Chat Bot Development

Agenda

Business Use-case
Technical Methodologies
Execution Steps
Chat Bot From Scratch

Chaitanya Patil

Deliverables

Chatbot using RASA Framework

Work In Progress Chatbot Built from scratch

Business Use Case

Helps the customers to reduce the wait time.

Guaranteed table alotment.

Enhances Customer experience

Smooth and Quick Booking

Assumptions:
The Customer can book for today, tomorrow and day after only.

Multiple bookings not allowed at same time

No Restrictions on number of people allowed per booking

Key Features in the designed Chat-Bot

1. **Chat bot can reserve the table using a single sentence from User**

Ex: User: Reserve a table for Tomorrow at 3 PM.

2. **Chat bot can reserve the table in a conversation flow.**

Ex: Bot: How can I help you
User: Need to book a table
Bot: For what time do you want me to book
User: 3PM
Bot: For when do you want me to book
User: Today

Key Features in the designed Chat-Bot

3. **Chat bot can Delete the existing Bookings**.

Ex: User: Delete the reservation

Bot: Please enter the Booking ID

User: 12344566

Bot: reservation Cancelled

4. **Chat Bot does not allow multiple Bookings at same time slot.**

Features

Scenario A : User Types Day and Time in a single sentence

Bot: Hi! How Can I help You
User: Please book a table for me at 8 PM for Today
Bot: Sure! Reservation Successful

Scenario B: User inputs days and time in seperate text messages

Bot: Hi! How Can I help You
User: Need to book a table
Bot: When Can I book for you, please mention AM/PM aswell
User: 8 PM
Bot: For When do you want me to book
User: Today
Bot: Reservation Successfull

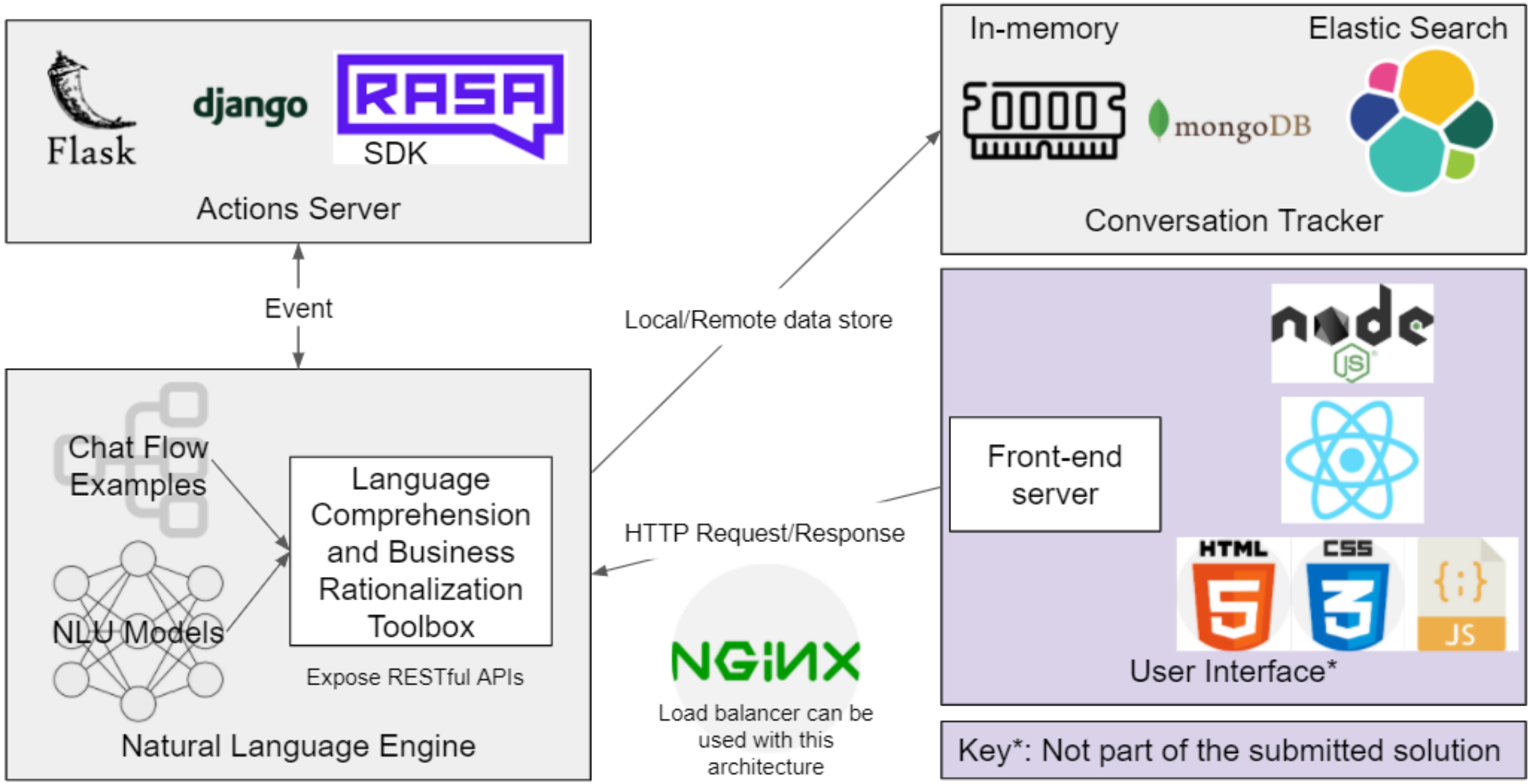
Scenario C: Reservation Alloted only if slot available

User: Please book a table for me today at 8 PM
Bot: Sorry! Table not available at this time. Please try other timings

Scenario D: Delete the Booking

Bot: How Can I help You
User: Delete My reservation
Bot: Please Enter the Booking ID
User: 12345678910
Bot: Booking Deleted

Chatbot Architecture



Rasa Key Components
NLU.yml

NLU.yml is required to train the entity classifier. The structure and different training examples are in this file.

As we can see in the image, GREET is the intent

And Examples [Hey, Hi, Hello There] are the

Training samples

Actions.py

- Actions.py includes the set of actions that should be taken based on the user inputs.

In this use case, Actions.py contains actions to Delete Bookings, Make Bookings,

- Check If slot Available.

```
class ActionHelloWorld(Action):

    def name(self) -> Text:
        return "action_delete_res"

    def run(self, dispatcher: CollectingDispatcher,
            tracker: Tracker,
            domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:
        id_ = tracker.get_slot('id')
        if id_ in Booking_IDs:
            Booking_IDs.pop(id_,0)
            dispatcher.utter_message(text="Slot Deleted!")
        else:
            dispatcher.utter_message(text="Invalid booking ID")

        dispatcher.utter_message(text='action_confirm_Day_Time_Given')
        return []
```

Stories.yml

- Stories.yml contains the structure of the conversations and guides the chatbot to have meaningful conversations.
- It Identifies intent of every user input and guides the chatbot with help of Actions.

This is the story when user wants to cancel the Booking. If the Intent is Delete Reservation it Initiates necessary actions to accomplish the user requirements.

```
- story: cancel
  steps:
  - intent: greet
  - action: utter_greet
  - action: utter_help
  - intent: delete_reservation
  - action: utter_ask_id
  - intent: id_check
    entities:
      - id: 1234567891

  - action: action_delete_res
  - action: utter_thankyou
```

Domain.yml

- Domain.yml exposes all
- the Dialogue flows,
- intents, slots and actions as APIs.

```yaml
actions:
    - action_hello_world
    - action_confirm_just_booking
    - action_confirm_Day_Time_Given
    - action_delete_res



slots:
  time:
    type: text
  number_of_people:
    type: text
  date:
    type: text
  id:
    type: float

responses:
  utter_greet:
  - text: "Hey! How are you?"

  utter_time:
  - text: "What time do you want to book (Mention am/pm)?"
  utter_help:
  - text: "How Can I help you?"
```

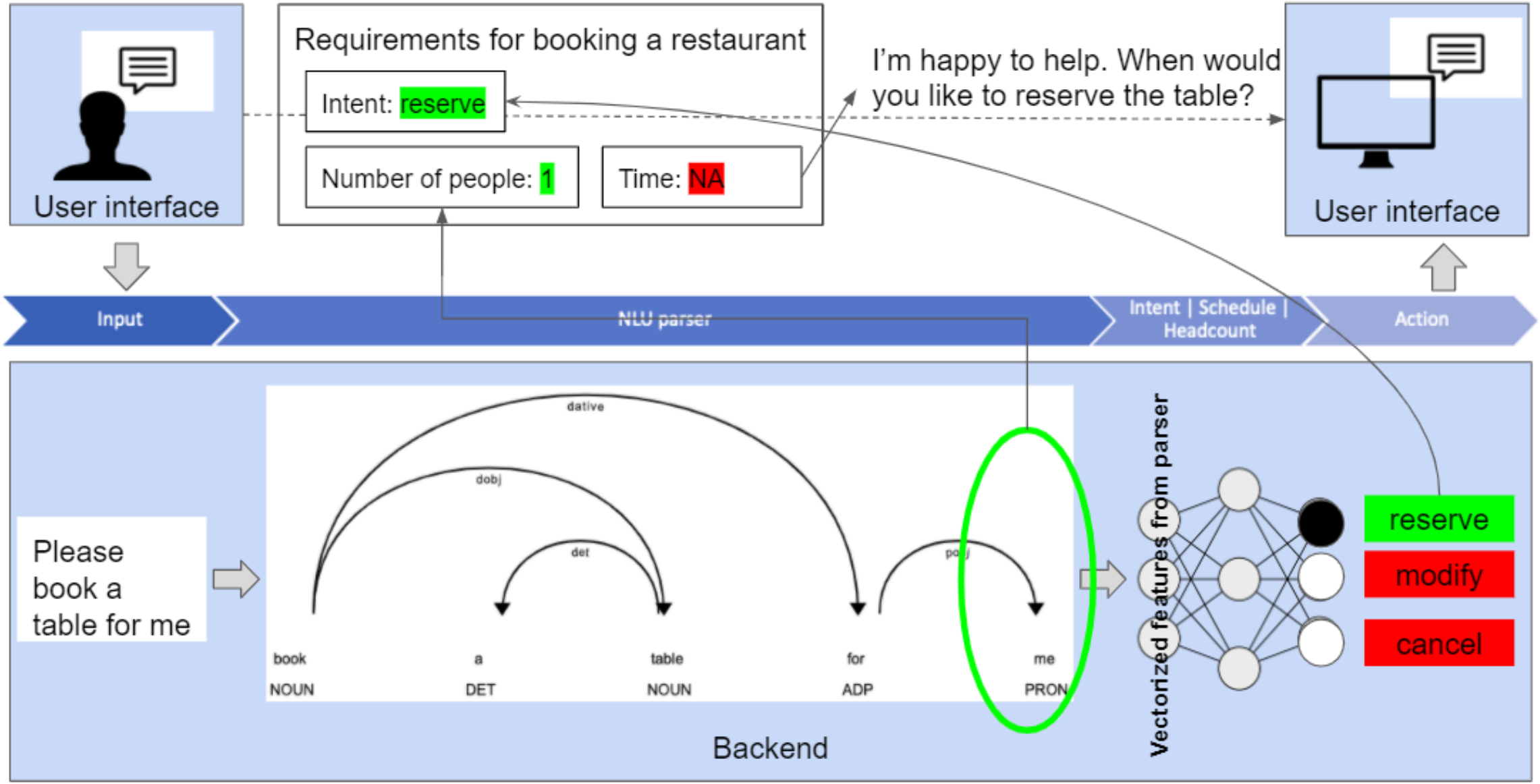
Execution Steps - RASA

1. Unzip the ChatBot_Solution File
2. conda create --name rasa python=3.6
3. conda activate rasa
4. pip install rasa
5. Mount the directory of the unzipped ChatBot_Solution File
6. rasa train (optional)
7. rasa shell on another command prompt window execute
   rasa run actions
8. Chatbot will be activated (ALWAYS START THE CHATBOT WITH WORD HI)

Chat-Bot From Scratch

- WIP
- In the following slides I have explained designing of Intent Classifier along with extracting key entities.

Building a chatbot from scratch



Reserving Table Execution by Chatbot

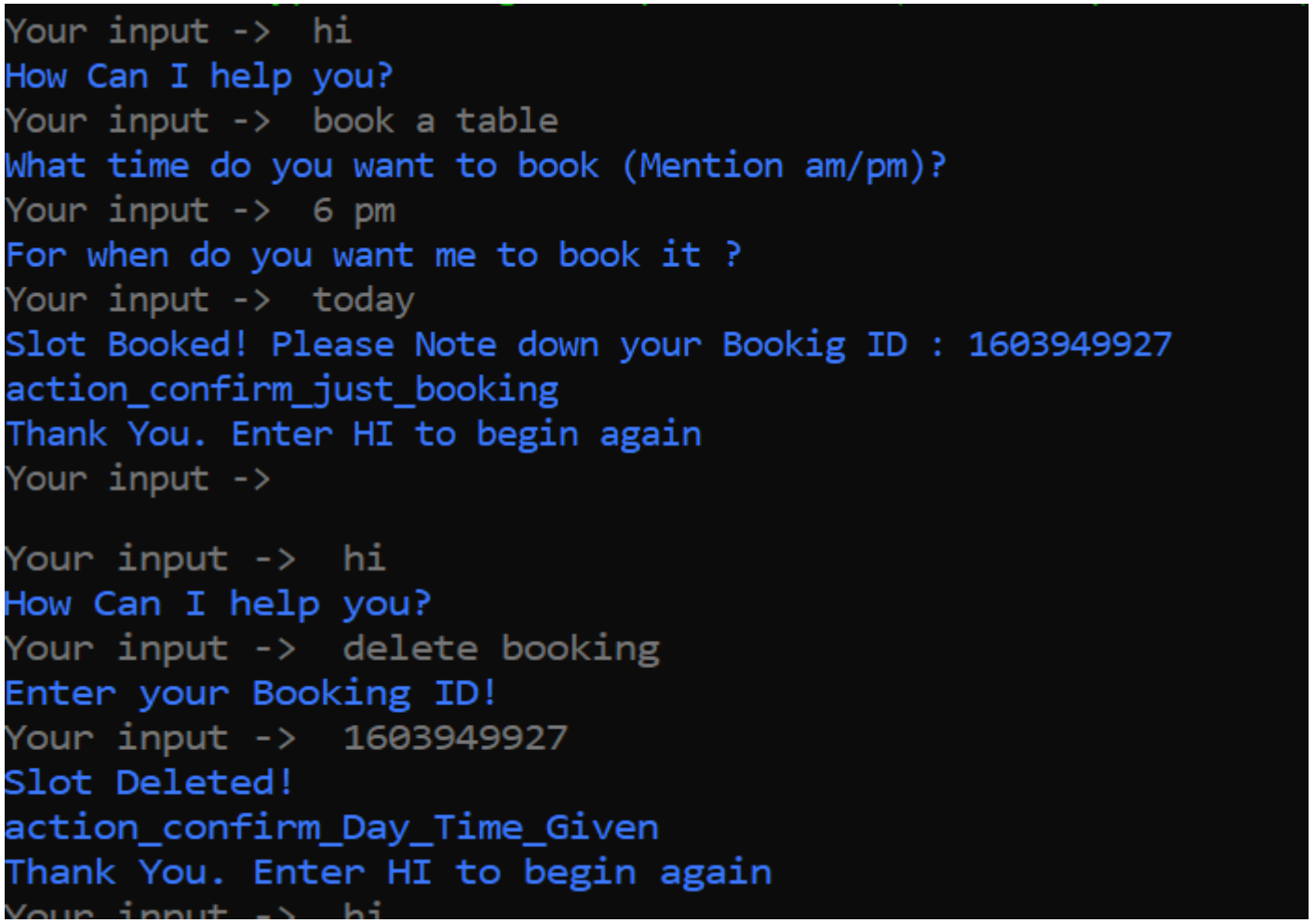
```
Bot loaded. Type a message and press enter (use '/stop' to
Your input ->  hi
Hey! How are you?
How Can I help you?
Your input ->  book a table for me
What time do you want to book (Mention am/pm)?
Your input ->  7 pm
For when do you want me to book it ?
Your input ->  today
Slot Booked! Please Note down your Bookig ID : 1603941330
action_confirm_just_booking
Thank You. Enter HI to begin again
Your input ->
```

Table Booking

```
Hey! How are you?
How Can I help you?
Your input ->  book a table for me
What time do you want to book (Mention am/pm)?
Your input ->  7 pm
For when do you want me to book it ?
Your input ->  today
Hi The Slot is not available! Please try for some other slots
action_confirm_just_booking
Thank You. Enter HI to begin again
Your input ->
```

Chat Bot Does not allow multiple bookings at same time

Deleting the Booking Action by Chatbot



Future Scope

- Increasing number of bookings at same time
- Handling booking for various dates and days
- Adding additional features to the bot like modifying booking, taking into account the number of people per tables/booking.

Reference

- https://rasa.com/docs/rasa