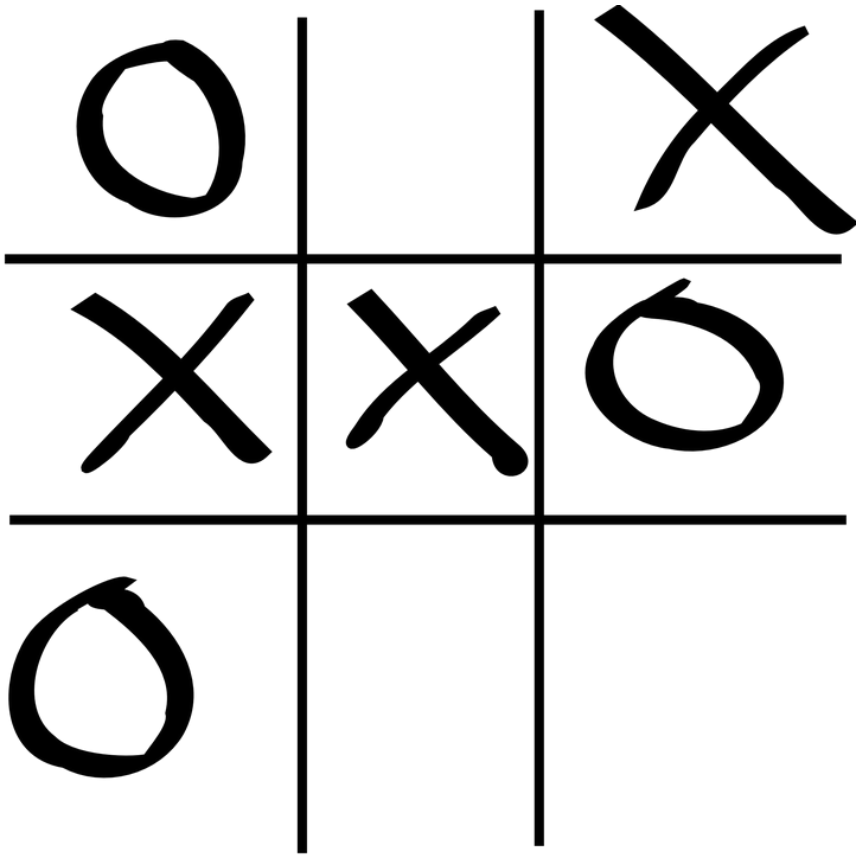


Alumno: Guerrero Azpitarte Adrian Eduardo

Documentación: Juego de Gato con conexión a AWS Lambda implementada a un API-Gateway de Amazón

Materia: Temas selectos de Ingeniería en Software

Profesor : Francisco Alfredo Adam Dajer

Semestre 2019-1 **Grupo #1**

Definición del Proyecto

Se requiere realizar una aplicación móvil para iOS o Android, que logré hacer una petición HTTP. Con un método Request a un API-Gateway de Amazón el cual deberá estar alojando una función Lambda que por medio de la petición, regresa una respuesta al dispositivo móvil. El desarrollo en el móvil implica un juego de “gato” o “3 en raya”. Depende de cada programar cómo hacer la llamada a Lambda.

Objetivos

Comprender, implementar y hacer uso de una función lambda de AWS (Amazon Web Services). Con ayuda de un API-Gateway en un juego para móviles. En este caso iOS.

Requerimientos Funcionales

1.-Internet: El sistema debe de contar con acceso a internet si se desea ver la respuesta de la función Lambda.
2.-Para poder acceder al Juego, se deben de llenar las 2 cajas de texto, de lo contrario no se podrá accesar al juego
3.-Para ganar o terminar el juego se deben de seguir las reglas ya conocida del juego de gato o 3 en raya.
4.-El sistema después de haber terminado una sesión o salir de la aplicación. No guardará los datos de la partida que se quedo suspendida.

5.-Se requiere un emulador en Xcode para un iPhone 8 Plus. O un dispositivo igual para poder visualizar la correcta ejecución de la aplicación.
6.- El juego es en un tablero de 9 casillas.
7.- A cada jugador se le asigna una ficha y debe jugar con ella, 1 ficha por turno.
8.- Se gana cuando algún jugador acumula 3 fichas del mismo tipo en una linea recta o inclinada.
9.-Es para 2 jugadores
10.- El juego esta diseñado para un total de 5 partidas
12.- Se siguen las reglas básicas del juego “3 en raya”, respetando un turno por cada jugador, no se vale encimar piezas en la misma posición de otra pieza.
13.-Tiene que correr en un dispositivo móvil.

Requerimientos No Funcionales

1.-Se puede usar en cualquier iPhone 8
2.- La aplicación es segura al hacer una llamada Lambda dentro de los servicios de AWS
3.- El tiempo de respuesta suele ser inmediato. De lo contrario no se podrá visualizar la respuesta

Lambda
4.-Backend corre en aws lambda con comunicación mediante API Gateway

Riesgos

Existe una gran cantidad de riesgos. Como el tiempo de entrega, el hecho de dependemos mucho de la documentación de AWS, y pueden existir variaciones en cuanto a versiones de Xcode o Android Studio. Es posible que no se alcance a cumplir todos los requisitos planteados.

Cada desarrollador esta elaborando su proyecto de manera individual, lo mas probable. es que no se llegue a una misma solución. Sino que haya mucha variación en la entrega final.

Entregable

Aplicación funcional del juego de Gato con llamada a una función lambda mediante un API-Gateway.

Beneficios del proyecto.

El desarrollador implementará un micro servicio de backend a su aplicación móvil con los servicios que nos puede proveer amazón. Ya que la función Lambda hace justamente eso, ejecutar código cuándo es llamado.

Análisis del Problema

Para poder realizar plenamente este proyecto escolar se requieren conocimientos acerca de los servicios de AWS Lambda, conceptos como: API-Gateway, JSON, programación en Java o algún otro lenguaje permitido por Lambda para poder hacer una función den sus servicios de la nube (backend). Además conocer la herramienta de Android Studio o Xcode (con conocimientos de lenguaje Swift) para implementar todo ello en una aplicación móvil.

La función lambda funge como nuestro backend, al mandar solamente la petición, ésta hace los cálculos necesarios y simplemente nos regresa una respuesta. Esa es la idea de la función lambda, Gracias a nuestro API Gateway, al enlazarlo a la función lambda, de hecho se puede ejecutar desde cualquier navegador o en este caso con una petición request desde el móvil. Desafortunadamente para esta entrega la función lambda hace funcionalidades básicas. De hecho, el usuario al hacer click en un botón para posicionar su pieza en el juego de gato. Sin saberlo esta haciendo la petición y como resultado le regresa una alerta con un mensaje ya preestablecido.

Diseño

Algoritmo:

El algoritmo principal, que maneja todo el flujo, se encuentra en la clase CatGameViewController. Dicha clase contiene entre otras cosas, 9 botones, 10 imágenes, 1 para el tablero y 9 para poner la ficha correspondiente. Cerca de 9 labels que despliegan información acerca durante el avance del juego, Y cada ciertas acciones ocurren alertas para los usuarios.

Empezando con el flujo de la clase nos encontraremos con los 9 botones. Cada botón al ser presionado manda a llamar una función llamada makingRequest(), se hablara de ella después, y comprueba con la ayuda de la variable auxiliar, de quien es el turno, de acuerdo al valor de auxiliar decidirá que imagen poner en el tablero. Después se desactiva el botón para que no pueda volver a ser utilizado.
Terminando eso manda a llamar una función llamada checkIfYouWin() que básicamente sirve para ver si ah ganado con ese movimiento. Sino aumenta en auxiliar en 1 movimiento para que el siguiente jugador realice su jugada. Y al final de cada botón llama a una función llamada displayMessage() de la cual se hablara también después. Esto para los 9 botones.

La siguiente función en ser llamada es checkIfYouWin(). Básicamente es una serie de if's anidados que comprueba 3 condiciones en cada if, para ver si las 3 imágenes que se han puesto nos anuncian un ganador. Un if para cada posibilidad de ganador y para las 2 diferentes "fichas". Si algún if entra a la condición significa que un jugadro ah ganado la partida. Lo que dispara una alerta anunciando el jugador que gano con su respectiva ficha con la función myAlertplayerOne/TwoWin(). Y, dependiendo el jugador que gano, aumenta en 1 una variable player1/2WinCounter para mostrar en un label las victorias que lleva con la función updateCounterO/X. Posteriormente se limpia el tablero y se habilitan los botones con la función clearBoardAFterWin(). Se aumenta una variable llamada matchCounter para llevar aumentar en 1 la victoria del jugador y de igual forma se aumenta la variable victoriaActual para llevar un conteo en un label de la partida que se esta llevando a cabo. Si se llena el tablero sin un ganador se dispara otra alerta anunciando empate y se limpia el tablero con ayuda de myAlertTie. Cuando termina, se manda a llamar la función llamada breakGame() de la cual se hablará después.

La siguiente función en ser llamada es clearBoardAFterWin(), que limpia las imagenes de las fichas actuales a imágenes en blanco, simulando que se reinician las piezas, se habilitan los botones nuevamente para poder usar el movimiento que se deseé. Y la variable auxiliar se reinicia a cero para que se renueve el ciclo de jugadas,

Las siguientes funciones son las diferentes alertas. Como la de ganador X o ganador 0, empate o incluso la alerta después de haber jugado las 5 partidas.

La función makingRequest() se encarga de hacer la llamada al API para procesar la función que alojamos en los servidores de amazón, que regresa un numero aleatorio. La URL donde esta alojada mi función lambda es "https://57272q9fsh.execute-api.us-east-1.amazonaws.com/Deploy/randomnumber/realrandomnumber"

Al final solo se manda a llamar un mensaje dependiendo de lo que recibió lambda

Diagrama de Flujo:

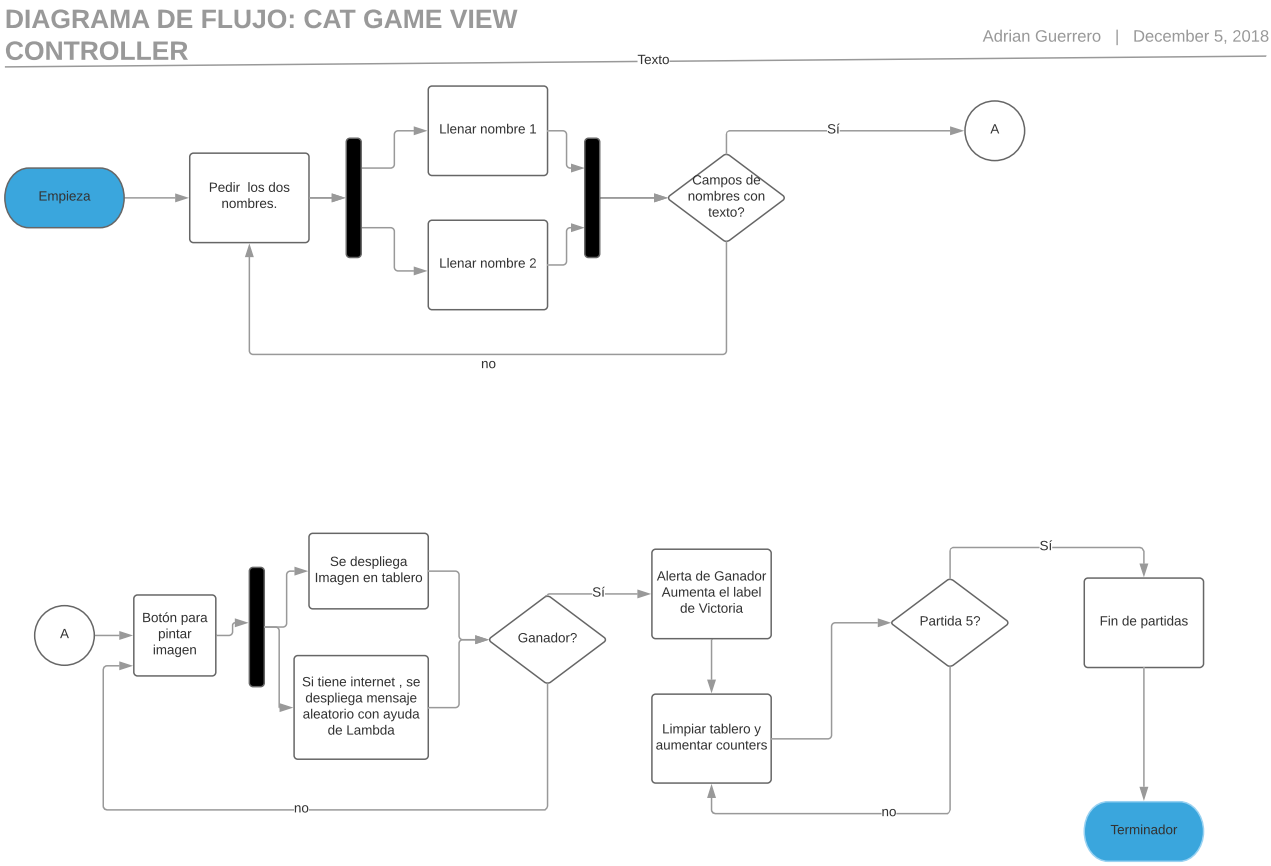


Diagrama de Actividades:

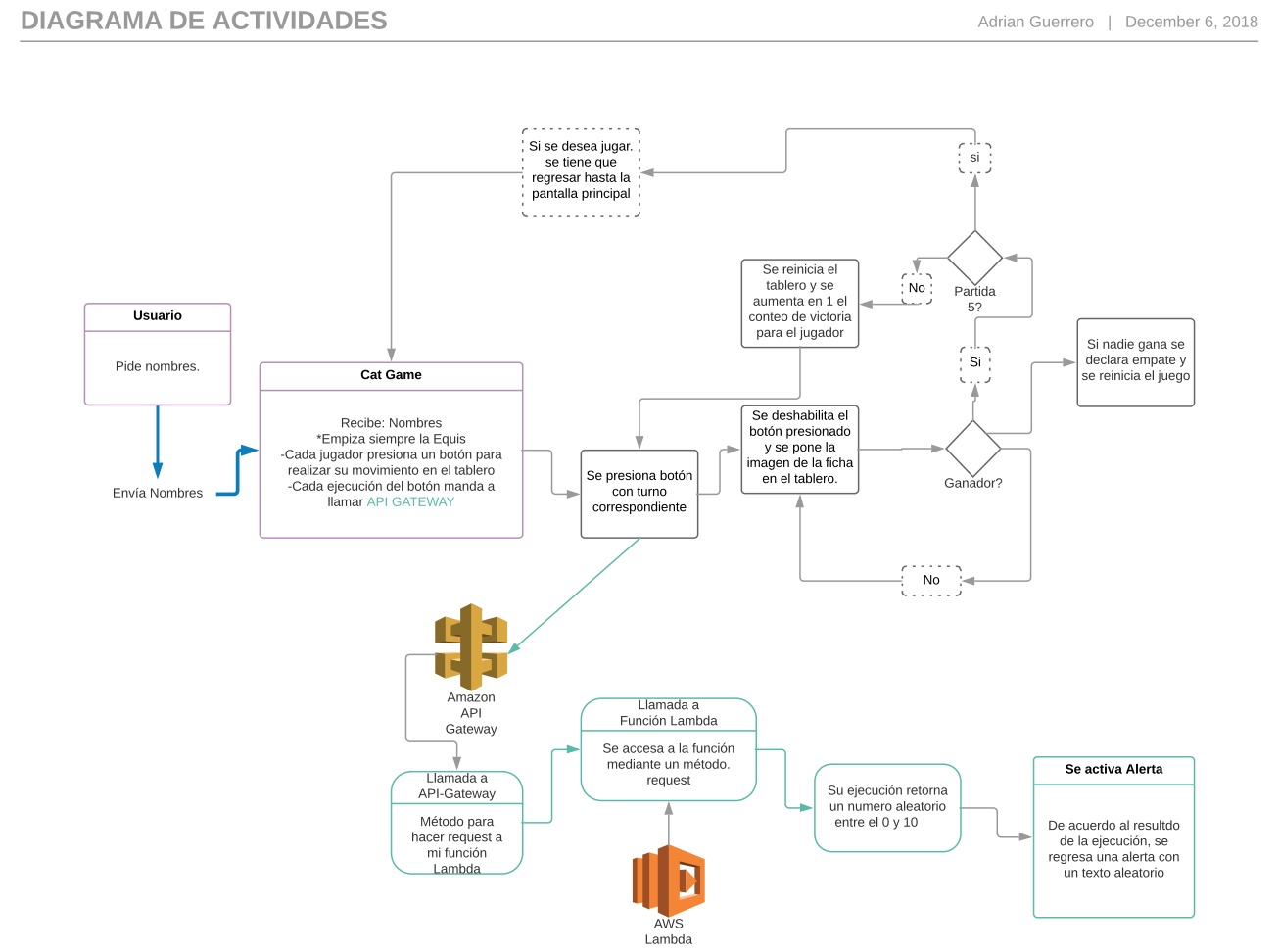
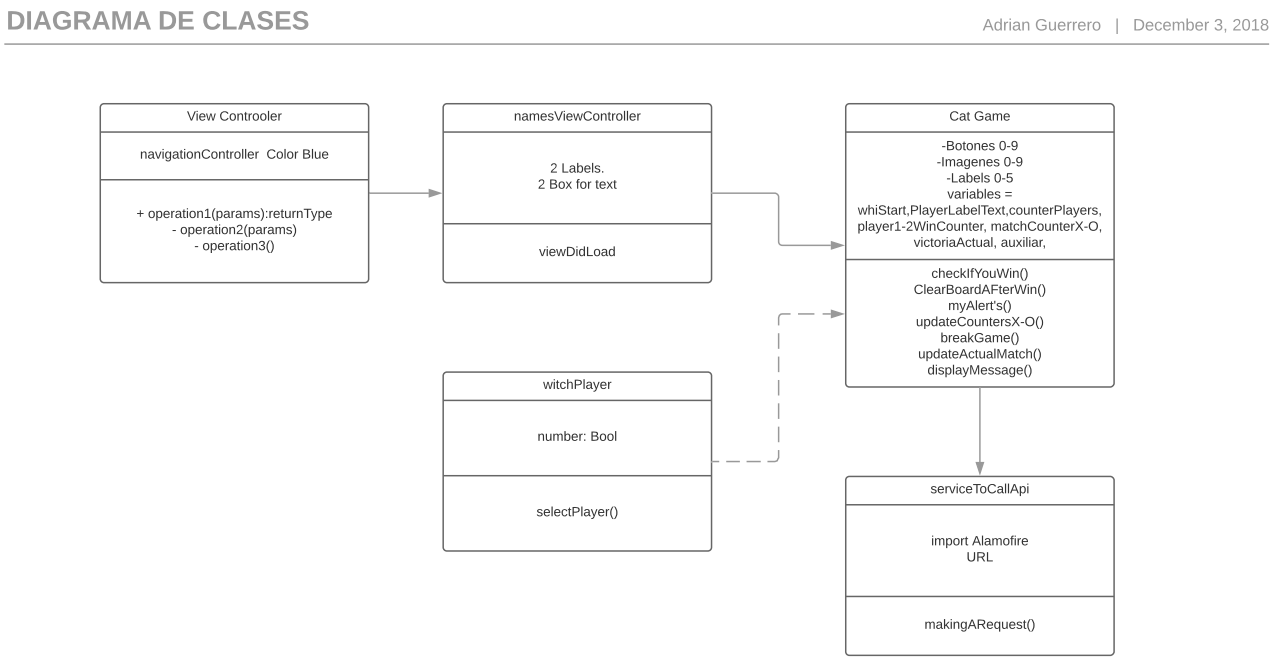
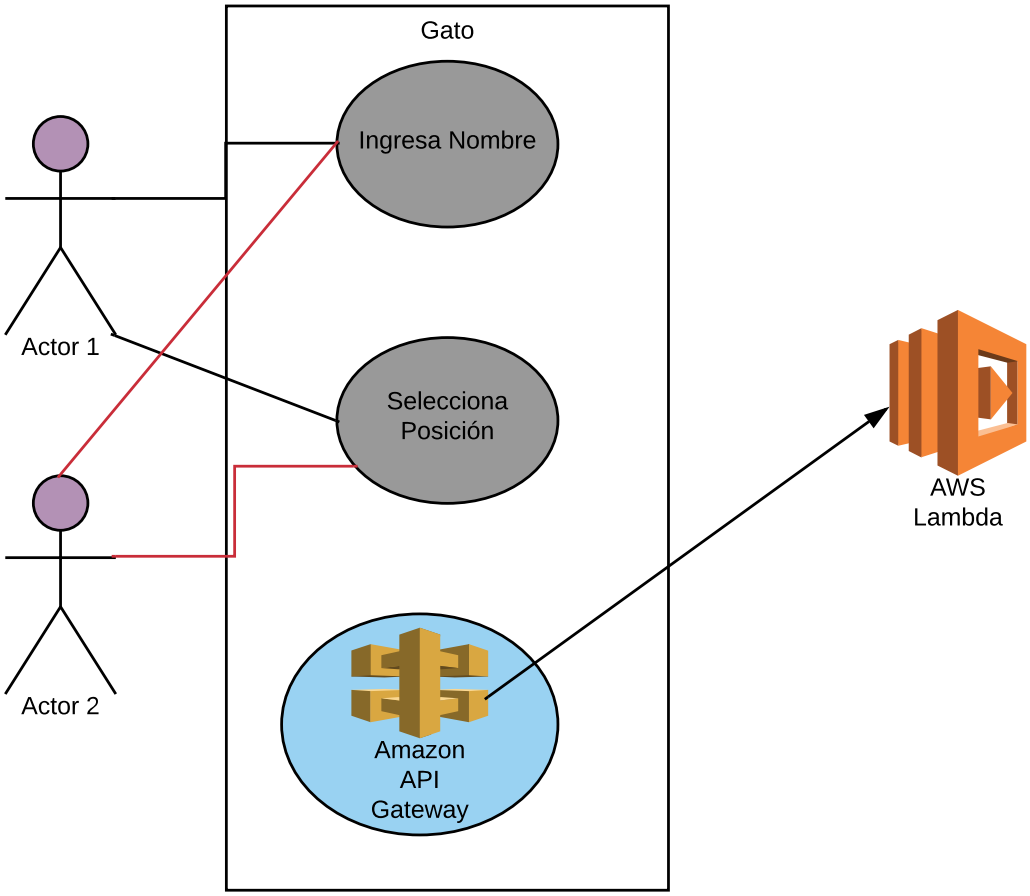


Diagrama de Clases

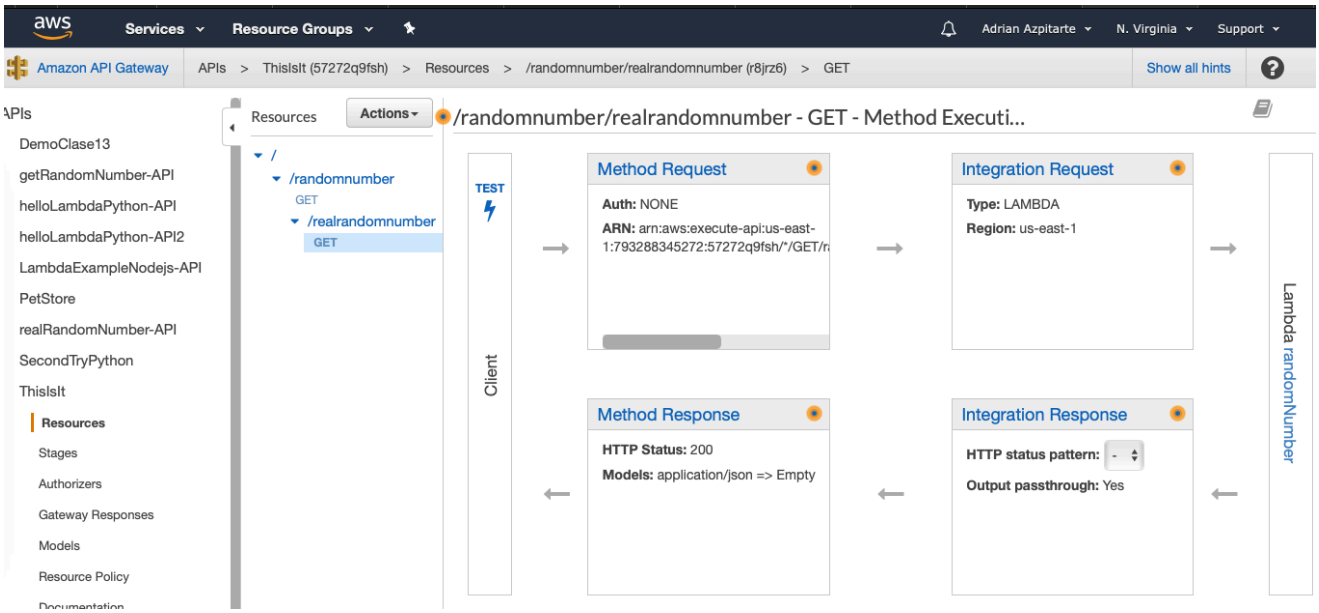


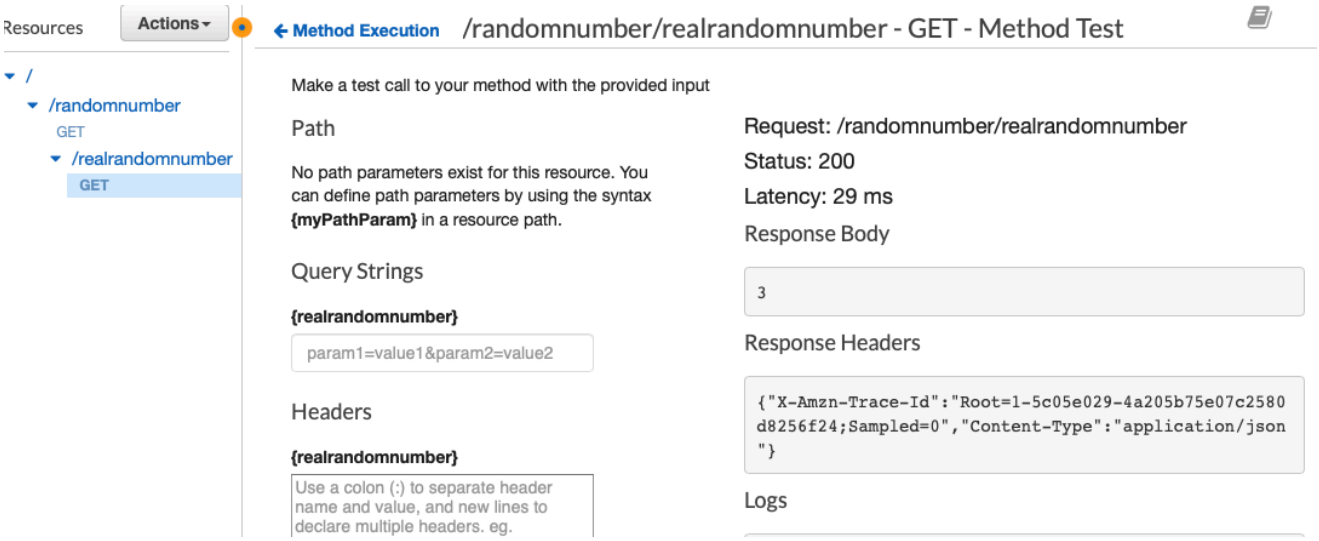
Casos de Uso


Gato
Ingresa Nombre
Actor 1
Selecciona
Posición
Actor 2
Amazon
API
Gateway
AWS
Lambda

API-Gateway

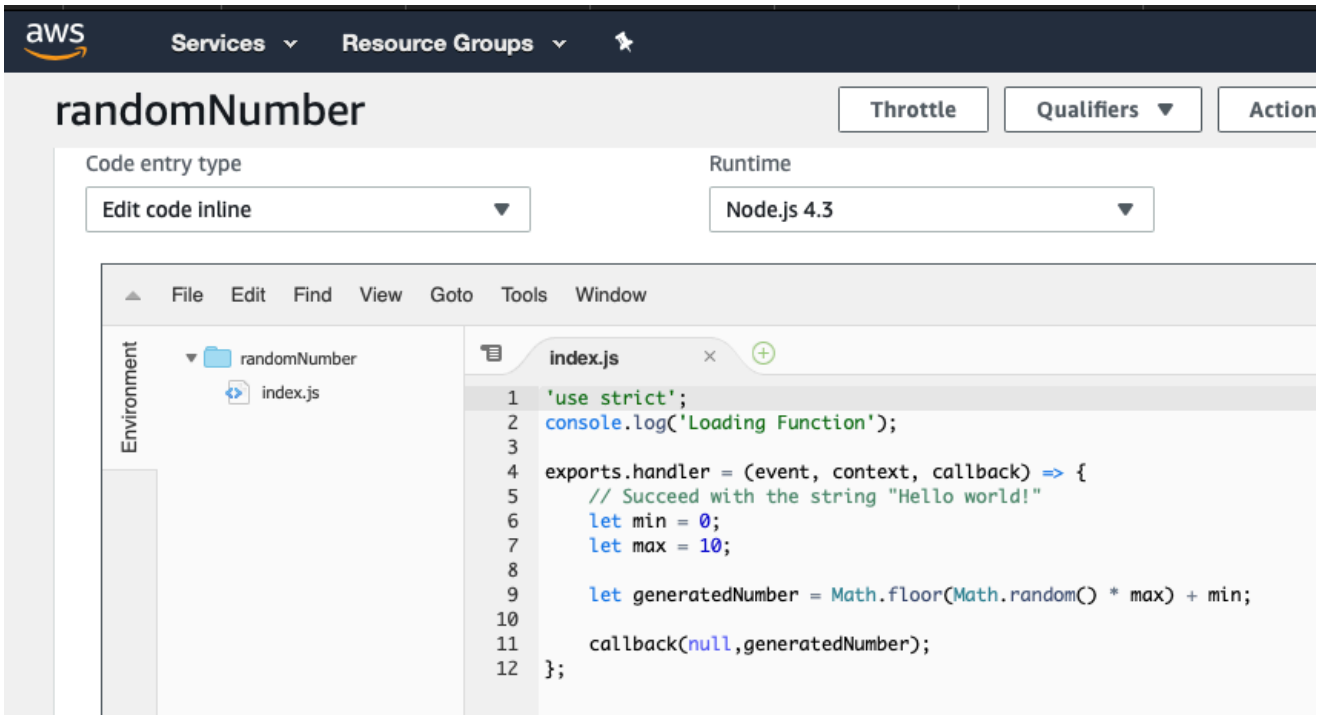
Después de muchos intentos se logro desplegar el API Gateway a la cual le logré enlazar mi función lambda.





Función Lambda

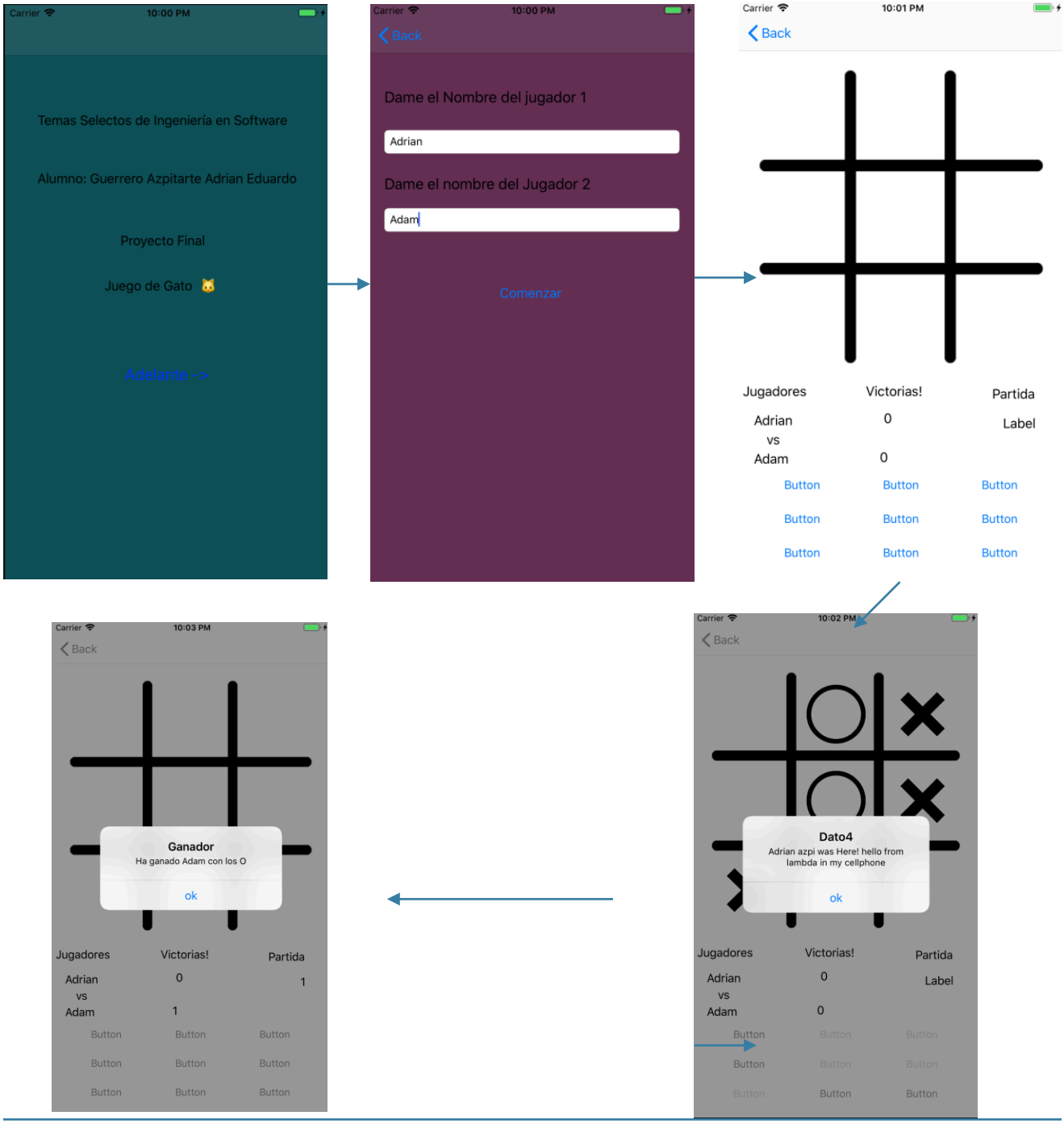
Mi función Lambda es bastante simple, pero aun así costo trabajo poder subirla y mas trabajo costo que mi api la reconociera y mandará los JSON correspondientes al valor random. Esta diseñada en Node.js4.3
Las 2 variables que posee es para hacer un numero random entre el mínimo valor y el máximo. Y con la ayuda de la función matemática, nos regresa el valor random entre esos limites.



La función Lambda así como el API Gateway. Fueron creados desde la consola de Amazon.

Definición de obeto JSON: Mi api gateway manda a llamar a Lambda y ésta regresa en forma de JSON solamente un número aleatorio entre el 0 y y el 10. Lo regresa en forma de String, y yo tomo ese valor en una variable. Cada botón al hacer su llamada. Regresa este objeto y yo lo comparo en una función para que arroje un mensaje determinado.

Mock up


Carrier
10:00 PM
Temas Selectos de Ingeniería en Software
Alumno: Guerrero Azpitarte Adrian Eduardo
Proyecto Final
Juego de Gato
Adelante ->
Back
Dame el Nombre del jugador 1
Adrian
Dame el nombre del Jugador 2
Adam
Comenzar
10:01 PM
Jugadores
Victorias!
Partida
Adrian
vs
Adam
0
0
Label
Button
10:02 PM
Dato4
Adrian azpi was Here! hello from lambda in my cellphone
ok
10:03 PM
Ganador
Ha ganado Adam con los O
ok
1

Referencias

https://prezi.com/toaespmdvhum/requerimientos-funcionales-y-no-funcionales-de-un-sistema-de/
https://aws.amazon.com/getting-started/projects/build-serverless-web-app-lambda-apigateway-s3-dynamodb-cognito/module-4/
https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-create-api.html
https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-create-model.html
https://es.wikipedia.org/wiki/Diagrama_de_clases
https://es.wikiversity.org/wiki/Plan_de_proyecto_software
https://www.youtube.com/watch?v=8U4RRw3PwGw&t=18s
https://www.youtube.com/watch?annotation_id=annotation_1811568499&feature=iv&src_vid=8U4RRw3PwGw&v=4ZyicyDpqE0
https://www.youtube.com/watch?v=UMgApUhg7ic&t=908s
https://www.youtube.com/watch?v=-qnz1GTQ7lg&t=179s
https://www.lucidchart.com