

AWS Project Build a Game with a Continuous Deployment Pipeline from GitHub to S3

Creating a continuous deployment pipeline for your memory matching game using AWS Code Pipeline and S3 involves several steps. This guide will walk you through the process, from setting up your S3 bucket for static website hosting to creating the deployment pipeline.

Step 1: Set Up Your S3 Bucket for Static Website Hosting

1. Log in to the AWS Management Console and navigate to the S3 service.
2. Create a new S3 bucket. Choose a unique name for your bucket.
3. Enable static website hosting** for your bucket. Under the "Properties" tab, find the "Static website hosting" section and click "Edit". Select "Enable" and specify the index document (usually `index.html`).
4. Set bucket policy to make your website publicly accessible. Go to the "Permissions" tab, click "Bucket Policy", and add a policy that grants public read access to your bucket.

*** bucket policy file is in my repository***

Amazon S3 > Buckets > Create bucket

Create bucket [Info](#)

Buckets are containers for data stored in S3.

General configuration

AWS Region

US East (N. Virginia) us-east-1

Bucket type [Info](#)

☒ **General purpose**

Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ **Directory - New**

Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [Info](#)

game-continous-deployment

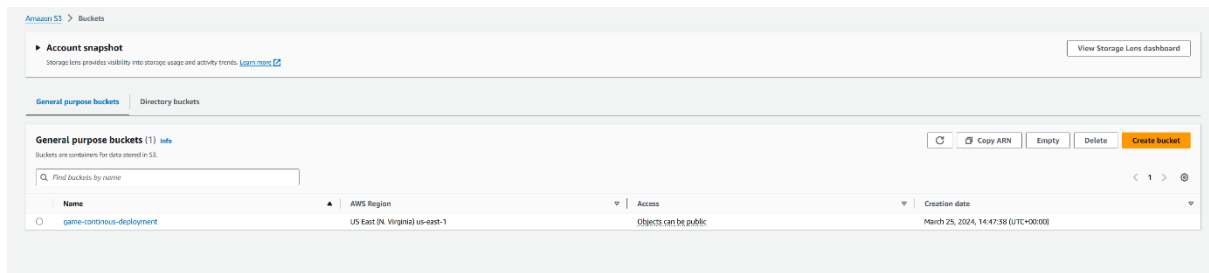
Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - *optional*

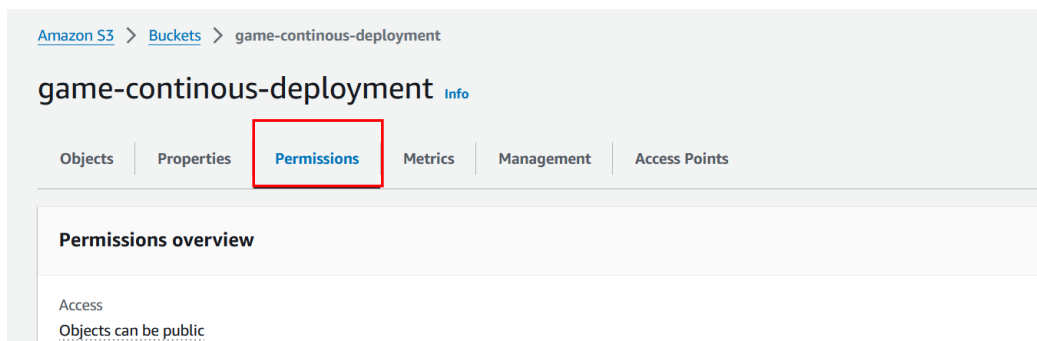
Only the bucket settings in the following configuration are copied.

Format: s3://bucket/prefix

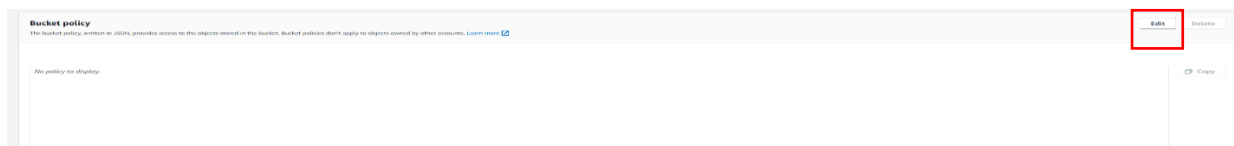
- Write unique S3 bucket name.
- Select create bucket.



- Bucket is created.
- Click on the bucket name.
- Go to permission tab.



- Select Bucket Policy.
- Click on edit.



- Write the bucket policy.
- This bucket policy sets get an object.
- Write your bucket ARN followed by '/*'.

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)

Bucket ARN
arn:aws:s3::game-continous-deployment

Policy

```
1 {  
2   "Id": "Policy1711378131075",  
3   "Version": "2012-10-17",  
4   "Statement": [  
5     {  
6       "Sid": "Stmt17113781310099",  
7       "Action": [  
8         "s3:GetObject"  
9       ],  
10      "Effect": "Allow",  
11      "Resource": "arn:aws:s3::game-continous-deployment/*",  
12      "Principal": "*"   
13    }  
14  ]  
15 }
```

Write you bucket ARN here.

- Click on save changes.
- Select the properties tab and scroll to bottom and select Static Web hosting.

Amazon S3 > Buckets > game-continous-deployment

game-continous-deployment [Info](#)

Objects **Properties** [Permissions](#) [Metrics](#) [Management](#) [Access Points](#)

Permissions overview

Access
Objects can be public

- Enable it .
- In index document tab write index.html.
- Click on save changes.
- After that you will get any website URL, which will be able to display the content of the website.

Amazon S3 > Buckets > game-continous-deployment > Edit static website hosting

Edit static website hosting [Info](#)

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

☐ Disable

☒ Enable

Hosting type

☒ Host a static website
Use the bucket endpoint as the web address. [Learn more](#)

☐ Redirect requests for an object
Redirect requests to another bucket or domain. [Learn more](#)

For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see [Using Amazon S3 Block Public Access](#)

Index document
Specify the home or default page of the website.

index.html

Error document - optional
This is returned when an error occurs.

error.html

Redirection rules - optional
Redirection rules, written in JSON, automatically redirect webpage requests for specific content. [Learn more](#)

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

Enabled

Hosting type

Bucket hosting

Bucket website endpoint

When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. [Learn more](#)

<http://game-continous-deployment-13-website-us-east-1.amazonaws.com/>

Step 2: Prepare Your Game Code

Ensure your game code (HTML, CSS, and JavaScript) is ready and organized in a GitHub repository. Your repository should have a structure that reflects the static website hosting setup, with `index.html` at the root.

*** the code for the website is on my GitHub repository and the images used. All the content used for the website is on my GitHub repository.

Step 3: Create an AWS CodePipeline

1. Navigate to the AWS CodePipeline service in the AWS Management Console.
2. Create a new pipeline. Name it appropriately, e.g., "MemoryMatchGameDeployment".
3. Choose the source provider. Select GitHub as your source provider and connect your GitHub account. Choose the repository and branch where your game code is hosted.
4. Choose the build provider. Since your game consists of static files (HTML, CSS, JavaScript), you don't need a build step. You can skip this step.
5. Choose the deploy provider. Select Amazon S3 as your deploy provider. Configure it to deploy to the S3 bucket you created earlier.
6. Review and create the pipeline. AWS CodePipeline will automatically detect changes in your GitHub repository and deploy them to your S3 bucket.

- Go to Codepipeline.
- Create Codepipeline.
- Write the name of the pipeline.
- Pipeline type P1.
- Select the new role, it will automatically create a new role.

Pipeline settings**Pipeline name**

Enter the pipeline name. You cannot edit the pipeline name after it is created.

game-continuous-deployment

No more than 100 characters

Pipeline typeThe pipeline type determines the pipeline structure and availability of parameters such as triggers. Pipeline type selection will impact features and pricing. [Which pipeline is right for me?](#)☒ V1☐ V2**Execution mode**

Choose the execution mode for your pipeline. This determines how the pipeline is run.

☒ Superseded

A more recent execution can overtake an older one. This is the default.

☐ Queued (Pipeline type V2 required)

Executions are processed one by one in the order that they are queued.

☐ Parallel (Pipeline type V2 required)

Executions don't wait for other runs to complete before starting or finishing.

Service role☒ New service role

Create a service role in your account

☐ Existing service role

Choose an existing service role from your account

Role name

AWSCodePipelineServiceRole-us-east-1-game-continuous-deployment

Type your service role name

☒ Allow AWS CodePipeline to create a service role so it can be used with this new pipeline**Variables**You can add variables at the pipeline level. You can choose to assign the value when you start the pipeline. Choosing this option requires pipeline type V2. [Learn more](#)

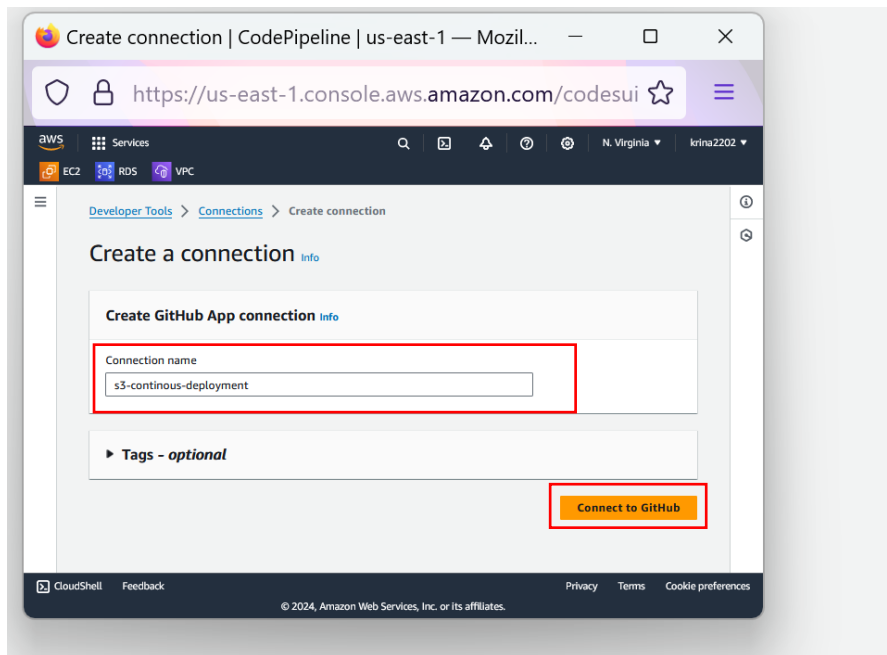
No variables defined at the pipeline level in this pipeline.

Add variable

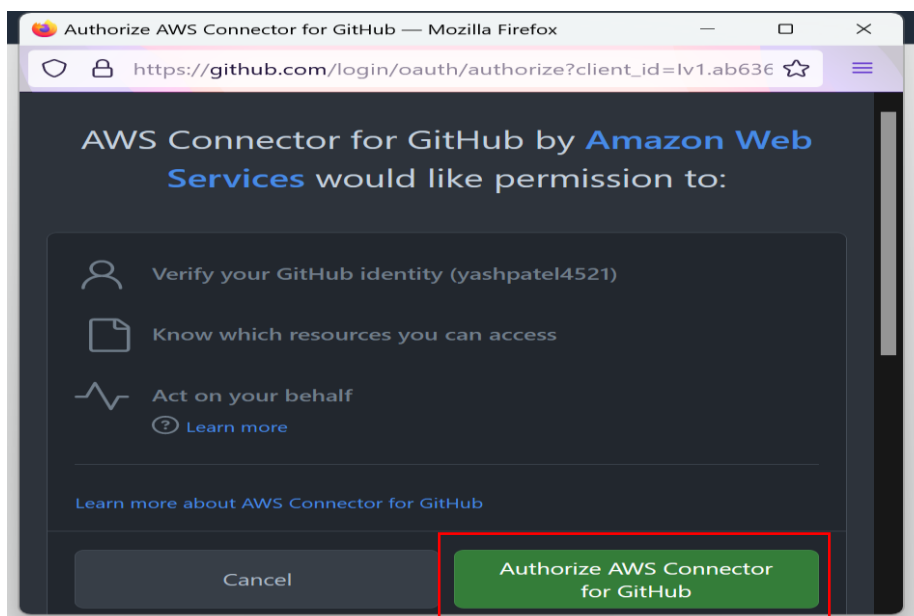
You can add up to 50 variables.

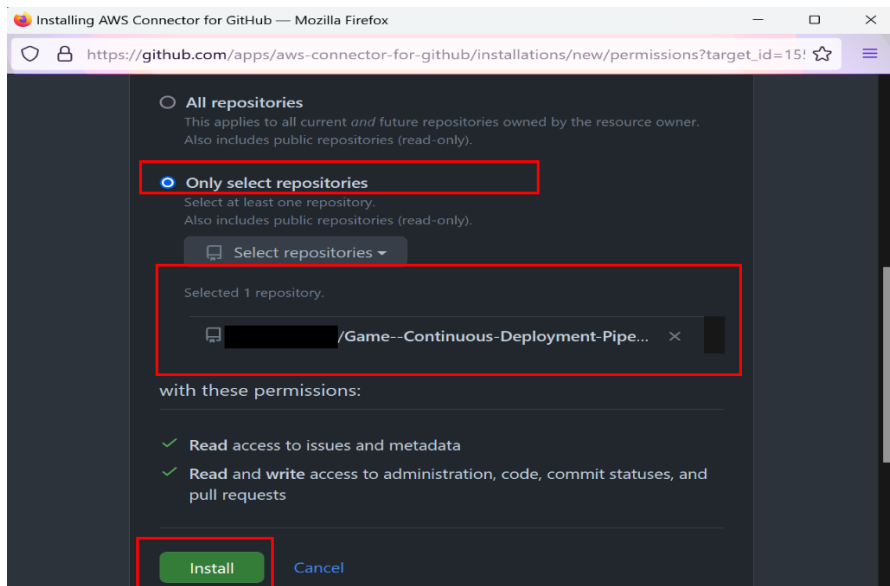
☒ The first pipeline execution will fail if variables have no default values.**► Advanced settings**

- Select next.
- Select the source provider, here we are using GitHub so select GitHub version2.
- When you click on next a window will pop up to get the connection from GitHub.
- Write the connection name.
- Click on connect to GitHub.

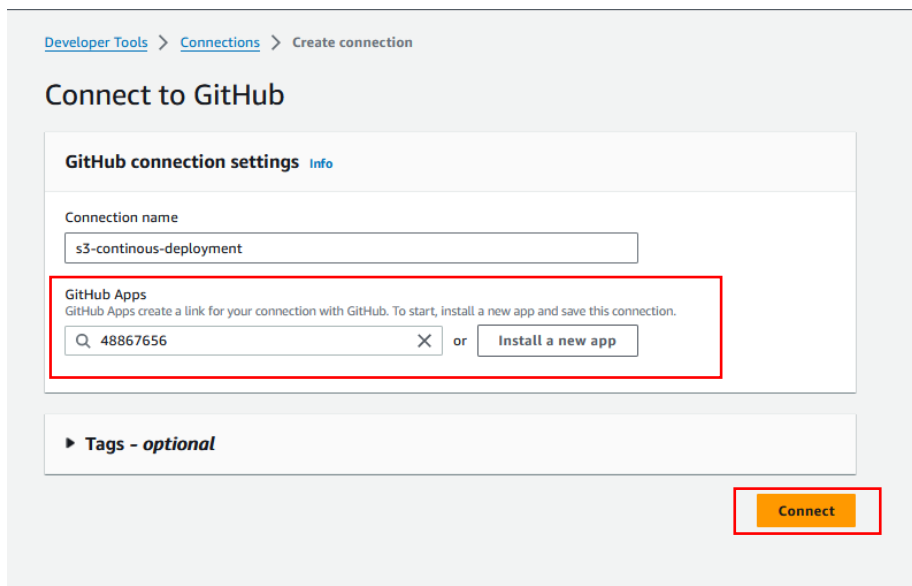


- Connect to your github account.
- Authorize it.
- Click on only repository and select the uploaded code repository.





- Click on install.
- The app will then install.
- You can see some numbers generated.
- Then click on connect.



Source

Source provider

This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.

GitHub (Version 2)



New GitHub version 2 (app-based) action

To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. [Learn more](#)

Connection

Choose an existing connection that you have already configured, or create a new one and then return to this task.

arn:aws:codestar-connections:us-east-1:397296617840:connection/a832f5e

or [Connect to GitHub](#)



Ready to connect

Your GitHub connection is ready for use.

Repository name

Choose a repository in your GitHub account.

Continuous-Deployment-Pipeline-from-GitHub-to-S3

You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.

Default branch

Default branch will be used only when pipeline execution starts from a different source or manually started.

main

Output artifact format

Choose the output artifact format.



CodePipeline default

AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include Git metadata about the repository.



Full clone

AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full Git clone. Only supported for AWS CodeBuild actions.

- Select the connection you created.
- Select the repository name.
- Default branch as main.
- Click next.
- Skip the built stage.
- Add Deploy, so we are deploying on S3 bucket so select s3 bucket.
- Select the bucket which you created.
- Tick the Extract file before deploy.
- Click next.

Info

Step 4 of 5



You cannot skip this stage

Pipelines must have at least two stages. Your second stage must be either a build or deployment stage. Choose a provider for either the build stage or deployment stage.

Deploy

Deploy provider

Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.

Amazon S3

Region

US East (N. Virginia) ▼

Bucket

Q game-continous-deployment X

Deployment path - *optional*

--

☒ Extract file before deploy

The deployed artifact will be unzipped before deployment.

► **Additional configuration**

Cancel

[Previous](#)

Next

- The deployment stage has begin deploying the resources into S3 bucket.

The screenshot shows the Argo CD web interface for an application named 'game-continous-deployment'. At the top, there's a breadcrumb trail: 'Dashboard' > 'Applications' > 'game-continous-deployment'. The application's status is 'SUCCEEDED'. Below this, there are two tabs: 'Source' and 'Deploy'. The 'Source' tab is selected, showing the repository 'game-continous-deployment' with a 'Refresh' button. The 'Deploy' tab is also visible, showing the deployment details. The application is in a 'SUCCEEDED' state.

Developer Tools > CodePipeline > Pipelines > game-continous-deployment

game-continous-deployment

Pipeline type: **V1** Execution mode: **SUPERSEDED**

Source Succeeded
Pipeline execution ID: [4db533c3-404f-4a52-9faa-9176c22e5ad1](#)

Source
[GitHub \(Version 2\)](#)
Succeeded - Just now
[ed459240](#)
[View details](#)

[ed459240](#) Source: Update script.js

Disable transition

Deploy Succeeded
Pipeline execution ID: [4db533c3-404f-4a52-9faa-9176c22e5ad1](#)

Deploy
[Amazon S3](#)
Succeeded - Just now
[View details](#)

[ed459240](#) Source: Update script.js

- Finally its been deployed !!! .
- Go to the static website URL and copy it to the browser and see its up and running the website.
- When you click on the start button you can now see that the game is running.

Static website hosting [Edit](#)

Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

Enabled

Hosting type

Bucket hosting

Bucket hosting endpoint

When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. [Learn more](#)

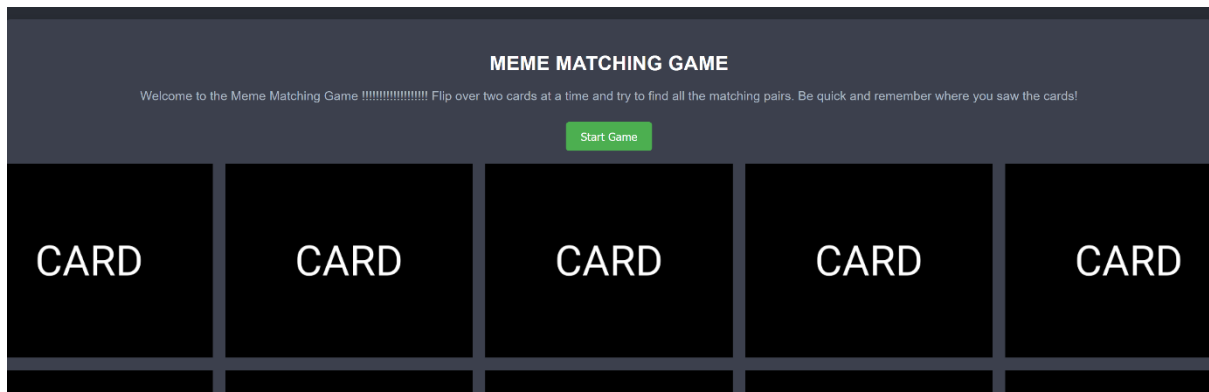
[http://game-continous-deployment.s3-website-us-east-1.amazonaws.com/](#)

game-continous-deployment.s3-website-us-east-1.amazonaws.com

MEME MATCHING GAME

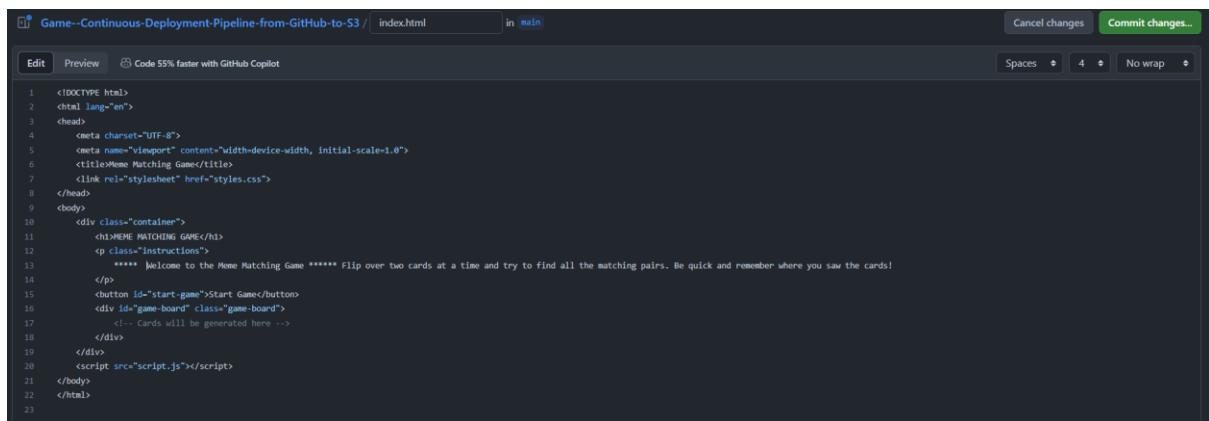
Welcome to the Meme Matching Game !!!!!!!!!!!!!!! Flip over two cards at a time and try to find all the matching pairs. Be quick and remember where you saw the cards!

[Start Game](#)

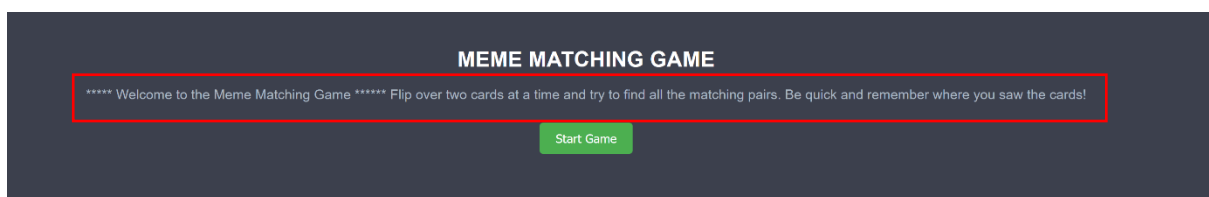
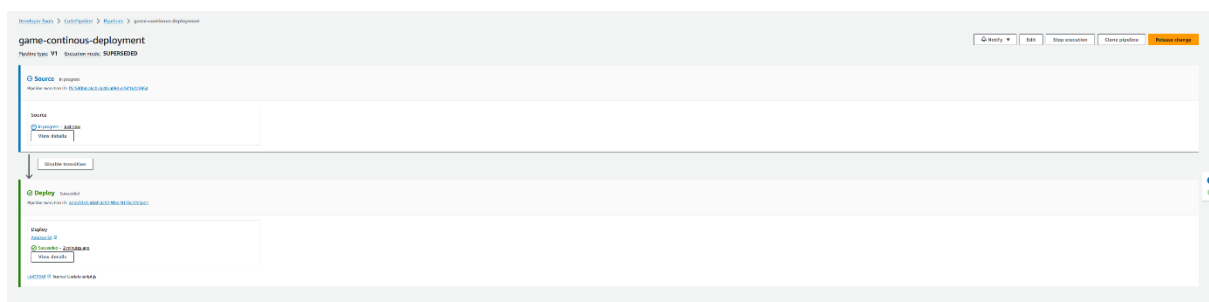


Step 4: Test Your Deployment

1. Make a change to your game code in your GitHub repository.
 2. Commit and push the change to the branch you're using with AWS CodePipeline.
 3. Monitor the pipeline execution in the AWS CodePipeline console. You should see the pipeline automatically triggering a deployment upon detecting the change.
 4. Access your game by navigating to the S3 bucket's static website endpoint (found in the bucket properties under "Static website hosting").
- Now we will make some changes to the index.html file and let's see how automatically it deploys on the S3 bucket.
 - Making the changes to the line no 13. (Like adding some **** welcome to the meme matching game *****) something like that.
 - Then commit the changes. It will automatically trigger the code pipeline to make the changes to the s3 bucket where it is deployed.
 - No refresh the web URL and see the changes have been done.



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Meme Matching Game</title>
7   <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10   <div class="container">
11     <h1>MEME MATCHING GAME</h1>
12     <p class="instructions">
13       ***** Welcome to the Meme Matching Game ***** Flip over two cards at a time and try to find all the matching pairs. Be quick and remember where you saw the cards!
14     </p>
15     <button id="start-game">Start Game</button>
16     <div id="game-board" class="game-board">
17       <!-- Cards will be generated here -->
18     </div>
19   </div>
20   <script src="script.js"></script>
21 </body>
22 </html>
```



Successfully !!!!!

Conclusion

By following these steps, you've set up a continuous deployment pipeline for your memory matching game. This setup allows you to easily update your game by pushing changes to your GitHub repository, and AWS CodePipeline will automatically deploy these changes to your S3 bucket. You can now focus on enhancing your game with additional features and improving the user experience.