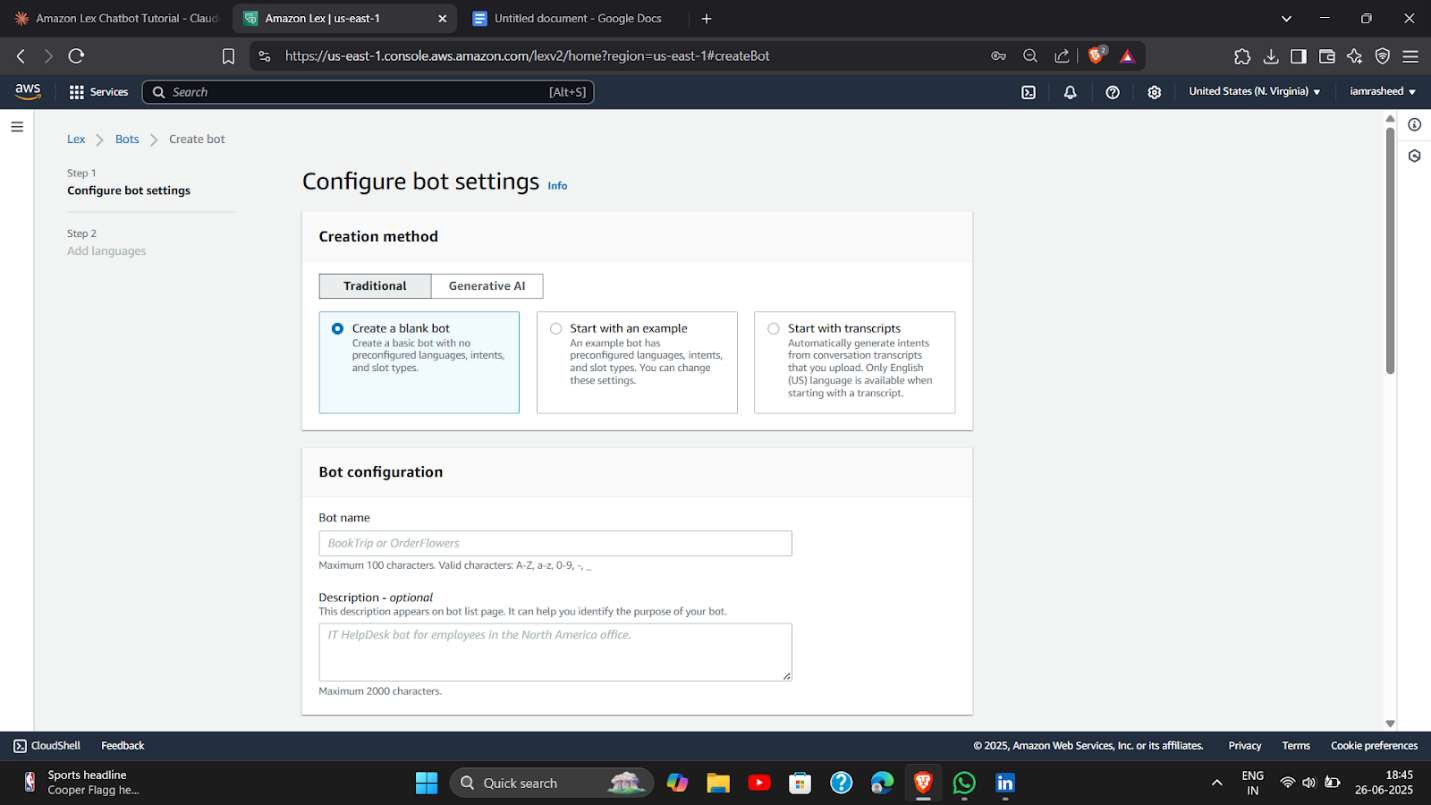
Amazon Lex is a service for building conversational interfaces using voice and text, powered by the same deep learning technologies as Alexa. It enables developers to create intelligent chatbots that can understand natural language and respond meaningfully.

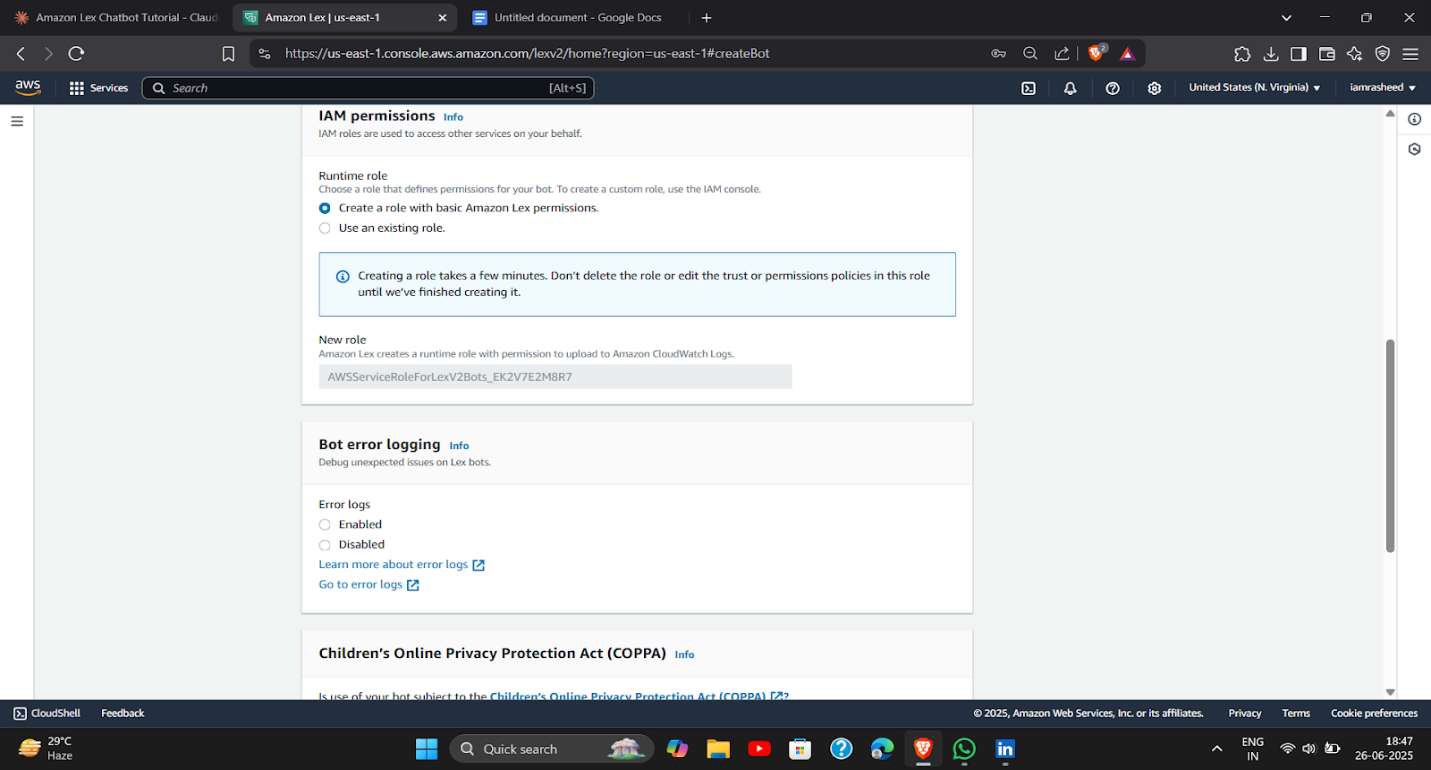
Go to AWS Console → Amazon Lex → Create Bot

Navigate to the AWS Management Console, search for “Amazon Lex” and click Create bot to start building your chatbot.



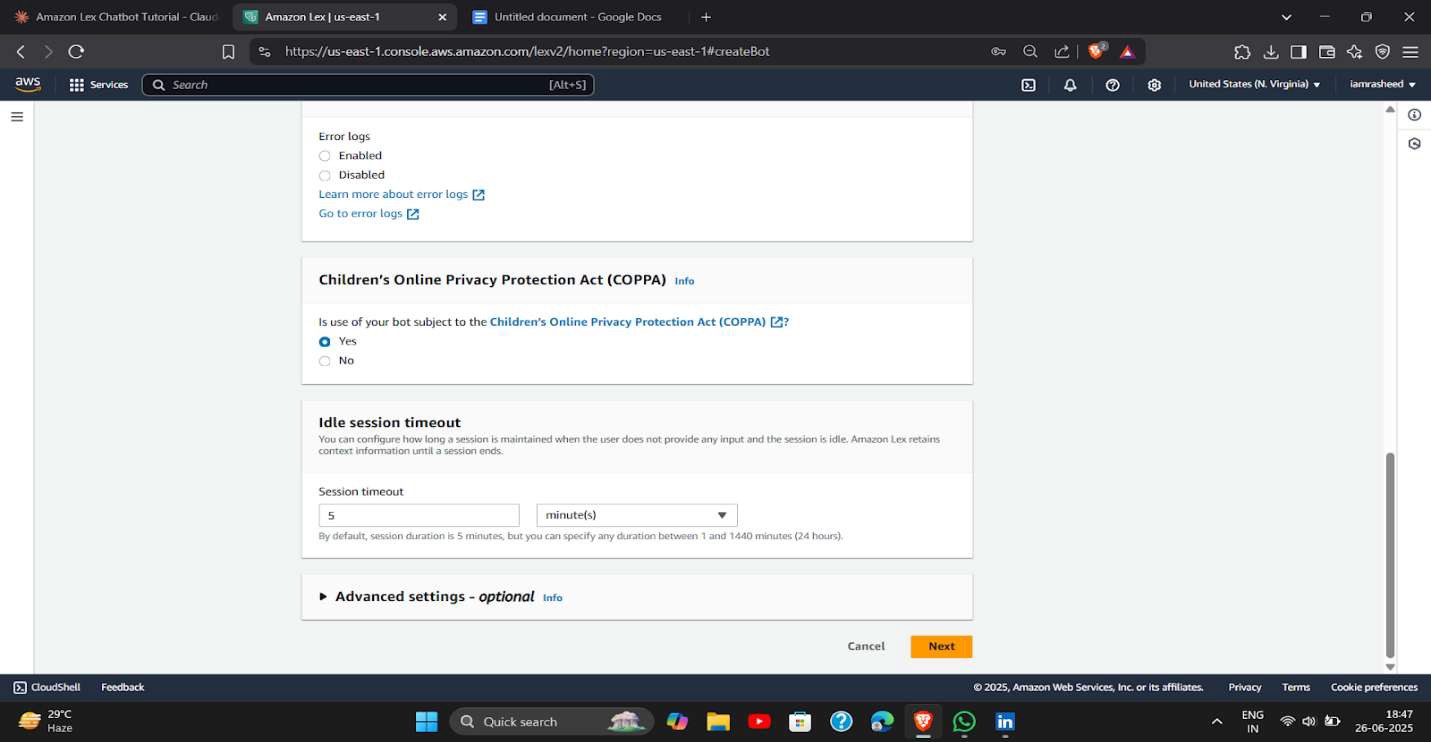
Creation Method - Traditional → Give Bot Name

Select the Traditional method (not the Lex V2 version), then enter a relevant bot name that represents your use case (e.g., HotelBot).



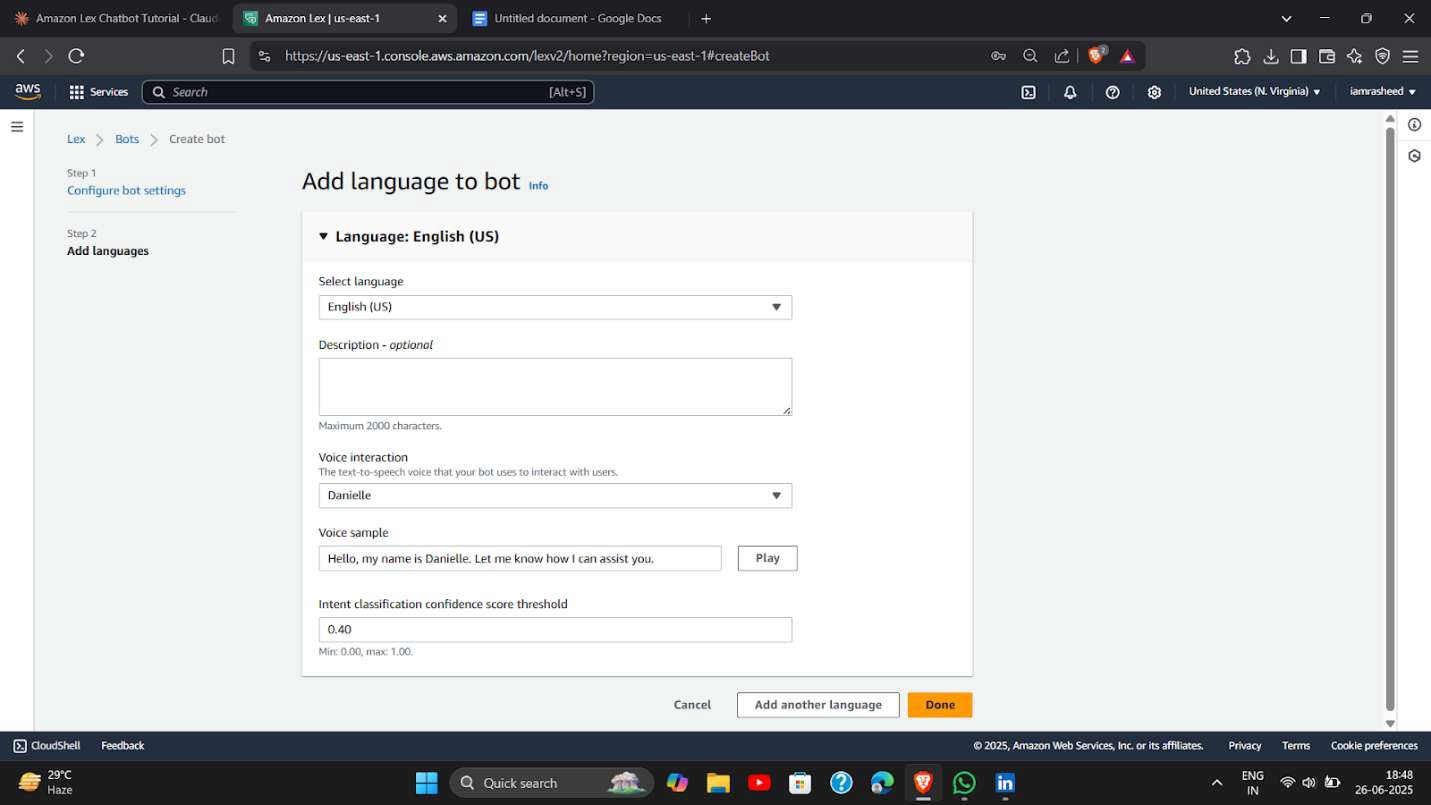
IAM Permissions - Create New Role → COPPA - Yes → Idle Session Timeout - Default 5 min

Allow Lex to create a new IAM role automatically for permissions, conﬁrm that your bot is not subject to COPPA, and keep the session timeout at 5 minutes.

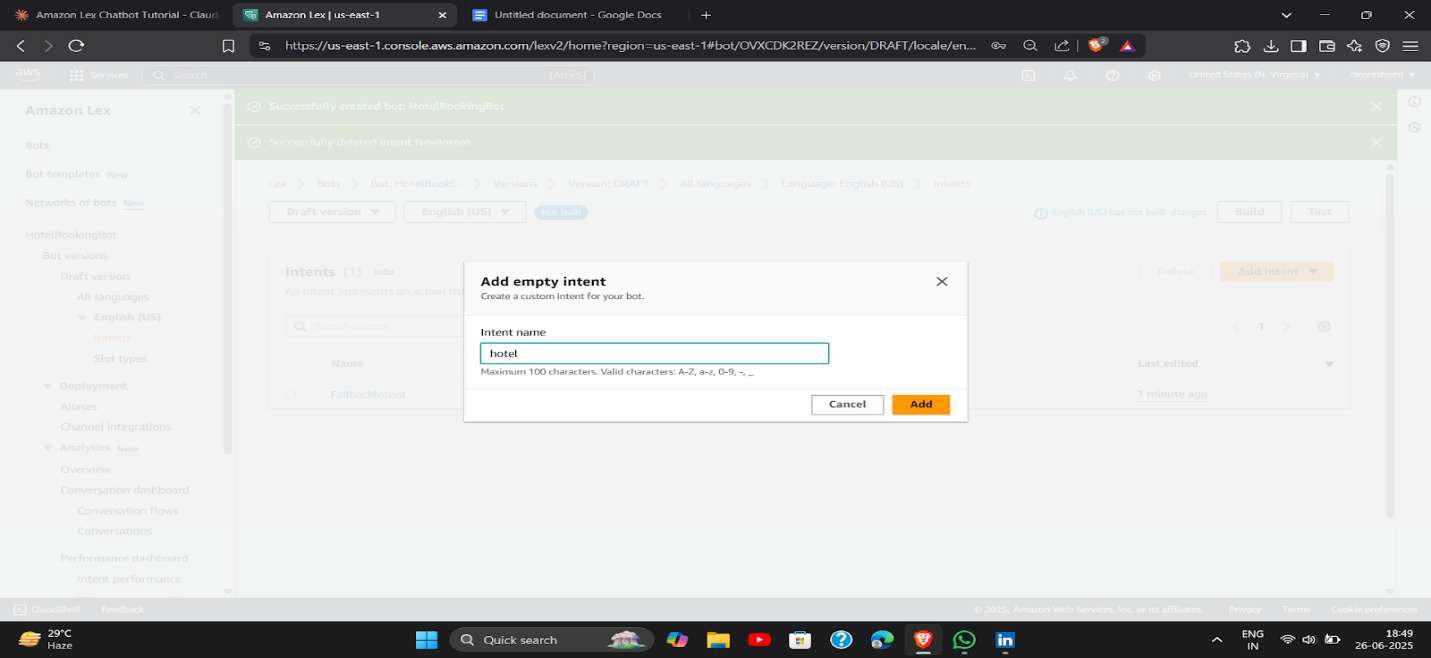


Add Language

Choose the language your chatbot will support (e.g., English), which determines how utterances and responses are processed.

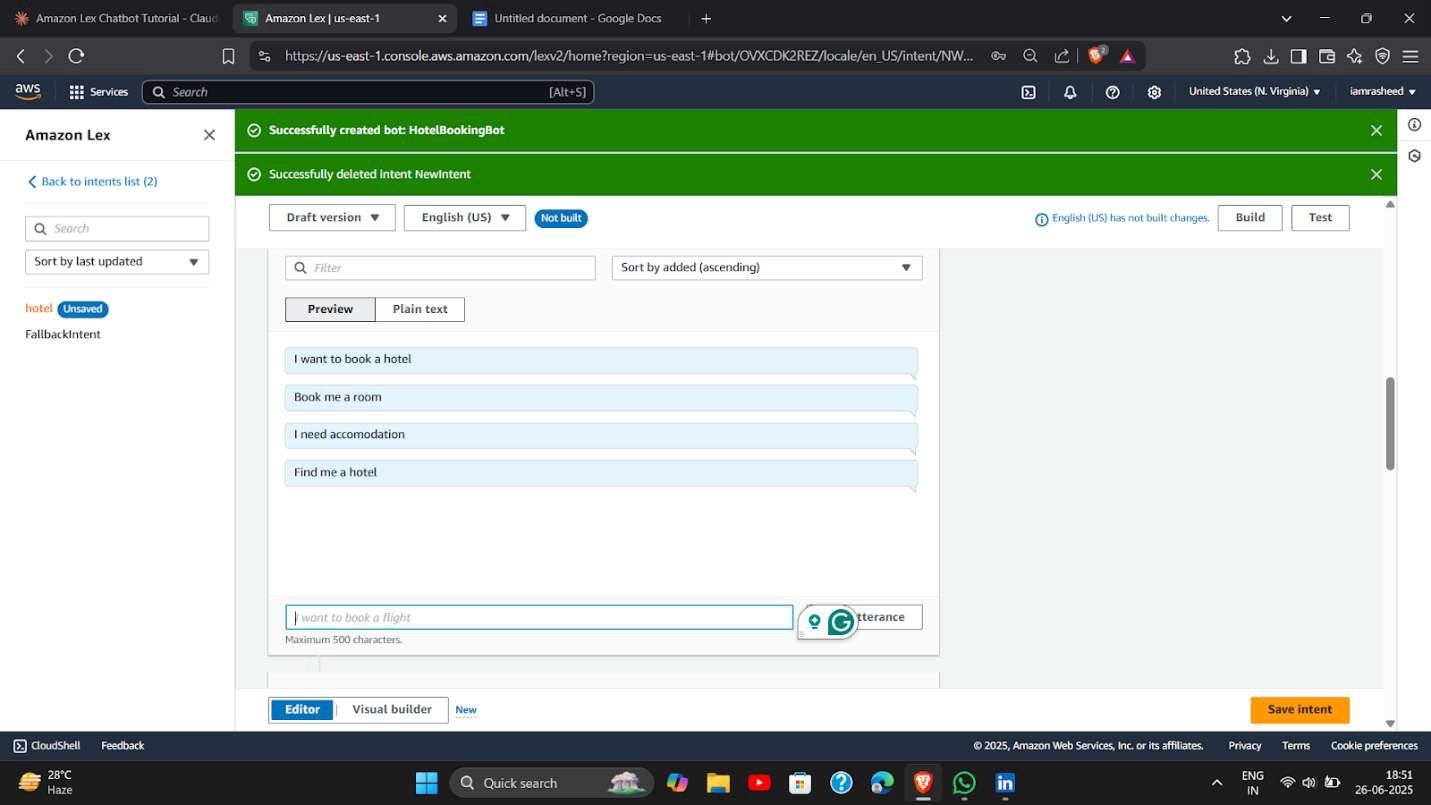


Go to the Bot → Intent → Add Intent Access your created bot and go to the Intents section; click Add intent to deﬁne the goal or purpose of a user interaction.



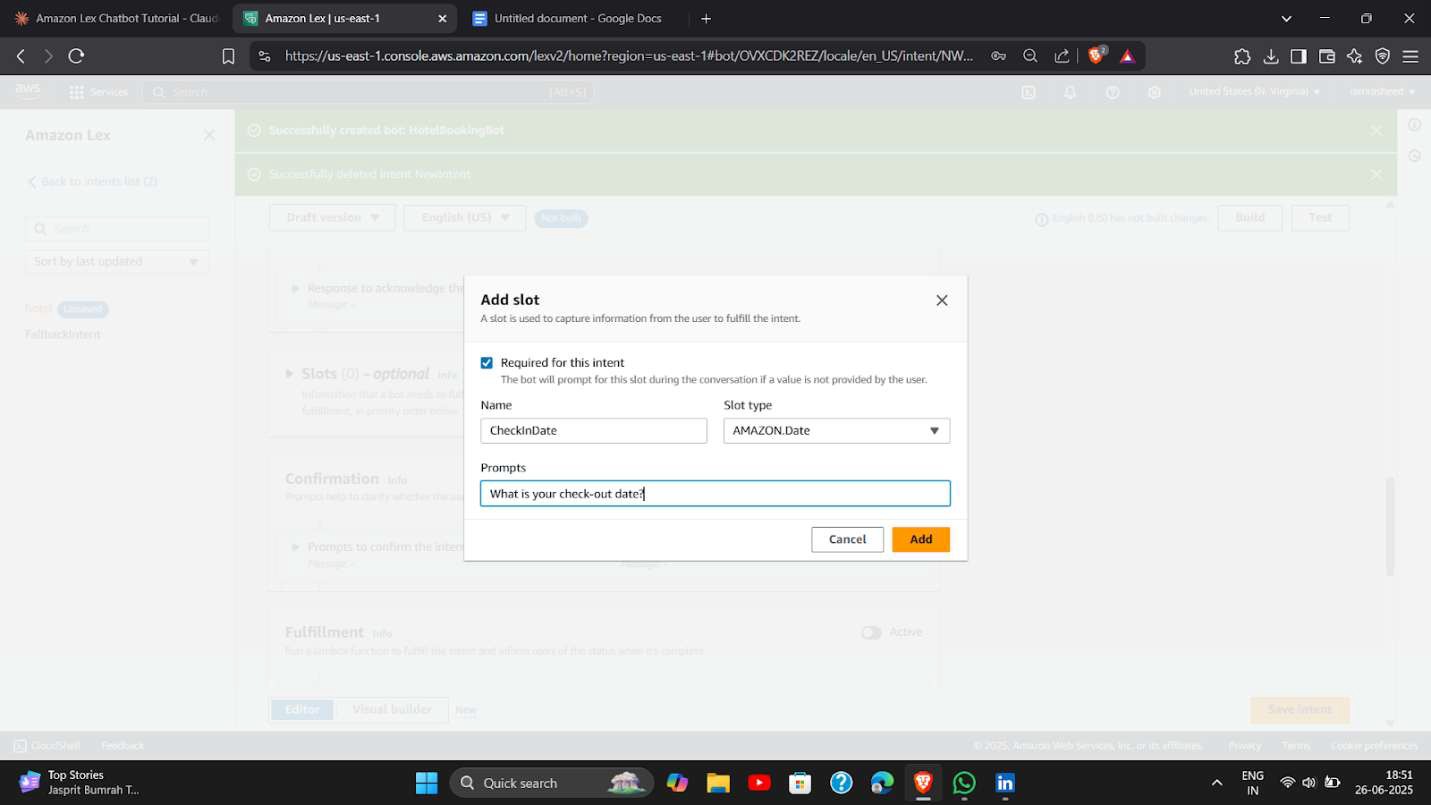
Scroll Down → Utterances → Add

In the intent, scroll down to the Sample Utterances section and add phrases users might say (e.g., “Book a hotel room”).

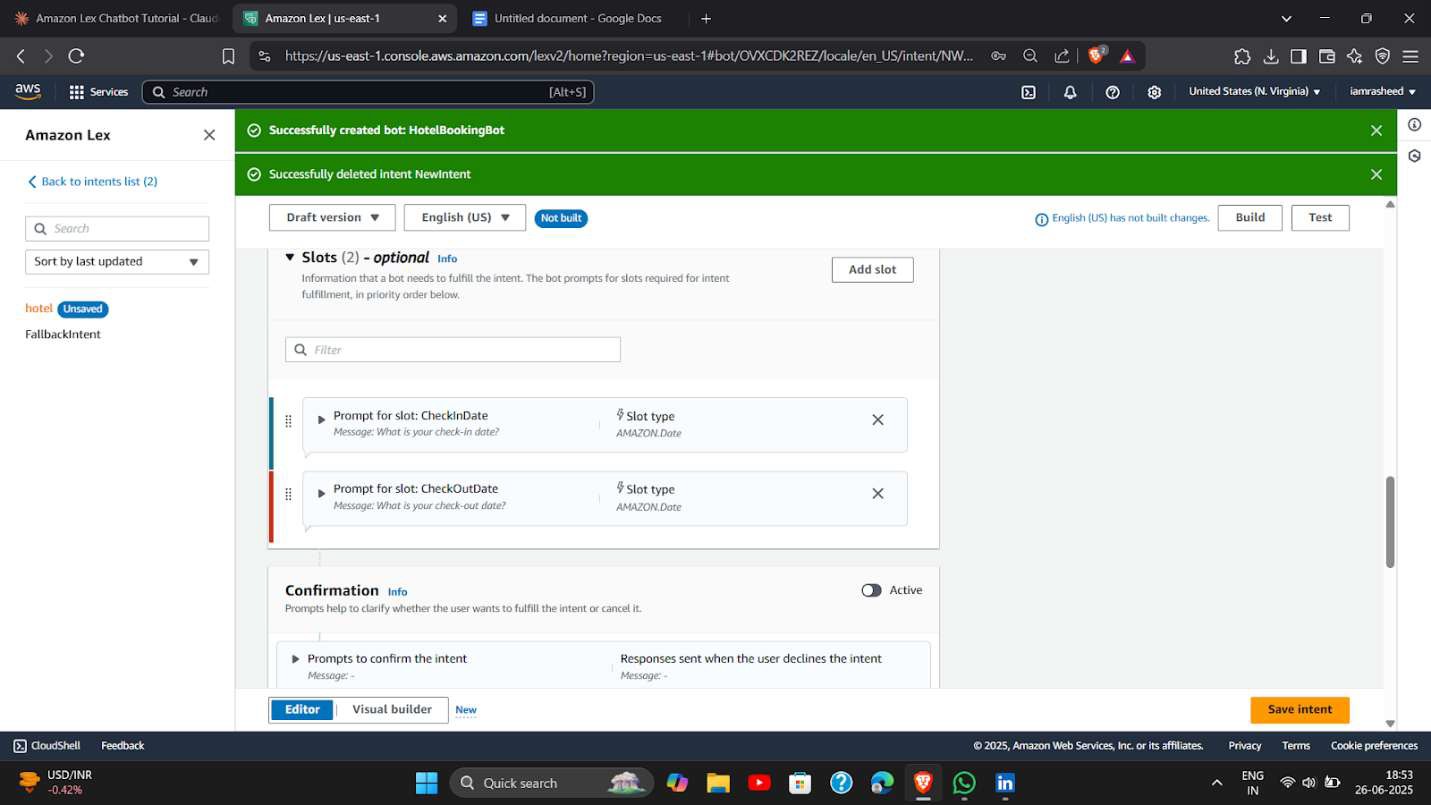
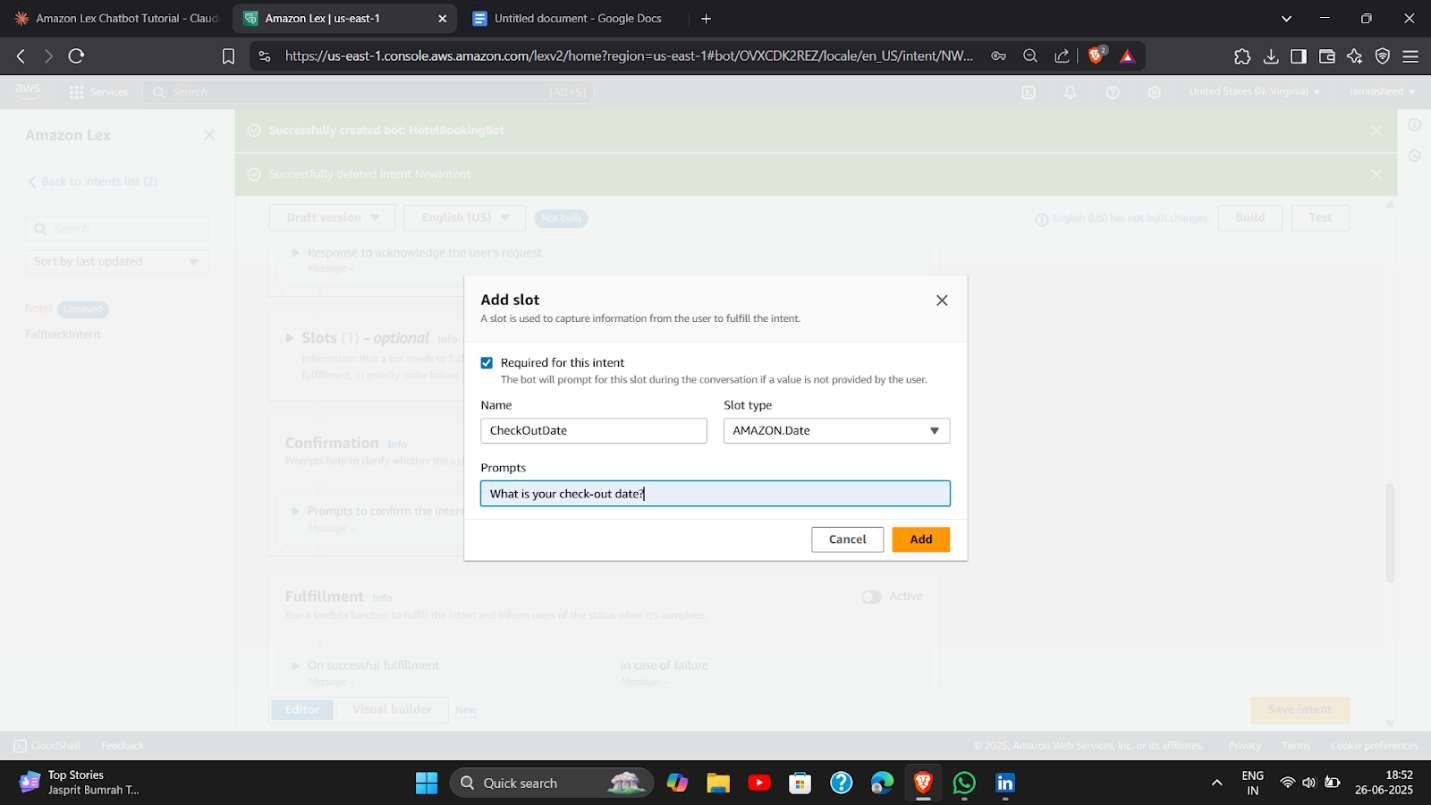


Slots → Add Slot → Name → Slot Type → Prompt Ex CheckInDate

CheckOutDate



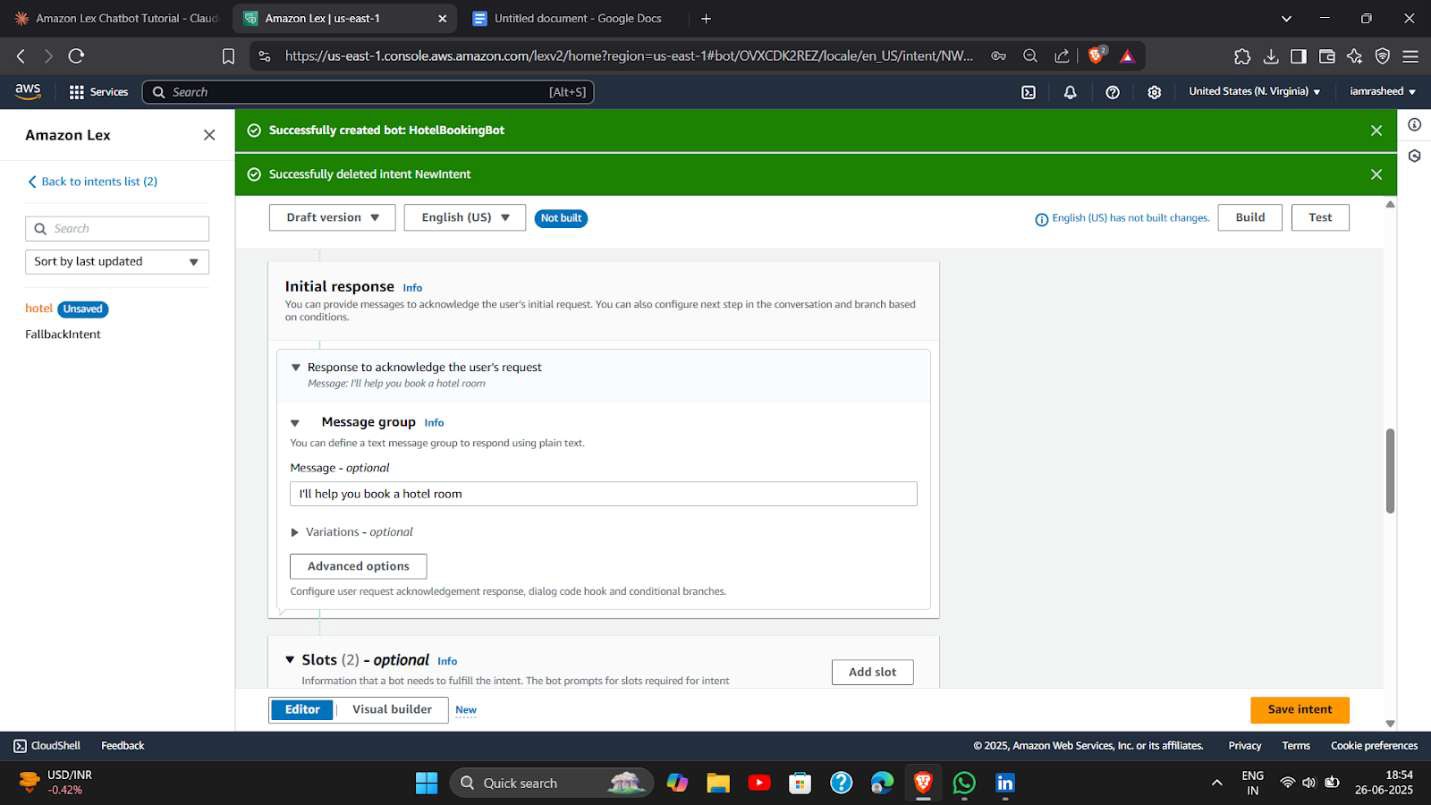
Create at least two slots, for example: CheckIn.. and checkOut.., each with its own prompt to collect necessary information from the user.



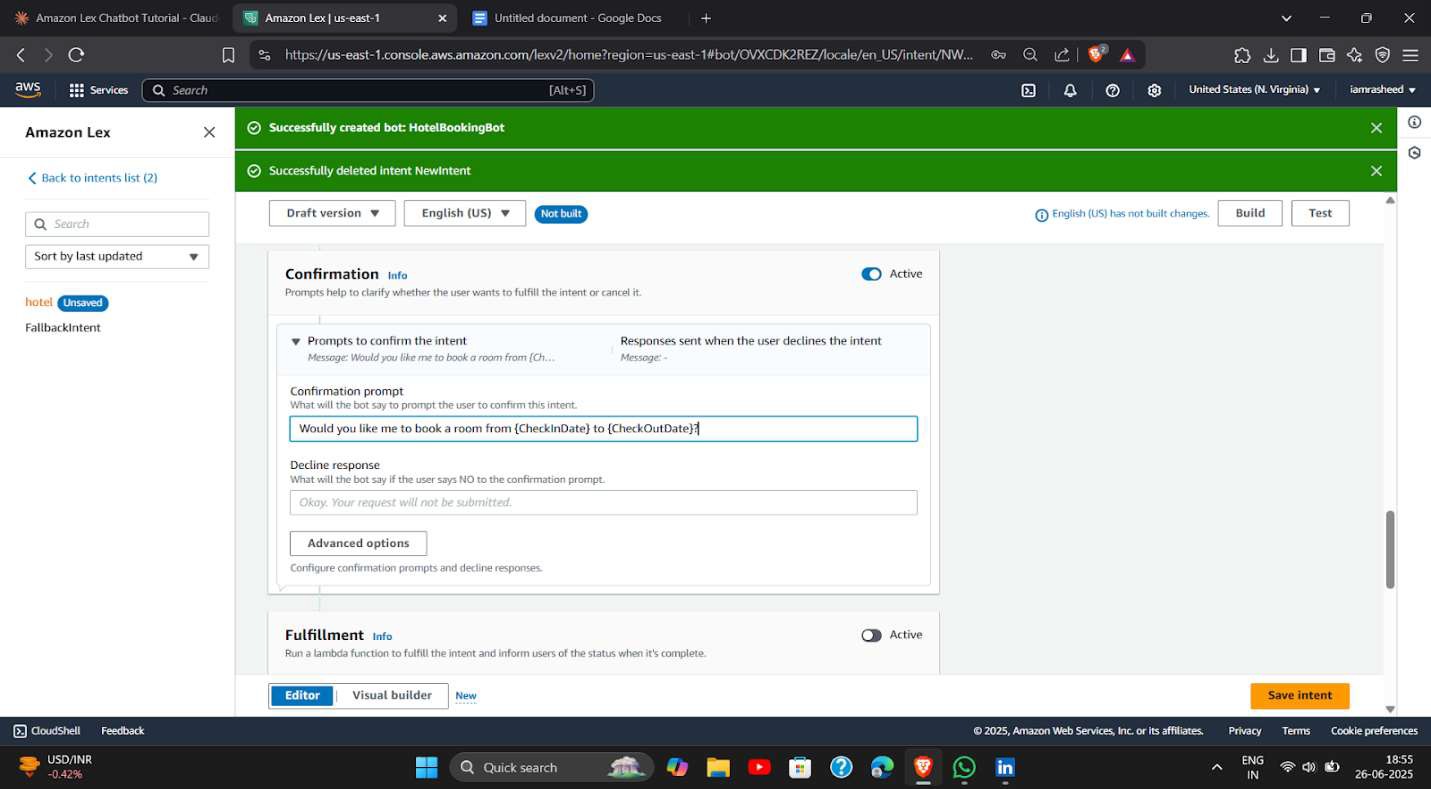
Slots are saved

Initial Response

Set an initial response that the bot gives when the intent is triggered (e.g., “Sure, I can help you book a hotel.”).

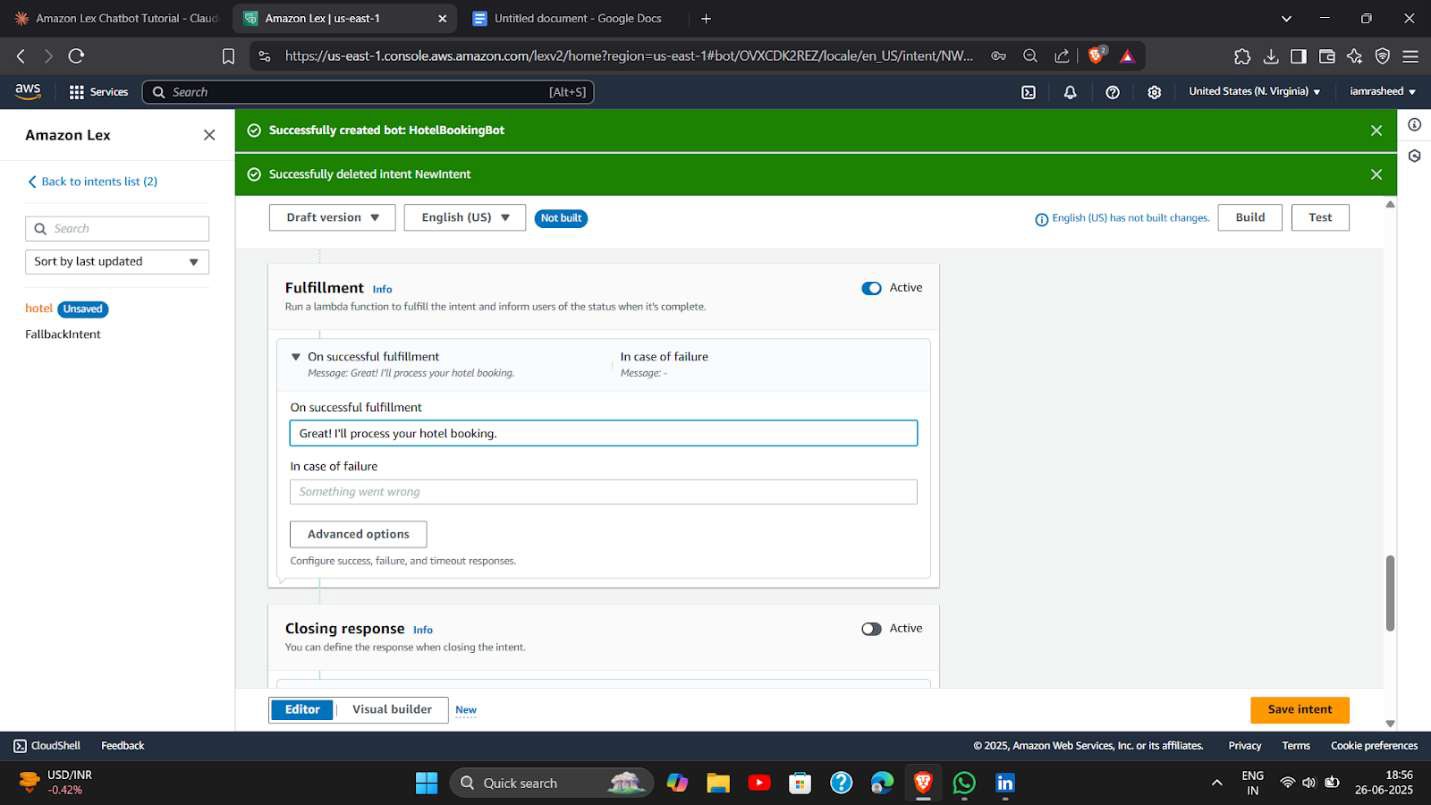


ConﬁrmationEnable conﬁrmation prompts to verify the user’s request before fulﬁllment (e.g., “Should I book a hotel



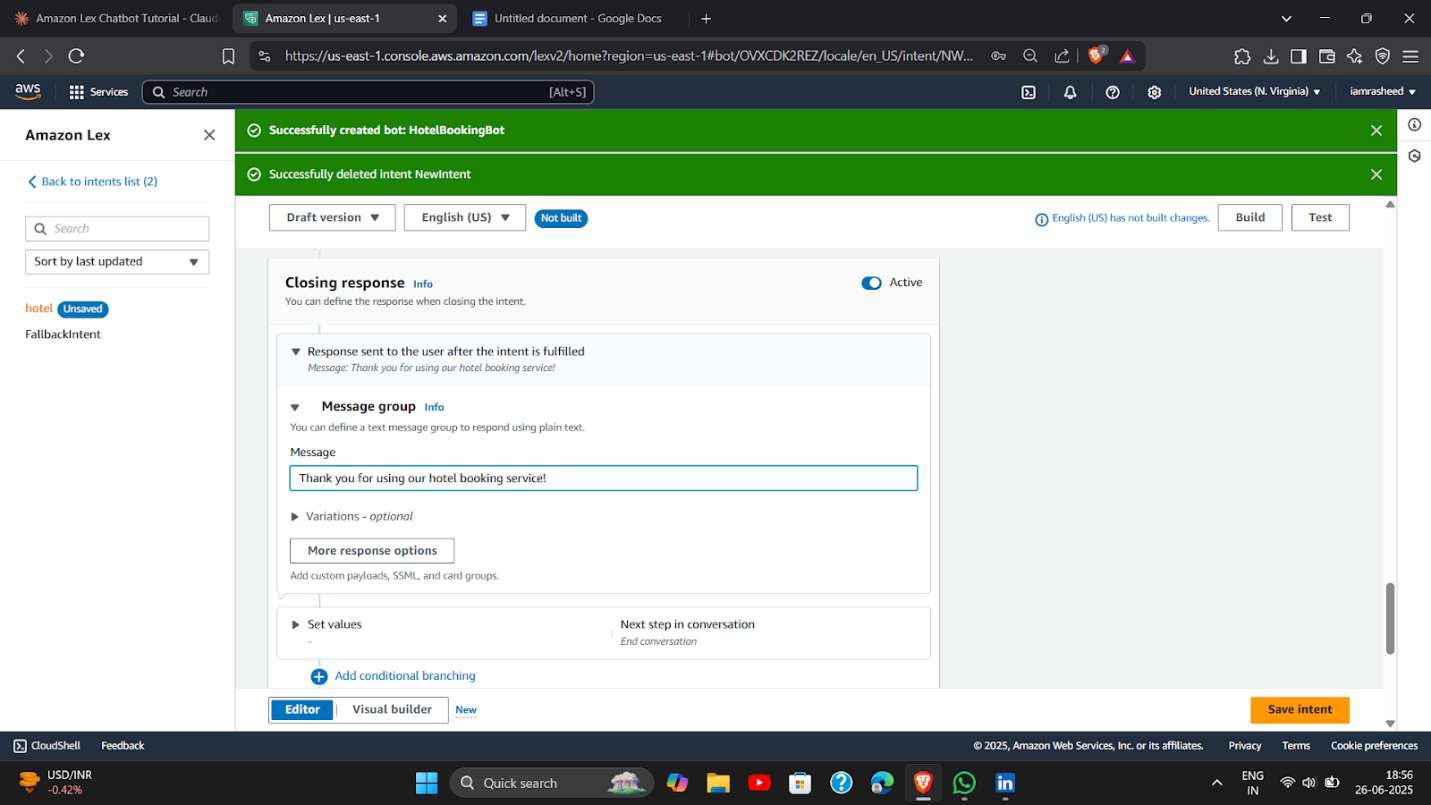
Fulﬁllment

Conﬁgure the fulﬁllment to trigger backend logic (Lambda or return response) after conﬁrmation, such as storing booking info or calling an API.

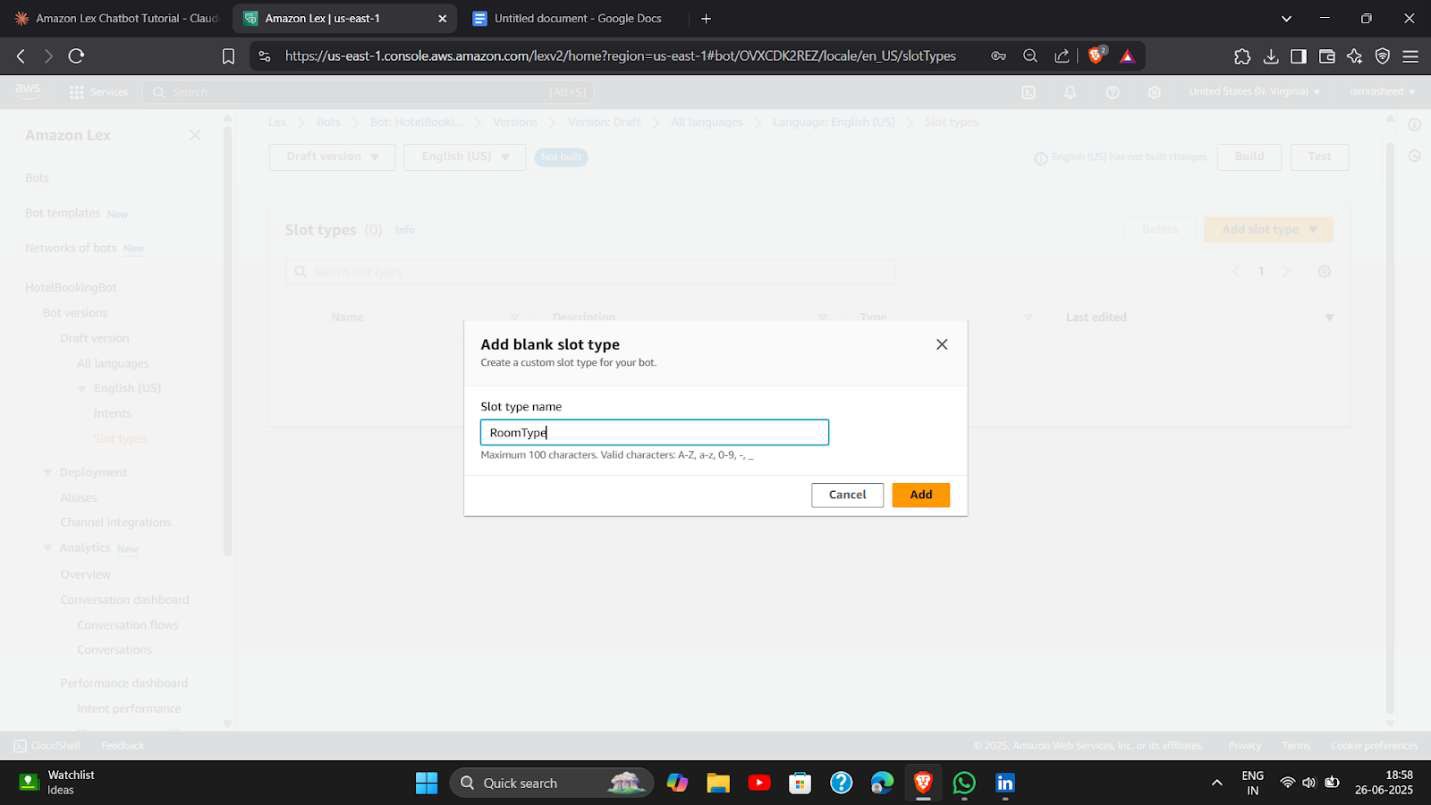


Closing Statement

Set a closing response that conﬁrms the completion of the task (e.g., “Your booking is conﬁrmed. Have a great stay!”).



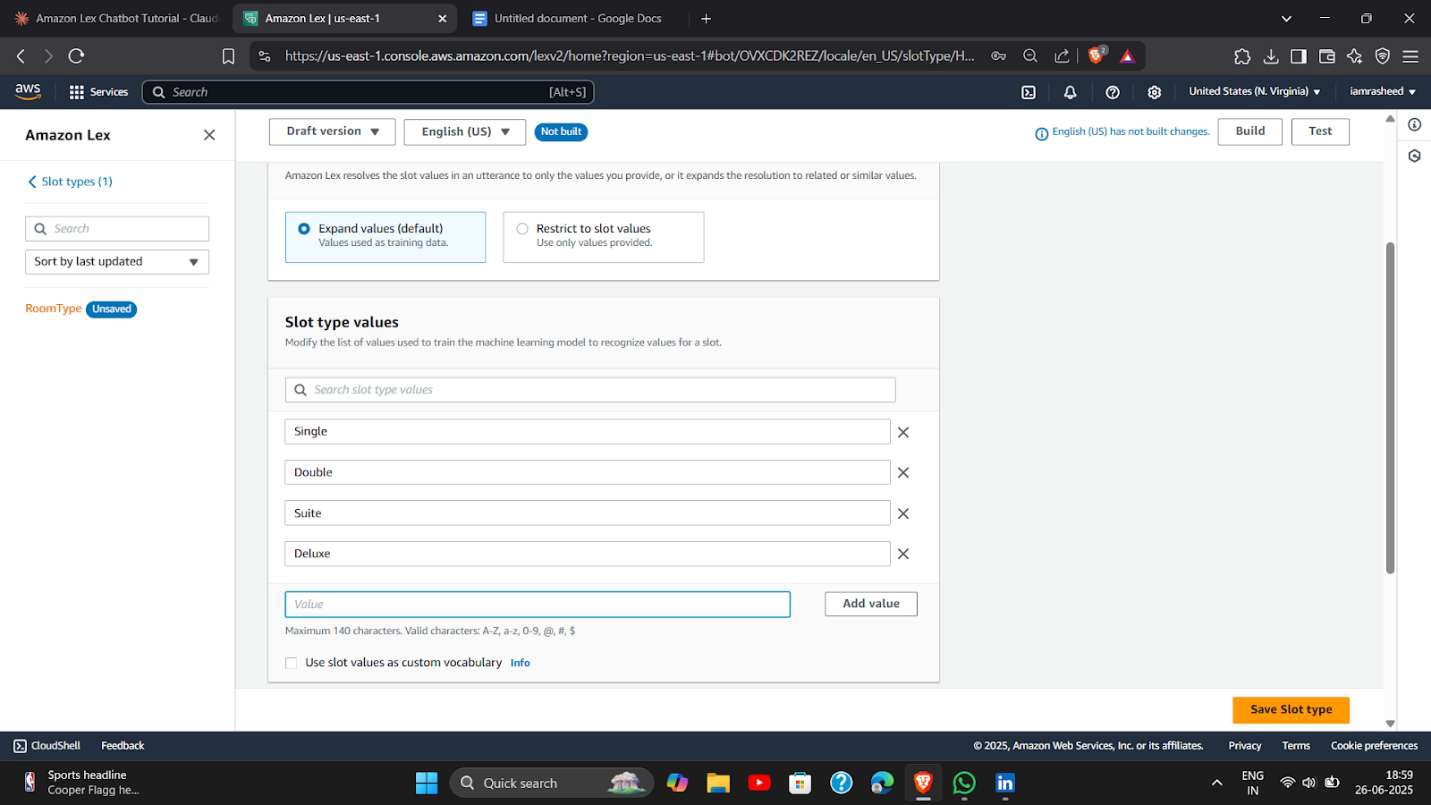
Bot → Slot Type → Add Slot Type Navigate to the Slot types section in the bot and click Add slot type to deﬁne a custom category of values.exRoomType



Slot Value Resolution → Expand → Add Slot Type Values → Save

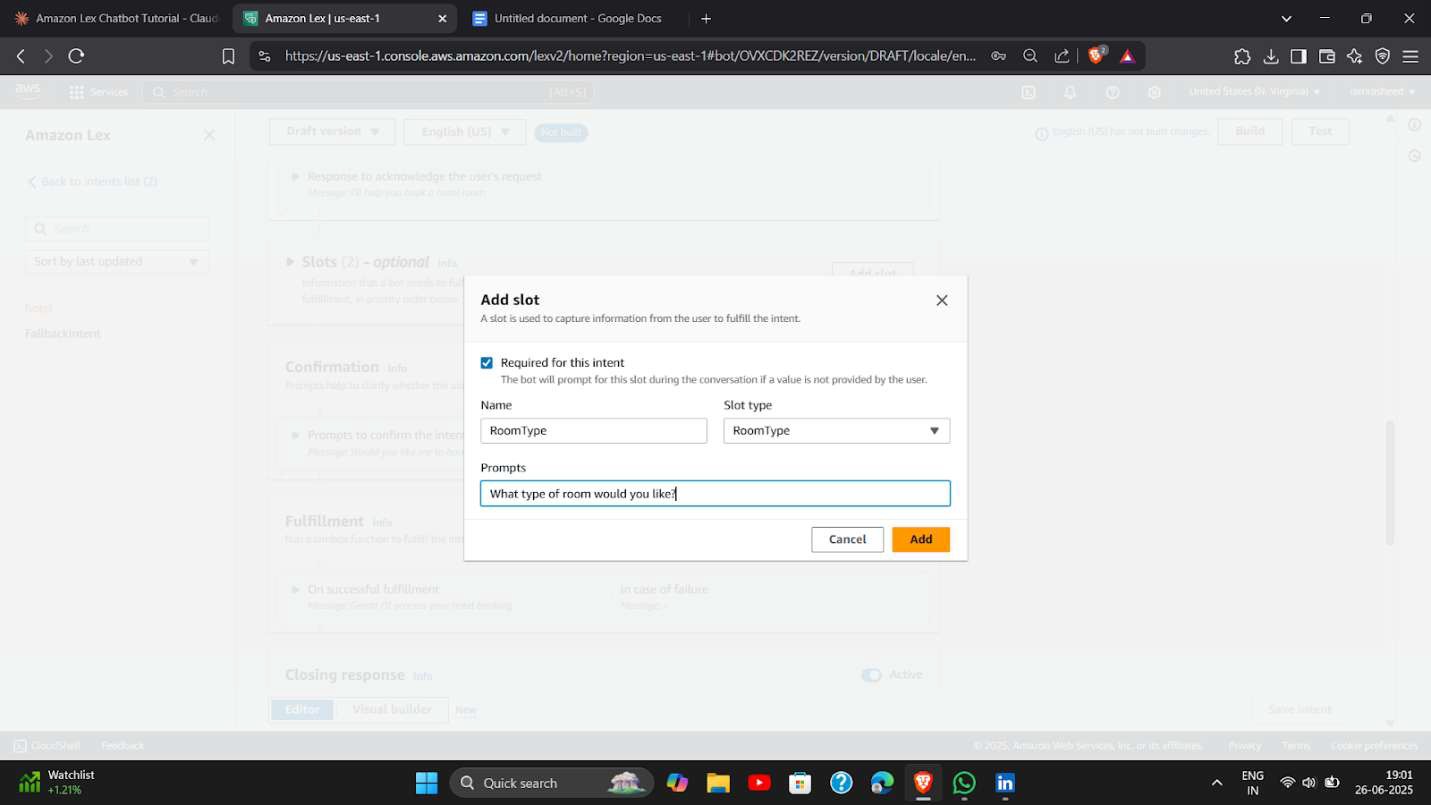
Expand the slot value resolution section and manually enter a list of values Add values

Single Double Suite Deluxe



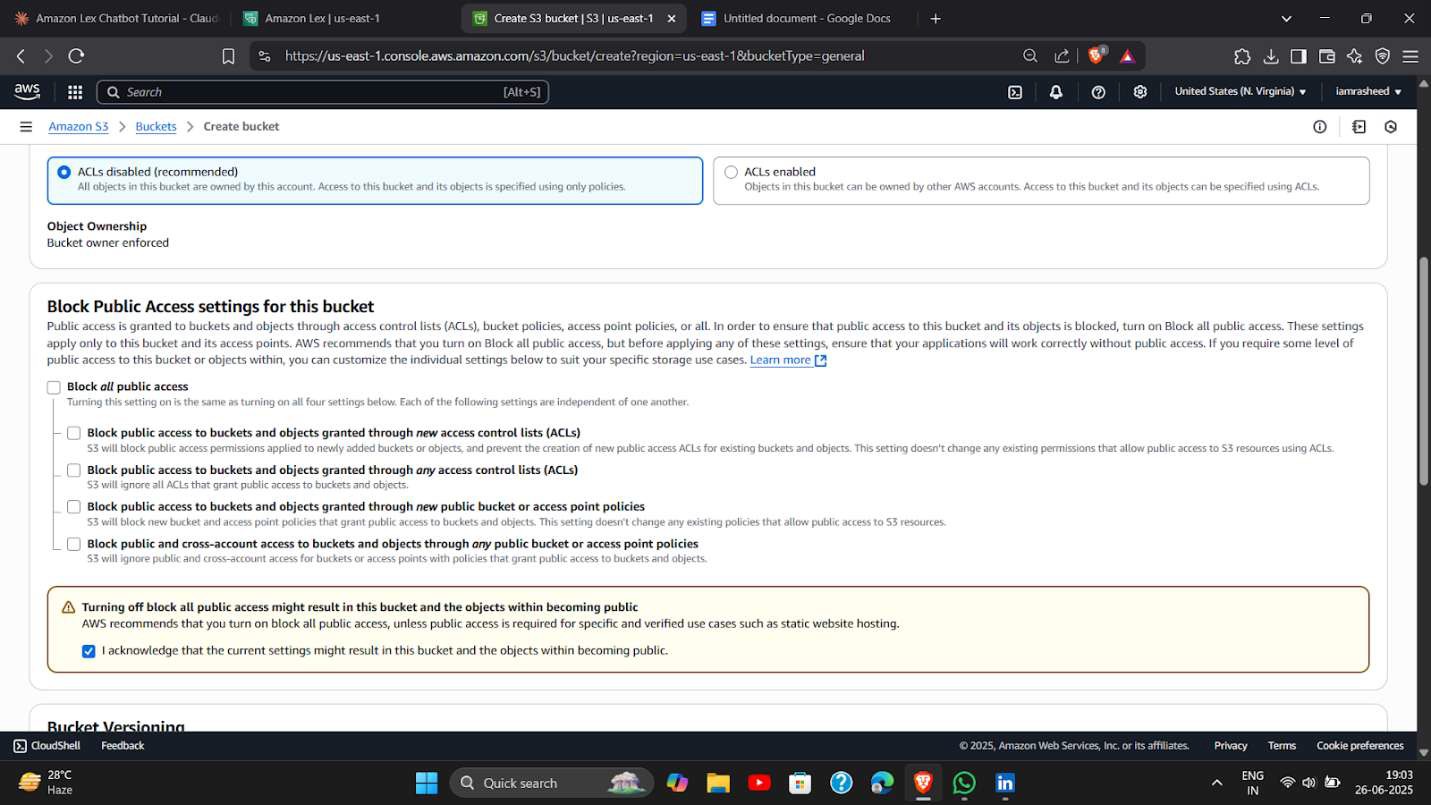
Go to the Intent → Add Slot → Add the Custom Slot Type

Return to the intent and add a slot that uses the custom slot type you just created

Go

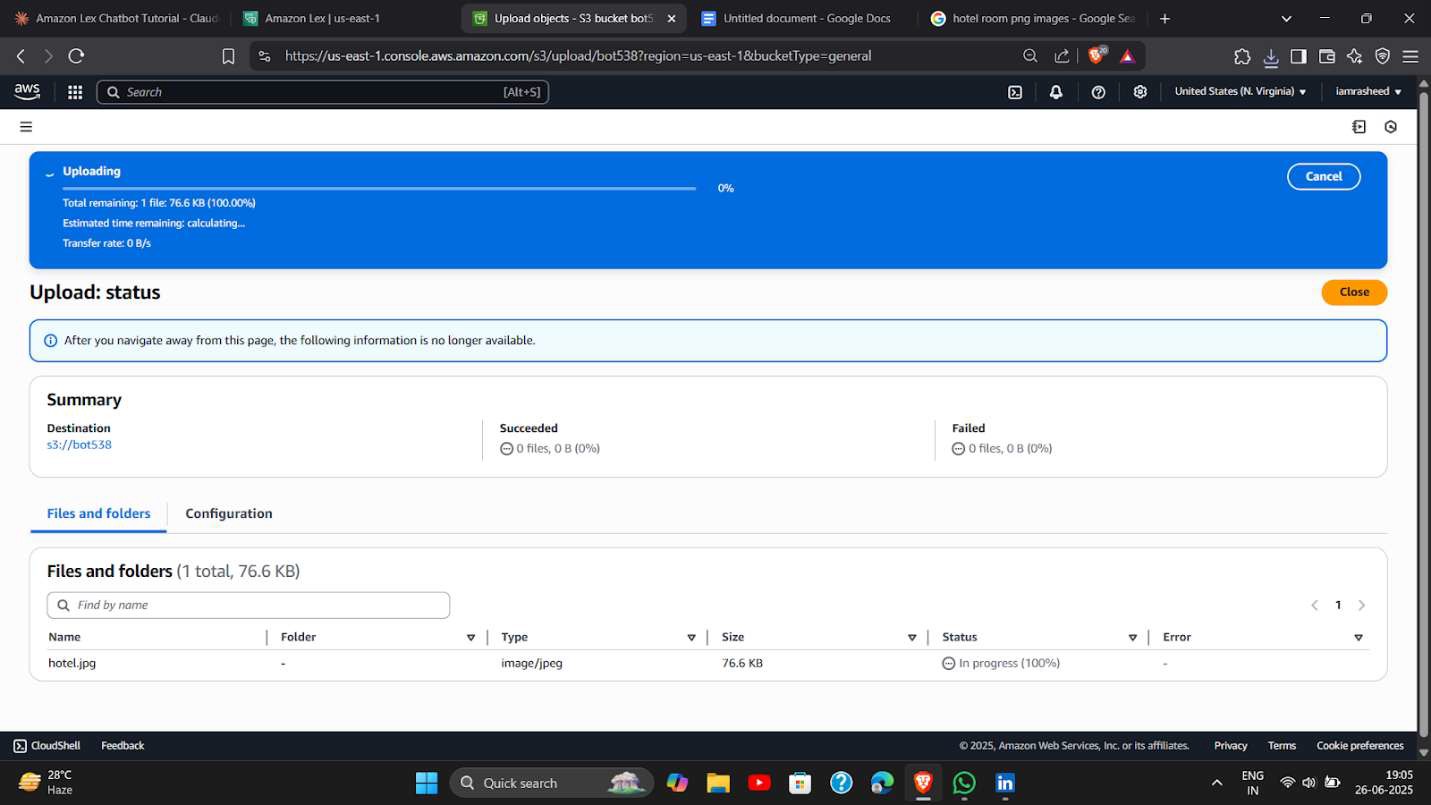
to S3 → Create Bucket

Go to the Amazon S3 service and create a new bucket where you will upload your images for use in the chatbot prompts.

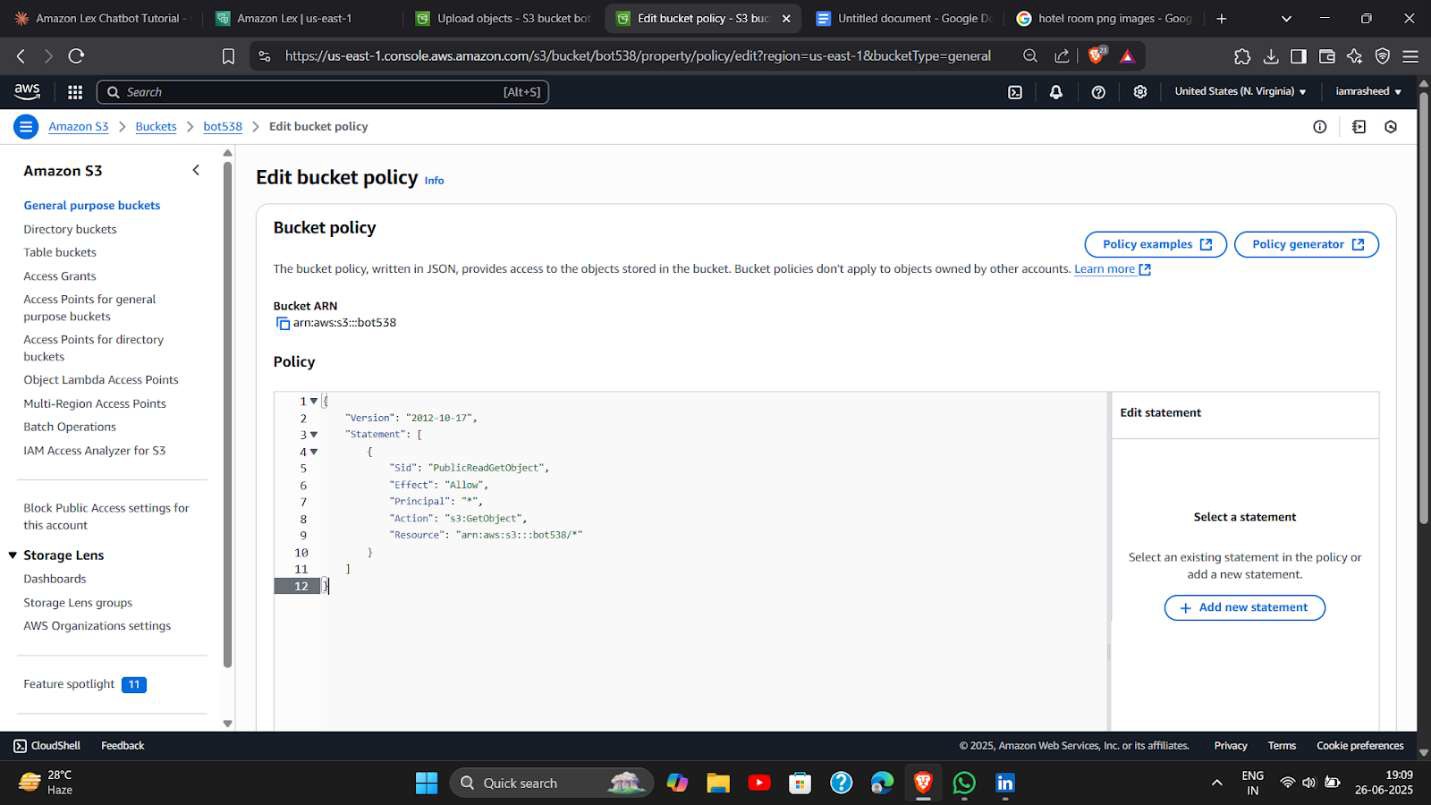
Add

an Image in the Bucket → Give All Permissions

Upload an image to the S3 bucket and make it publicly accessible by adjusting the permissions (ACL or bucket policy)



Editing bucket policy



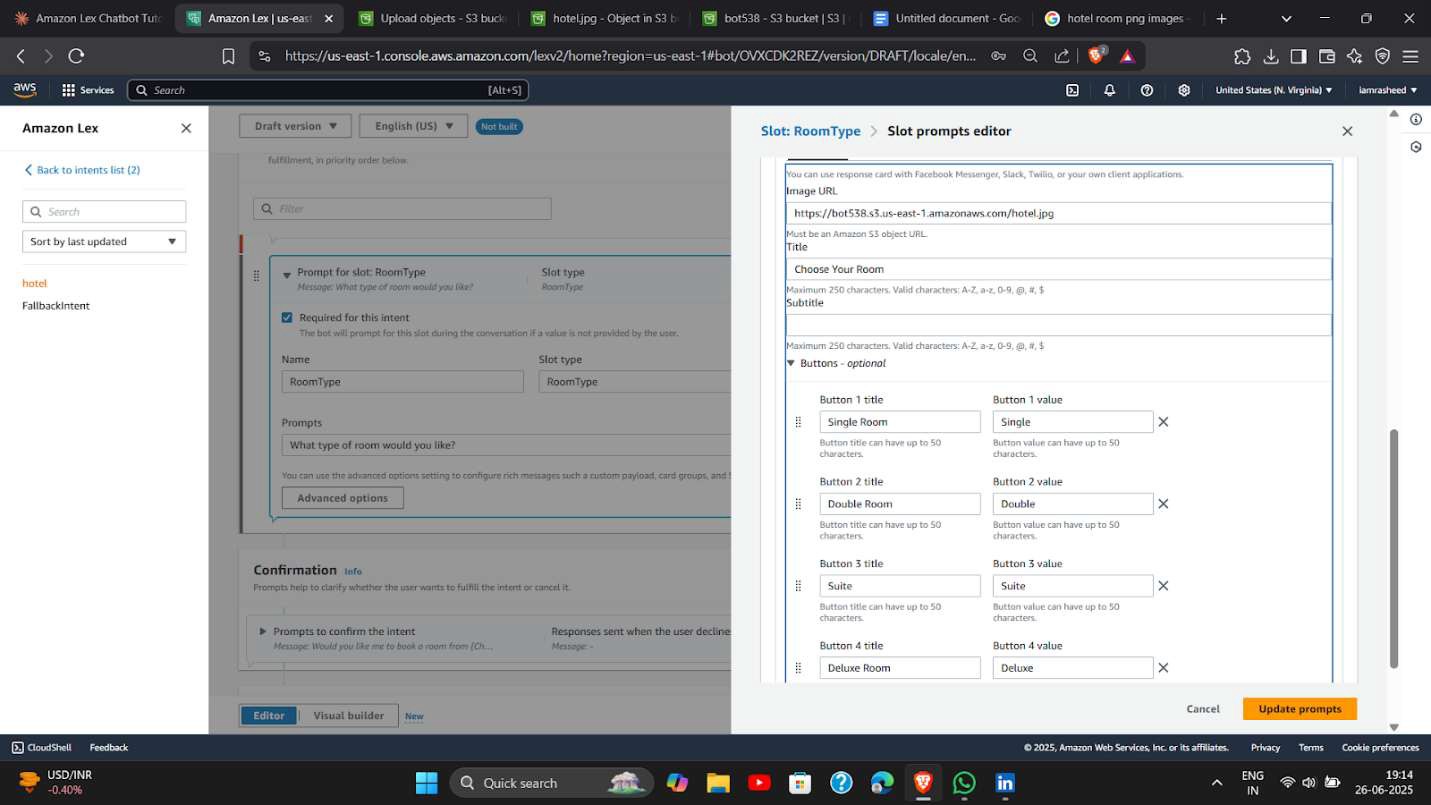
Click on the Slot → Advanced → Slot Prompts → More Prompt Options → Again Slot Prompts

Open the slot conﬁguration, go to Advanced options, select Slot prompts, and then choose More prompt options to enable rich card responses.Add → Add Card Group

In prompt options, click Add card group to design a visual card with an image, title, and buttons as a user- friendly prompt.Add the Image URL from Bucket → Title

Paste the public image URL from S3 into the card, then provide a meaningful title related to the image (e.g., “Deluxe Room”).Add Button → Update Prompt → Update Slot

Add buttons to the card (e.g., “Select Room”), update the prompt conﬁguration, and save the updated slot with this enhanced user interaction.



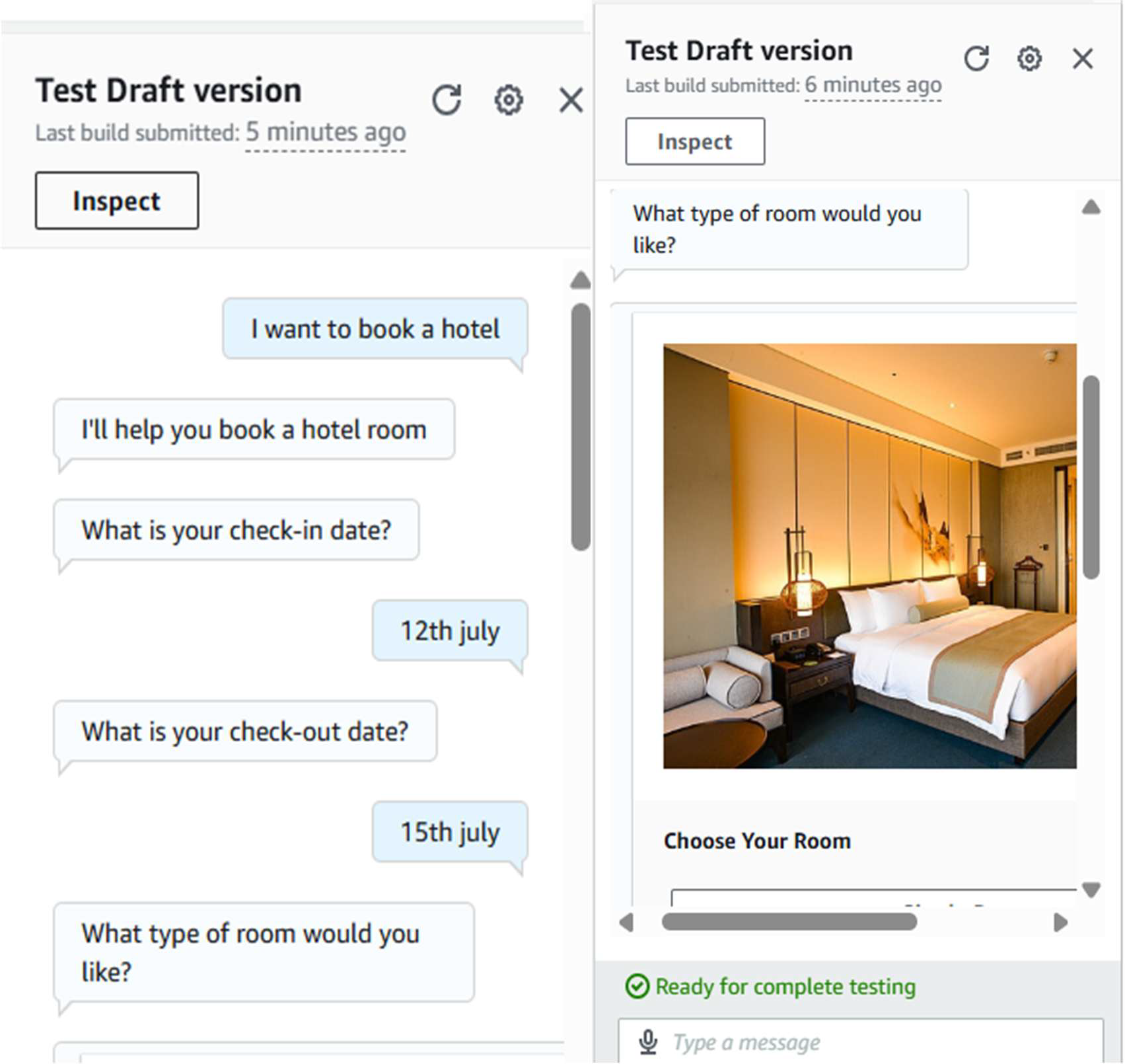
Fulﬁllment

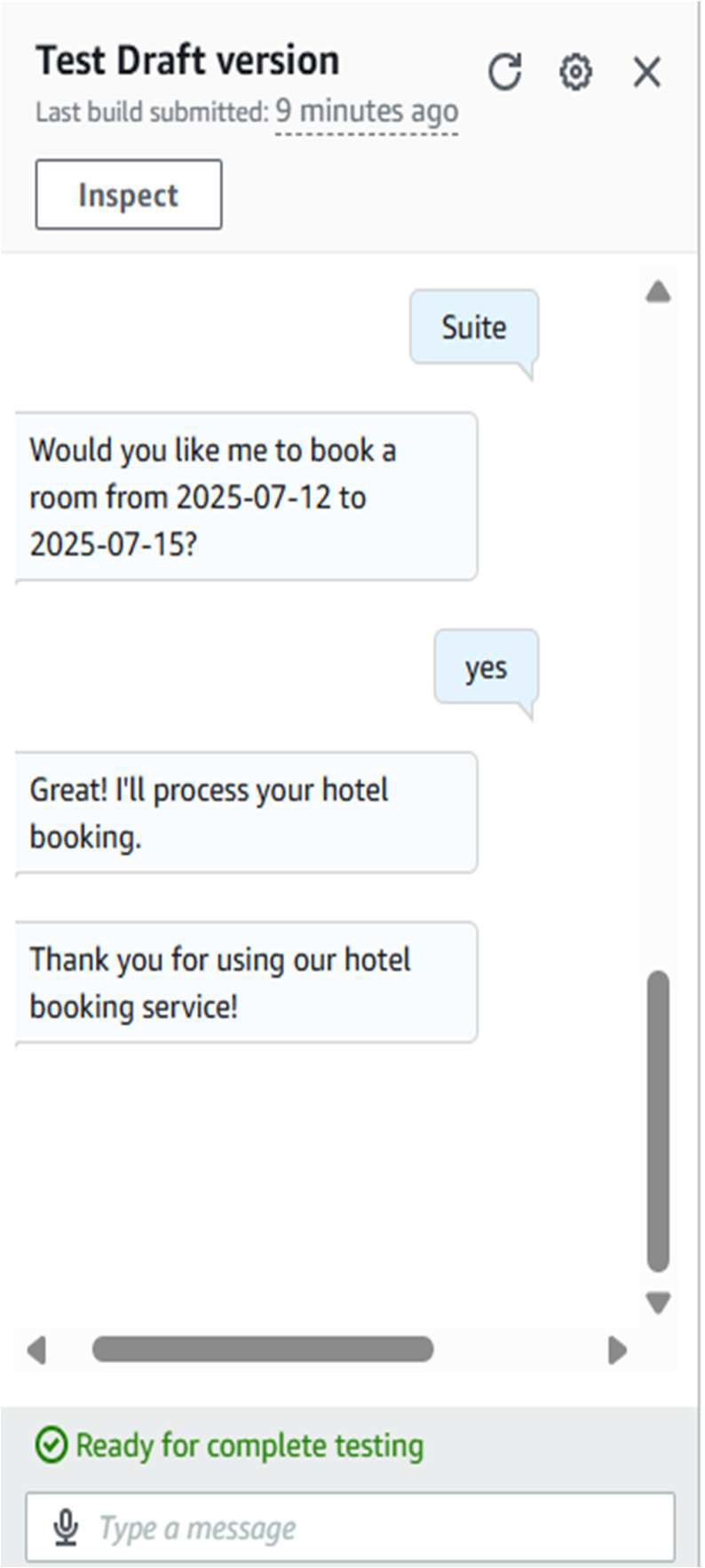
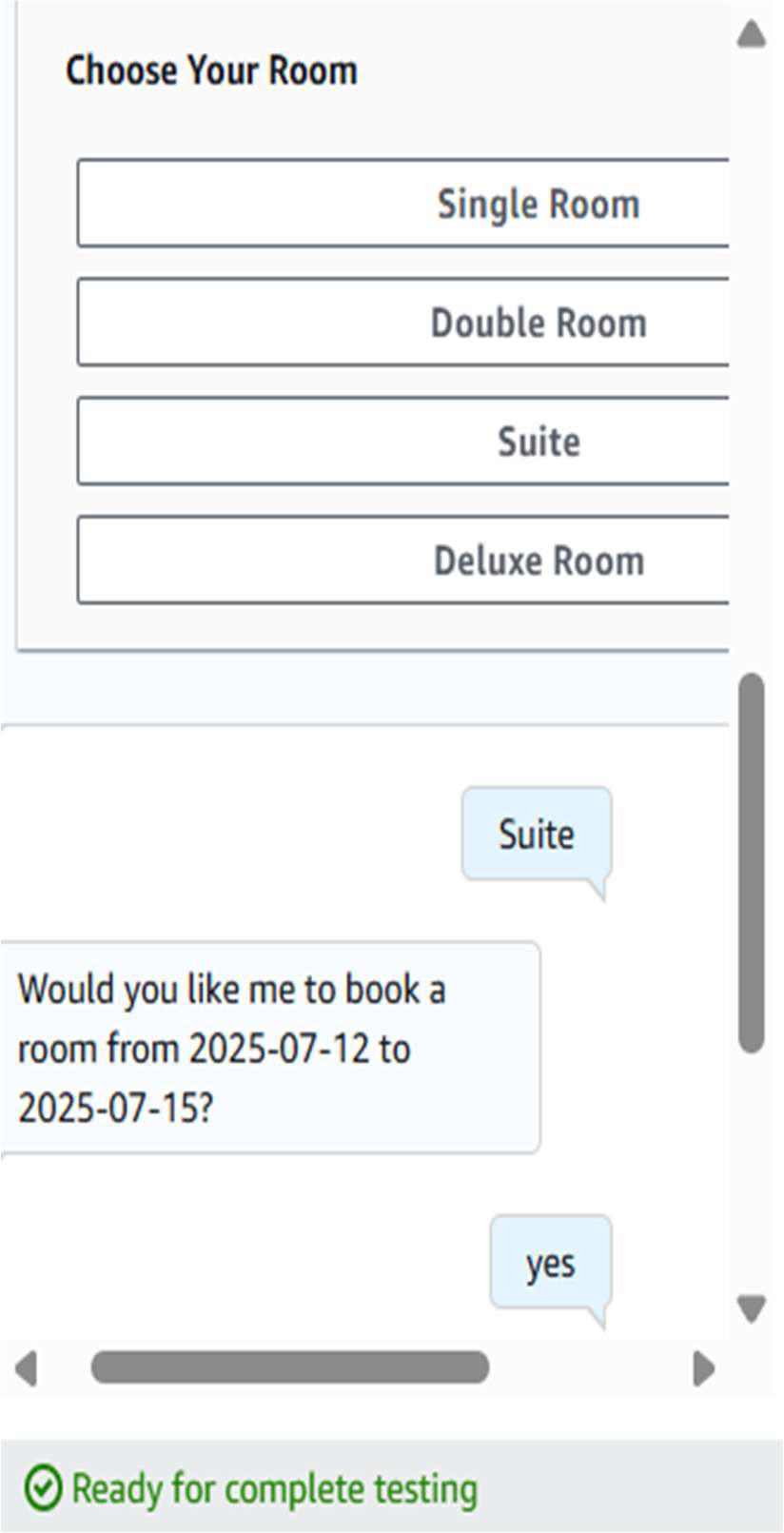
Set up the fulﬁllment section again to process the user’s choices using a Lambda function or return a static message based on slot values.

Build → Test

Click Build to compile your bot conﬁguration, then go to the Test tab to interact with your bot and verify its responses and logic.

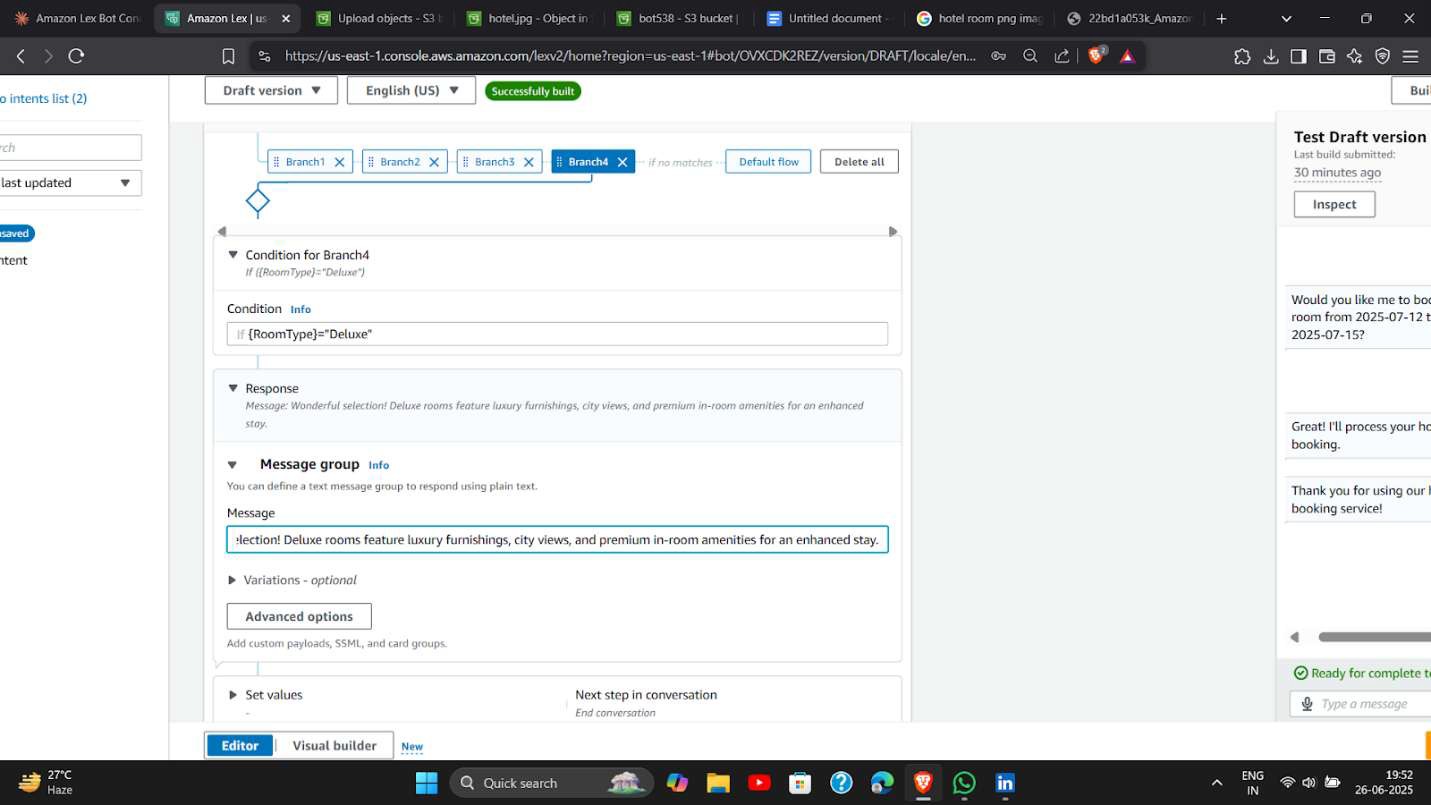
Now communicate with the chatbot You will receive responses as followed





Scroll Down → Conditional Rendering → Add Condition → Add Message

Inside an intent or slot, scroll to Conditional branching, click Add condition, and specify logic to show dynamic responses (e.g., if city = Delhi, show special message).



Add Branch for "Suite" Room

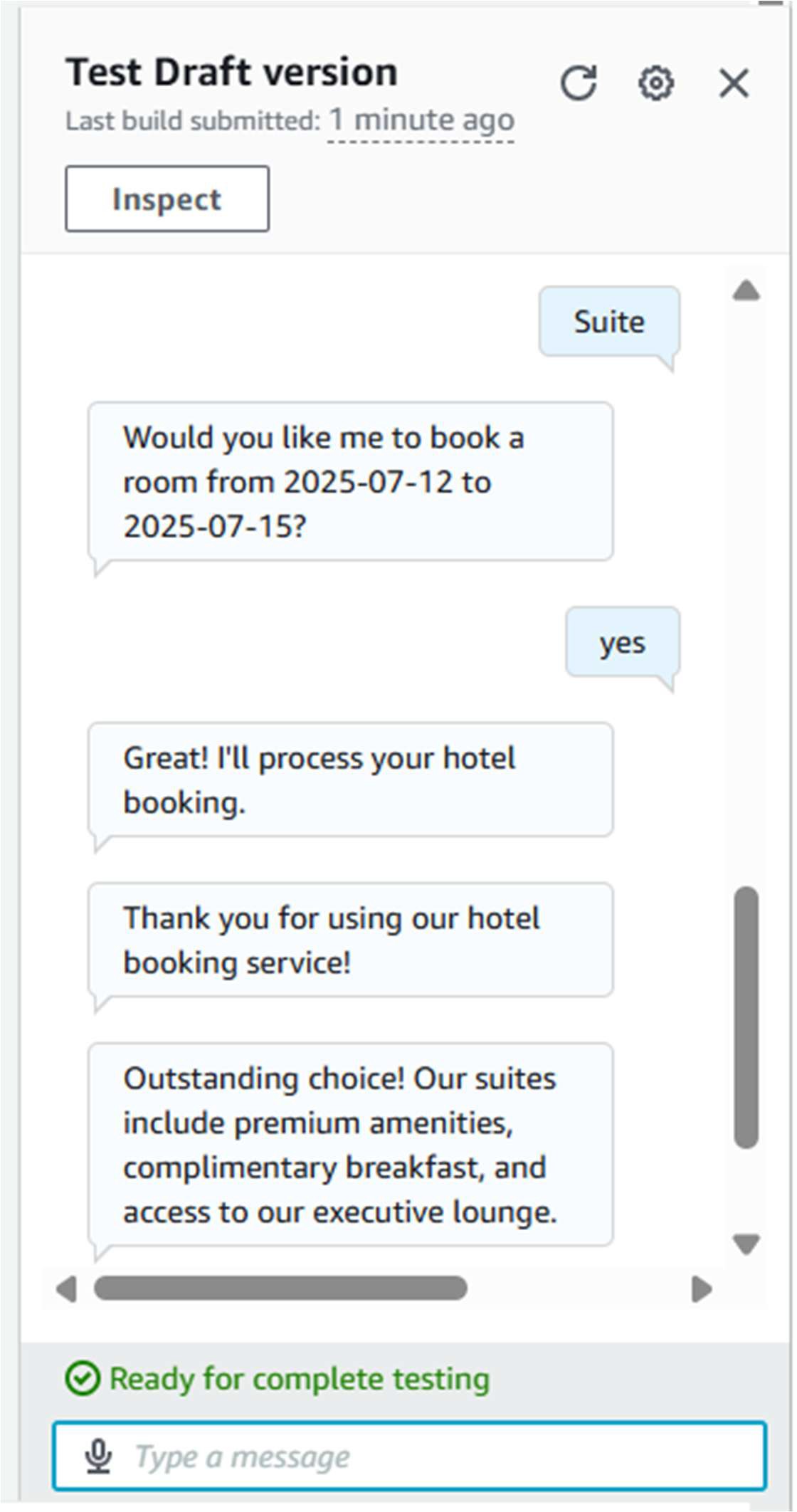
1. Click "Add branch" again
2. In the condition ﬁeld, type: If {RoomType}="Suite"
3. In the Message ﬁeld, type:

Outstanding choice! Our suites include premium amenities, complimentary breakfast, and access to our executive lounge.

Similarly we add branch for remaining options

Build - Test

Rebuild the bot after adding conditions, and thoroughly test it in the chatbot UI to validate all ﬂows and slot resolutions.



After adding conditional rendering

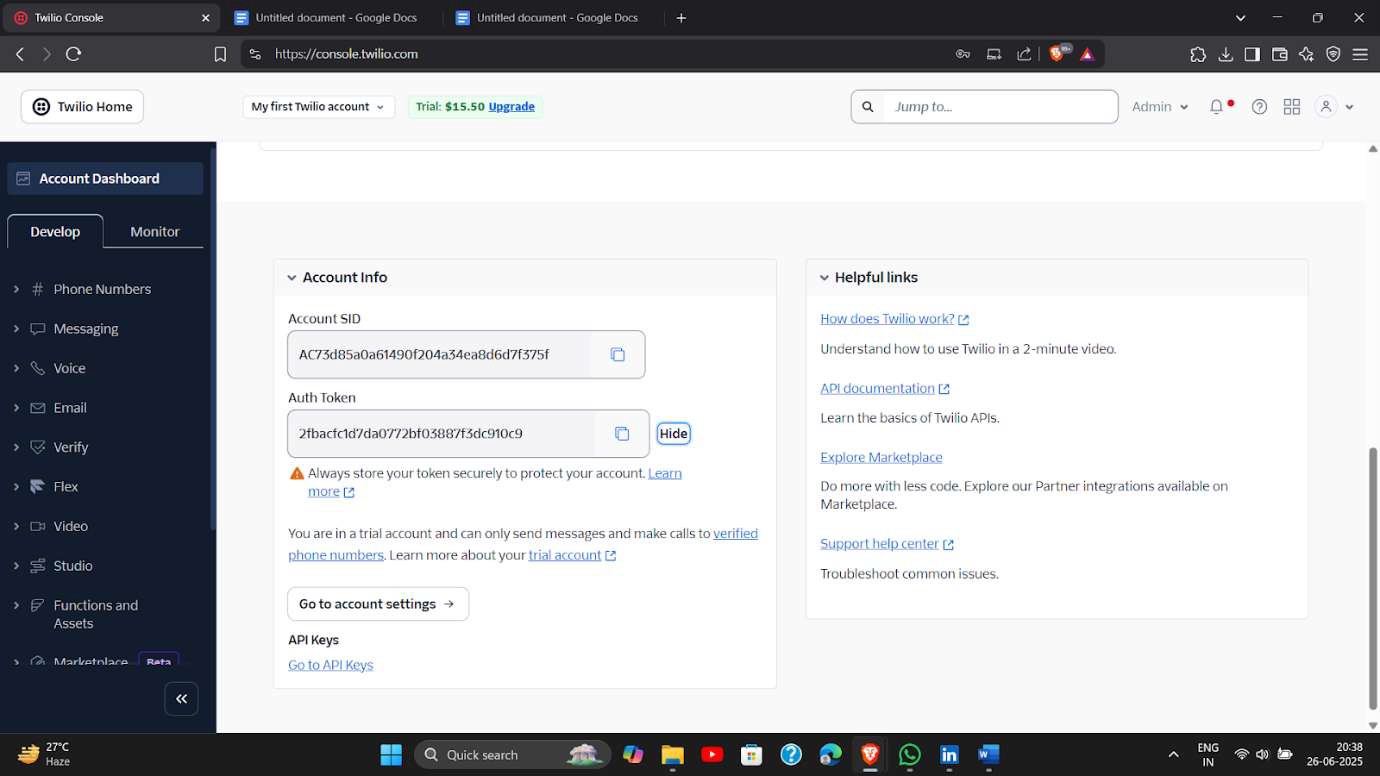
The conﬁrmation is given based on selection of RoomType

## Connect Amazon Lex to WhatsApp via Twilio

Amazon Lex chatbot is already ready, let's connect it to WhatsApp using the direct channel integration method (much simpler than Lambda approach).

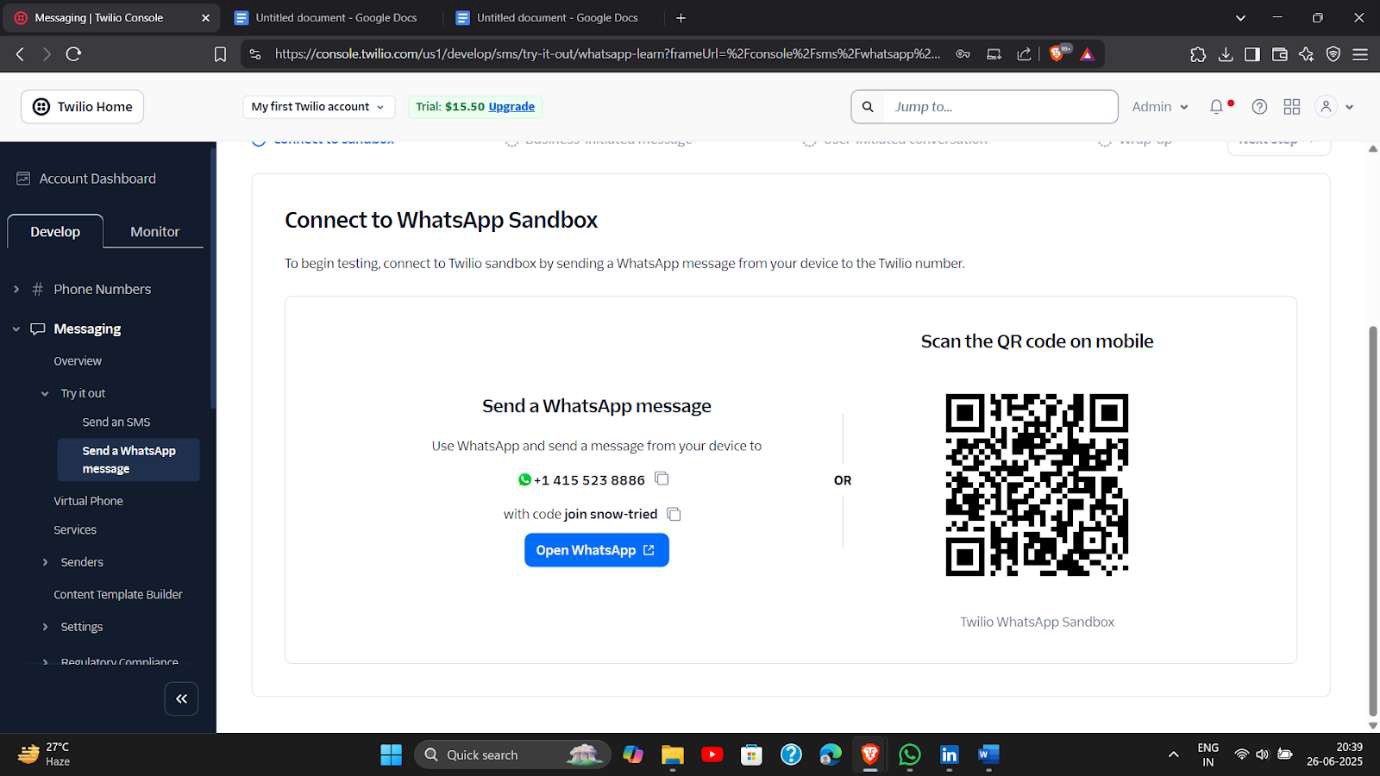
Get Your Twilio Credentials After veriﬁcation, go to Twilio Console Dashboard Important: Copy and save these values: Account SID (starts with AC...) Auth Token (click the eye icon to reveal)

Keep these safe - you'll need them in the next steps!

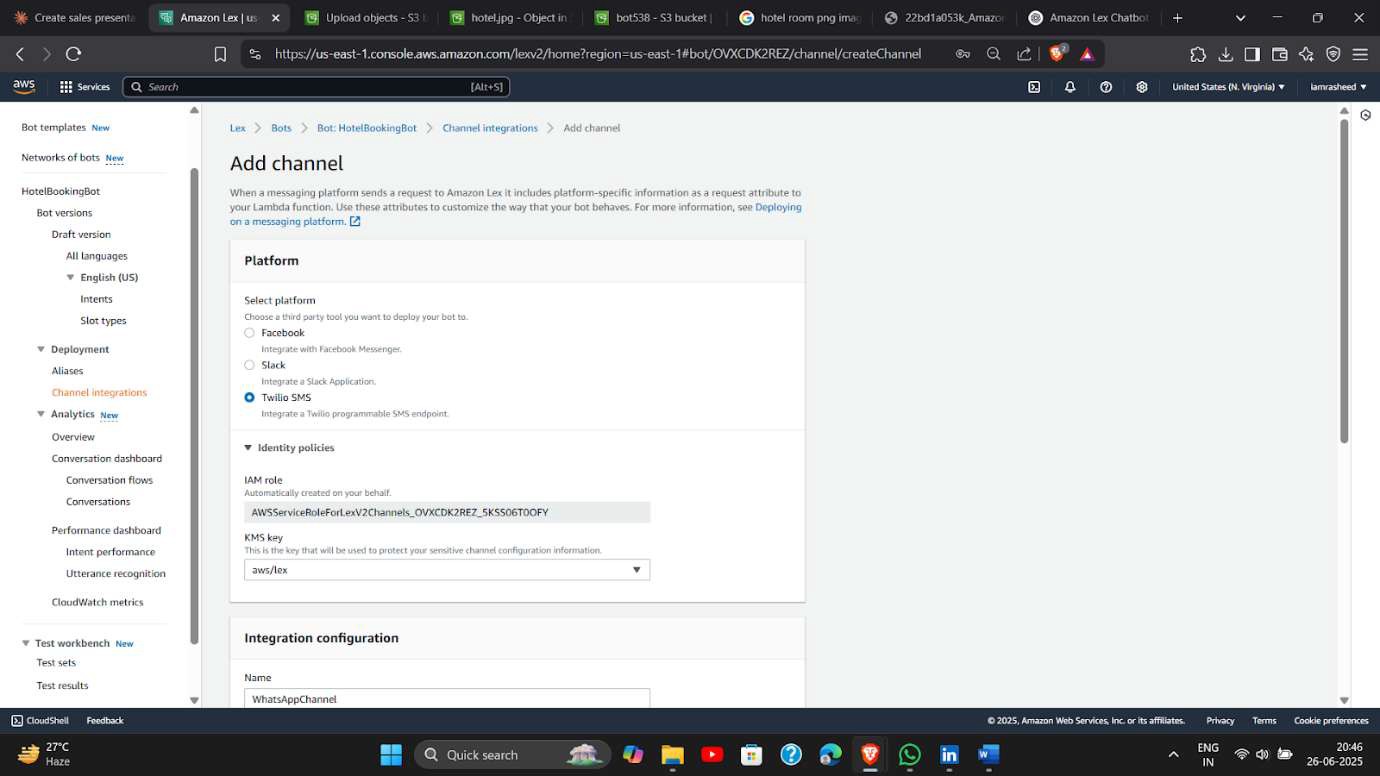


Access WhatsApp Sandbox In Twilio Console,

go to "Messaging" → "Try it out" → "Send a WhatsApp message"You'll see the WhatsApp Sandbox page with:A sandbox phone number (like: +1 415 523 8886)A join code (like: "join neighbor-brass")

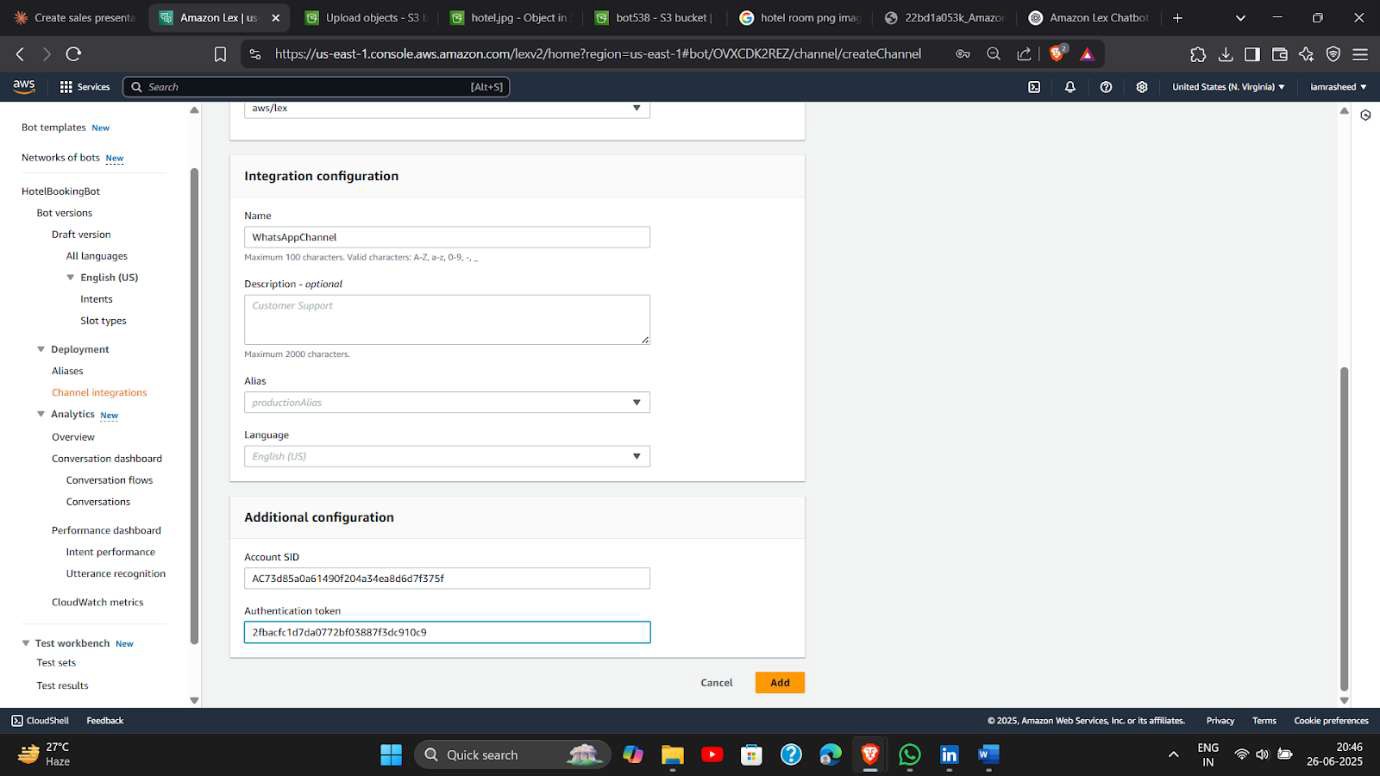


Add Twilio SMS Channel In your bot console, look for "Channels" in the left sidebar Click "Channels" Click "Add channel"

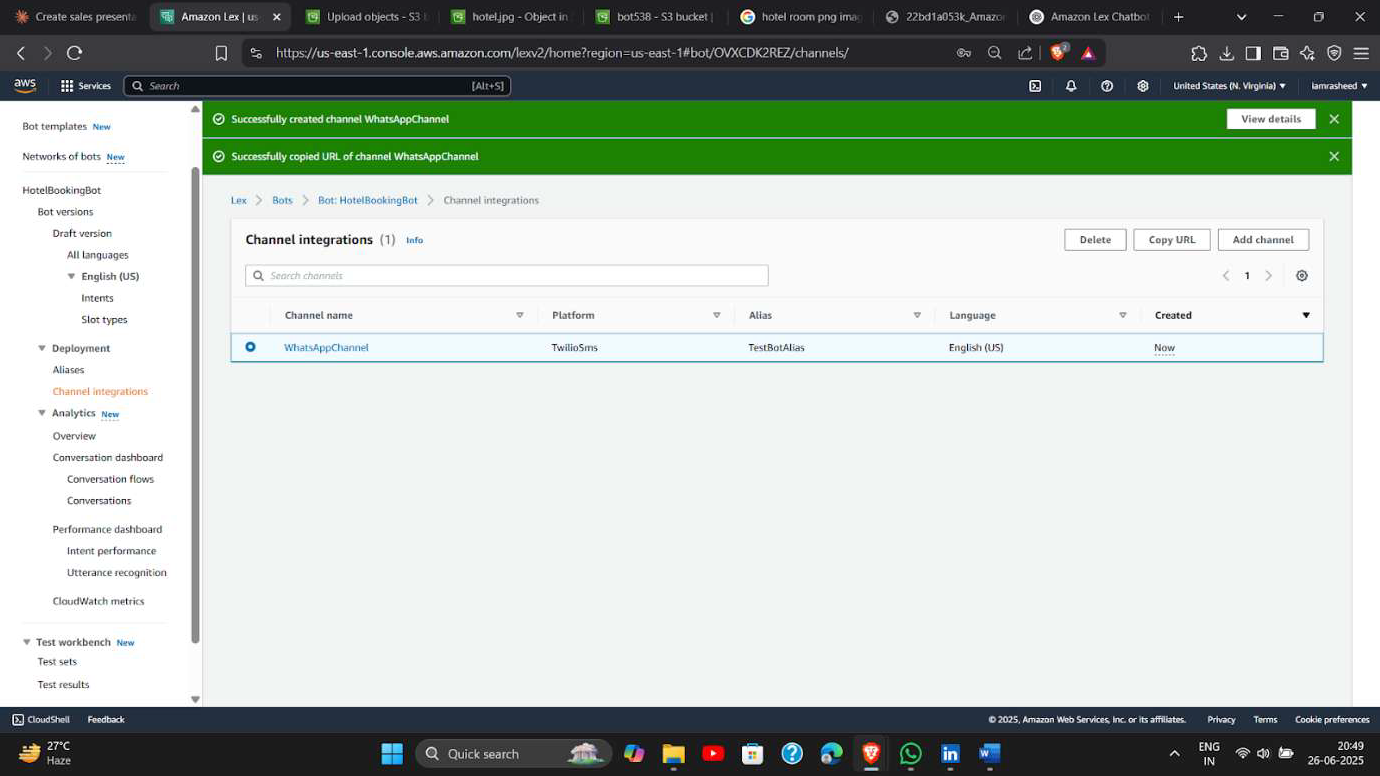


Add the account ssd

And auth token from the above steps

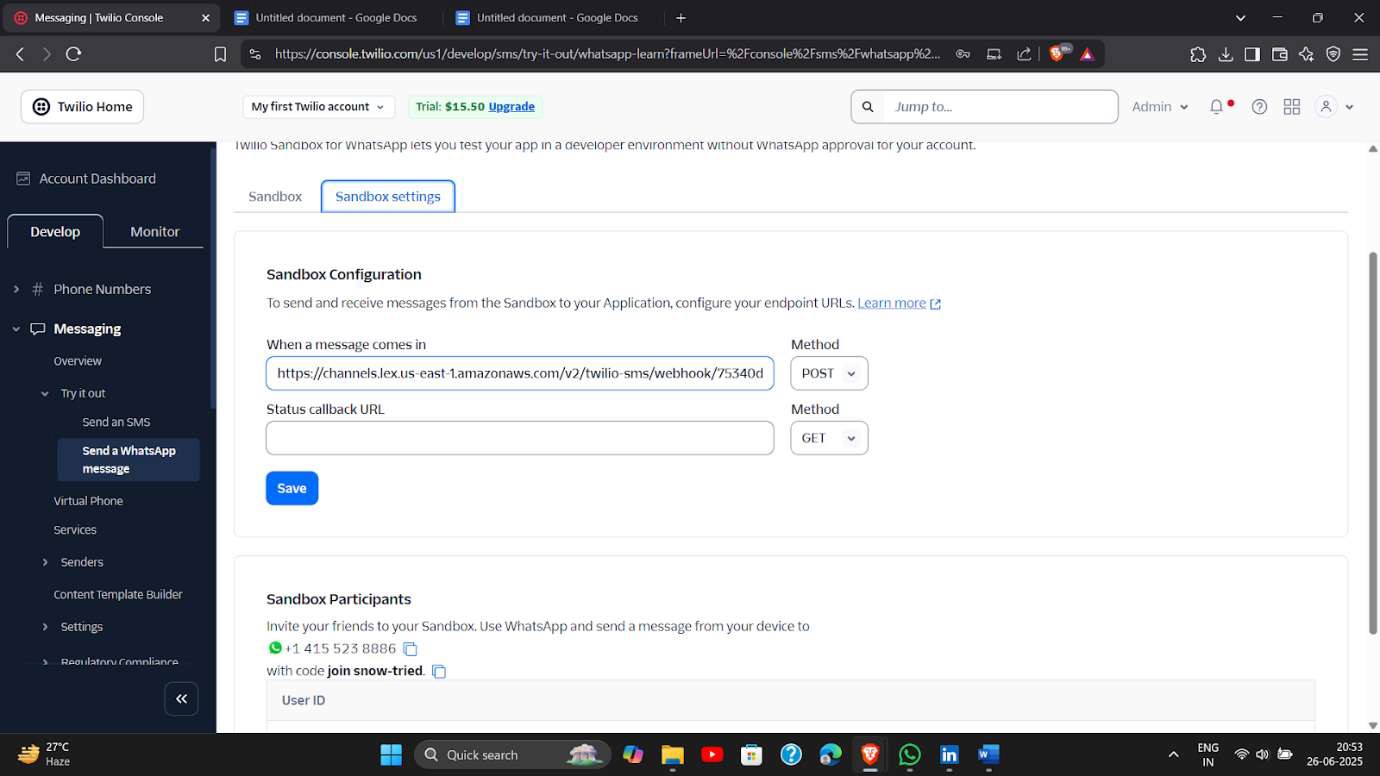


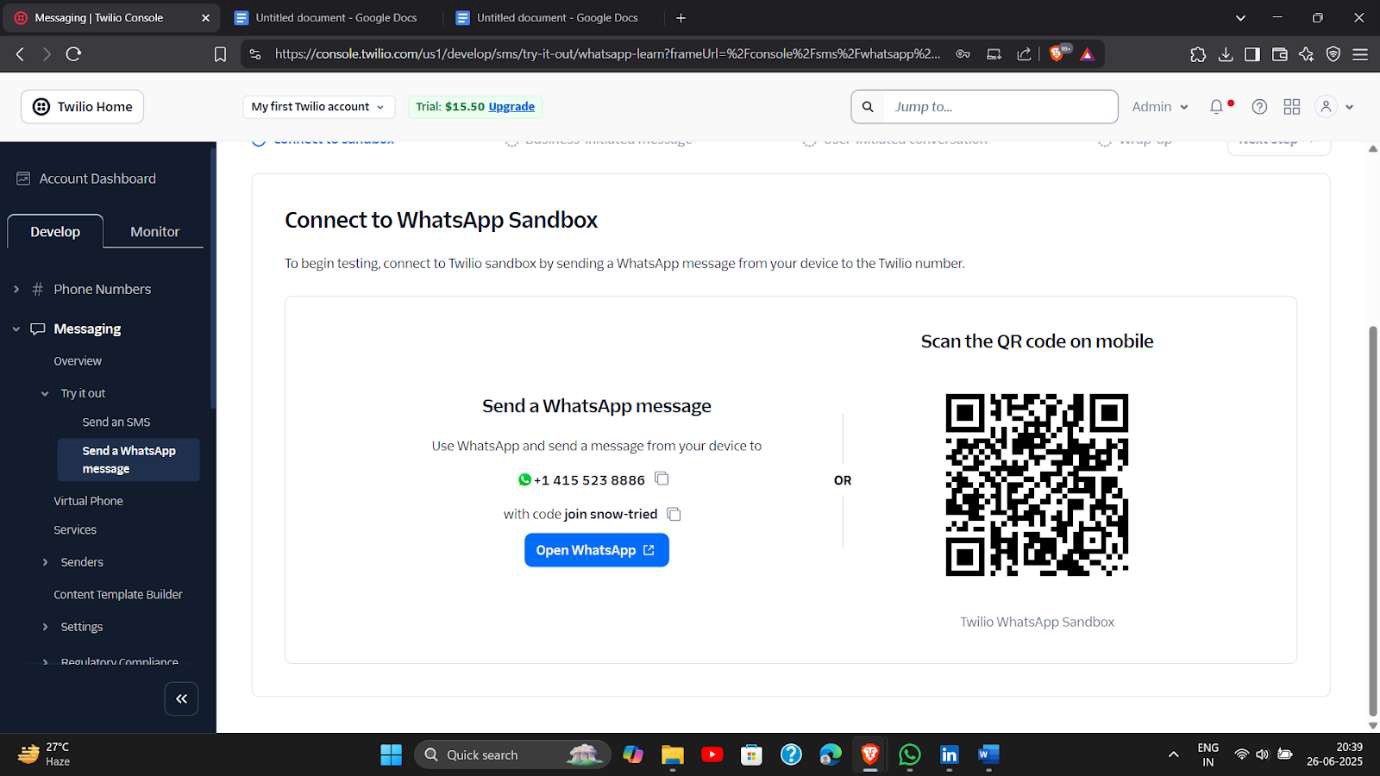
Select the channel and copy url



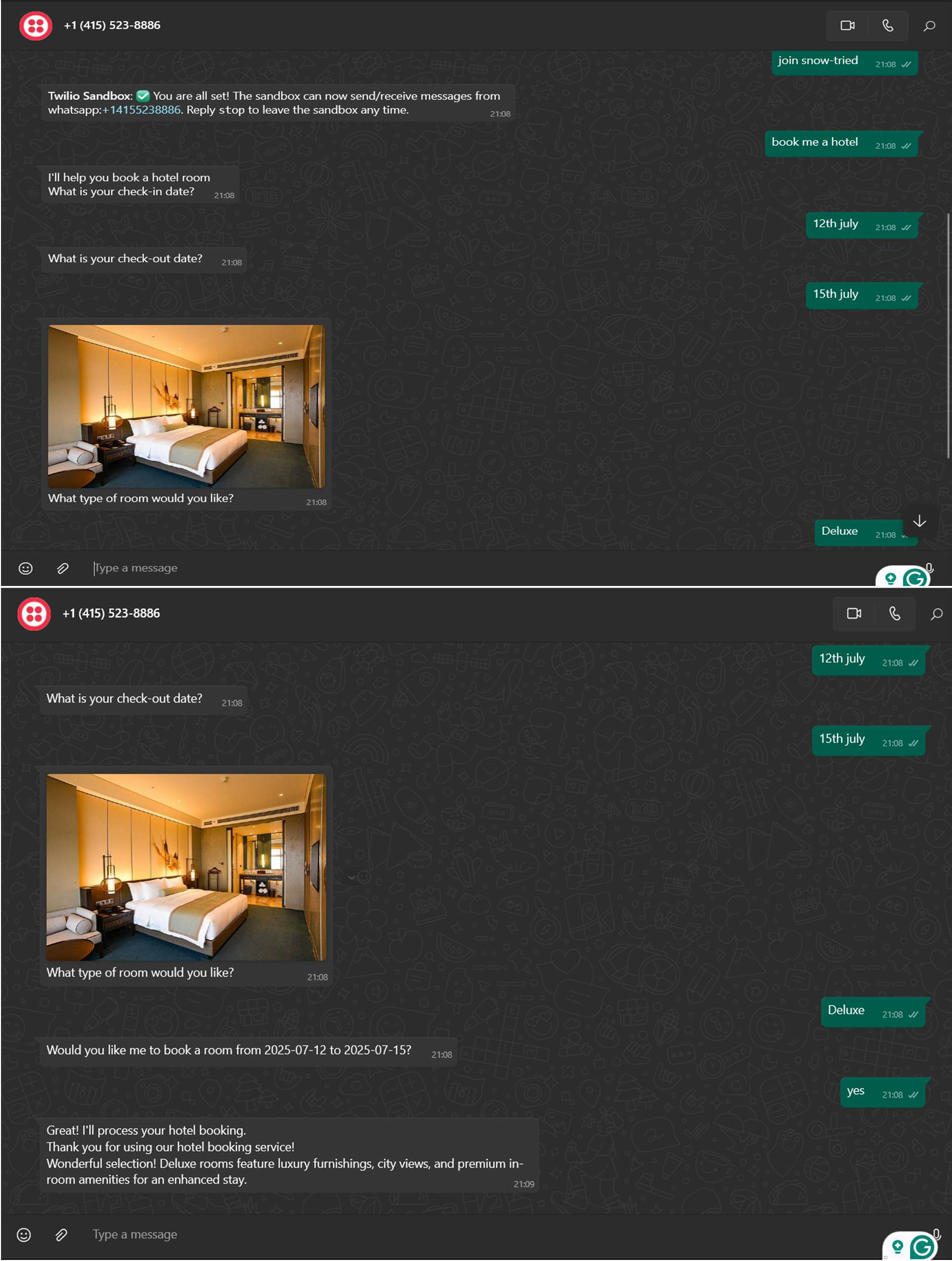
Paste the url here in twillo Sandbox settings

Make sure the method is post



Click on open Whatsapp (laptop)or scan qr(MOBILE)

And then send the message as your chatbot will read You can see you aws lex running here



WhatsApp sandbox