Icon

Description automatically generated with low confidence

**IOT Data Science Project**

**ASSIGNMENT 2**

|  |  |
| --- | --- |
| **Student Name** | **Student ID** |
| Ngui Jia Le Sherlena | S10227488A |

Contents

[Project Problem Statement 3](#_Toc127350681)

[Data Preparation 3](#_Toc127350682)

[Model Selection 5](#_Toc127350683)

[Model Training 6](#_Toc127350684)

[Hyper-parameter Tuning 9](#_Toc127350685)

[Model Deployment 9](#_Toc127350686)

[Performance Analysis 13](#_Toc127350687)

[Challenges Faced 13](#_Toc127350688)

[Conclusion 13](#_Toc127350689)

# Project Problem Statement

The main project problem statement was to help users which are unsure of how to book a trip for travelling to paranormal locations through a website. As many may either want to book multiple trips and or even are unsure how to book the trips, it will be slightly easier to make use of services for users which are not so well versed with the booking on the web page.

# Data Preparation

In order to prepare the data required by the site, a manual inspection of the website was done to scrape the information required by the site and to take note of the various data input formats required by the site. Some examples are the name of the user, the contact number, the date of tour and the tour that the user would like to embark on.

This was to ensure that when the chatbot sends the questions to the users and the users reply, the chatbot is able to retrieve the information and format it in the correct structure before placing the data into the database and within an organization use the sending of the notification for approval using the SNS service in AWS.

Graphical user interface, text

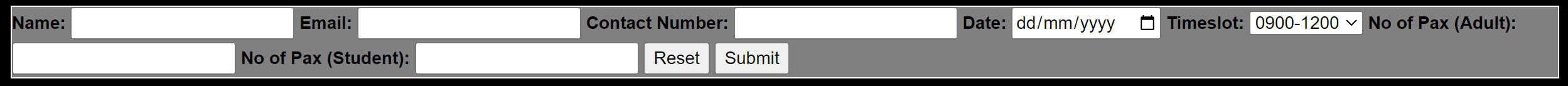
Description automatically generated

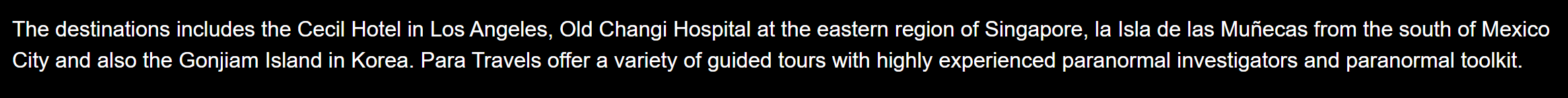
Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated





# Model Selection

The following services was used during the project, Amazon Lex, AWS Lambda and, Amazon S3. These services are all platform as a service as such the configuration of the base system is all handled by AWS and services configuration is handled by the customer.

The Amazon Lex service provides a deep functionality which is flexible in understanding natural language. It is able to quickly build up a conversational chatbot quickly. The Amazon Lex is also able to manage dialogues between the various users as well as to dynamically adjust the response in the conversation based on the user’s interaction.

The AWS Lambda service is a service which is able to support various programming language such as Python 3.9 and even Node.js. The Lambda service is also able to be used between the various services to configure a trigger as well as an output from the service. The lambda function is also able to support multiple runs concurrently and is only paid within the time the service is running. The Lambda service has a maximum run time of 15 minutes as such the python code configured will be required to run completely within 15 minutes.

The Amazon Simple Storage Service also known as Amazon S3, is an object storage service and is able to be configured to host a static website. This static website using the Amazon S3 features is backed up regularly and is able to be integrated with services like the AWS Lambda function.

The Amazon Cognito is a service which offers user pools as well as identity pools. This will allow the provision of AWS credentials and grant user access for AWS services such as Amazon Lex. The creation of roles which would allow the users to provision the rules required to a specific user using the Identity and Access Management (IAM) console. s

# Model Training

To create the model, the template to book a trip was used. The below settings were configured as follows.

Graphical user interface

Description automatically generated

Next, to conduct the model training, the following was the list of sample utterances that was configured to the system.

Graphical user interface, text, application, email

Description automatically generated

These sample utterances were configured to trigger when the user typed the following into the chat bot prompt.

There were various slots that were configured to prompt the user the various questions in order to record the information required by the site in order to process a booking. The configuration of the options was as follow.

Graphical user interface

Description automatically generated with medium confidence

The BookingLocation intent as well as the Response intent was configured consisting of the various valid responses expected by the user.

Graphical user interface, text, application, email

Description automatically generated

# Hyper-parameter Tuning

To ensure that the data is more useful compared to just the basic information requested by the bot, a hyper-parameter tuning was implemented to add additional fields like the contact name as well as the contact number of the user booking. The formatting of how the data should be placed for the date was added. This was added as during the process of testing, various date formats were tried such as when the user is to input the day followed by month and year, the system throws back an error message to specify that the details of date was incorrect and to input an appropriate date.

Graphical user interface, application

Description automatically generated

# Model Deployment

The model was deployed with the following configurations.

Firstly, the bot was build and published with the Alias of the following. This alias was then added into the code for the Lambda function.

Graphical user interface, application

Description automatically generated

Secondly, Cognito was used to manage identity pools and added. The identity pool enabled the unauthenticated role for users to access the service without having to be authenticated.

Graphical user interface, text, application, email

Description automatically generated

Thirdly, the roles were modified using the identity and access management to add the reading and execution permission for Amazon Lex for the roles.

Next, an AWS Lambda function was created using the default template of Book a trip with Lex. This Lambda function code was than copied out and modified to meet the needs of the chatbot being configured for the booking of tours at the paranormal website. The code for the Lambda function is attached as BookTourCodeHook.py.



After which, the Amazon Lex was connected to the AWS Lambda that was configured to check for the initialization and validation as well as the fulfillment of the process.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

A website is than created and uploaded to the S3 bucket created. This S3 bucket had the static hosting enabled along with the index page and custom error page being pointed at the various files. The site was also added to the website to ensure that the users are able to access the page. The files can be found from this link: <https://github.com/C4RR0T02/IOT-DSP-Chatbot>.

Graphical user interface, text, application, email

Description automatically generated

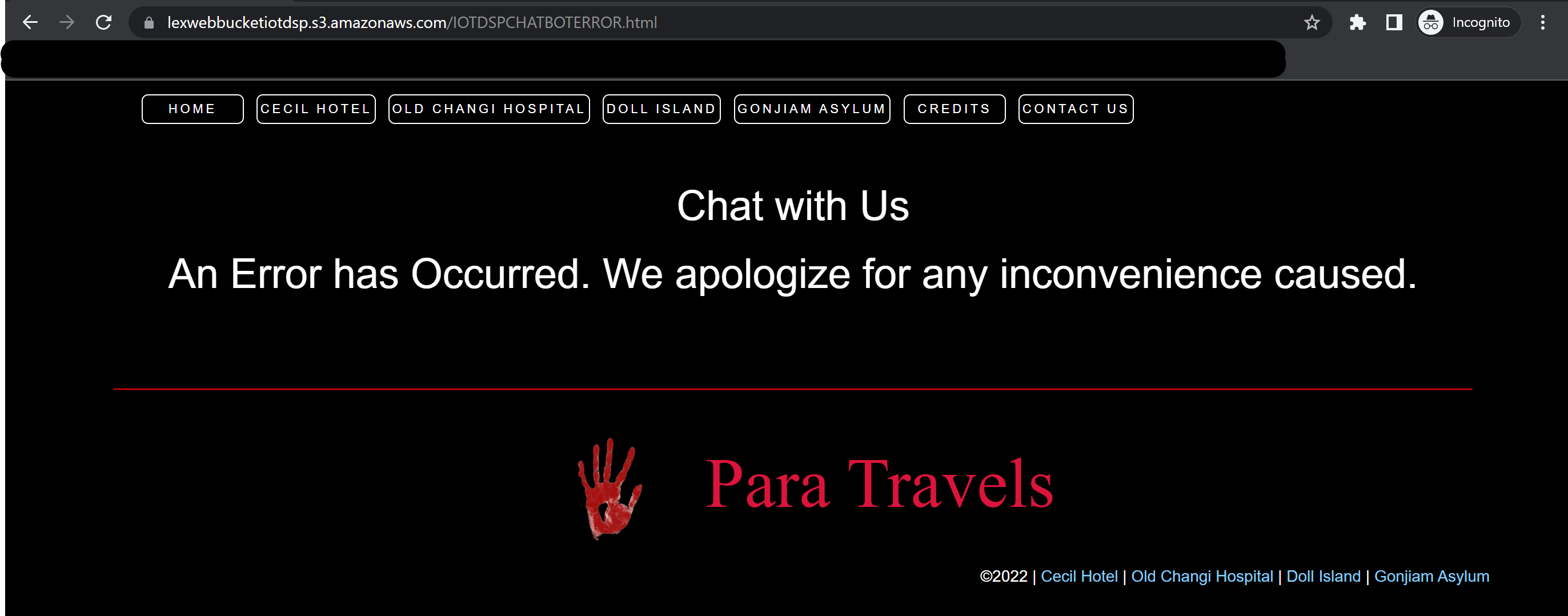
Graphical user interface, text, application, email

Description automatically generated

With the static hosting website, the link to the chatbot page was attempted to be accessed and ran. On running of the chatbot, it successfully returned the message configured.

Graphical user interface, text, application, chat or text message

Description automatically generated



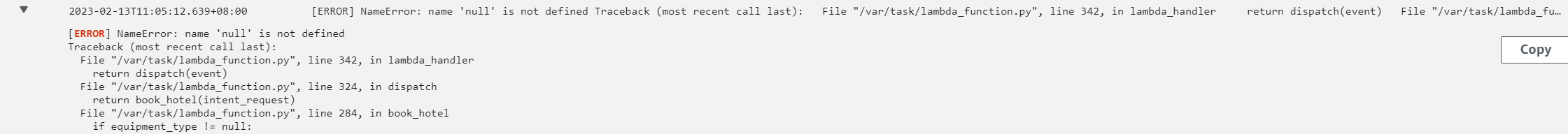
This showed that the sites configured are working as intended.

# Performance Analysis

Overall, the bot was able to perform the task of allowing users to be able to book a tour and prompt questions appropriately as per intended. However, there are still some hiccups from the bot for example the bot does not return an error to ask prompt the users to retype the answer for any values unintended. This could likely be due to the Lambda code not having the function called properly to validate the prompt properly.

# Challenges Faced

Along the way of the project, there were many challenges faced such as the AWS account limitations. One instance is using the AWS Academy Learner Lab [28535] due to the limitation of the permissions for the account, several roles were already created with the necessary permissions, however during the implementation, I was unable to make use of the already created LabRole for the implementation of the various permissions to deploy and host the web server as such I switched to an AWS account which I had the permission to create roles as per needed. This will also ensure that the implementation will also not be destroyed after the set time frame when choosing to use the lab exercise environment. Another challenge faced was the deploying of the Lambda code. As I was not very familiar with the way the Amazon Lex Lambda code is created, a lot of studying and troubleshooting within the running of the Lambda was done and monitored through the CloudWatch to ensure that the information was being sent over.



# Conclusion

Through this project, a bot was implemented with the purpose of creating a bot to assist users with the booking. This function was achieved as the bot was able to book a tour for the users. Through the usage of AWS Lambda, S3, Cognito and Amazon Lex. Even though the input validation is not completely implemented, this information retrieved can still be stored in a NoSQL database like DynamoDB.