Complete Dialogflow ES Food Delivery Chatbot Blueprint

Table of Contents

Overview

This blueprint provides a complete implementation guide for creating a food delivery chatbot using Google Dialogflow ES that supports both takeaway and door delivery orders.

Key Features

- **Multi-channel support**: Web, mobile, voice assistants
- **Order management**: Add, modify, remove items from cart
- **Delivery options**: Home delivery and takeaway
- **Customer management**: Store customer details and preferences
- **Order tracking**: Real-time order status updates
- **Payment integration**: Multiple payment methods
- **Natural language processing**: Understanding varied user expressions

Technology Stack

- **NLU Platform**: Google Dialogflow ES
- **Backend**: FastAPI (Python) or Node.js
- **Database**: MySQL or PostgreSQL

- **Session Storage**: Redis (recommended)
- **Deployment**: Google Cloud Platform / AWS / Azure
- **Tunneling**: Ngrok (for development)

Agent Configuration

Basic Settings

```
Agent Name: Food Delivery Assistant
Default Language: English (en)
Time Zone: Your local timezone
Description: AI assistant for food ordering and delivery
```

Advanced Settings

```
Classification Threshold: 0.3
API Version: V2
Beta Features: Enabled
Auto Speech Adaptation: Enabled
Spell Checking: Enabled
```

Intents Structure

Core Intents (25 Total)

1. Conversation Management

- **Default Welcome Intent**
  - Training Phrases: "Hi", "Hello", "Start ordering"
  - Response: Welcome message with menu options
  - Output Context: `welcome-followup` (lifespan: 2)
- **Default Fallback Intent**
  - Handles unrecognized inputs
  - Response: Clarification request
- **Goodbye Intent**
  - Training Phrases: "Thank you", "Goodbye", "That's all"
  - Response: Farewell message
  - End conversation

2. Menu & Discovery

- **Show Menu Intent**
  - Action: `show.menu`
  - Parameters: `food-category` (optional)
  - Fulfillment: Yes
  - Training Phrases: "Show menu", "What do you have"
- **Recommendations Intent**
  - Action: `get.recommendations`
  - Fulfillment: Yes
  - Training Phrases: "What do you recommend", "Popular items"
- **Dietary Restrictions Intent**
  - Action: `check.dietary.restrictions`
  - Parameters: `dietary-restriction`
  - Fulfillment: Yes

3. Order Management

- **Order Food Intent**
  - Action: `start.order`
  - Output Context: `order-followup`
  - Training Phrases: "I want to order", "Start ordering"
- **Add Item to Cart Intent**
  - Action: `add.item.to.cart`
  - Parameters: `food-item` (required), `quantity` (required), `size` (optional)
  - Fulfillment: Yes
  - Prompts: "What would you like to order?" (for food-item)
- **Remove Item from Cart Intent**
  - Action: `remove.item.from.cart`
  - Parameters: `food-item` (required)
  - Fulfillment: Yes
- **Modify Order Intent**
  - Action: `modify.order`
  - Parameters: `food-item`, `modification-type`, `new-value`
  - Fulfillment: Yes

4. Order Specifications

- **Delivery Option Selection Intent**
  - Action: `select.delivery.type`
  - Parameters: `delivery-type`
  - Training Phrases: "Delivery please", "I want it delivered"
- **Takeaway Option Selection Intent**
  - Action: `select.takeaway.type`
  - Training Phrases: "Takeaway", "I'll pick it up"
- **Special Instructions Intent**
  - Action: `add.special.instructions`
  - Parameters: `instructions`, `spice-level`, `cooking-preference`
  - Fulfillment: Yes

5. Customer Information

- **Customer Details Collection Intent**
  - Action: `collect.customer.details`
  - Parameters: `customer-name` (required), `phone-number` (required)
  - Fulfillment: Yes
  - Prompts: "Please provide your name" (for customer-name)
- **Address Collection Intent**
  - Action: `collect.address`
  - Parameters: `address-components` (required)
  - Fulfillment: Yes
  - Input Context: `delivery-selected-followup`

6. Order Finalization

- **Order Confirmation Intent**
  - Action: `confirm.order`
  - Fulfillment: Yes
  - Training Phrases: "Confirm order", "Yes, place the order"
- **Payment Information Intent**
  - Action: `collect.payment`
  - Parameters: `payment-method` (required)
  - Fulfillment: Yes
- **Order Total Intent**

- Action: `calculate.total`
- Fulfillment: Yes

7. Post-Order Services

- **Order Status Inquiry Intent**
  - Action: `track.order`
  - Parameters: `order-id` (optional)
  - Fulfillment: Yes
- **Cancel Order Intent**
  - Action: `cancel.order`
  - Parameters: `order-id` (optional)
  - Fulfillment: Yes
- **Delivery Time Estimate Intent**
  - Action: `estimate.delivery.time`
  - Fulfillment: Yes

8. Support & Information

- **Restaurant Hours Intent**
  - Action: `get.restaurant.hours`
  - Fulfillment: Yes
- **Contact Information Intent**
  - Response: Static contact details
- **Complaint or Feedback Intent**
  - Action: `process.feedback`
  - Parameters: `feedback-type`, `message`
  - Fulfillment: Yes

Entities Design

System Entities

- `@sys.number` - For quantities
- `@sys.person` - For customer names
- `@sys.phone-number` - For contact numbers
- `@sys.location` - For addresses
- `@sys.date-time` - For delivery times

Custom Entities

Food-Related Entities

```
@food-item:
  - pizza: pie, italian bread
  - burger: sandwich, patty
  - pasta: spaghetti, noodles
  - chicken: poultry, bird
  - salad: greens, vegetables

@food-category:
  - main course: entree, primary dish
  - appetizer: starter, snack
  - dessert: sweet, pudding
  - beverage: drink, liquid

@size:
  - small: S, mini, personal
  - medium: M, regular
  - large: L, big, family
  - extra large: XL, jumbo, super

@pizza-toppings:
  - pepperoni: salami
  - mushrooms: fungi
  - cheese: mozzarella, cheddar
  - olives: black olives, green olives
```

Service-Related Entities

```
@delivery-type:
  - delivery: home delivery, door delivery
  - takeaway: pickup, self pickup, collect

@payment-method:
  - cash: money, bills, cash on delivery
  - card: credit card, debit card
  - online: digital payment, UPI, PayPal

@spice-level:
  - mild: less spicy, light
  - medium: normal, regular
  - spicy: hot, extra hot
  - no spice: bland, plain

@dietary-restrictions:
  - vegetarian: veg, veggie
  - vegan: plant-based
  - gluten-free: no gluten
  - dairy-free: no dairy, lactose-free
```

Conversation Flow

Main Conversation Path

1. **Welcome** → Show greeting and options
2. **Menu Display** → Show available items
3. **Item Selection** → Add items to cart
4. **Order Building** → Modify quantities, sizes, instructions
5. **Delivery Type** → Choose delivery or takeaway
6. **Customer Details** → Collect name and contact
7. **Address** (if delivery) → Collect delivery address
8. **Order Review** → Show order summary
9. **Payment** → Select payment method
10. **Confirmation** → Confirm and place order
11. **Tracking** → Provide order tracking info

Context Management

```
Context Lifespans:
  - welcome-followup: 2
  - menu-followup: 5
  - order-followup: 10
  - item-selected-followup: 8
  - order-building-followup: 10
  - delivery-selected-followup: 5
  - customer-details-followup: 3
  - order-review-followup: 5
```

Follow-up Intents Structure

```
Welcome Intent
├── Show Menu (yes follow-up)
├── Order Food (custom follow-up)
└── Restaurant Hours (custom follow-up)

Add Item Intent
├── Add More Items (yes follow-up)
├── Modify Order (custom follow-up)
├── Checkout (no follow-up)
└── Remove Item (custom follow-up)

Order Review Intent
├── Confirm Order (yes follow-up)
├── Modify Order (no follow-up)
└── Add Payment (custom follow-up)
```

Webhook Implementation

Webhook URL Structure

```
POST https://your-domain.com/webhook
Content-Type: application/json
```

Request Format

```
{
  "queryResult": {
    "action": "add.item.to.cart",
    "parameters": {
      "food-item": "pizza",
      "quantity": 2,
      "size": "large"
    },
    "queryText": "Add 2 large pizzas",
    "intent": {
      "displayName": "Add Item to Cart"
    }
  },
  "session": "projects/PROJECT_ID/agent/sessions/SESSION_ID"
}
```

Response Format

```
{
  "fulfillmentText": "Added 2 large pizzas to your cart. Anything else?",
  "outputContexts": [
    {
      "name": "projects/PROJECT_ID/agent/sessions/SESSION_ID/contexts/order-building-foll
      "lifespanCount": 8
    }
  ]
}
```

Key Webhook Functions

1. **Menu Management**: Fetch and display menu items
2. **Cart Operations**: Add, remove, modify cart items
3. **Order Processing**: Create and manage orders
4. **Customer Management**: Store customer information
5. **Payment Processing**: Handle payment methods
6. **Order Tracking**: Provide real-time status updates

Database Schema

Core Tables

```
-- Customers
CREATE TABLE customers (
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(200) NOT NULL,
    phone VARCHAR(20) UNIQUE NOT NULL,
    email VARCHAR(200),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Menu Categories
CREATE TABLE food_categories (
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    description TEXT,
    display_order INT DEFAULT 0,
    active BOOLEAN DEFAULT TRUE
);

-- Menu Items
CREATE TABLE menu_items (
    id INT PRIMARY KEY AUTO_INCREMENT,
    category_id INT,
    name VARCHAR(200) NOT NULL,
    description TEXT,
    price DECIMAL(10,2) NOT NULL,
    available BOOLEAN DEFAULT TRUE,
    dietary_info JSON,
    FOREIGN KEY (category_id) REFERENCES food_categories(id)
);

-- Orders
CREATE TABLE orders (
    id INT PRIMARY KEY AUTO_INCREMENT,
    order_id VARCHAR(20) UNIQUE NOT NULL,
    customer_id INT,
    status ENUM('confirmed', 'preparing', 'out_for_delivery', 'delivered', 'cancelled'),
    delivery_type ENUM('delivery', 'takeaway'),
    total_amount DECIMAL(10,2),
    delivery_address TEXT,
    special_instructions TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (customer_id) REFERENCES customers(id)
);

-- Order Items
CREATE TABLE order_items (
    id INT PRIMARY KEY AUTO_INCREMENT,
    order_id VARCHAR(20),
    menu_item_id INT,
    quantity INT NOT NULL,
    size VARCHAR(50),
```

```
    special_instructions TEXT,
    unit_price DECIMAL(10,2),
    total_price DECIMAL(10,2),
    FOREIGN KEY (order_id) REFERENCES orders(order_id),
    FOREIGN KEY (menu_item_id) REFERENCES menu_items(id)
);
```

Integration Guide

1. Web Integration

```
<!-- Add to your website -->
<iframe src="https://console.dialogflow.com/api-client/demo/embedded/PROJECT_ID"></iframe
```

2. Mobile App Integration

```
// React Native example
import { Dialogflow_V2 } from 'react-native-dialogflow';

Dialogflow_V2.setConfiguration({
  type: 'service_account',
  projectId: 'your-project-id',
  private_key: 'your-private-key',
  client_email: 'your-client-email'
});

const sendMessage = async (message) => {
  const result = await Dialogflow_V2.requestQuery(message);
  return result;
};
```

3. Voice Integration

```
// Google Assistant Actions integration
const { dialogflow } = require('actions-on-google');
const app = dialogflow();

app.intent('Default Welcome Intent', conv => {
  conv.ask('Welcome! What would you like to order?');
});
```

Testing Strategy

1. Intent Testing

- Test each intent with multiple training phrase variations
- Verify entity extraction accuracy
- Check context flow between intents

2. Conversation Flow Testing

```
Test Scenarios:
1. Complete order flow (delivery)
2. Complete order flow (takeaway)
3. Order modification scenarios
4. Error handling and recovery
5. Multi-item orders
6. Special dietary requirements
```

3. Integration Testing

- Webhook response validation
- Database operations testing
- Payment flow testing
- Order tracking functionality

4. User Acceptance Testing

- Natural conversation testing
- Edge case handling
- Performance testing

Deployment Checklist

Pre-Deployment

- [ ] All intents configured and tested
- [ ] Entities properly defined with synonyms
- [ ] Webhook implementation complete
- [ ] Database schema deployed
- [ ] SSL certificate configured
- [ ] Environment variables set

Production Setup

- [ ] Agent versioning enabled
- [ ] Audit logging enabled
- [ ] Error monitoring configured
- [ ] Performance monitoring setup
- [ ] Backup procedures established
- [ ] Security measures implemented

Post-Deployment

- [ ] Integration testing in production
- [ ] User training completed
- [ ] Support documentation ready
- [ ] Monitoring dashboards configured
- [ ] Escalation procedures defined

Best Practices

Intent Design

1. Use descriptive intent names
2. Provide diverse training phrases (15-20 per intent)
3. Implement proper fallback mechanisms
4. Use follow-up intents for complex flows

Entity Management

1. Create comprehensive synonym lists
2. Use system entities when possible
3. Implement fuzzy matching for food items
4. Regular entity updates based on menu changes

Context Handling

1. Set appropriate context lifespans
2. Clear contexts when conversation ends
3. Use context for conversation state management
4. Implement context validation in webhook

Error Handling

1. Graceful degradation for webhook failures
2. Clear error messages for users
3. Retry mechanisms for transient failures
4. Logging and monitoring for debugging

Performance Optimization

1. Cache frequently accessed data
2. Optimize database queries
3. Use connection pooling
4. Implement rate limiting

This blueprint provides a comprehensive foundation for building a robust food delivery chatbot using Dialogflow ES. Customize the intents, entities, and responses based on your specific restaurant's needs and menu items.