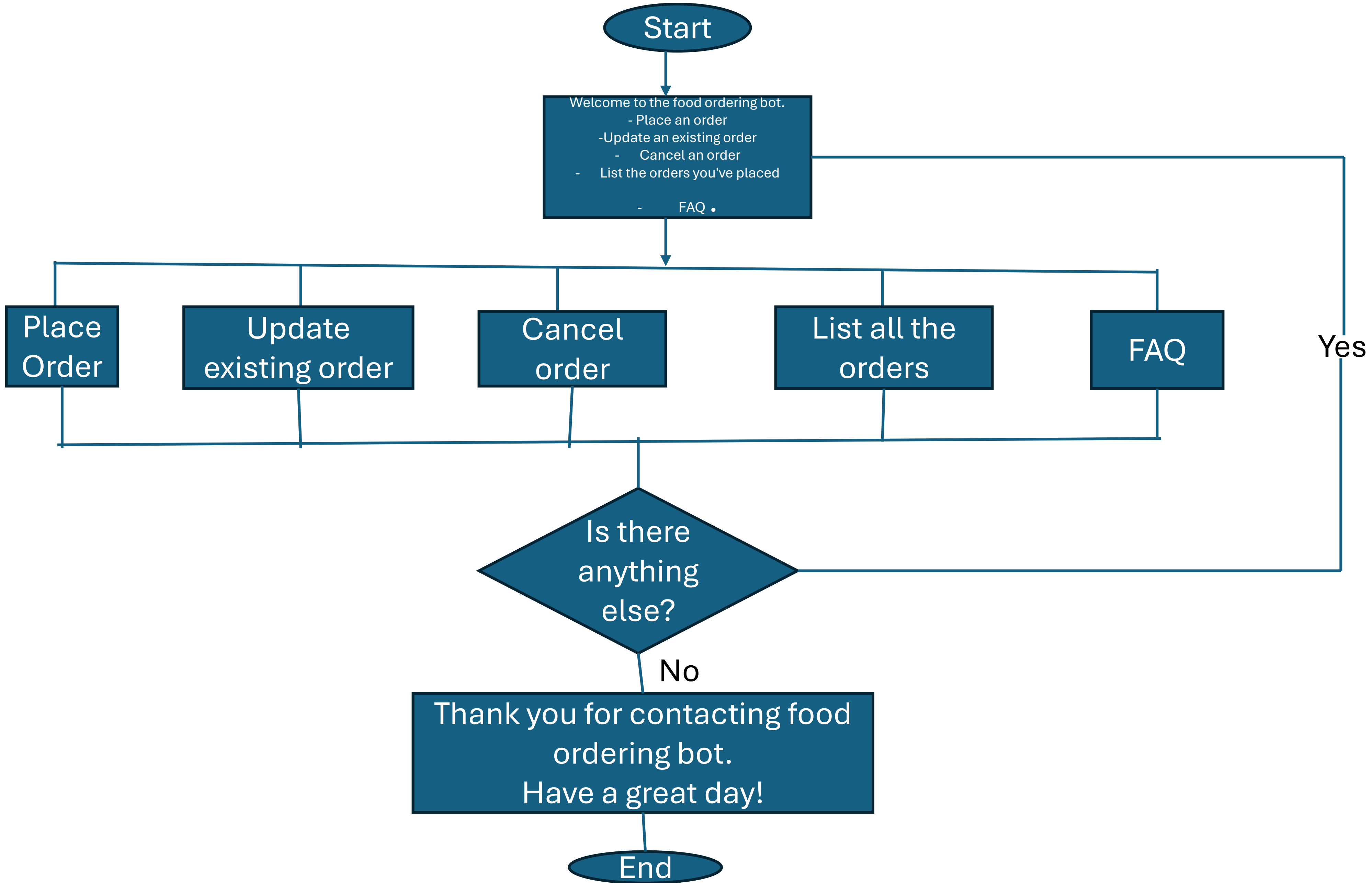
Food Ordering Bot
Using Google Dialogflow

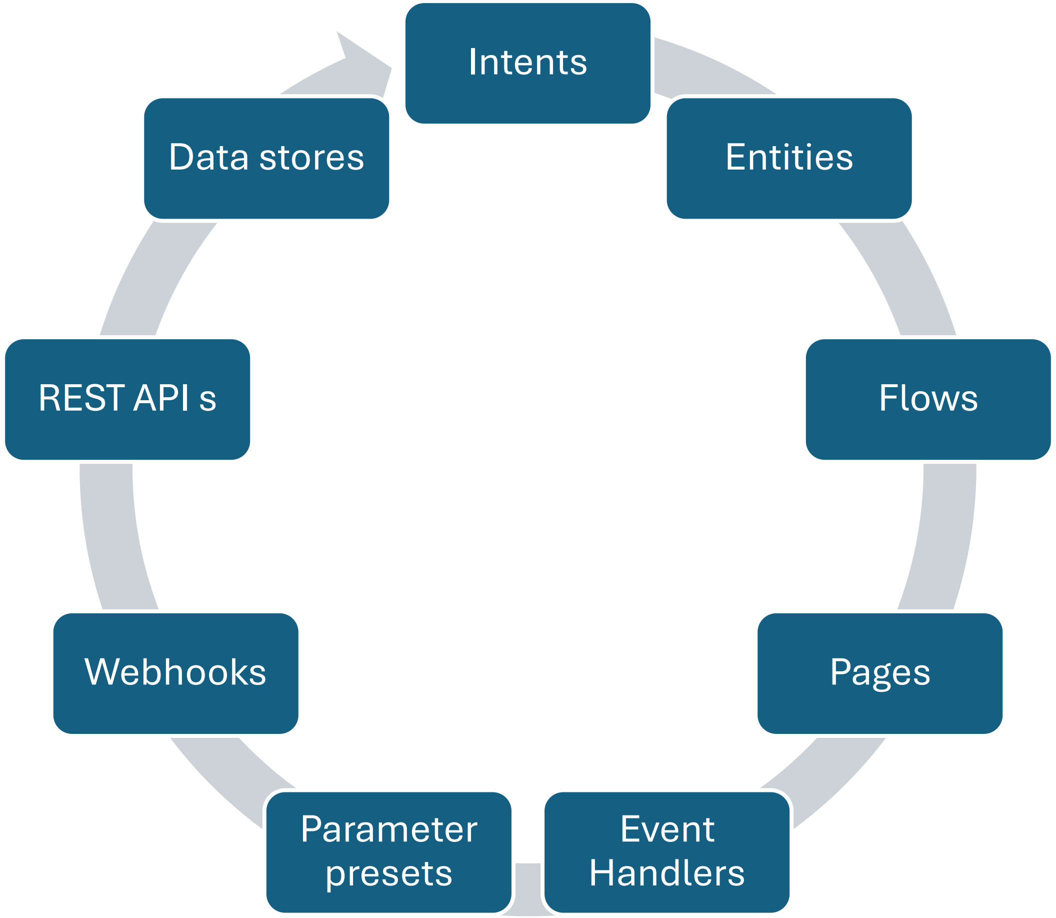
Dialogflow CX

Flow Chart :



Start
Welcome to the food ordering bot.
- Place an order
-Update an existing order
- Cancel an order
- List the orders you've placed
- FAQ .
Place Order
Update existing order
Cancel order
List all the orders
FAQ
Yes
Is there anything else?
No
Thank you for contacting food ordering bot.
Have a great day!
End

TOPICS COVERED :



Intents

Entities

Flows

Pages

Event Handlers

Parameter presets

Webhooks

REST API s

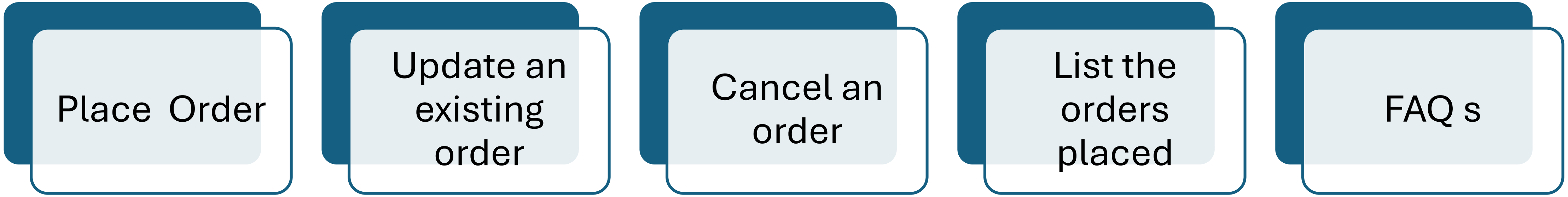
Data stores

Created agent named “Food Ordering Bot”.

In default start flow’s start page added welcome message.

Using Custom Payload, Created flow names (Place Order, Update order, Cancel order, List the order, FAQ) as buttons in start page and routed all the flows.

FLOWS :



Place Order

Update an existing order

Cancel an order

List the orders placed

FAQ s

PLACE ORDER :

Created intents for placing order , Payment mode , Confirmation and added training phrases to train the bot.

Created entity for the food item names, Card name, City name, quantity, Card number, user name .

Created pages like Input details, Payments, Confirmation etc.

In input details page, Parameters like user_name, user_city, user_item, user_quantity, user_city were created to store the user input details.

Created a mock API and list of food items were updated into it using POST HTTP method and written code for GET HTTP method in Google Cloud Run .

Webhook named “List of items” was created and linked cloud run function to it. This webhook contains the list of items.

Contd.

The user details like his/her name, item he/she wanted to order, quantity, user address were collected .

Later the user will be routed to payment page and asked to select the mode of payment. There were 3 options i.e COD, UPI, Card.

Once the details were collected, using session parameters the details were displayed to the user and asked for confirmation.

>> If user chooses **COD**, He/ She will be routed to Confirmation page.

>> If user chooses **UPI** , He/ She will be asked to enter UPI id and routed to confirmation page.

>> If user chooses **Card** , He/ She will be asked to enter card number, card expiry date, card CVV, OTP and then routed to confirmation page.

>> REGEX was used to capture the card number (12-digit numbers) and Event Handlers were used to handle the errors.

If user confirms it as “yes” ,user will be routed to the next page which displays thank you message along with the order id which will be generated using Webhook using GET HTTP method. Later user will be redirected to “ anything else” page.

If user types “No”, user preferences will be erased using parameters presets and redirected to the user input page, where he will again be asked to enter input parameters(item name, quantity, city, mode of payment).

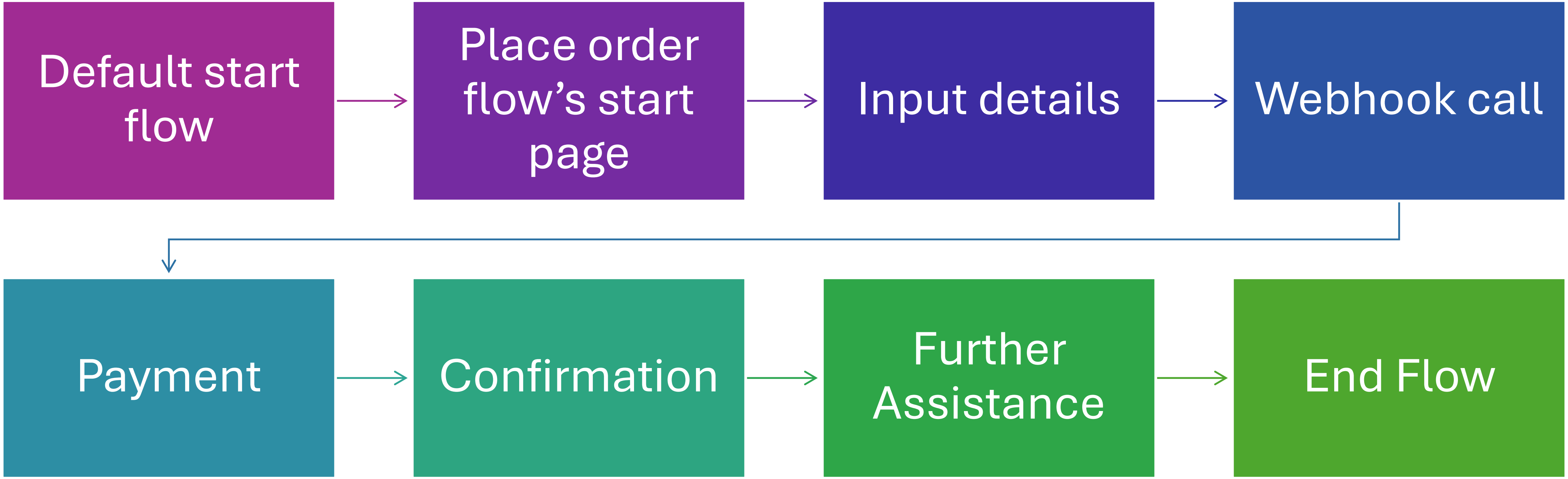
Contd.

- The user will be asked for any assistance.

i. If user types “Yes” , user will be asked to choose one of the options from the flows (Place order, Update order, Cancel order, List the order, FAQ) displayed. Once he selects the flow, user will be redirected to that particular flow.

ii. If user types “NO”, Thank you message will be displayed, and flow will be ended.

Flow Design:


Default start flow
Place order flow’s start page
Input details
Webhook call
Payment
Confirmation
Further Assistance
End Flow

UPDATE ORDER:

- The Update Order flow allows users to make changes to an already placed order directly through the chatbot.
- It starts by capturing the order number, followed by the updated item and quantity.
- Dialogflow intents handle user inputs, while webhook tags manage the logic and API communication.
- The flow ensures order verification before updating, preventing invalid changes.
- This is a key part of making the bot more dynamic, user-aware, and conversationally rich.

Approach:

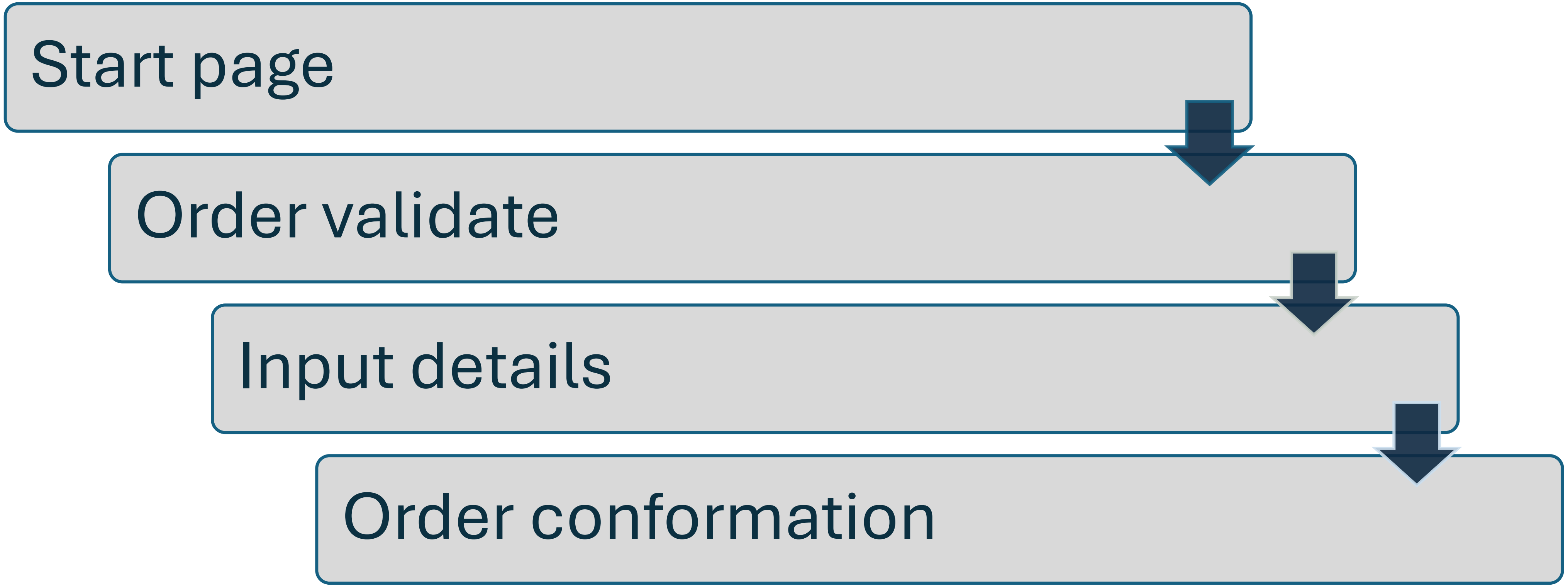
Order ID Verification: The chatbot prompts the user to enter their order ID.

Order Validation: If the order ID exists in the API, the chatbot proceeds with the update using a PUT call. If the order ID is invalid, the user is asked to recheck and re-enter the correct order ID.

Order Details Retrieval: Upon entering a valid order ID**,** the chatbot fetches the previous order details and displays them to the user.

Order Update: The user can update the order details. The chatbot then updates the order using a PUT API call.

Flow Design:



Start page

Order validate

Input details

Order conformation

Webhook logic:

The webhook uses two tags to manage the update flow:

validate_order: Sends a GET request to check if the order number is valid in the **Orders** API.

update_order: Sends a PUT request to update the item and quantity for the given order.

Sample API Response:

```
{
   "id": 234567,
   "name": “john", "items": "Pizza",
   "quantity": 9,  "address": “city",
   "mode_of_payment": "COD"
}
```

This approach ensures that only valid orders are updated and item details are preserved during the update process.

Challenges and Solutions:

Challenge: Validating invalid or non-existent order numbers

Solution: Implemented a GET request in the webhook to check order validity before proceeding

Challenge: Managing user input variations (e.g., typos in item names)

Solution: Used Dialogflow's entity mapping to handle synonyms and improve item recognition

Challenge: Handling missing parameters (e.g., quantity not provided)

Solution: Added fallback responses and parameter re-prompts in Dialogflow

Summary:

The "Update Your Order" flow enhances the overall user experience by allowing customers to modify their existing food orders using natural language. It validates the provided order number through a webhook and updates the selected item and quantity in real-time.

This flow improves the bot’s flexibility, reduces manual intervention, and ensures smoother interactions, making the ordering system more efficient and user-friendly.

CANCEL ORDER :

Created intents for Cancel order, Confirmation and added training phrases to train the bot.

Created @sys.number and @sys.any entity to capture order number and user input for cancellation reason.

Created pages for Cancellation reason and confirmation page to cancel order.

Created a mock API and list of food items were updated into it using POST HTTP method and written code for GET HTTP method in Google Cloud Run .

Webhook named “Cancel order” was created and linked cloud run function to it. This webhook contains the list of orders and their details.

Cancel order webhook communicate with the backend and send order number to the API and it will cancel the order from the data base and that order number details will be deleted from the database.

Default Start flow

If the end user provides training phrases like "cancel order" or "I want to cancel my order," or selects the "cancel order" option from the default start flow, it will proceed to the cancel order flow.

Cancel order flow

GDF console will prompt the user to enter the order number to cancel the order. If the user input is valid and found in the database, it will display the order details and ask for the cancellation reason. Upon receiving the user's input, it will ask for confirmation to cancel the order. Upon confirmation, the order details will be deleted from the database.

Order Not Found

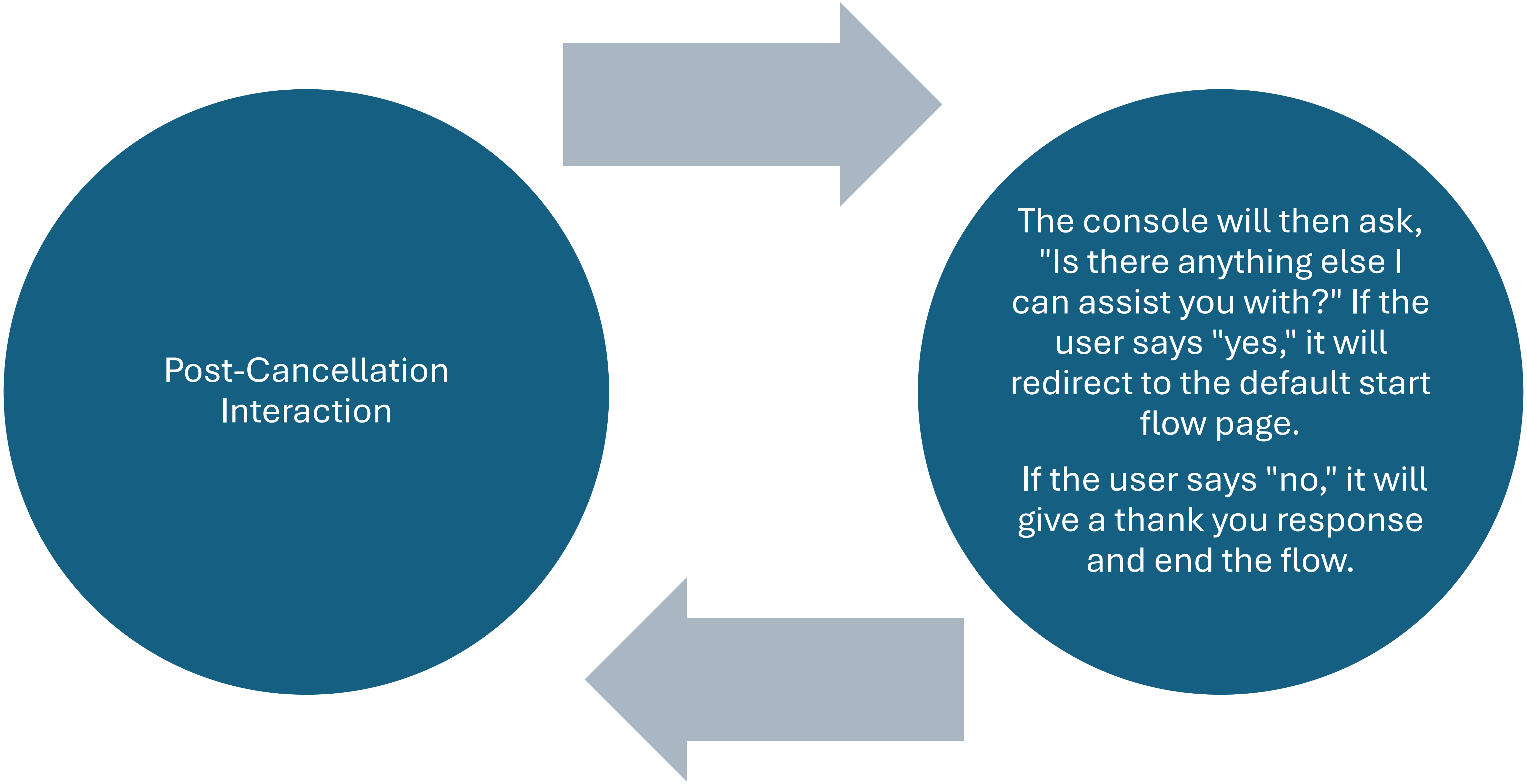
If the order number is not found, the system will respond with "No matching order number found. Please enter a valid order number."

Order Cancellation Confirmation

If the user confirms "Yes," it will delete the order details from the database and display the message "Sorry, your order has been cancelled successfully.“

If the user confirms "No," the order number and cancellation reason input parameters will be erased using parameter presets, and the user will be redirected to the start page of the cancel order flow, where they will be asked again to enter the order number to cancel the order.

Contd.



Post-Cancellation Interaction

The console will then ask, "Is there anything else I can assist you with?" If the user says "yes," it will redirect to the default start flow page.

If the user says "no," it will give a thank you response and end the flow.