/coldfusion-2023-update-4.html>.

The "Available Versions" dropdown is set to "ColdFusion 2023 Update 4". At the bottom, there are three buttons: "Download", "Download and Install" (which has a red arrow pointing to it), and "Close".

- Follow the prompts until the installation is complete.

## Alternatively, Download and Install ColdFusion 2023 Updates using Java

Note: The Oracle Java CPU update released in July 2023 introduced an additional validation of ZIP64 extra fields that is known to cause an issue while downloading and/or applying the ColdFusion updates that were released on July 19, 2023 and Aug 17, 2023. The “-Djdk.util.zip.disableZip64ExtraFieldValidation=true” directive below allows the hotfix to be applied without issue.

- Run the following commands:

```
cd /opt/coldfusion2023/cfusion/bin  
.coldfusion stop  
cd /home/ubuntu  
wget  
https://cfdownload.adobe.com/pub/adobe/coldfusion/2023/updates/hotfix-006-330617  
.jar  
java -Djdk.util.zip.disableZip64ExtraFieldValidation=true -jar  
hotfix-006-330617.jar
```

- Accept the terms of the licensing agreement.

- Once the hotfix has been installed, clean up the hotfix files by running the following command:

```
rm -rf /home/ubuntu/hotfix-004-330500.jar
```

## Configure Apache using the Web Server Connector

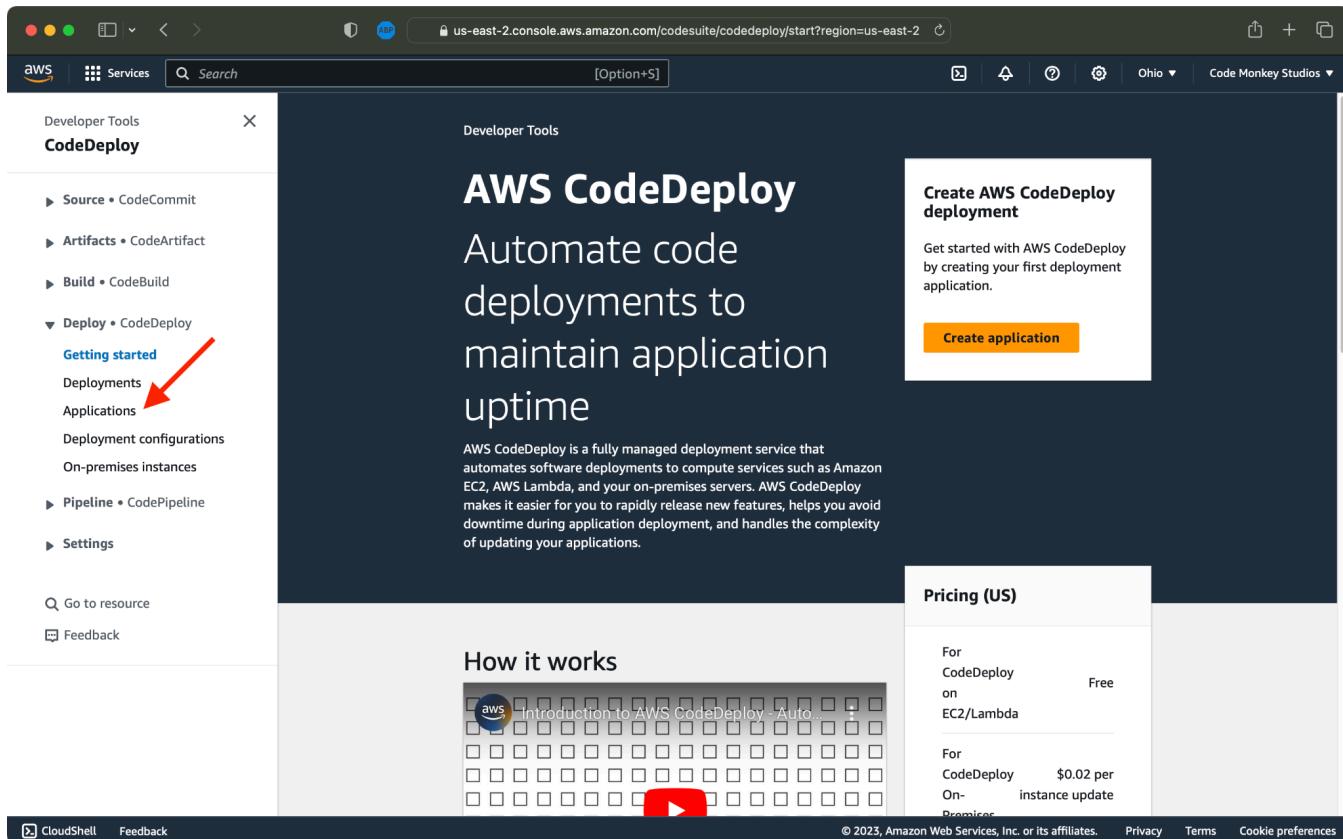
- Run the following commands:

```
cd /opt/coldfusion2023/cfusion/runtime/bin/  
.wsconfig -ws Apache -dir /etc/apache2 -v
```

# Part 5 - AWS CodeDeploy Configuration

## Step 1 - Create an Application

- Go to the AWS CodeDeploy service.
- Click on the Application link in the left column.



- Click the Create Application button.

The screenshot shows the AWS CodeDeploy Applications page. On the left, there is a navigation sidebar with the following menu items:

- Source • CodeCommit
- Artifacts • CodeArtifact
- Build • CodeBuild
- Deploy • CodeDeploy
  - Getting started
  - Deployments
  - Applications**
  - Deployment configurations
  - On-premises instances
- Pipeline • CodePipeline
- Settings

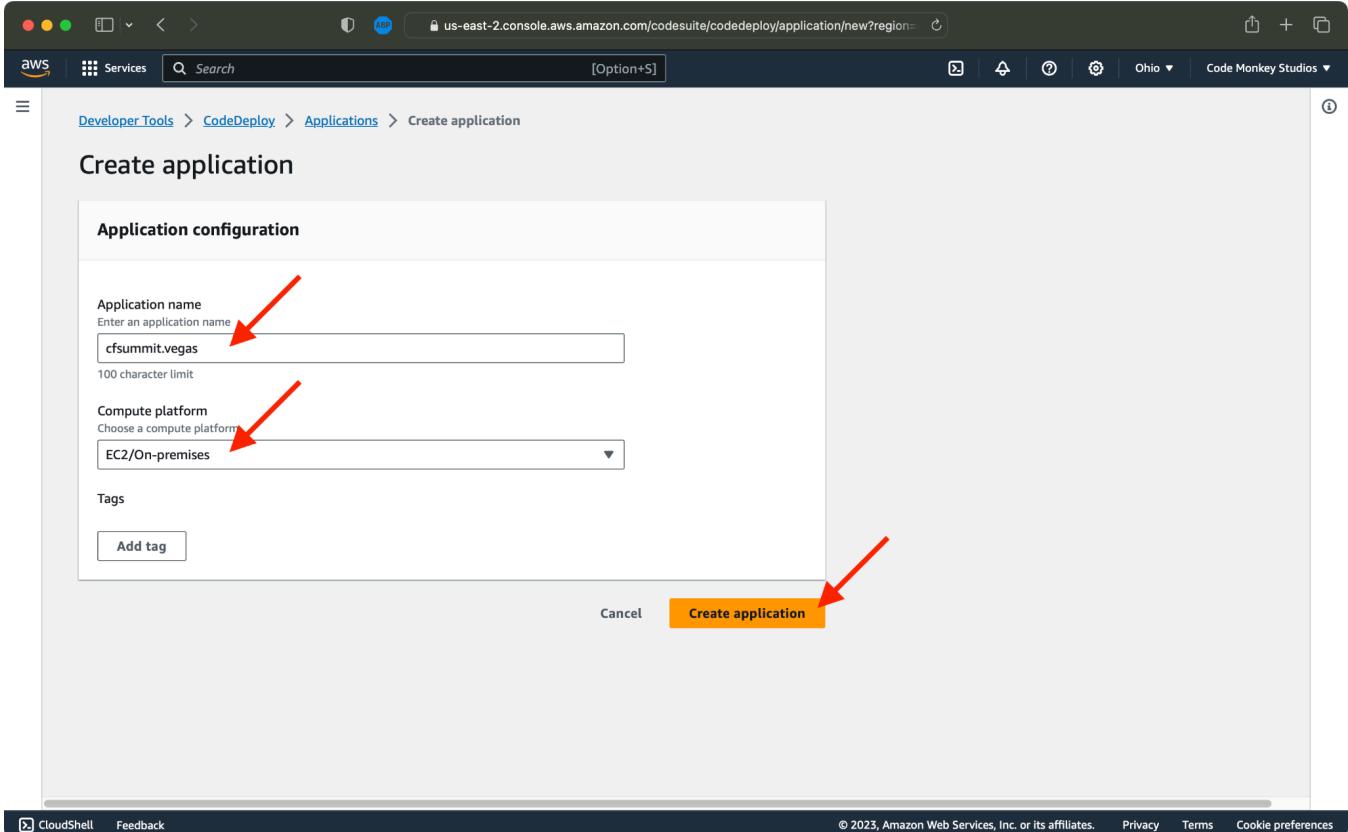
Below the sidebar are two links: "Go to resource" and "Feedback".

The main content area has a header "Applications" with a search bar, a "Notify" dropdown, a "View details" button, a "Deploy application" button (which is highlighted with a red arrow), and a "Create application" button.

The table below the header has columns for "Application name", "Compute platform", and "Created". A message "No results" and "There are no results to display." is shown.

At the bottom of the page, there are links for CloudShell, Feedback, Privacy, Terms, and Cookie preferences.

- Name the application **cfsu**mmi-vegas-production.
- Select EC2/On Premises from the Compute Platform dropdown menu.
- Click the Create Application button.



## Step 2 - Create a Deployment Group

- Click on the Create Deployment Group button.

The screenshot shows the AWS CodeDeploy application details page for 'cfsummit.vegas'. The left sidebar shows navigation options like Source, Artifacts, Build, Deploy, Application, Settings, Deployment configurations, On-premises instances, Pipeline, and Settings. The main area displays application details: Name (cfsummit.vegas) and Compute platform (EC2/On-premises). Below this, the 'Deployment groups' tab is selected, showing a table with columns: Name, Status, Last attempted depl..., Last successful depl..., and Trigger count. A search bar and 'View details' and 'Edit' buttons are at the top of the table. A large red arrow points to the 'Create deployment group' button at the bottom right of the table area.

- In the section titled Deployment Group Name, name the deployment group **cfsummit-vegas-production-deployment-group**.

This screenshot shows the 'Deployment group name' configuration screen. It has a title 'Deployment group name' and a sub-instruction 'Enter a deployment group name'. Below this is a text input field containing 'cfsummit.vegas\_deployment\_group'. A note below the input field states '100 character limit'.

- In the section titled Service Role, select the AWSCodeDeployRole service role from the Service Role dropdown.

This screenshot shows the 'Service role' configuration screen. It has a title 'Service role' and a sub-instruction 'Enter a service role'. Below this is a text input field containing 'arn:aws:iam::894991230159:role/AWSCodeDeployRole'. A note below the input field states 'Enter a service role with CodeDeploy permissions that grants AWS CodeDeploy access to your target instances.'

- In the section titled Deployment Type, select the In-Place radio button.

## Deployment type

Choose how to deploy your application

In-place

Updates the instances in the deployment group with the latest application revisions. During a deployment, each instance will be briefly taken offline for its update

Blue/green

Replaces the instances in the deployment group with new instances and deploys the latest application revision to them. After instances in the replacement environment are registered with a load balancer, instances from the original environment are deregistered and can be terminated.

- In the section titled Environment Configuration:
  - Select the Amazon EC2 checkbox.
  - Select the tag Name.
  - Select the value cfsummit.vegas.

## Environment configuration

Select any combination of Amazon EC2 Auto Scaling groups, Amazon EC2 instances, and on-premises instances to add to this deployment

Amazon EC2 Auto Scaling groups

Amazon EC2 instances

2 unique matched instances. [Click here for details](#)

You can add up to three groups of tags for EC2 instances to this deployment group.

**One tag group:** Any instance identified by the tag group will be deployed to.

**Multiple tag groups:** Only instances identified by all the tag groups will be deployed to.

Tag group 1

Key

Name X

Value - optional

cfsummit.vegas X

Remove tag

Add tag

+ Add tag group

On-premises instances

### Matching instances

2 unique matched instances. [Click here for details](#)

- In the section titled Agent Configuration with AWS Systems Manager, leave the default values.

## Agent configuration with AWS Systems Manager Info



**Complete the required prerequisites before AWS Systems Manager can install the CodeDeploy Agent.**

Make sure the AWS Systems Manager Agent is installed on all instances and attach the required IAM policies to them. [Learn more](#)

### Install AWS CodeDeploy Agent

- Never
- Only once
- Now and schedule updates

**Basic scheduler**

**Cron expression**

14



Days



- In the section titled Deployment Settings, leave the default value.  
(CodeDeployDefault.AllAtOnce)

## Deployment settings

### Deployment configuration

Choose from a list of default and custom deployment configurations. A deployment configuration is a set of rules that determines how fast an application is deployed and the success or failure conditions for a deployment.

CodeDeployDefault.AllAtOnce



or

**Create deployment configuration**

- In the section labeled Load Balancer, uncheck the Enable Load Balancing checkbox.

## Load balancer

Select a load balancer to manage incoming traffic during the deployment process. The load balancer blocks traffic from each instance while it's being deployed to and allows traffic to it again after the deployment succeeds.

- Enable load balancing

- Click the Create Deployment Group button at the bottom of the page.

# Part 6 - BitBucket Pipeline Configuration

## Step 1 - Create the `appspec.yml` and `bitbucket-pipelines.yml` files

- In the root directory of the code repository, create an `appspec.yml` file with the following contents:

```
version: 0.0
os: linux
files:
- source: app/
  destination: /var/www/cfsummit.vegas
```

- Create a `bitbucket-pipelines.yml` file with the following contents:

```
image: atlassian/default-image:3

pipelines:
  branches:
    production:
      - step:
          name: Deploy to cfsummit.vegas Production
          deployment: production
          script:
            - apt-get update
            - apt-get install -y zip
            - zip -r cfsummit-vegas-production.zip .
            - pipe: atlassian/aws-code-deploy:1.4.0
              variables:
                AWS_ACCESS_KEY_ID: $AWS_ACCESS_KEY_ID
                AWS_SECRET_ACCESS_KEY: $AWS_SECRET_ACCESS_KEY
                AWS_DEFAULT_REGION: $AWS_DEFAULT_REGION
                COMMAND: 'upload'
                APPLICATION_NAME: 'cfsummit-vegas-production'
                BUNDLE_TYPE: 'zip'
                ZIP_FILE: 'cfsummit-vegas-production.zip'
          - pipe: atlassian/aws-code-deploy:1.4.0
              variables:
                AWS_ACCESS_KEY_ID: $AWS_ACCESS_KEY_ID
                AWS_SECRET_ACCESS_KEY: $AWS_SECRET_ACCESS_KEY
                AWS_DEFAULT_REGION: $AWS_DEFAULT_REGION
                COMMAND: 'deploy'
```

```
APPLICATION_NAME: 'cfsummit-vegas-production'  
DEPLOYMENT_GROUP: 'cfsummit-vegas-production-deployment-group'  
WAIT: 'true'  
FILE_EXISTS_BEHAVIOR: 'OVERWRITE'  
IGNORE_APPLICATION_STOP_FAILURES: 'true'
```

## Step 2 - Create the Repository Variables in BitBucket

- From our repository in BitBucket, click the Repository Settings link in the left column.

The screenshot shows the BitBucket interface for the repository 'cfsummit'. On the left, there's a sidebar with various navigation links: Source, Commits, Branches, Pull requests, Pipelines, Deployments, Jira issues, Security, Downloads, and Repository settings. A red arrow points to the 'Repository settings' link. The main content area shows the repository details: Last updated 2023-09-18, Language ColdFusion, Open pull requests 0, Branches 1, Watchers 1, Forks 0, and Access level Admin. It also displays a section for Bitbucket Pipelines with a message about not having a build tool configured yet, and a 'Set up a pipeline' button.

- Scroll to the bottom of the left column and click on the Settings link.
- Toggle the Enable Pipelines toggle.

A screenshot of the Bitbucket repository settings page for the 'cfsummit' repository. The left sidebar shows various repository features like Git LFS, Wiki, and Issue tracker. Under the 'PIPELINES' section, the 'Repository variables' link is highlighted with a red arrow. On the right, the 'Settings' tab is selected. A prominent red arrow points to the 'Enable Pipelines' toggle switch, which is currently off (grey). Below it is a button labeled 'Configure bitbucket-pipelines.yml'.

- Next, click the Repository Variables link in the left column.

A screenshot of the Bitbucket repository settings page for the 'cfsummit' repository. The left sidebar shows various repository features like Git LFS, Wiki, and Issue tracker. Under the 'PIPELINES' section, the 'Repository variables' link is highlighted with a red arrow. On the right, the 'Settings' tab is selected. The 'Enable Pipelines' toggle switch is now turned on, indicated by a green checkmark icon. Below it is a message: 'Next, configure and commit a **bitbucket-pipelines.yml** file into your repository. You can configure your own or choose from the example configuration we provide in our getting started guide.' A blue button labeled 'Configure bitbucket-pipelines.yml' is visible.

- From the Repository Variables page, create four variables:
  - Non-secure variable:
    - AWS\_DEFAULT\_REGION, with a value of us-east-2.
  - Secure variables:
    - AWS\_ACCESS\_KEY\_ID, with a value copied from the .csv credentials file we downloaded earlier.
    - AWS\_SECRET\_ACCESS\_KEY, with a value copied from the .csv credentials file we downloaded earlier.

The screenshot shows the Bitbucket Repository Variables page for the 'cfsummit' repository. The left sidebar lists various settings like Git LFS, Wiki, and Integrations, with 'Repository variables' selected. The main content area shows a table of environment variables:

Name	Value	Secured	Actions
AWS_ACCESS_KEY_ID	.....	<input checked="" type="checkbox"/>	
AWS_SECRET_ACCESS_KEY	.....	<input checked="" type="checkbox"/>	
AWS_DEFAULT_REGION	us-east-2	<input type="checkbox"/>	

Red arrows point to the 'Value' column of the first three rows (AWS\_ACCESS\_KEY\_ID, AWS\_SECRET\_ACCESS\_KEY, and AWS\_DEFAULT\_REGION), indicating the values to be copied from the CSV file.

## Part 7 - Moment of Truth

If we've done everything right, we should see a successful deployment, and our code will be pushed live to the server.

