CSCI 5410
Serverless Data Processing

Assignment 2
Part C
Building a Chatbot

Ritva Katrodiya
B00930131

Table of Contents

Table of figures 3

Building chatbot on Amazon lex 4

Summary about the operations performed 28

References 29

Table of figures

Figure 1 Amazon lex service dashboard with no chatbots [1] 4
Figure 2 Creation of chatbot named RideRequest [1] 5
Figure 3 Creation of chatbot named RideRequest [1] 5
Figure 4 Creation of chatbot named RideRequest [1] 6
Figure 5 Creation of chatbot named RideRequest [1] 6
Figure 6 RideRequest chatbot successfully created on amazon lex dashboard with status available [1] 7
Figure 7 RideRequest chatbot dashboard with description [1] 7
Figure 8 RideRequest chatbot dashboard with description [1] 8
Figure 9 RideRequest chatbot dashboard with description [1] 8
Figure 10 RequestRideIntent for Taxi and car rental service [1] 9
Figure 11 Created two custom slots named RideType and VehicleType [1] 10
Figure 12 RideType slot type for Taxi or Self-Drive option [1] 11
Figure 13 RideType slot type for Taxi or Self-Drive option [1] 12
Figure 14 RideType slot type for SUV, Sedan, or minivan options [1] 12
Figure 15 RideType slot type for SUV, Sedan, or minivan options [1] 13
Figure 16 RideRequest Conversation flow [1] 13
Figure 17 RideRequest Intent details [1] 14
Figure 18 RideRequest utterances are used by user to request for taxi or self drive [1] 14
Figure 19 Set the Initial response for start the conversation [1] 15
Figure 20 RideRequest slots configured to accept Vehicle Type, Quantity, Ride Type, and Arrival Time [1] 15
Figure 21 RideRequest slots configured to accept address, pick up date and time [1] 16
Figure 22 RideRequest Confirmation prompt [1] 16
Figure 23 RideRequest Fulfillment Response [1] 17
Figure 24 RideRequest chatbot built successfully [1] 17
Figure 25 RideRequest chatbot testing conversation in working [1] 18
Figure 26 RideRequest chatbot testing conversation in working [1] 19
Figure 27 RideRequest chatbot testing conversation in working [1] 19
Figure 28 RideRequest chatbot testing conversation in working [1] 20
Figure 29 RideRequest chatbot testing conversation in working [1] 20
Figure 30 RideRequest chatbot testing conversation in working [1] 21
Figure 31 RideRequest chatbot testing conversation in working [1] 21
Figure 32 RideRequest chatbot testing conversation in working [1] 22
Figure 33 RideRequest chatbot testing conversation in working [1] 23
Figure 34 RideRequest chatbot testing conversation in working [1] 24
Figure 35 RideRequest chatbot testing conversation in working [1] 24
Figure 36 RideRequest chatbot testing conversation in working [1] 25
Figure 37 RideRequest chatbot testing conversation in working [1] 25
Figure 38 RideRequest chatbot testing conversation in working [1] 26
Figure 39 RideRequest chatbot testing conversation in working [1] 26
Figure 40 RideRequest chatbot testing conversation in working [1] 27

Building chatbot on Amazon lex

Amazon Lex [1] is an Amazon Web Services (AWS) service that allows developers to incorporate conversational interfaces, or chatbots, into their applications. Amazon Lex enables developers to build interactive chatbots that can be integrated with a variety of messaging platforms, mobile apps, and voice-enabled devices [1]. It provides tools and resources for designing conversation flows, defining intents and slot types, and managing the chatbot's dialogue logic [1].

To understand the amazon lex, this assignment focuses on building a chatbot the "Taxi and Car rental service," operates by providing options for taxi service or self-drive rental. For taxi service, the chatbot collects customer details such as address, pickup date, and time. Alternatively, for self-drive rental, assuming the same day, the chatbot inquires about the customer's arrival time. These inputs allow the chatbot to gather the necessary information to facilitate the booking process efficiently.

Figures 1,2,3,4 and 5 display all the steps required to create chatbot named **RideRequest** on amazon lex. Initially, there is not any chatbot created in lex dashboard.

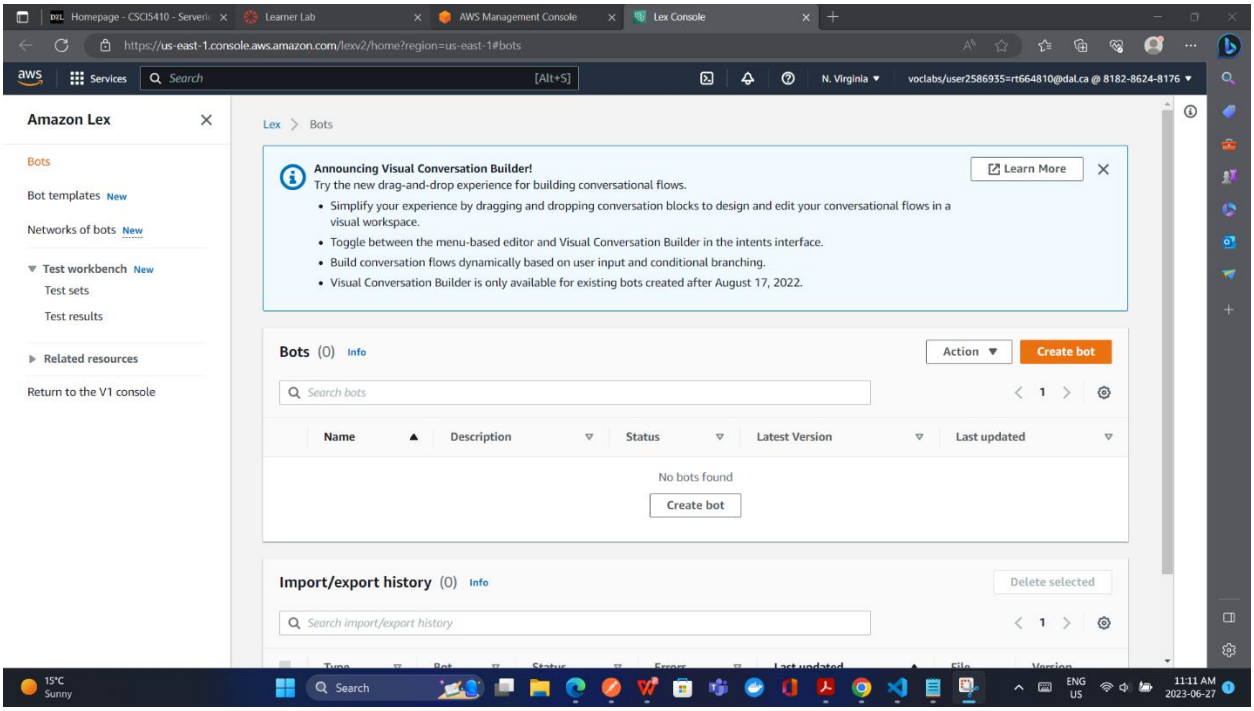

Figure 1 Amazon lex service dashboard with no chatbots [1]

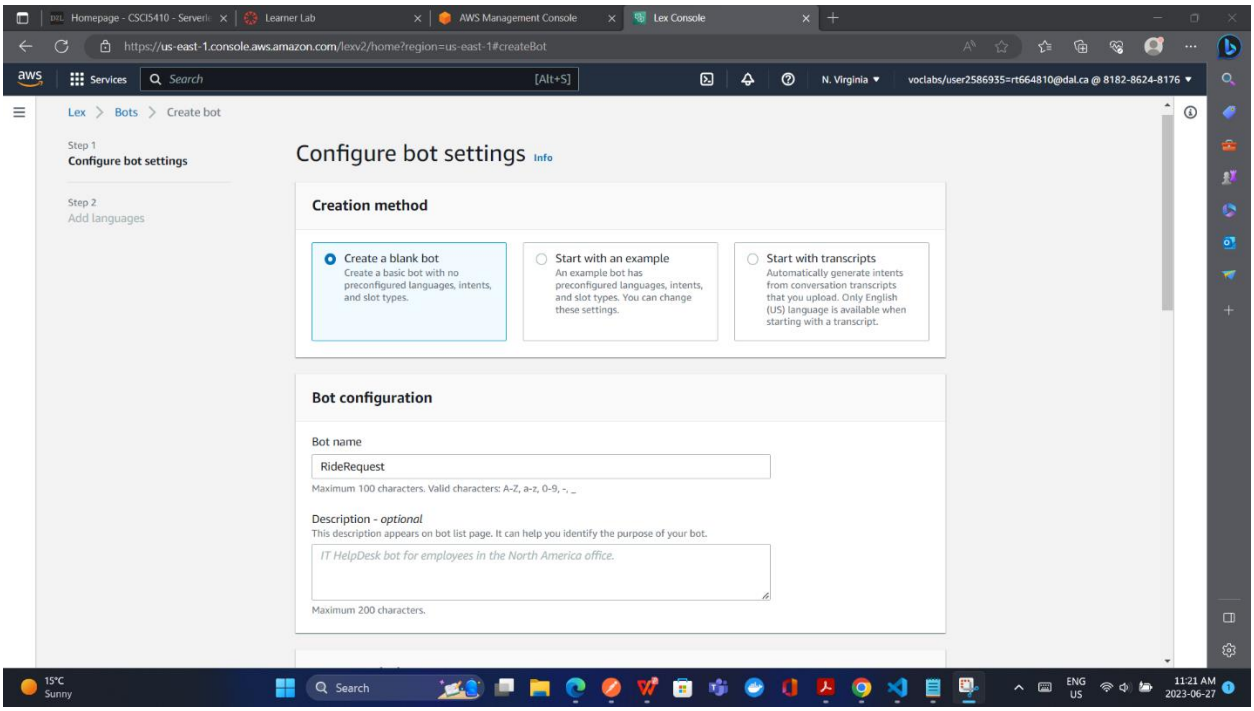

Figure 2 Creation of chatbot named RideRequest [1]

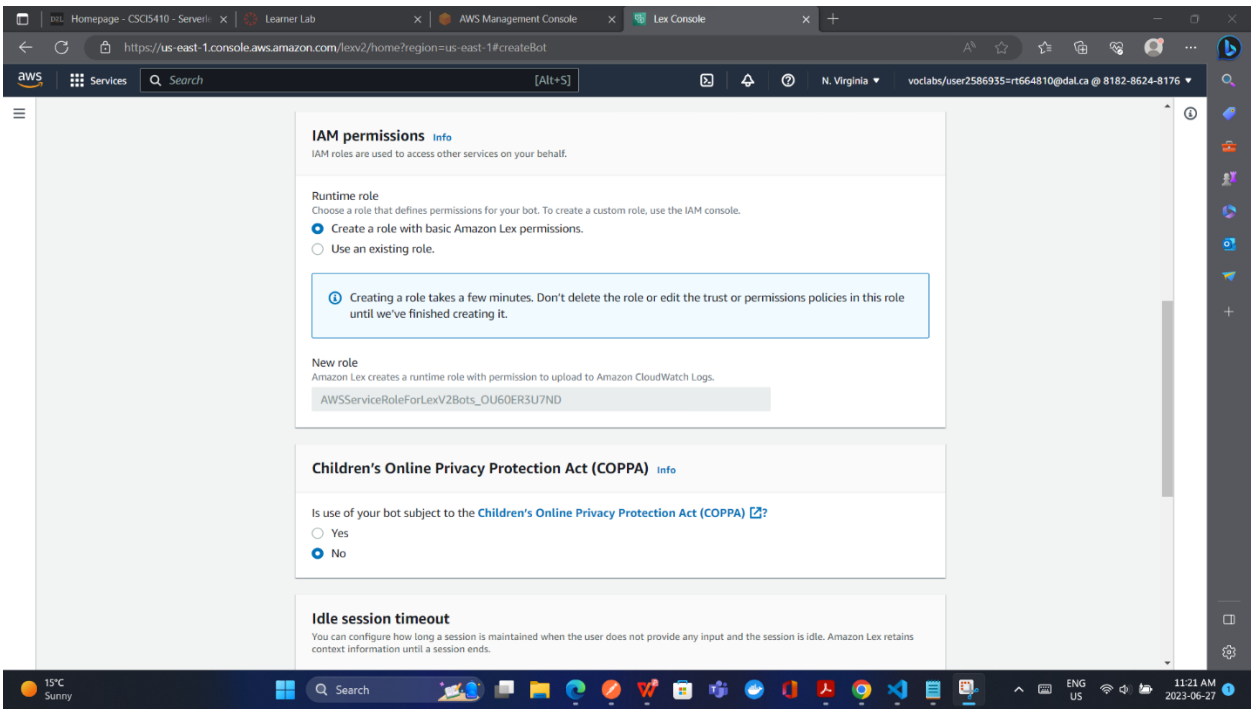

Figure 3 Creation of chatbot named RideRequest [1]

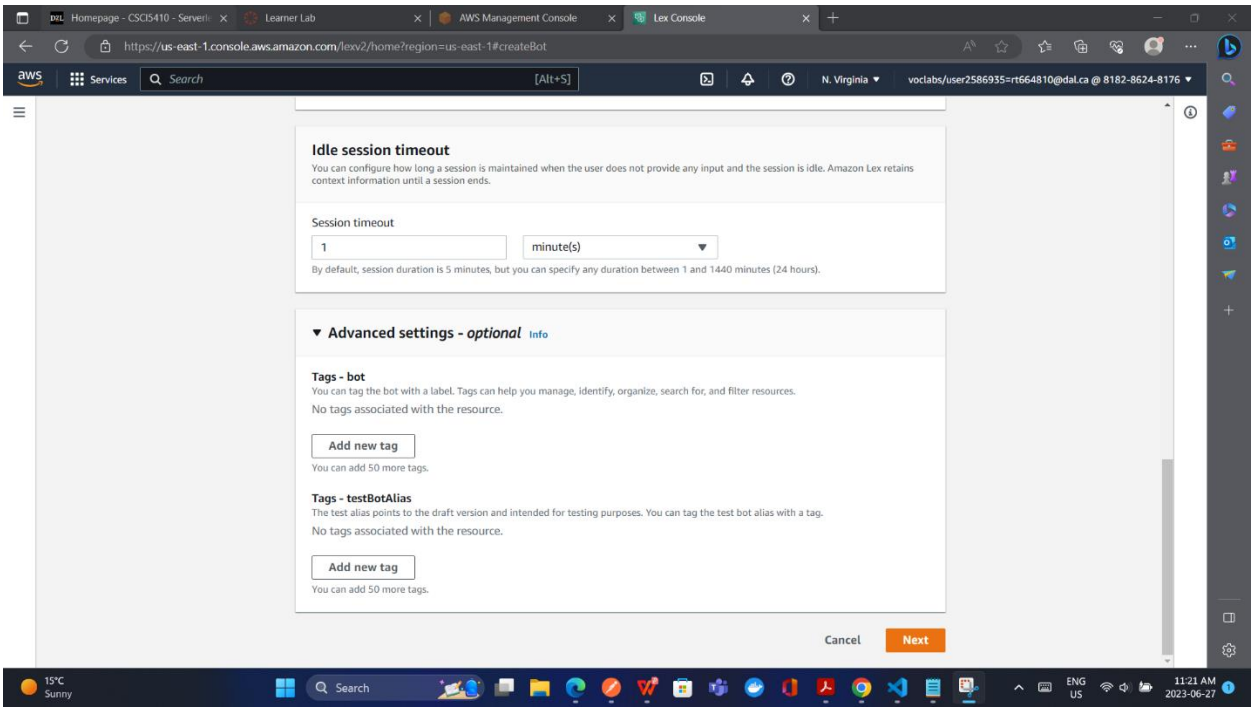

Figure 4 Creation of chatbot named RideRequest [1]

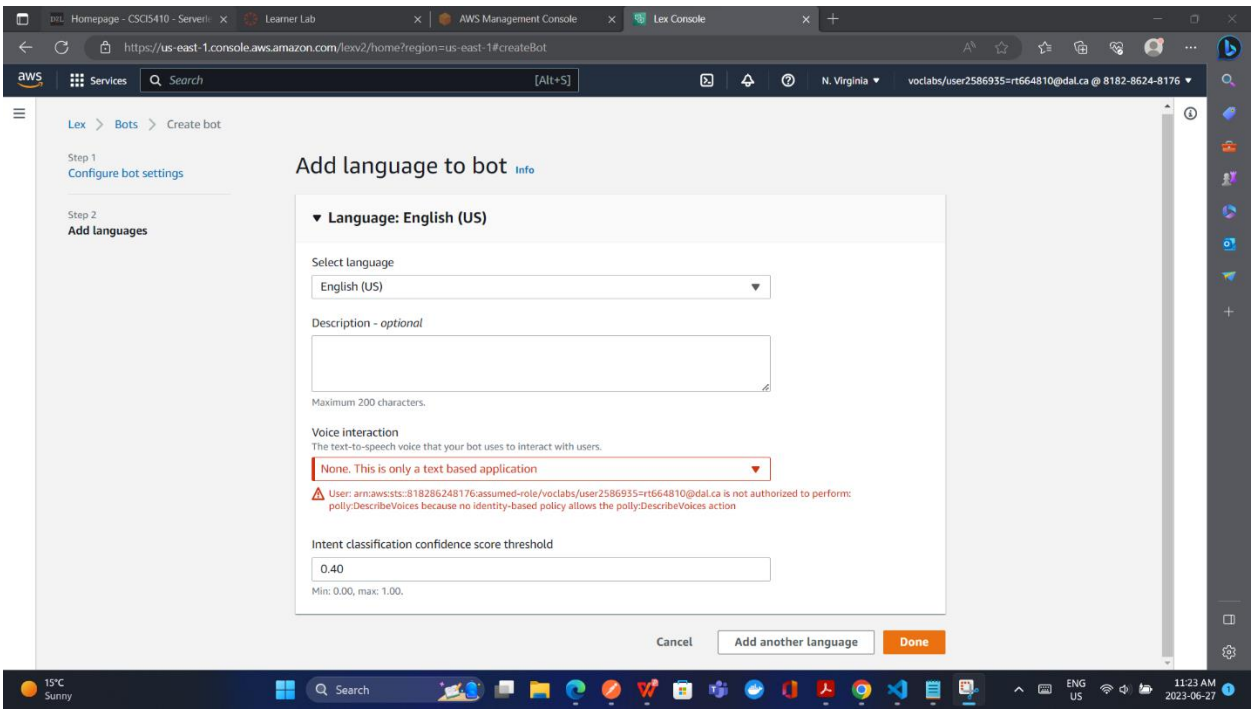

Figure 5 Creation of chatbot named RideRequest [1]

Figure 6 defines that chatbot named **RideRequest** sucessfully created on amazon lex dashboard.

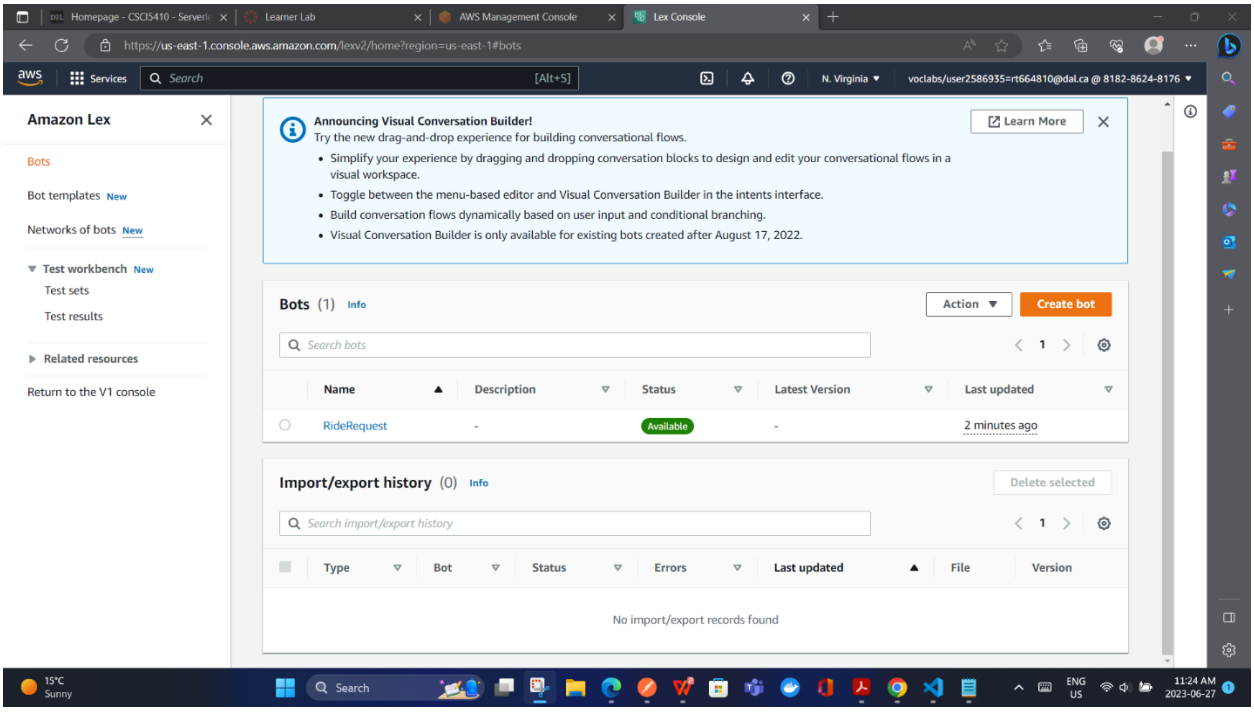

Figure 6 RideRequest chatbot successfully created on amazon lex dashboard with status available [1]

Figures 7, 8 and 9 defines the description of created chatbot **RideRequest**.

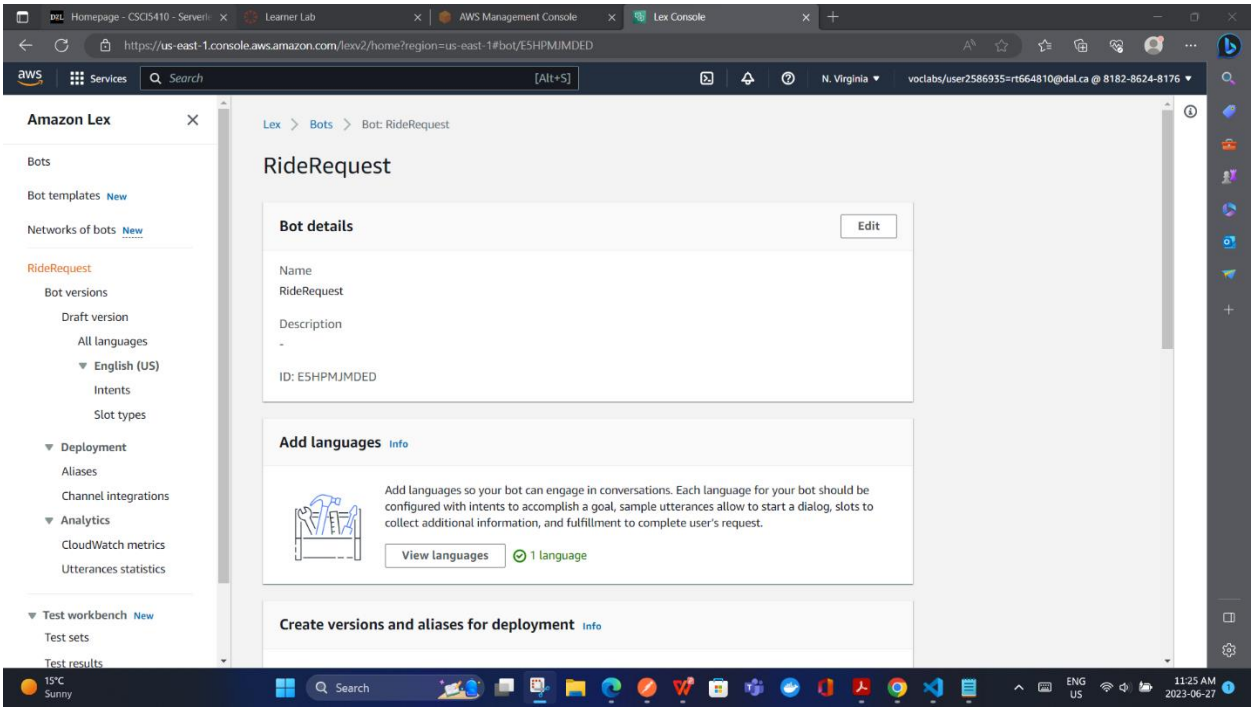

Figure 7 RideRequest chatbot dashboard with description [1]

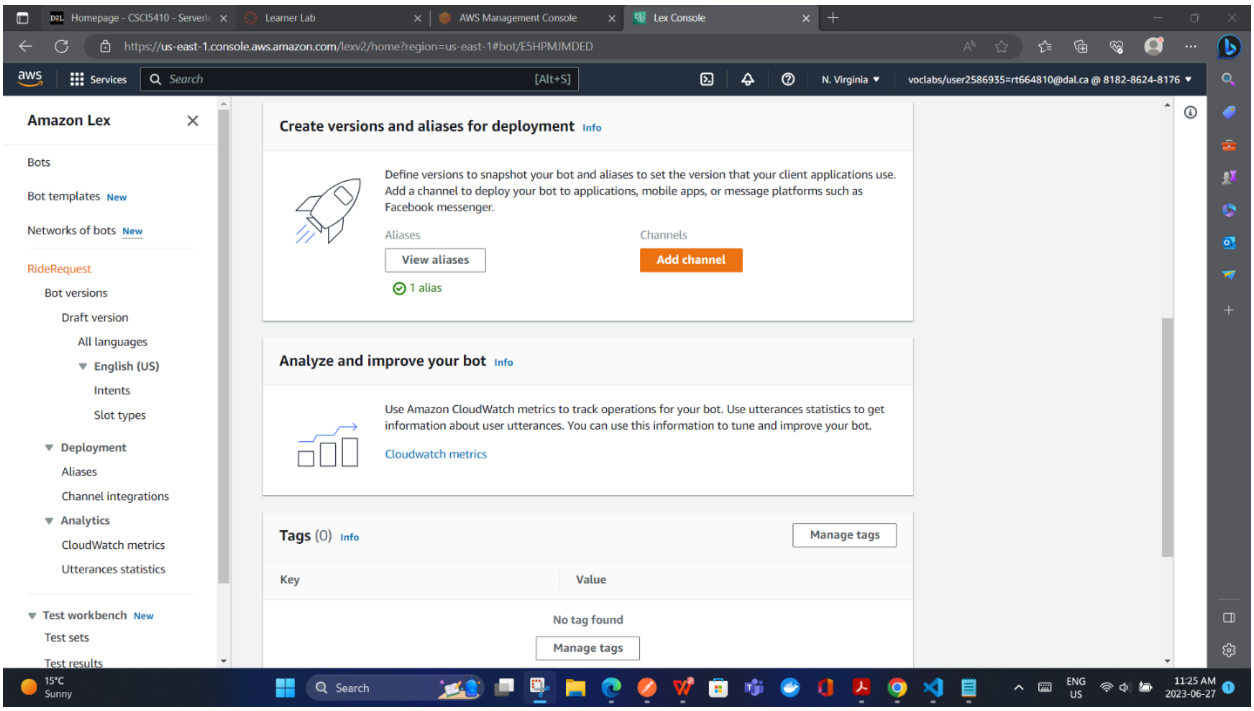

Figure 8 RideRequest chatbot dashboard with description [1]

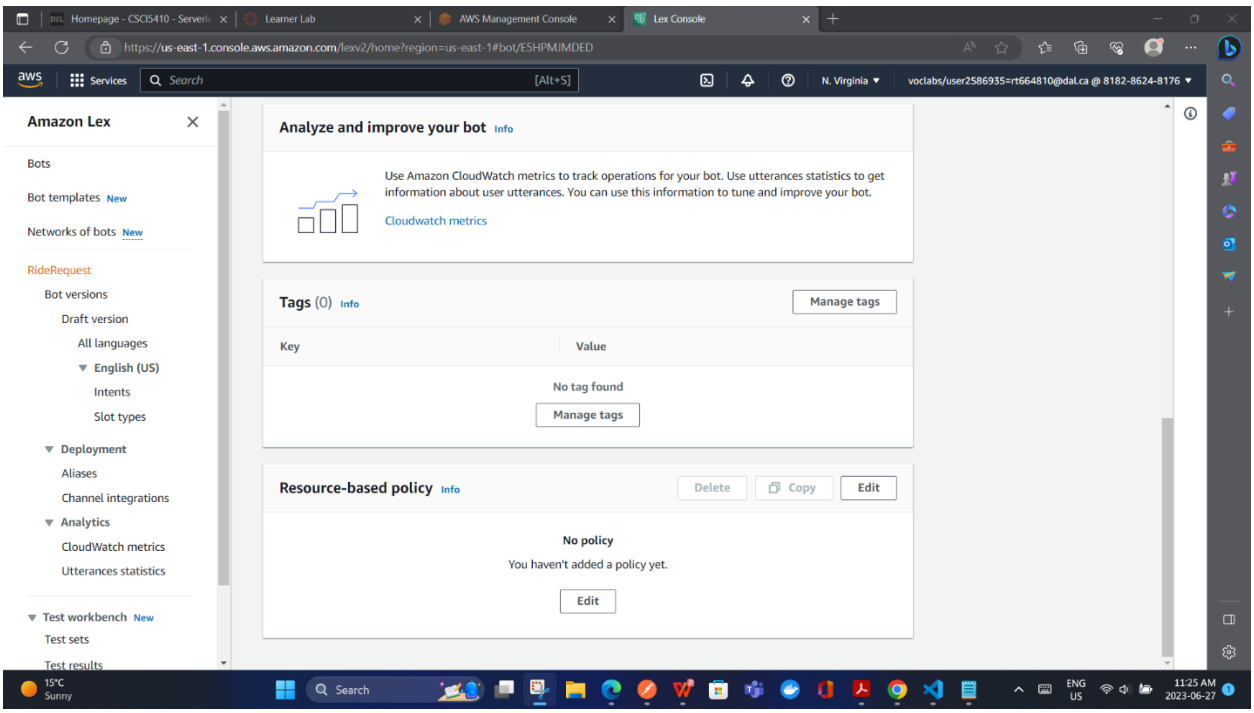

Figure 9 RideRequest chatbot dashboard with description [1]

Figure 10 shows intent RequestRideIntent for taxi and car rental services. This intent is configured as it is responsible for conversation related address, time, pick up date, taxi or self-drive option and vehicle options. In addition, fallbackintent intent is responsible to handle any false requests from user side.

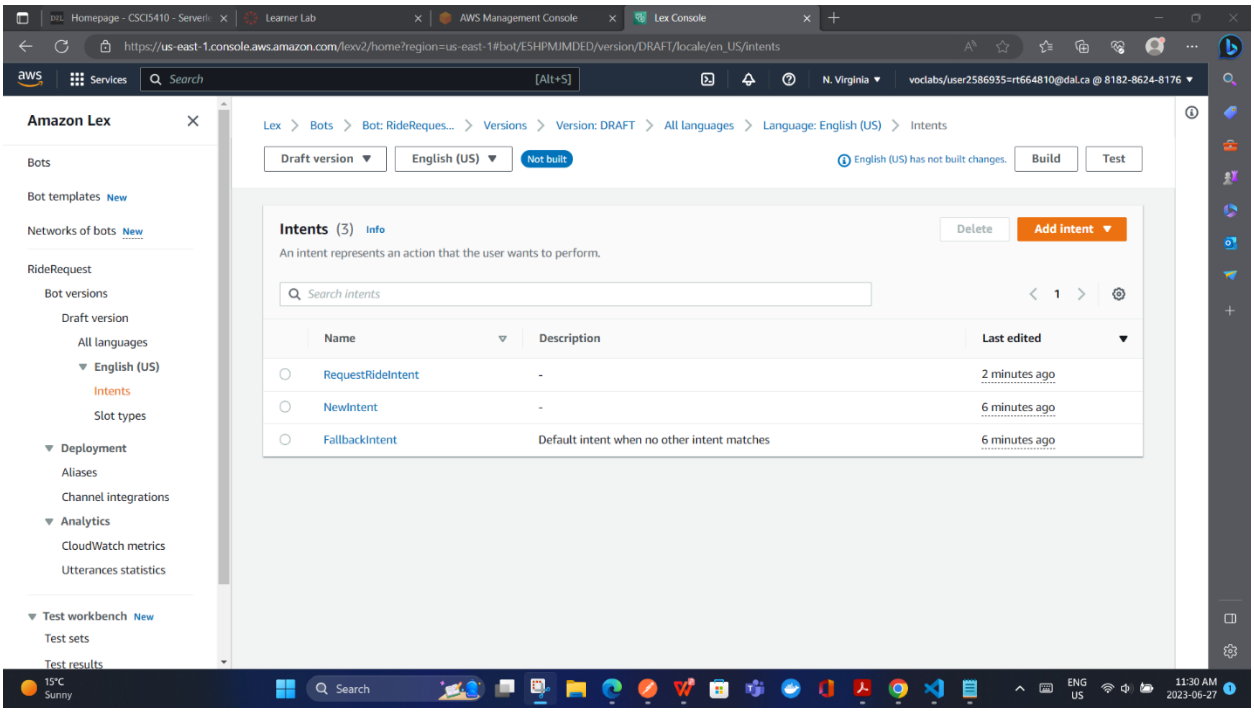

Figure 10 RequestRideIntent for Taxi and car rental service [1]

Figure11 displays the creation of two custom slots type named RideType and VehicleType.which are configured with RequestRideIntent for a conversation with user.

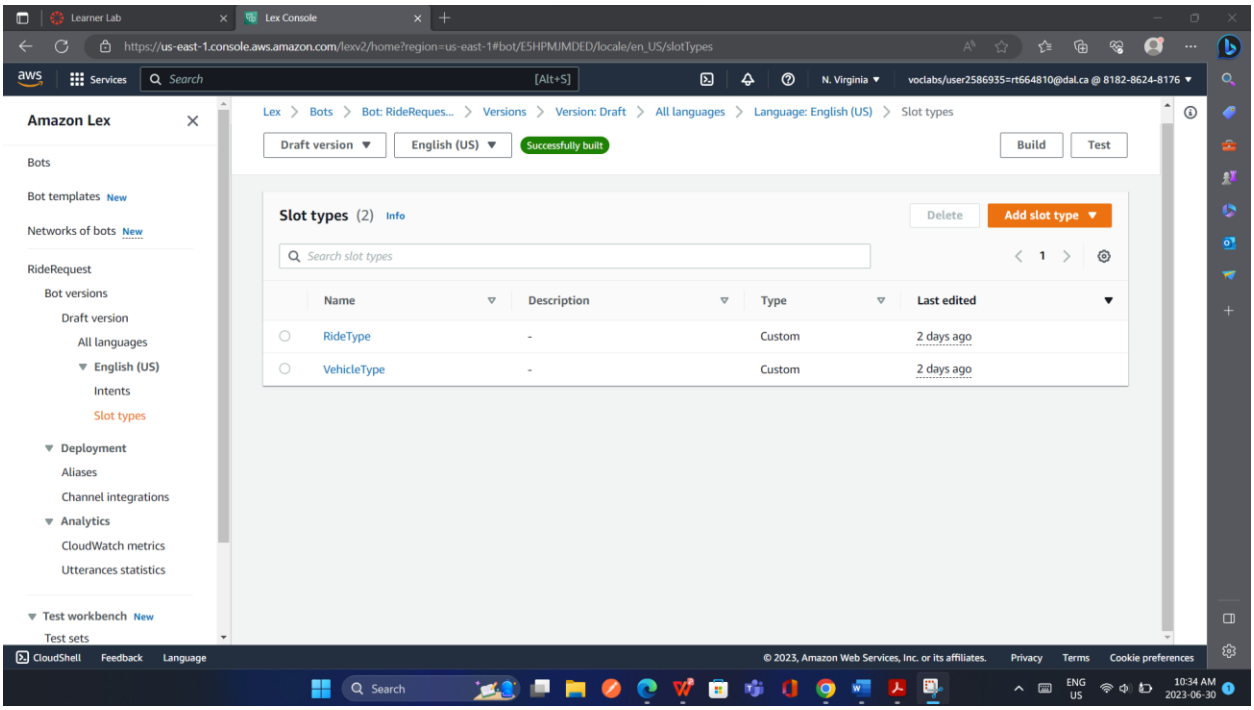

Figure 11 Created two custom slots named RideType and VehicleType [1]

Figure 12 and 13 depict the RideType slot type, which offers users the choice between selecting a taxi or a self-drive option. Depending on the user's reply, the conversation flow will differ. If the customer selects "taxi," they will be prompted to provide additional details such as their address, pick-up time, and date. On the other hand, if the user selects "self drive," they will be asked to specify the time they plan to arrive to collect the vehicle. To handle both scenarios, a custom slot type called RideType has been created.

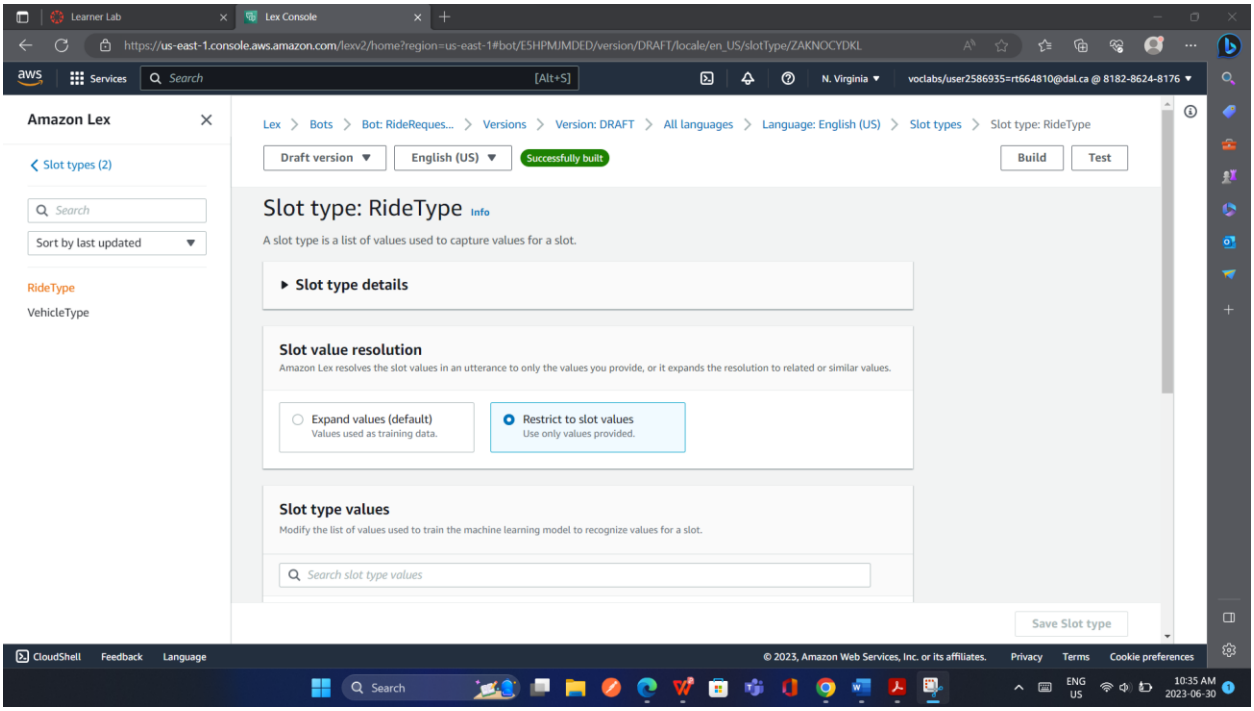

Figure 12 RideType slot type for Taxi or Self-Drive option [1]

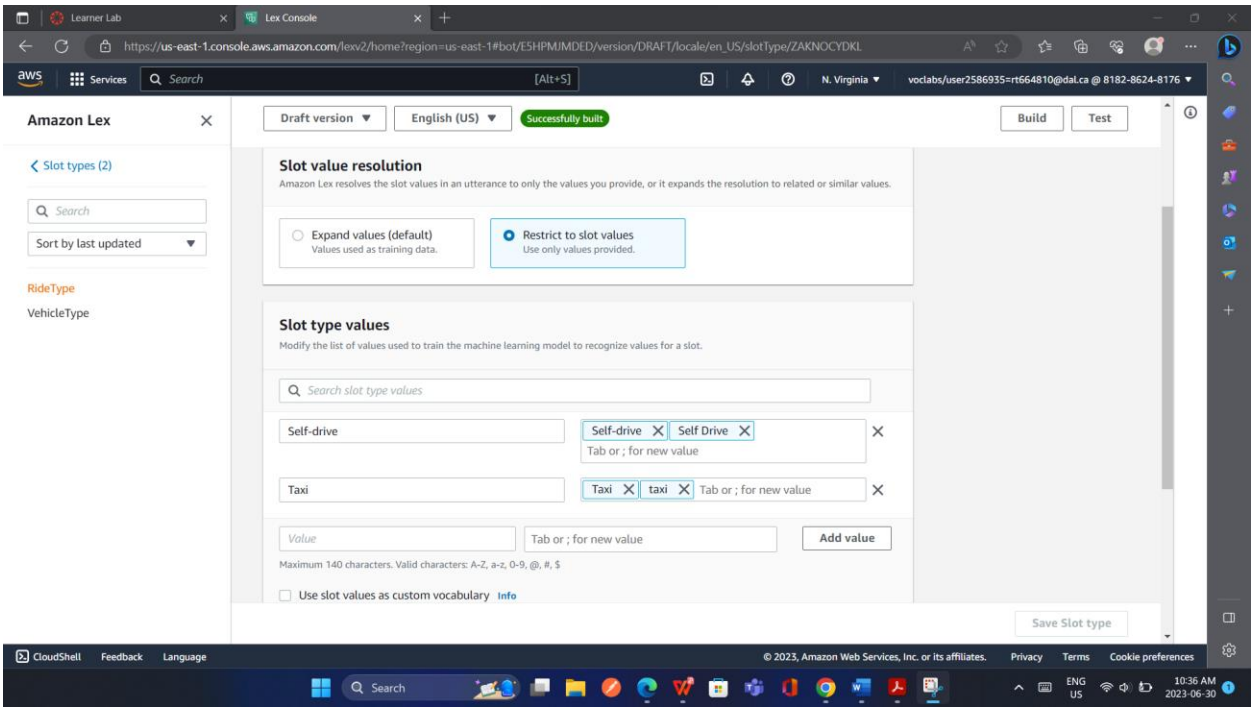

Figure 13 RideType slot type for Taxi or Self-Drive option [1]

Figures 14 and 15 depict the VehicleType slot type, which offers users the choices for selecting a suv or a minivan or a Sedan.

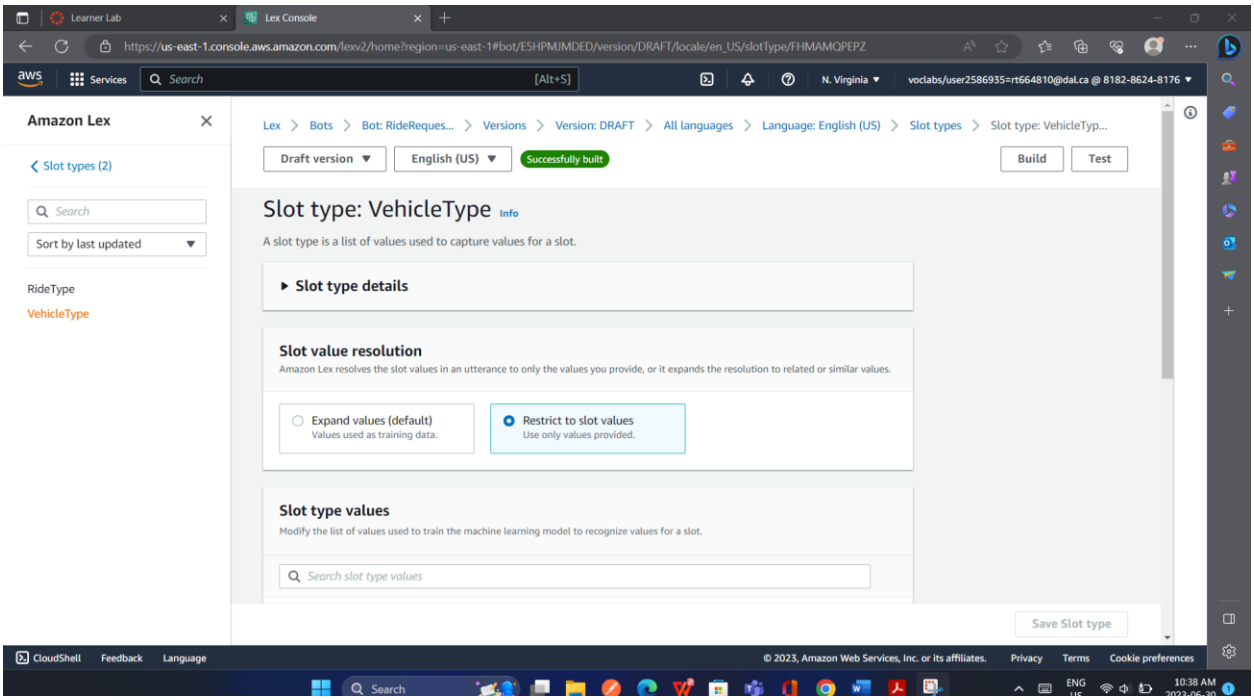

Figure 14 RideType slot type for SUV, Sedan, or minivan options [1]

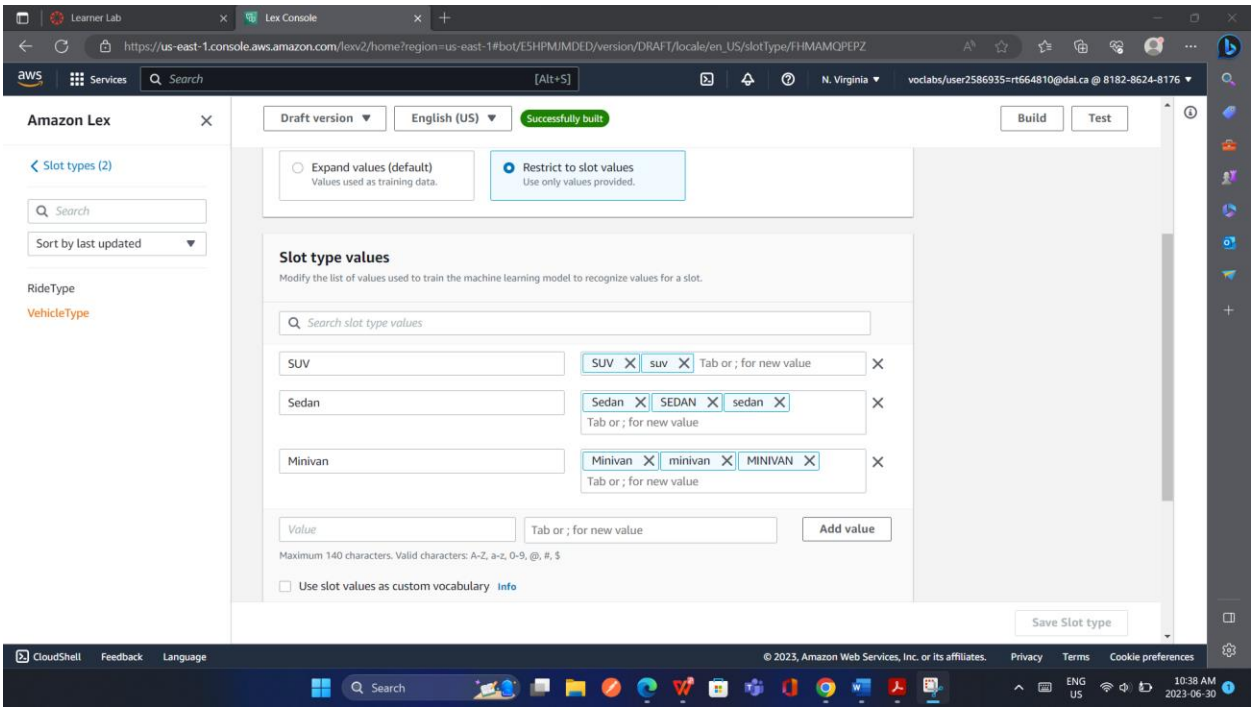

Figure 15 RideType slot type for SUV, Sedan, or minivan options [1]

Figures 16,17,18,19,20,21,22 and 23 shows the configuration set of **RequestRideIntent** which consists of Conversation flow, Intent Details, utterances are used by user to request for taxi or self drive, Initial Response, Required slots type, Confirmation prompt and fulfillment response.

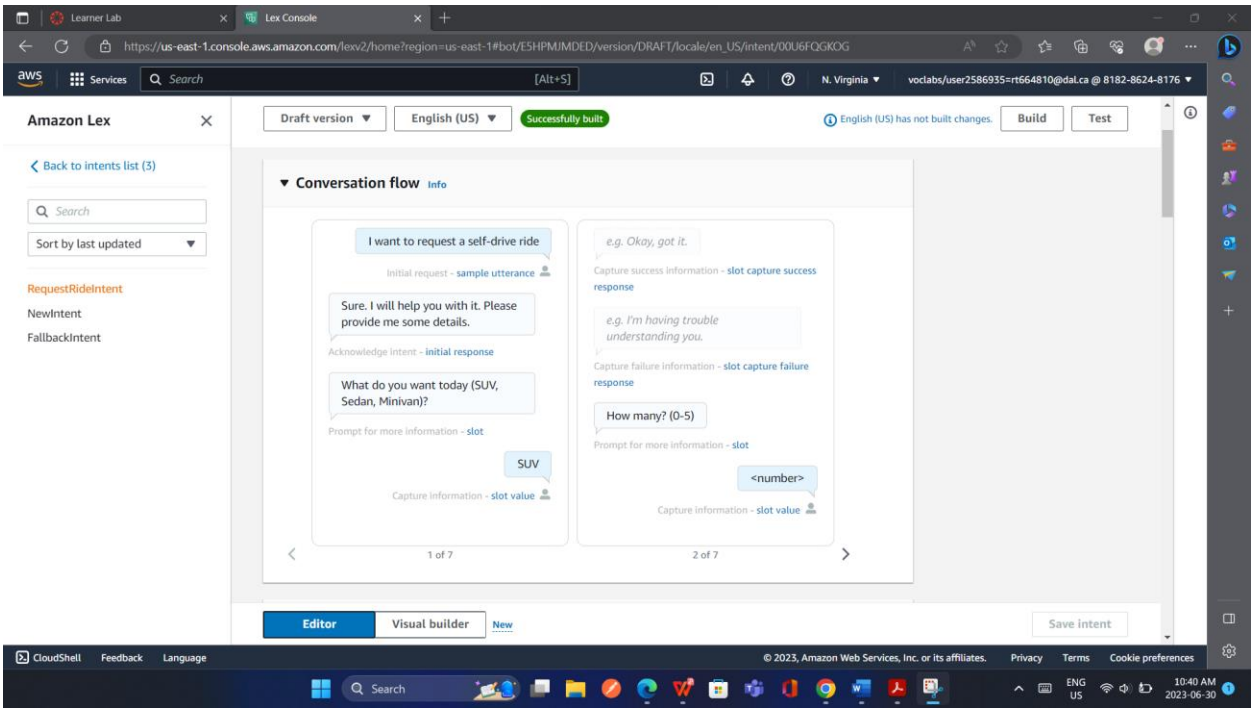

Figure 16 RideRequest Conversation flow [1]

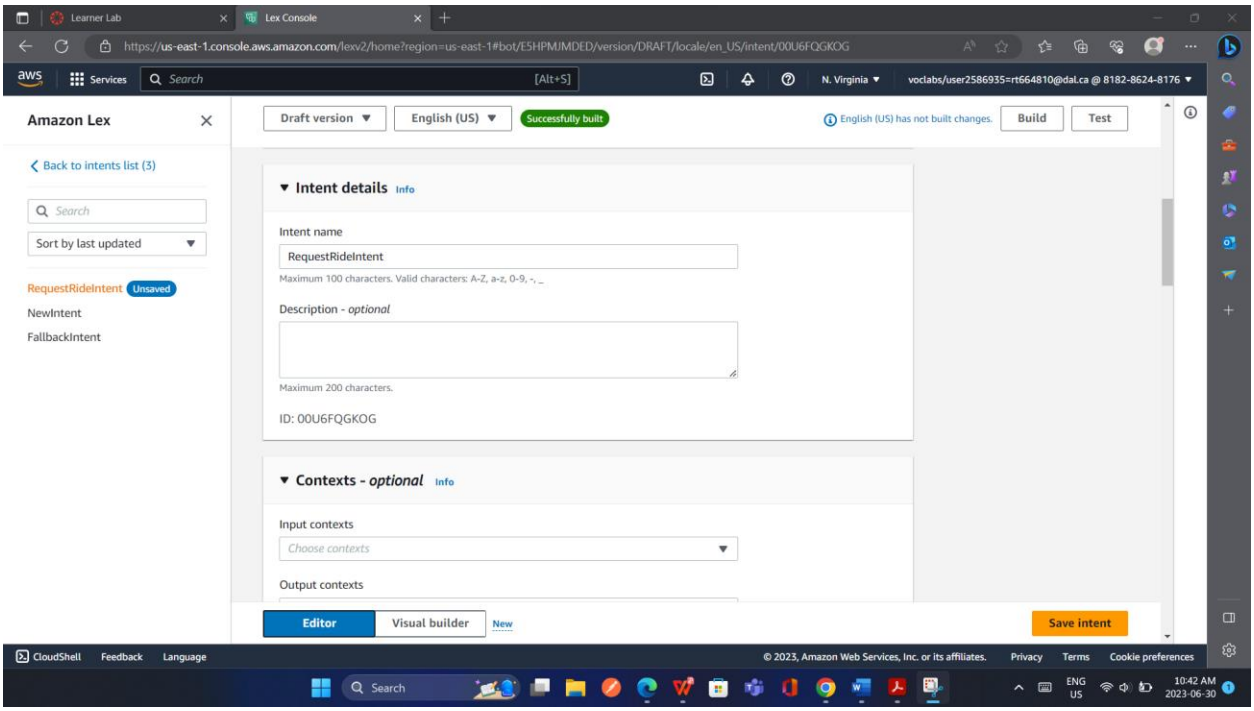

Figure 17 RideRequest Intent details [1]

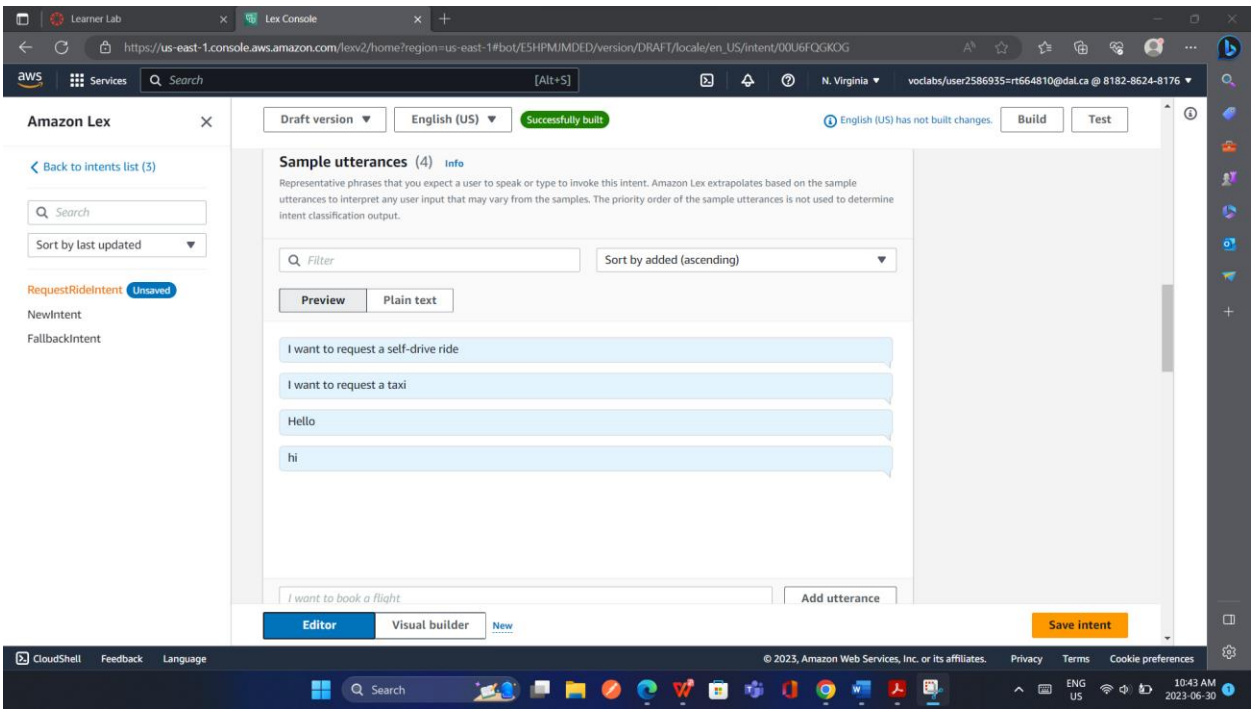

Figure 18 RideRequest utterances are used by user to request for taxi or self drive [1]

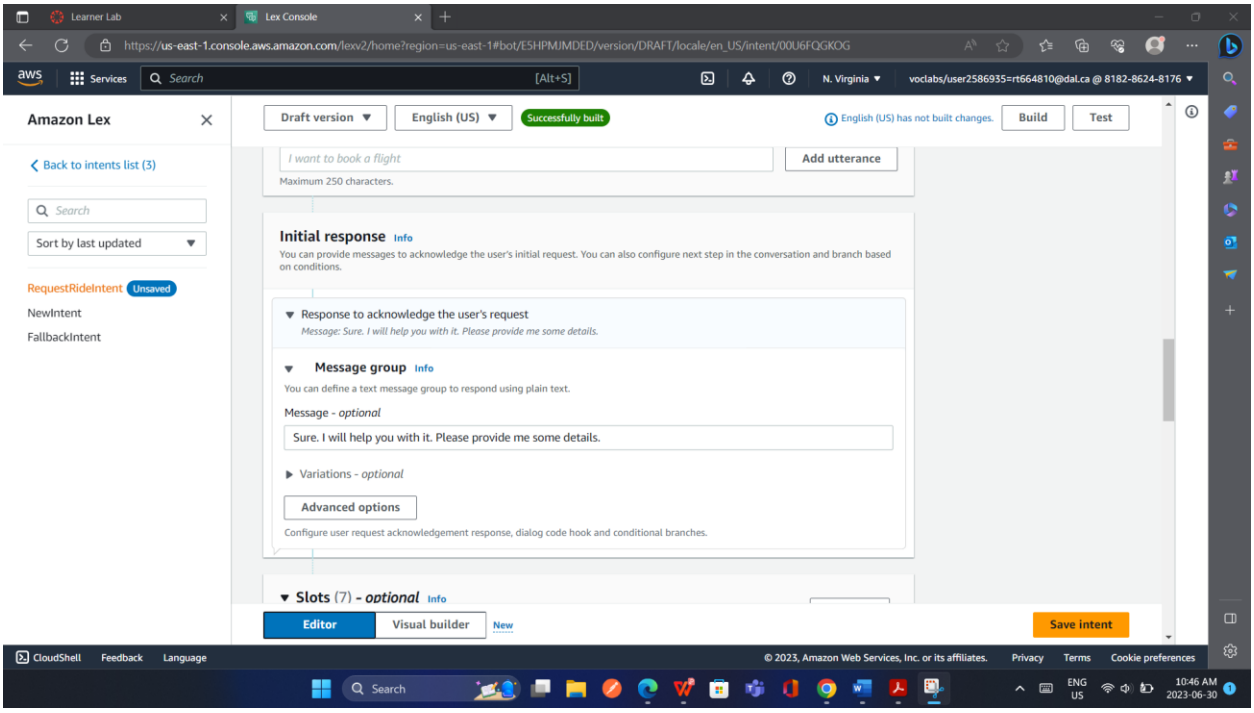

Figure 19 Set the Initial response for start the conversation [1]

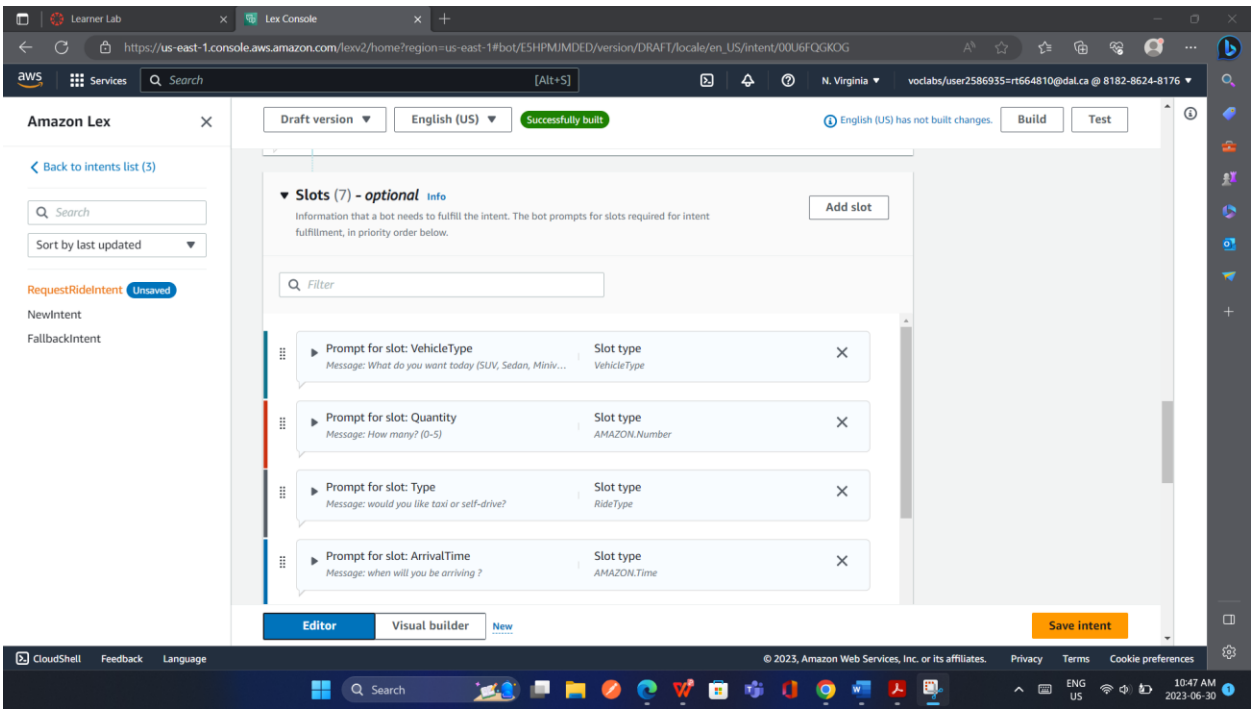

Figure 20 RideRequest slots configured to accept Vehicle Type, Quantity, Ride Type, and Arrival Time [1]

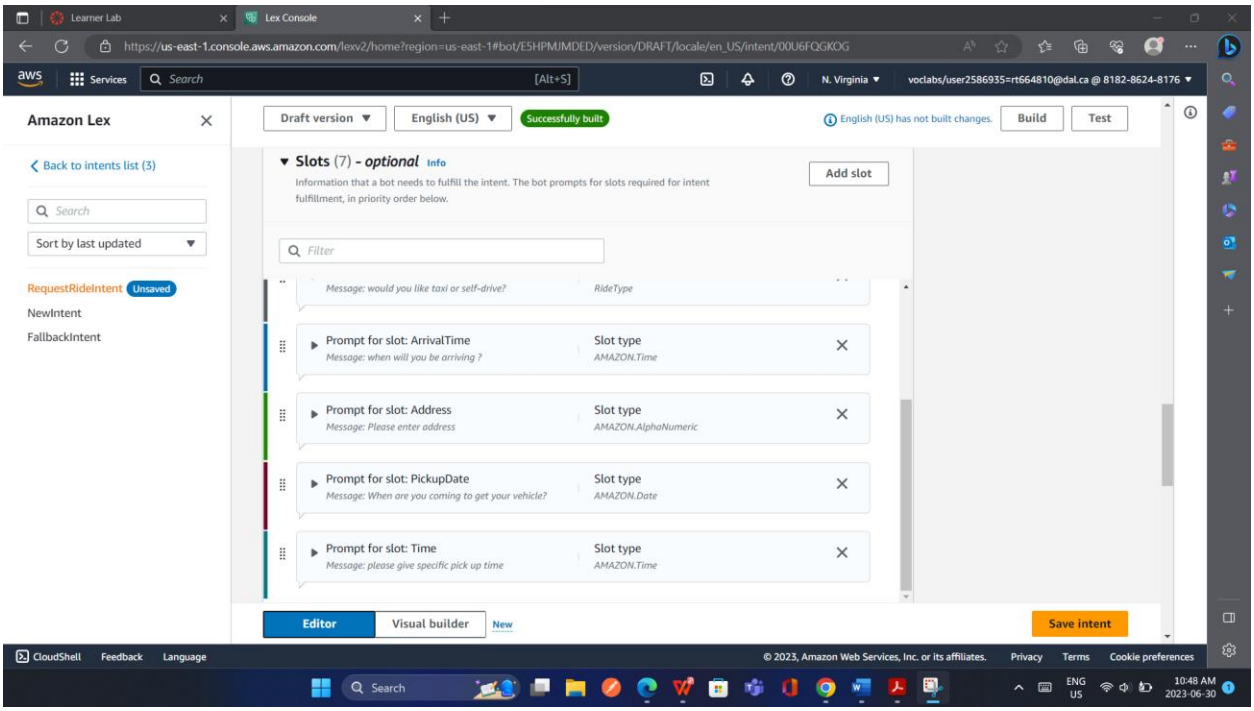

Figure 21 RideRequest slots configured to accept address, pick up date and time [1]

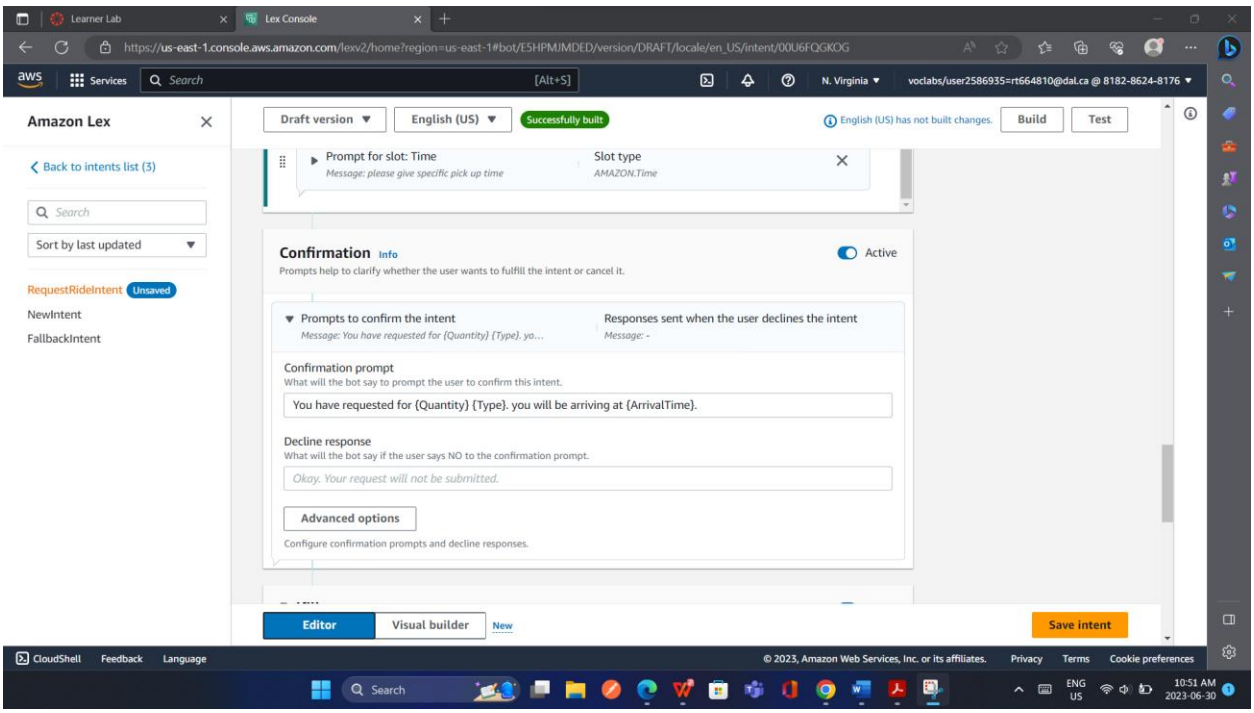

Figure 22 RideRequest Confirmation prompt [1]

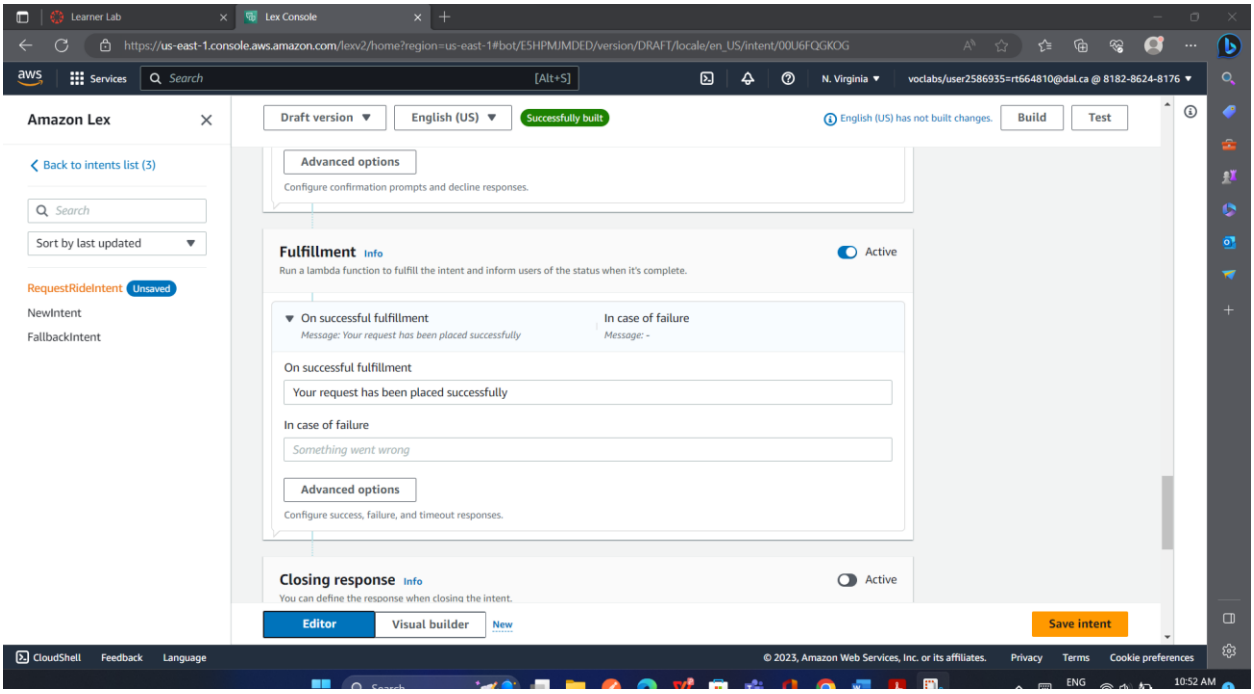

Figure 23 RideRequest Fulfillment Response [1]

Figure 24 displays that after completing the configuration set up, chatbot **RideRequest** Successfully built.

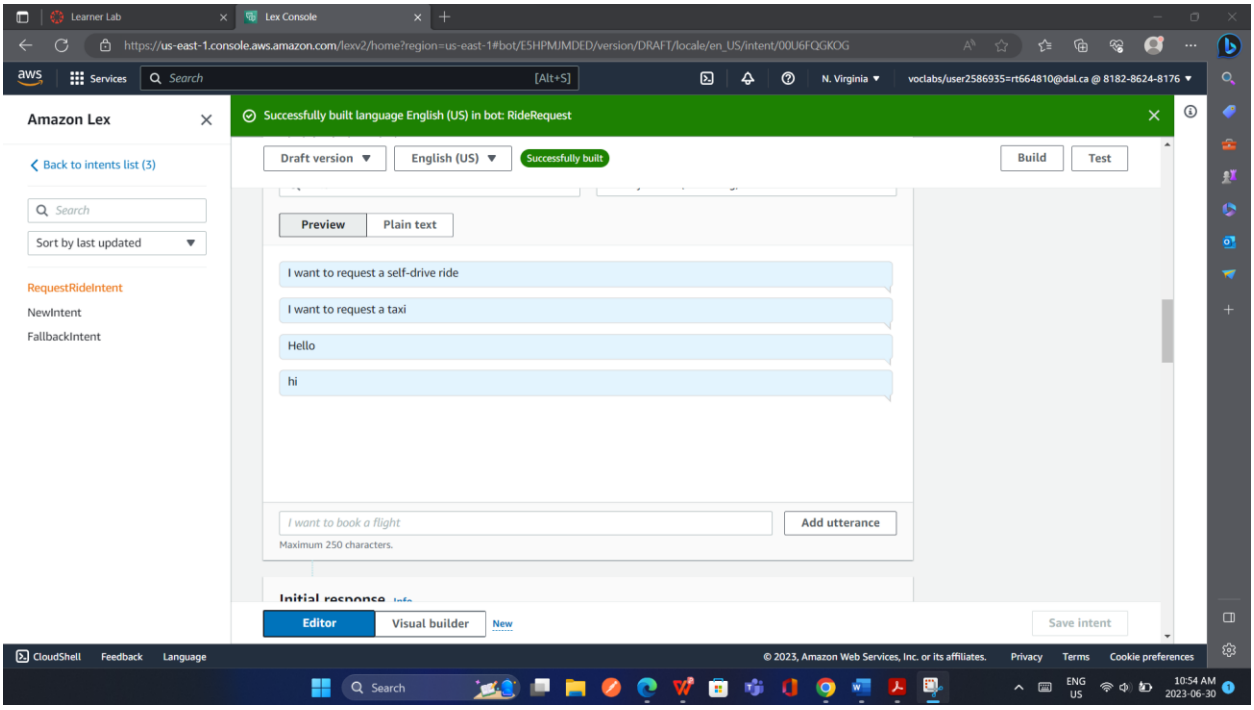

Figure 24 RideRequest chatbot built successfully [1]

Figures 25, 26, 27, 28, 29, 30, 31, and 32 depict the conversation flow when a user requests a self-drive option. The images demonstrate that the process of asking the user about vehicle types and quantity remains the same in both scenarios. However, the flow diverges when the user is asked about their preference for a taxi or self-drive. In the case of a self-drive request, the user will be prompted to provide the arriving time to collect the vehicle, followed by a confirmation message.

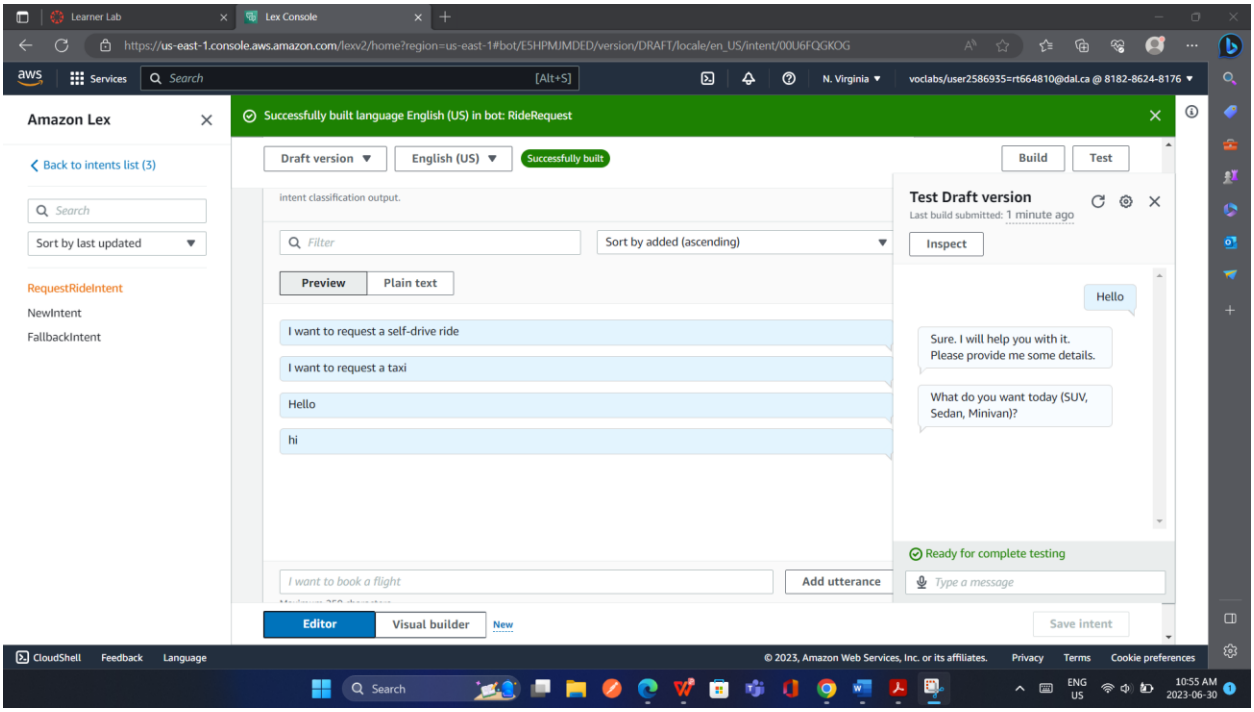

Figure 25 RideRequest chatbot testing conversation in working [1]

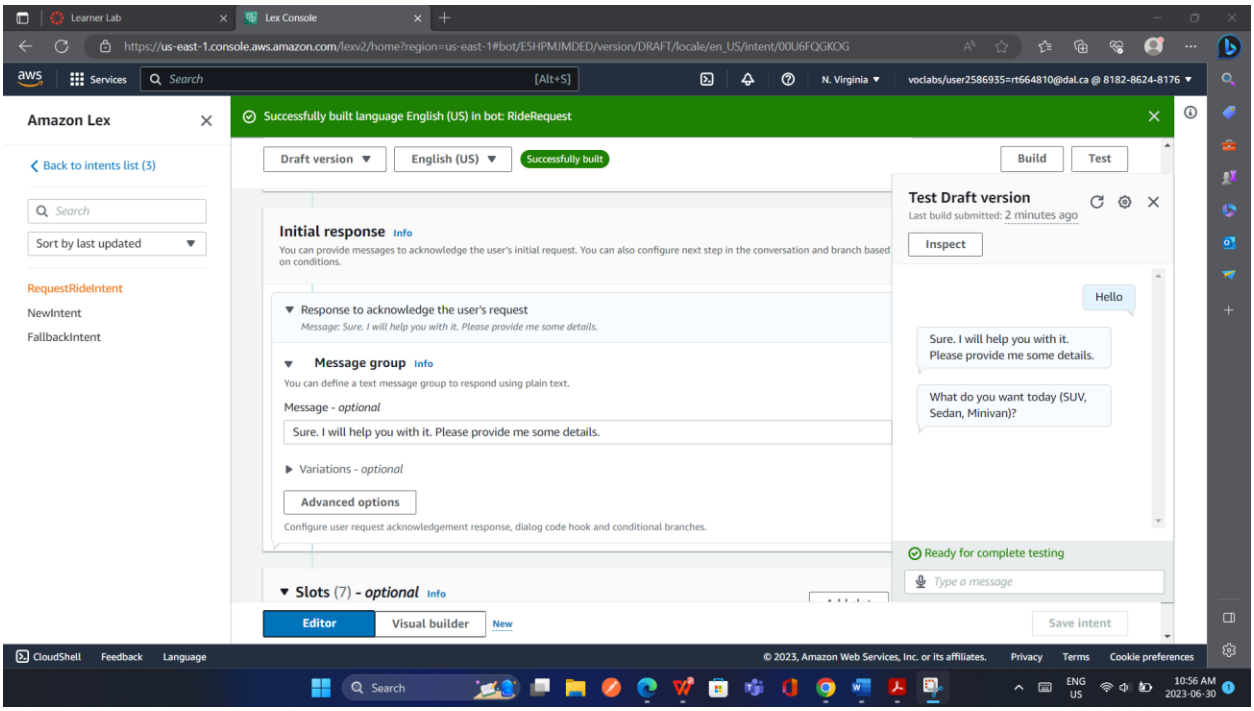

Figure 26 RideRequest chatbot testing conversation in working [1]

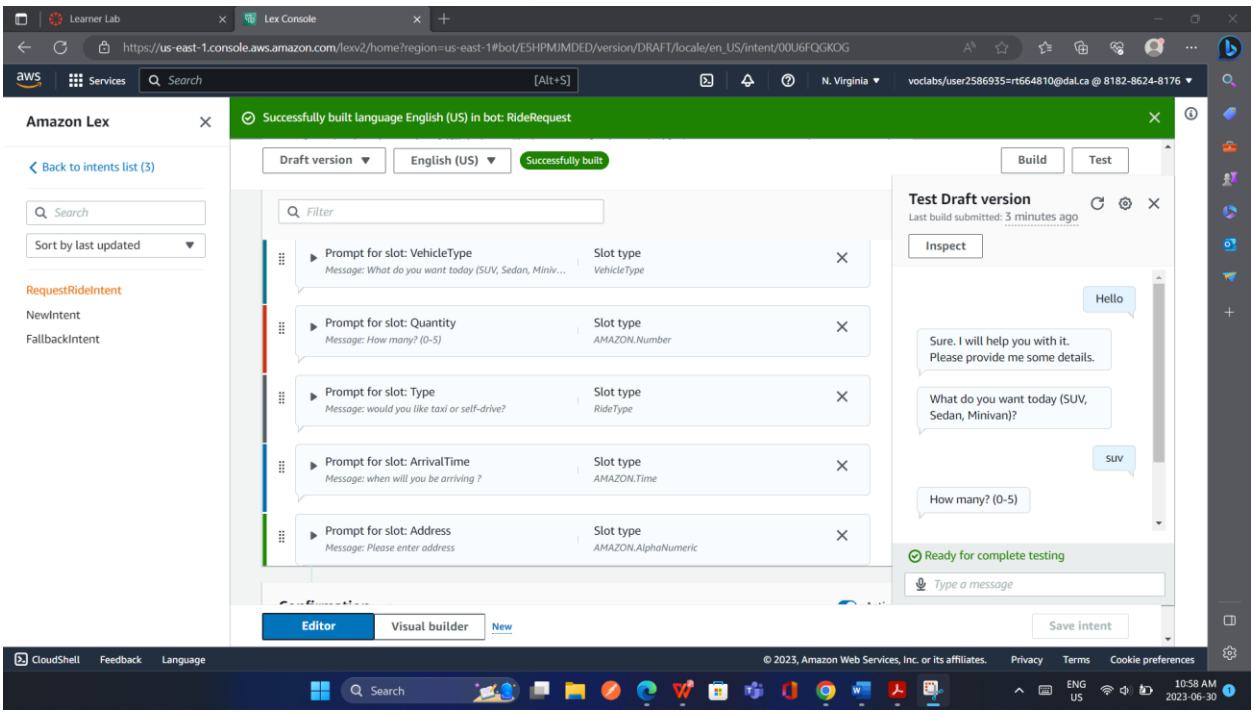

Figure 27 RideRequest chatbot testing conversation in working [1]

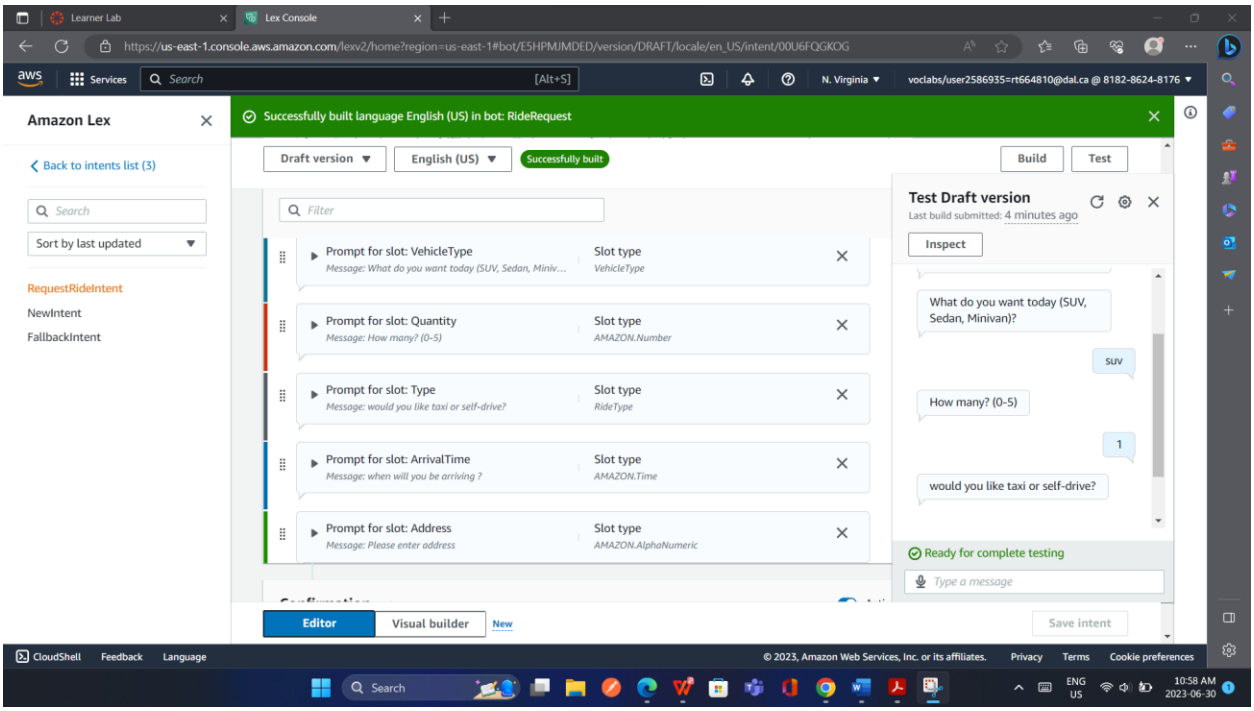

Figure 28 RideRequest chatbot testing conversation in working [1]

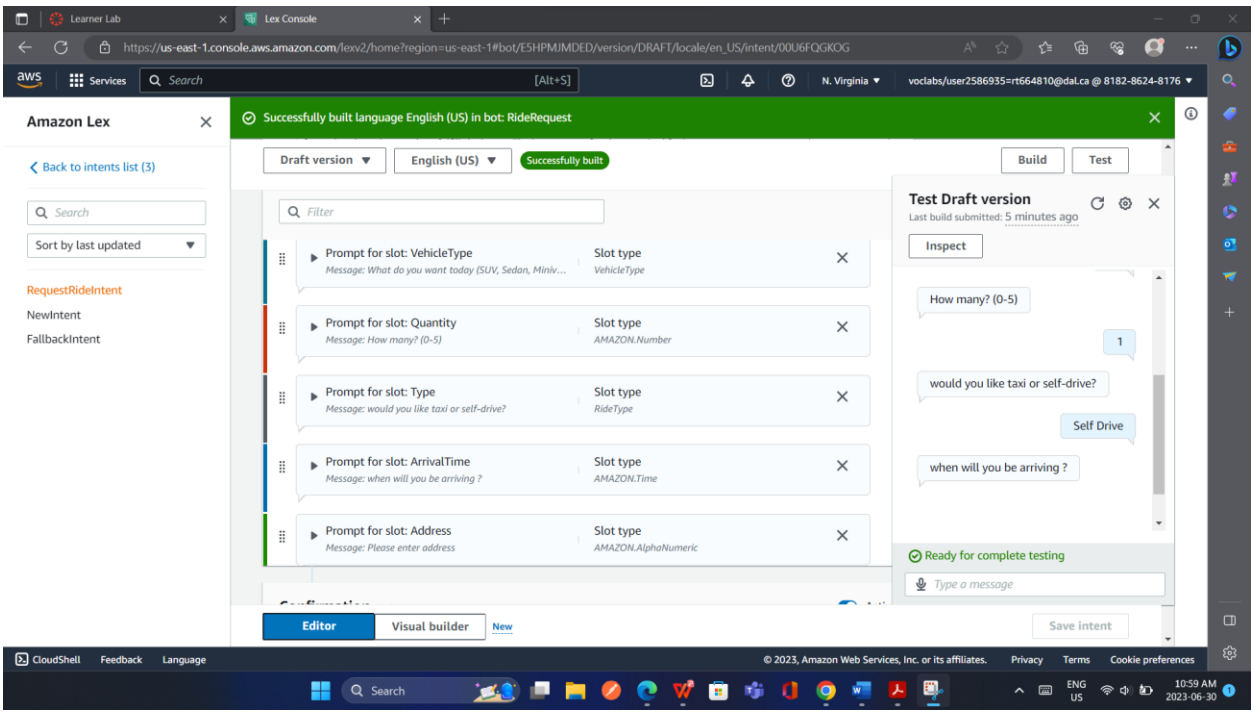

Figure 29 RideRequest chatbot testing conversation in working [1]

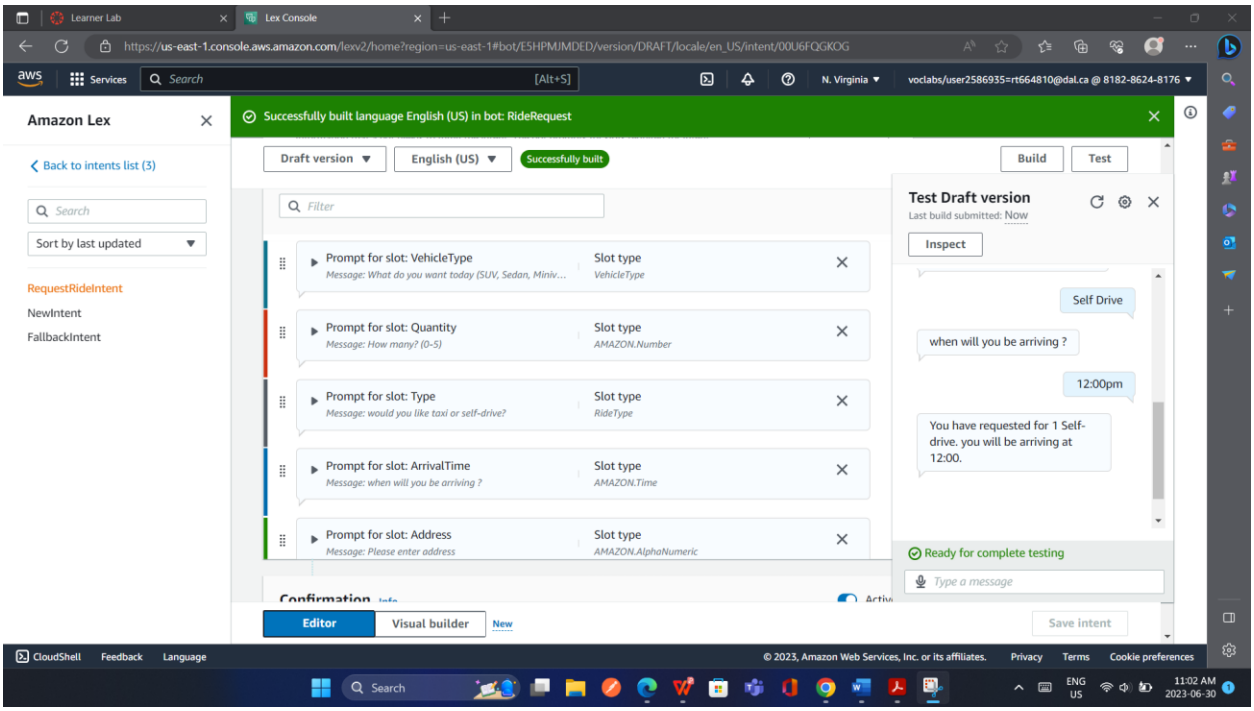

Figure 30 RideRequest chatbot testing conversation in working [1]

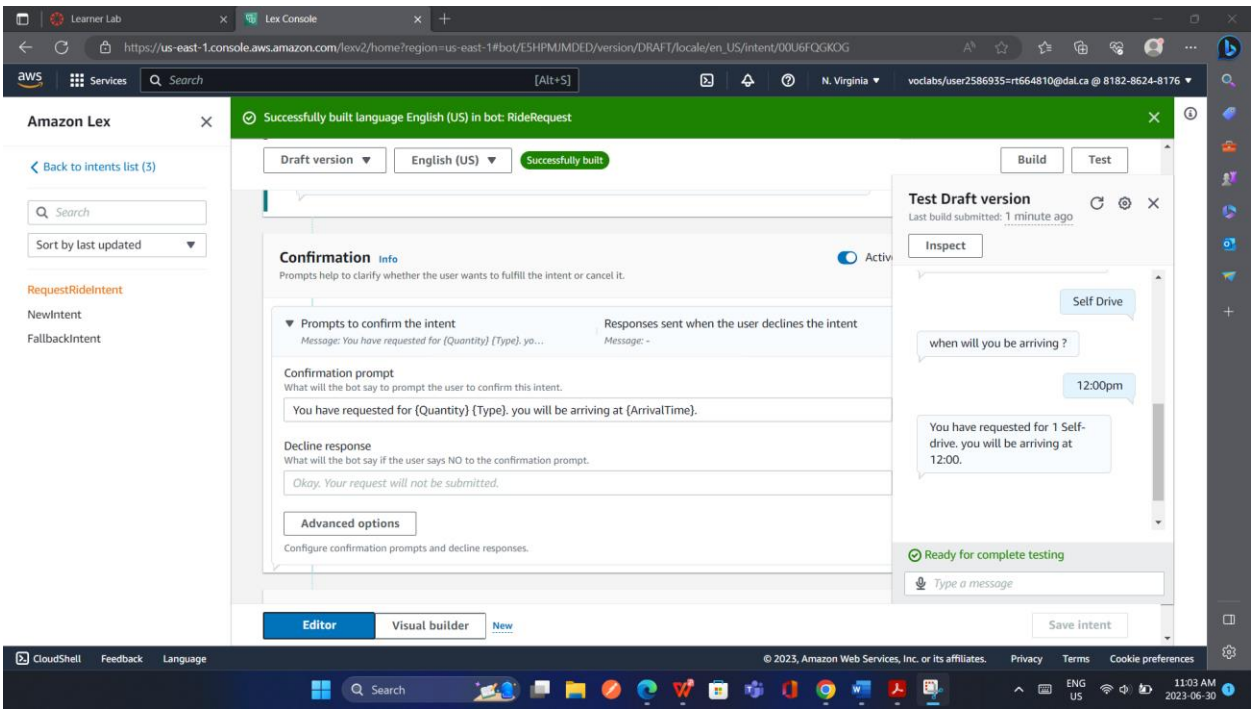

Figure 31 RideRequest chatbot testing conversation in working [1]

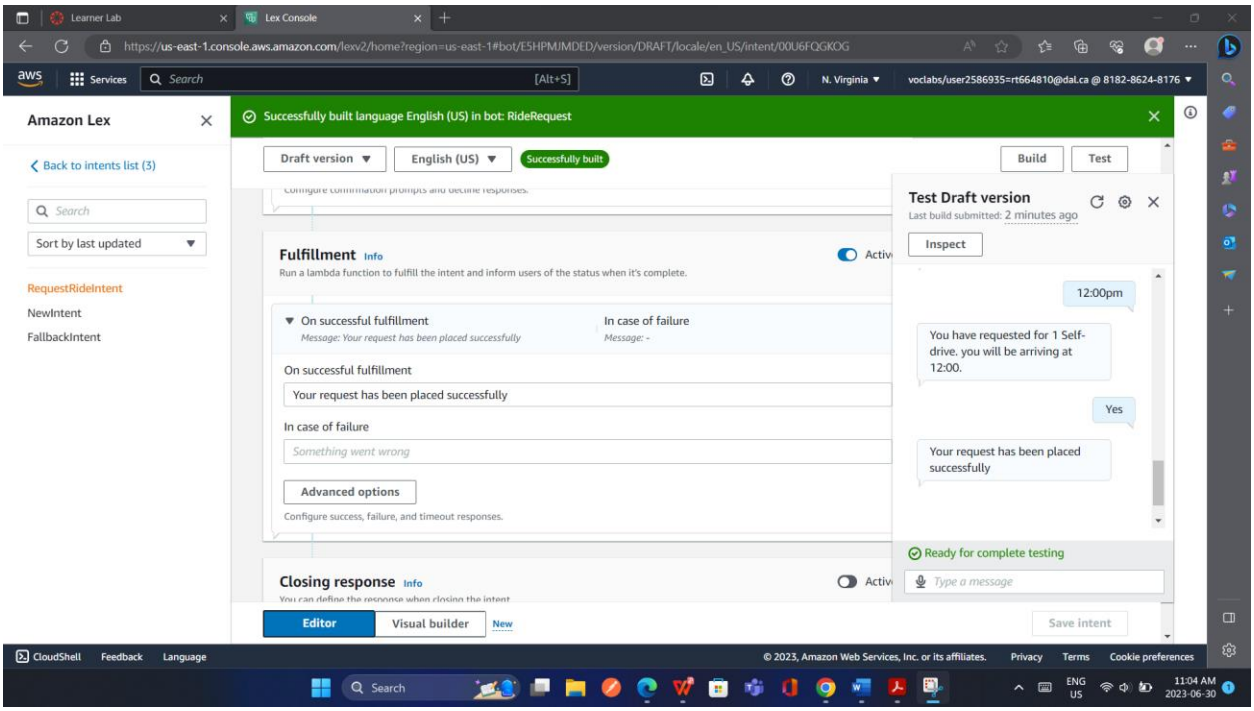

Figure 32 RideRequest chatbot testing conversation in working [1]

Figures 33, 34, 35, 36, 37, 38, 39, and 40 depict the conversation flow when a user requests a taxi option. The images demonstrate that the process of asking the user about vehicle types and quantity remains the same in both scenarios. However, the flow diverges when the user is asked about their preference for a taxi or self-drive. In the case of a taxi request, the user will be prompted to provide the address, pick up date and time, followed by a confirmation message.

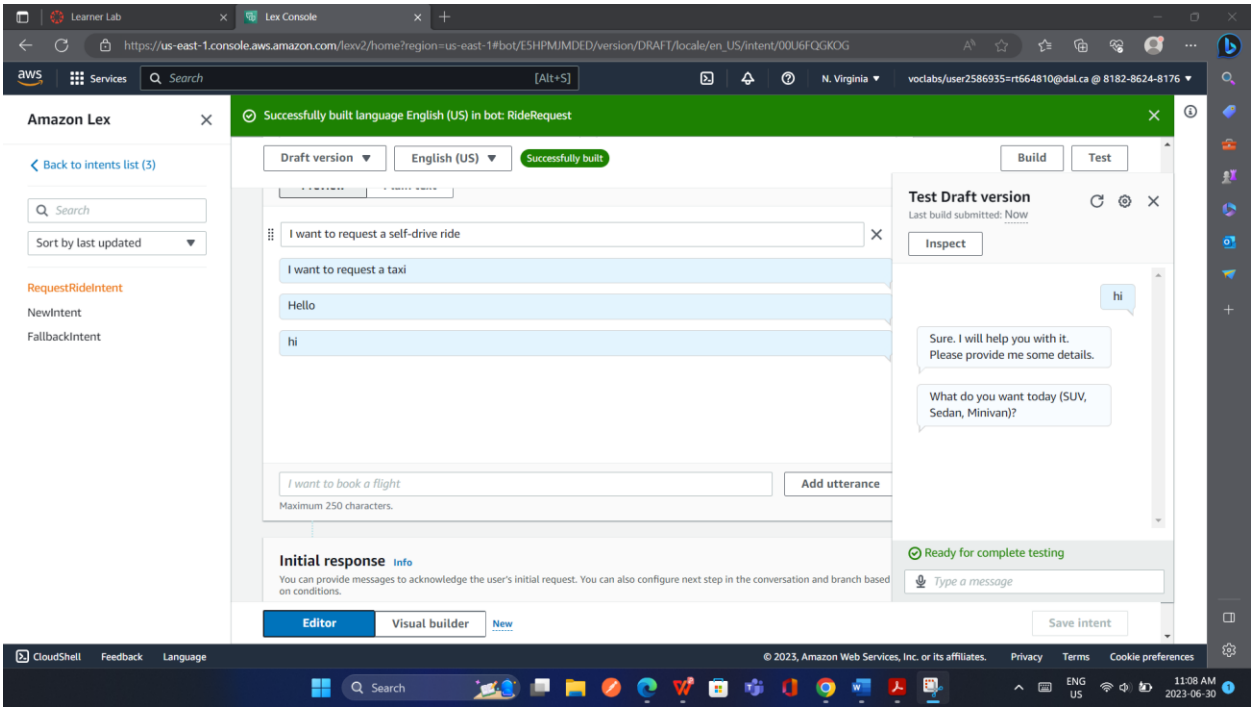

Figure 33 RideRequest chatbot testing conversation in working [1]

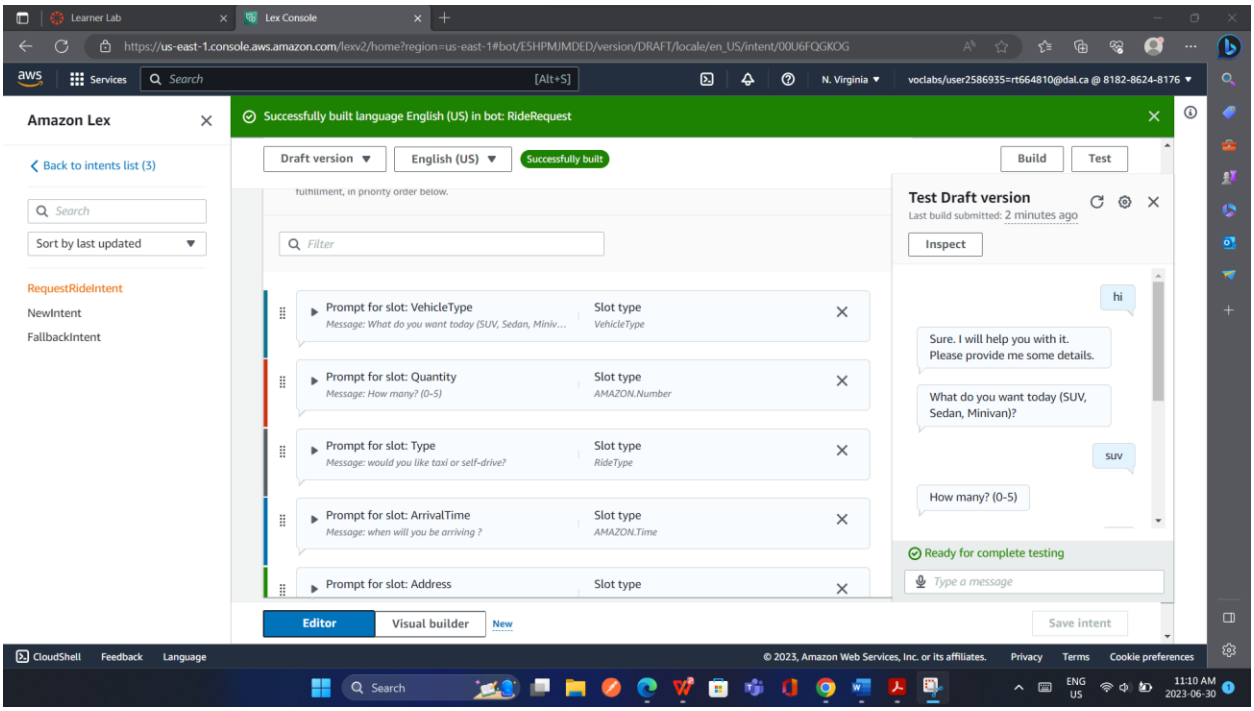

Figure 34 RideRequest chatbot testing conversation in working [1]

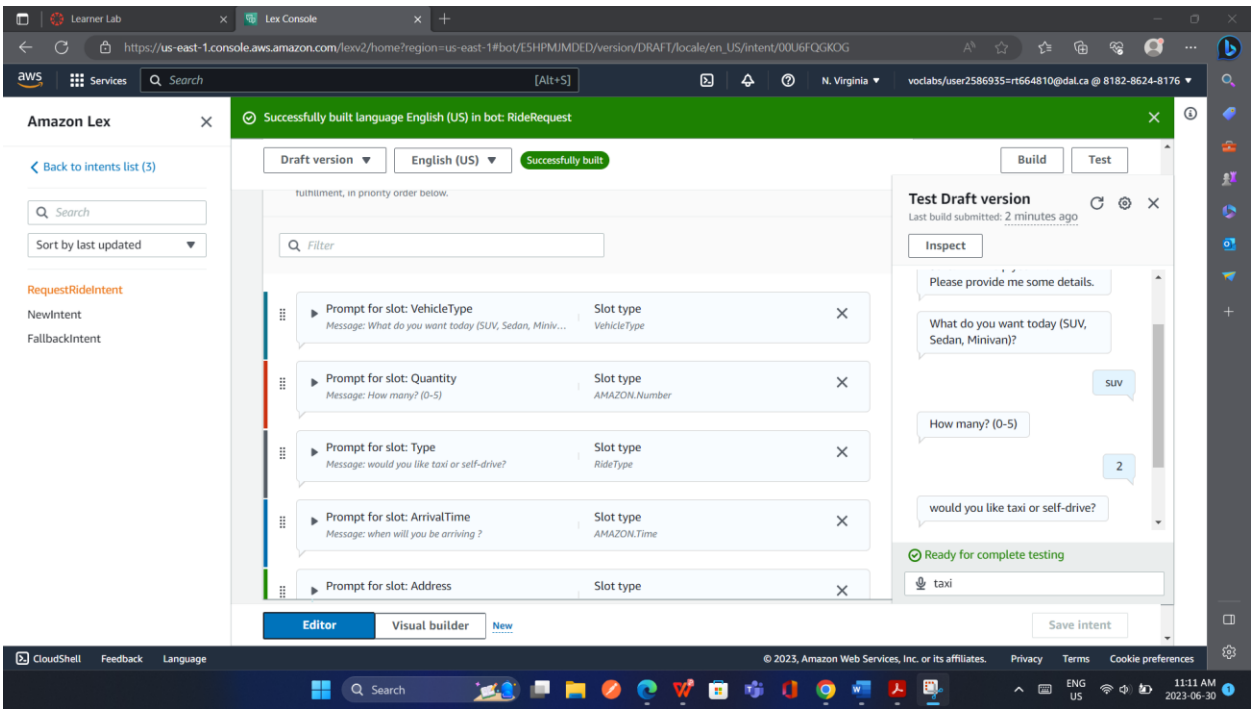

Figure 35 RideRequest chatbot testing conversation in working [1]

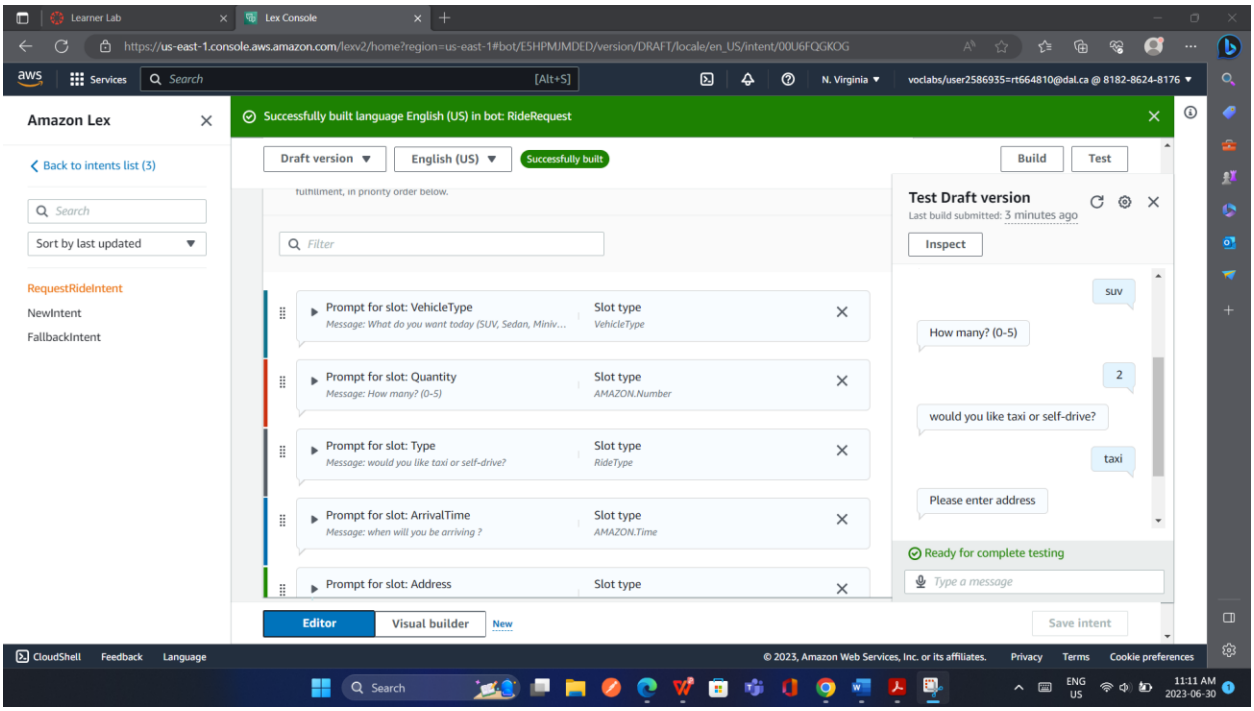

Figure 36 RideRequest chatbot testing conversation in working [1]

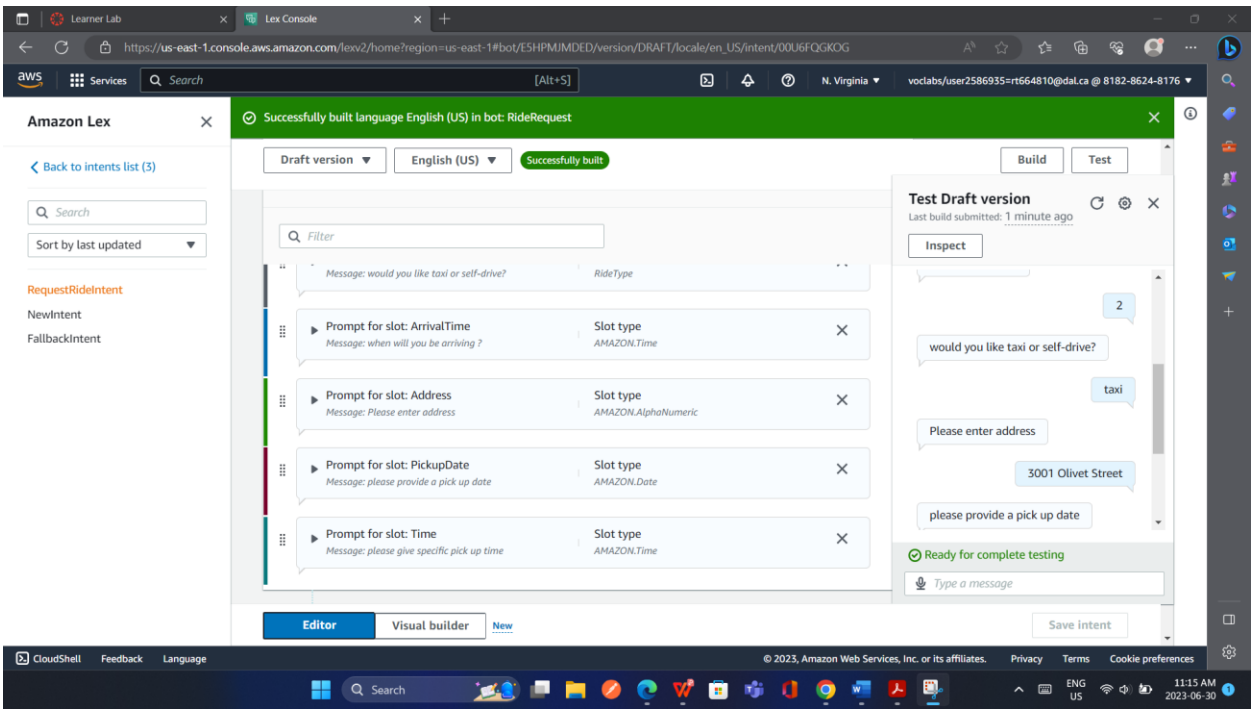

Figure 37 RideRequest chatbot testing conversation in working [1]

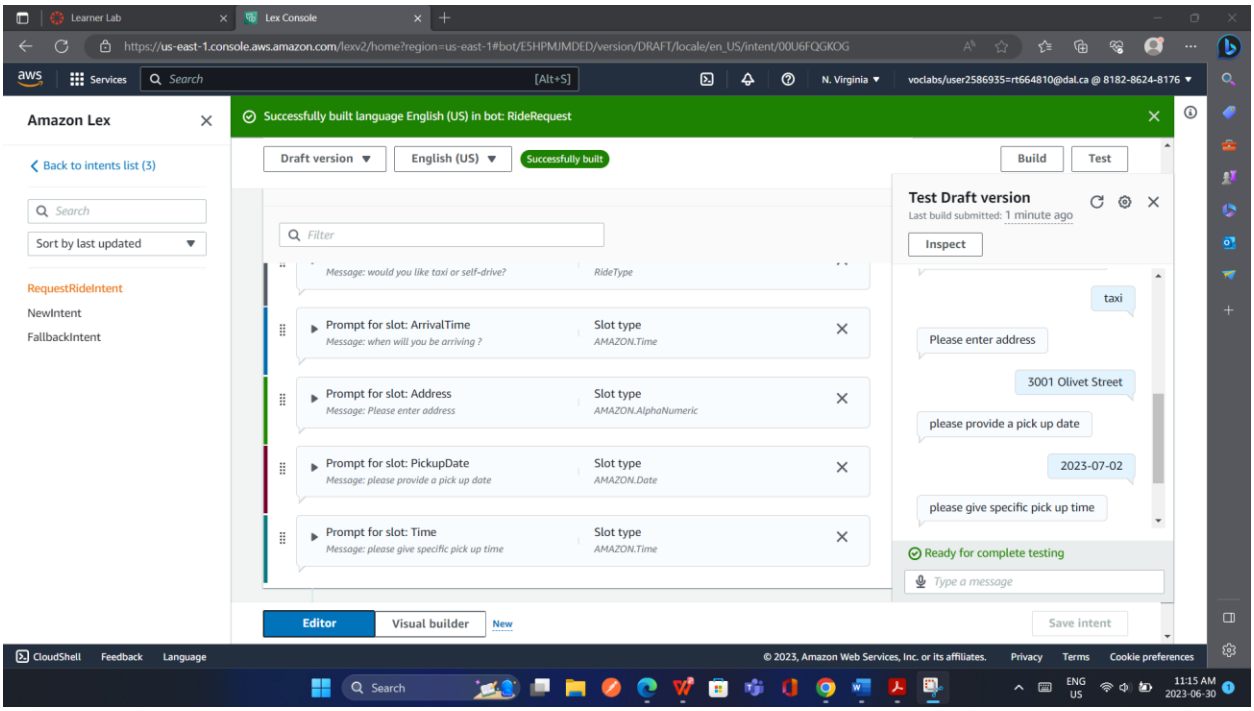

Figure 38 RideRequest chatbot testing conversation in working [1]

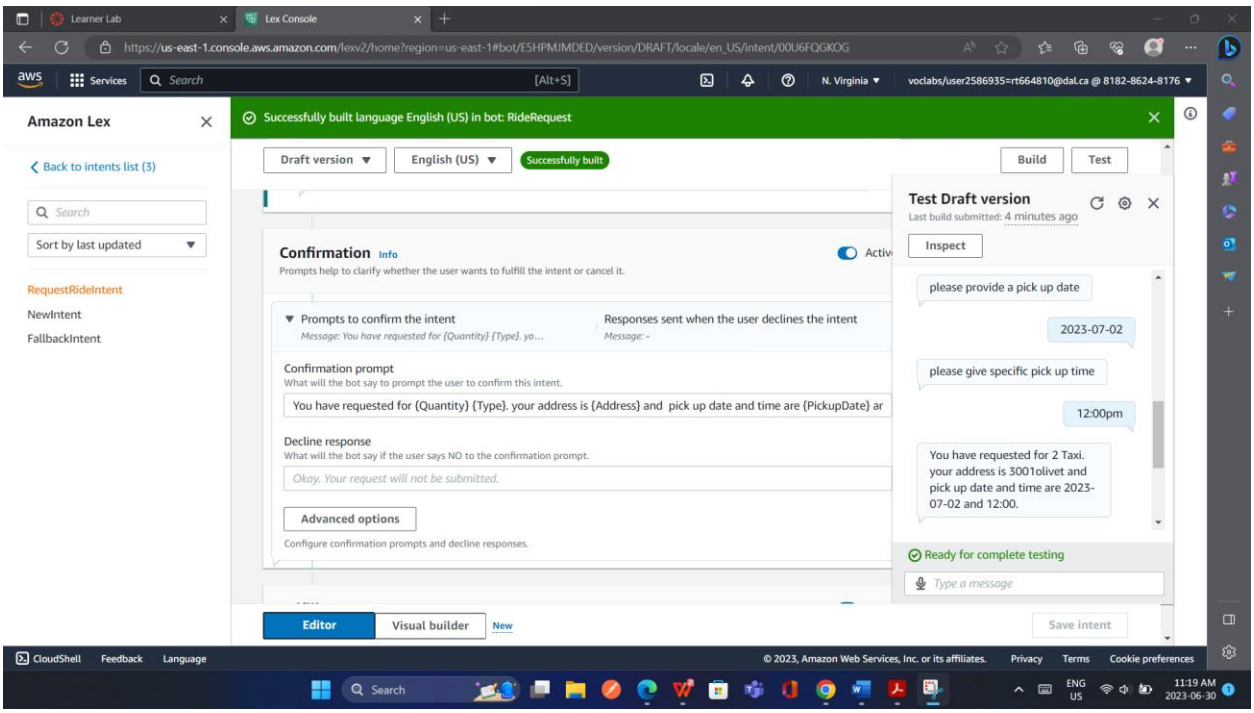

Figure 39 RideRequest chatbot testing conversation in working [1]

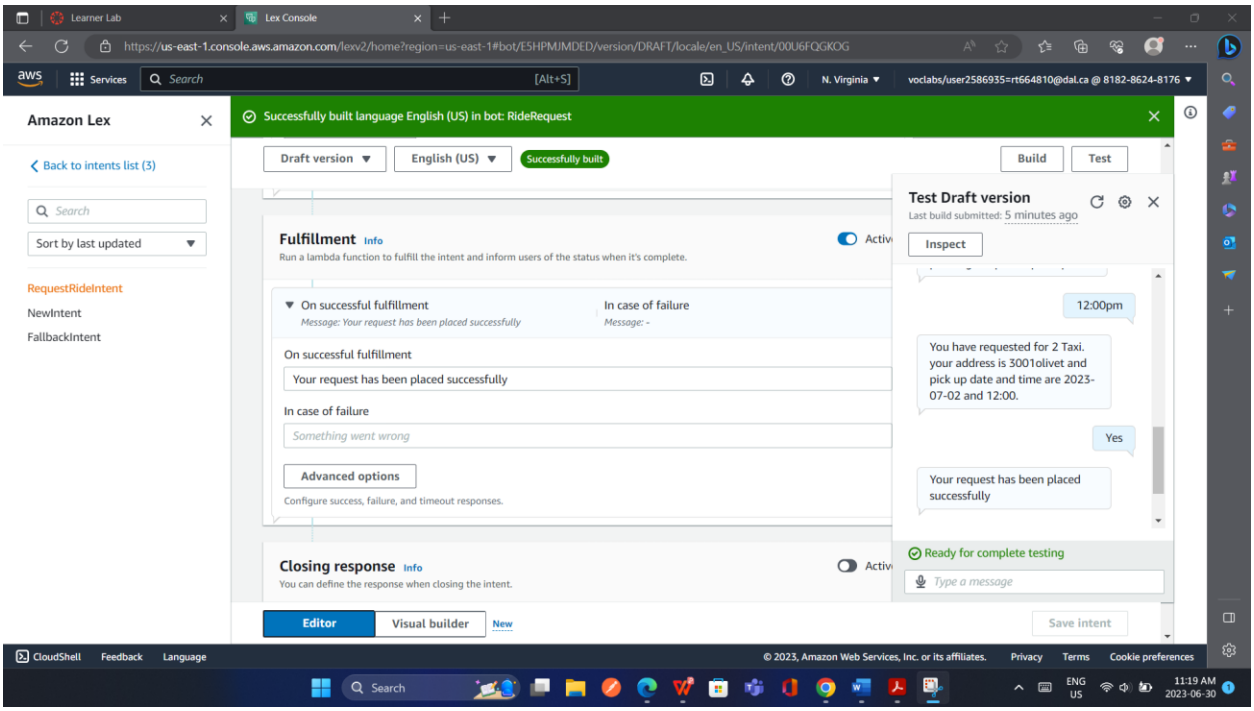

Figure 40 RideRequest chatbot testing conversation in working [1]

Summary about the operations performed

The RideRequest chatbot is desined using Amazon Lex, which is a service offred by Amazon Web Services (AWS) specifically designed for creating organized conversational chatbots. The configuration of the chatbot involves specifying details such as the chatbot's name,language set up for the bot and other things as well . The creation process includes the setup of intents, and for this assignment, a RequestRideIntent is created. This intent handle user requests for taxi or self-drive services and encompasses various tasks such as utterances, intent details, slot types, confirmation prompts, and fulfillment messages. Utterances are predefined phrases or sentences used by users to initiate conversations with the chatbot. Sample utterances are set up to enable the chatbot to understand the expected messages from users. Slot types play a crucial role in extracting specific information during interactions. In this task, built-in slot types for time, date, and address are utilized. Additionally, custom slot types are created to provide choices for ride types and vehicle types. A confirmation message is utilized to confirm that the user has provided all the necessary details. Once the user confirms the information, the chatbot sends a fulfillment message containing the user's details as per the conversation. All these components are designed and implemented using the Amazon Lex service, which is part of the suite of services offered by Amazon Web Services.

References

[1] AWS, “Amazon Lex – Build Conversation Bots,” Amazon Web Services, Inc., 2019. [Online]. Available: https://aws.amazon.com/lex/. [Accessed 30 June 2023].