Capital Guessing Game

**SPL-BE-200-DVCTGG-1 - Version 1.0.3**

© 2024 Amazon Web Services, Inc. or its affiliates. All rights reserved. This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited. All trademarks are the property of their owners.

Note: Do not include any personal, identifying, or confidential information into the lab environment. Information entered may be visible to others.

Corrections, feedback, or other questions? Contact us at [*AWS Training and Certification*](https://support.aws.amazon.com/#/contacts/aws-training).

**Lab Overview**

This lab demonstrates how you can build and run a Capital Guessing Game by using state machines to start games, add participants, ask questions, keep scores, and more. State machines are a way of defining and visualizing the flow of your application. They specify how the application should behave under different conditions, and how it should transition between states. With AWS Step Functions, you can define state machines by using a JSON-based language, visualize and test them by using the AWS Management Console, and integrate them with other AWS services and third-party applications.

You update the application Step Functions code snippet that is responsible for starting the game. After you add the code, you redeploy the application by using the AWS Serverless Application Model (AWS SAM). AWS SAM is an open-source framework for building serverless applications on AWS. It extends AWS CloudFormation to provide a simplified way of defining the Amazon API Gateway APIs, AWS Lambda functions, and Amazon DynamoDB tables that your serverless application needs.

After AWS SAM finishes redeploying the application, you start a game and open a second browser tab to simulate a second participant who’s playing the trivia game. Then, as the game progresses, you observe the state machines and the workflows that they follow.

OBJECTIVES

By the end of this lab, you will be able to do the following:

* Test the functionality of the Capital Guessing Game.
* Update the game’s Step Functions start\_execution code snippet.
* Deploy the updated application.
* Visualize how the application uses Step Functions by observing the state machine’s workflow.

TECHNICAL KNOWLEDGE PREREQUISITES

To successfully complete this lab:

* Familiarity with the basic navigation of the AWS Management Console.
* Versed in editing and running scripts using an AWS Cloud9 code editor and terminal.
* A basic understanding of and familiarity with Amazon API Gateway, AWS Serverless Application Model (AWS SAM), AWS Lambda, AWS CloudFormation, and AWS Step Functions.
* Prior experience with AWS services and serverless computing is helpful, but isn’t required.

DURATION

This lab requires *60* minutes to complete.

ICON KEY

Various icons are used throughout this lab to call attention to different types of instructions and notes. The following list explains the purpose for each icon:

* **Command:** A command that you must run.
* **Expected output:** A sample output that you can use to verify the output of a command or edited file.
* **Note:** A hint, tip, or important guidance.
* **Consider:** A moment to pause to consider how you might apply a concept in your own environment or to initiate a conversation about the topic at hand.

**Start lab**

1. To launch the lab, at the top of the page, choose **Start lab**.

**Caution:** You must wait for the provisioned AWS services to be ready before you can continue.

1. To open the lab, choose **Open Console**.

You are automatically signed in to the AWS Management Console in a new web browser tab.

**WARNING:** **Do not change the Region unless instructed.**

COMMON SIGN-IN ERRORS

**Error: You must first sign out**



If you see the message, **You must first log out before logging into a different AWS account:**

* Choose the **click here** link.
* Close your **Amazon Web Services Sign In** web browser tab and return to your initial lab page.
* Choose **Open Console** again.

**Error: Choosing Start Lab has no effect**

In some cases, certain pop-up or script blocker web browser extensions might prevent the **Start Lab** button from working as intended. If you experience an issue starting the lab:

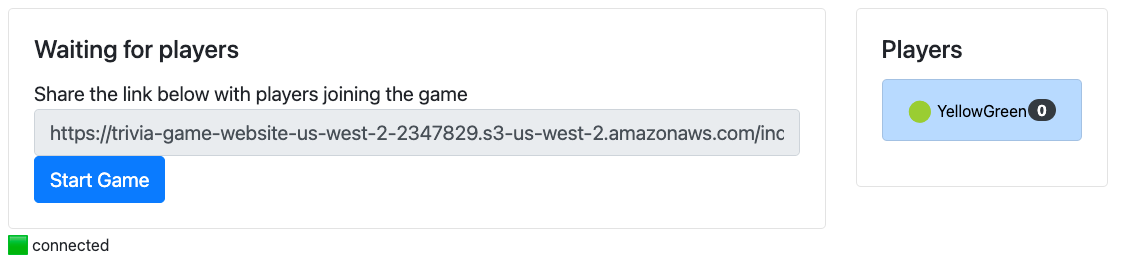
* Add the lab domain name to your pop-up or script blocker’s allow list or turn it off.
* Refresh the page and try again.

**Task 1: Test the Capital Guessing Game for functionality**

In this task, you test the game to see if it works.

1. From the navigation pane at the left of these instructions, copy the **WebsiteURL** value and in a new browser tab, paste the URL.
2. From the **Get Started** section, choose **Create a New Game** .

**Expected output:**

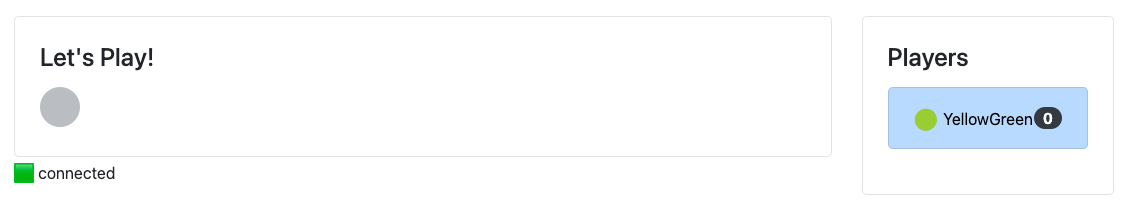


*Image depicts the current status of the game. It contains a link to share the game with other players so they can join the game. It shows a Start Game button and shows the names of current players.*

Notice that it adds a player with a random name to the right. You use this player to participate in the game. You can start the game with only one player.

1. Choose **Start Game** .

**Expected output:**



*Image depicts the current status of the game as “Let’s Play!” It shows names of the current players.*

You see *Let’s Play!* with an animated circle.

There seems to an issue with the application code to start the game because the status never moves on from this state. You address this issue next.

 Congratulations! You have successfully tested the game and determined that a code update is required to allow the game to start.

**Challenge A: Update the Capital Guessing Game’s state machine**

Welcome to the code challenge. In this challenge, you update a code snippet for the *trivia\_startgame* function, which is responsible for starting a game. Your challenge is to add code between the *CHALLENGE-A START* and *CHALLENGE-A END* comments. Then, you redeploy the application and observe how the application uses the state machines. If you get stuck, you can refer to three hints and a solution code snippet.

Connect to the AWS Cloud9 environment.

1. Copy the **Cloud9Environment** value from the navigation pane at the left of these instructions, and paste it into a new browser tab.
2. Close the **Welcome** tab.
3. The following challenge details outline what you need to know to get started:

* The application file you need to update is *back-end-python/gameactions/app.py*. A full solutions file (named *app-solution.py*) is also in the same directory as the *app.py* file.
* You need to add three parameters.
* The code you need to complete is prefaced with CHALLENGE-A START. It calls the *STEPFUNCTIONS.start\_execution* to start a state machine execution with the given input.

**Note:** The following snippet is an example of the code that you will update.

# Student implements the start execution function

STEPFUNCTIONS.start\_execution(

# CHALLENGE-A START

...

# CHALLENGE-A END

)

1. To figure out what code you need to add, refer to the **AWS Step Functions** documentation for the **start\_execution** function. See the link in the [Additional resources](https://labs.skillbuilder.aws/sa/lab/arn%3Aaws%3Alearningcontent%3Aus-east-1%3A470679935125%3Ablueprintversion%2FSPL-BE-200-DVCTGG-1%3A1.0.3-3c28a1c2/en-US#ar) section.

**Hint 1**

**Hint 2**

**Hint 3**

**Solution**

**Note:** After you update the code snippet and it matches the solution, be sure to save your changes.

 Congratulations! You have updated the code the required to start the game.

**Task 2: Deploy the application with the latest updates**

Now, you are ready to redeploy the application by using AWS SAM.

1. **Command:** Switch directories to the **~/environment** directory, build the deployment files, and redeploy the updated application by running the following batch of commands:

cd ~/environment; sam build; sam deploy

**Expected output:** Output has been truncated.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\* This is OUTPUT ONLY. \*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

CloudFormation outputs from deployed stack

--------------------------------------------------------------------------------------------------

Outputs

--------------------------------------------------------------------------------------------------

Key TriviaWebSocketApi

Description API Gateway websockets endpoint

Value wss://k4fsmp02c0.execute-api.us-west-2.amazonaws.com/Prod

--------------------------------------------------------------------------------------------------

Successfully created/updated stack - sam-game in us-west-2

 Congratulations! You have successfully deployed the changes to update the application by adding the Step Functions code snippet to start the game.

**Task 3: Start the game and review the state machine workflow**

1. Switch to the browser tab that’s opened to the AWS Management Console.
2. At the top of the AWS Management Console, in the search bar, search for and choose

Step Functions

.

The browser takes you to the *Step Functions > State Machines* page.

**Note:** You see values for **Total**, **Running**, **Succeeded**, **Failed**, **Timeout**, and **Aborted**. These values will be populated while players answer questions and play games.

1. From the **State machines** section, choose the **TriviaStateMachine** link.

Links will populate under the *Executions* section after games start. In this section, you will observe the state machine as the application starts.

1. Switch back to the browser tab that’s opened to the game’s frontend and refresh this browser tab.

**Note:**

* If this tab was closed by accident, copy the **WebSiteURL** value (from the navigation pane to the left of these instructions) and in a new browser tab, paste the URL.
* This game is best observed with two monitors. Use one monitor to engage the frontend application, and use the other monitor for observing the state machine as the application starts.

1. Choose **Create a New Game** .

You see that one player has joined.

1. In the **Share the link below with players joining the game** box, copy the link and in a new browser tab, paste the URL.
2. Simulate a second player joining the game by choosing **Join**.

You now see two players listed on the frontend application tab.

**Consider:** If you want to brush up on your knowledge of country capitals before you start the game, expand the following *Questions and Answers* section.

**Questions and Answers**

**Note:** You get only 5 seconds to answer each question, so be prepared to choose your answers quickly.

1. When ready, choose **Start Game** .

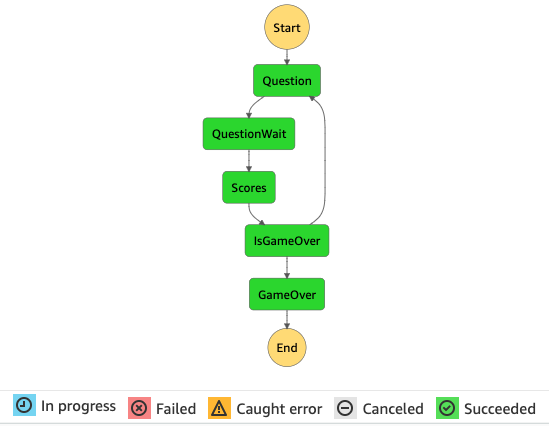
**Expected output:**

The game now starts asking questions, such as “*What is the capital of Mongolia?*” When you answer questions correctly, it starts adding points for the specific user.

1. Go to the **State machines** page and choose the  refresh button.
2. In **Executions** section, choose the link that starts with **game-**.

The browser takes you to the **Execution** page. In this page, you can see a graph view of the various steps that occurred as the game continued.

You can choose each step and see the code that was invoked to the right of the screen. Each tab shows various details for the *Input*, *Output*, *Details*, *Definition*, and *Events* that are invoked.



*The image depicts a Step Functions state machine workflow for a trivia game. It begins with a function that starts the game > a function that asks a question > a function that adds a score > it checks to see if the game is over > if not, it loops back to the functions to ask a question and add a score. When the function for IsGameOver is invoked, it calls the GameOver function and the game ends.* All steps are invoked in this diagram.

1. If you want to play the game again to try and improve your score, choose the **Restart** > **Create a New Game** > **Start Game** .

 Congratulations! You have successfully started a game and reviewed the state machine’s workflow.

**Conclusion**

 Congratulations! You now have successfully:

* Tested the functionality of the Capital Guessing Game.
* Updated the game’s Step Functions start\_execution code snippet.
* Deployed the updated application.
* Visualized how the application uses Step Functions by observing the state machine’s workflow.

**End lab**

Follow these steps to close the console and end your lab.

1. Return to the **AWS Management Console**.
2. At the upper-right corner of the page, choose **AWSLabsUser**, and then choose **Sign out**.
3. Choose **End lab** and then confirm that you want to end your lab.

**Additional resources**

* To learn more, see the **AWS Step Functions** documentation for the [start\_execution](https://docs.aws.amazon.com/step-functions/latest/apireference/API_StartExecution.html) operation.

For more information about AWS Training and Certification, see [*https://aws.amazon.com/training/*](https://aws.amazon.com/training/).

*Your feedback is welcome and appreciated.*  
If you would like to share any feedback, suggestions, or corrections, please provide the details in our [*AWS Training and Certification Contact Form*](https://support.aws.amazon.com/#/contacts/aws-training).