```
import spacy
from transformers import pipeline

nlp = spacy.load("en_core_web_sm")

intent_classifier = pipeline("zero-shot-classification", model="facebook/bart-large-mnli")

def process_text(text):
    """
    Processes input text: tokenizes the text, extracts named entities, and predicts intent using zero-shot classification.
    """
    doc = nlp(text)

    tokens = [token.text for token in doc]

    entities = [(ent.text, ent.label_) for ent in doc.ents]

    candidate_intents = ["BookFlight", "CancelReservation", "OrderFood", "AskWeather"]

    intent_result = intent_classifier(text, candidate_intents)

    return tokens, entities, intent_result

def run_chatbot_with_context():
    """
    Runs an interactive, context-aware chatbot. For the "BookFlight" intent,
    it collects required information via slot filling. For other intents, it provides a direct response.
    """
    print("Welcome to the Intelligent Customer Support Chatbot (Enhanced)!")
    print("You can ask for services like flight booking, reservation cancellation, or food ordering.")
    print("Type 'exit' or 'quit' at any time to end the session.\n")

    conversation_context = {'intent': None, 'slots': {}}

    while True:
        user_input = input("Customer: ")
        if user_input.lower() in ["exit", "quit"]:
            print("ChatBot: Thank you for contacting support. Goodbye!")
            break

        tokens, entities, intent = process_text(user_input)
        predicted_intent = intent['labels'][0]

        if conversation_context['intent'] is None:

            conversation_context['intent'] = predicted_intent

        if conversation_context['intent'] == "BookFlight":
            required_slots = ['departure', 'destination', 'date']

            for ent_text, ent_label in entities:
                if ent_label == "GPE":
                    if 'departure' not in conversation_context['slots']:
                        conversation_context['slots']['departure'] = ent_text
                    elif 'destination' not in conversation_context['slots']:
                        conversation_context['slots']['destination'] = ent_text
                if ent_label == "DATE" and 'date' not in conversation_context['slots']:
                    conversation_context['slots']['date'] = ent_text

            missing_slots = [slot for slot in required_slots if slot not in conversation_context['slots']]

            if missing_slots:
                response = "To book your flight, please provide your " + ", ".join(missing_slots) + "."
            else:
                response = (f"Great! Here are the details I got:\n"
                            f"Departure: {conversation_context['slots']['departure']}\n"
                            f"Destination: {conversation_context['slots']['destination']}\n"
                            f"Date: {conversation_context['slots']['date']}\n"
                            "I will now proceed to book your flight.")
                conversation_context = {'intent': None, 'slots': {}}

        else:
            if predicted_intent == "CancelReservation":
                response = "Please provide your reservation number so I can assist with cancellation."
            elif predicted_intent == "OrderFood":
                response = "What cuisine or specific dish would you like to order?"
            elif predicted_intent == "AskWeather":
                response = "Which city's weather forecast do you need?"
            else:
                response = "I'm not entirely sure how to help with that. Could you please clarify?"
```

```
            conversation_context = {'intent': None, 'slots': {}}

        print("\n--- Debug Information ---")
        print("Tokens:", tokens)
        print("Entities:", entities)
        print("Detected Intent:", intent['labels'][0], "(score: {:.2f})".format(intent['scores'][0]))
        print("Current Context:", conversation_context)
        print("----------------------------")

        print("ChatBot:", response, "\n")

run_chatbot_with_context()
```

```
Device set to use cpu
Welcome to the Intelligent Customer Support Chatbot (Enhanced)!
You can ask for services like flight booking, reservation cancellation, or food ordering.
Type 'exit' or 'quit' at any time to end the session.

Customer: I want to book flight ticket from Newyork to Paris on next Friday

--- Debug Information ---
Tokens: ['I', 'want', 'to', 'book', 'flight', 'ticket', 'from', 'Newyork', 'to', 'Paris', 'on', 'next', 'Friday']
Entities: [('Newyork', 'PERSON'), ('Paris', 'GPE'), ('next Friday', 'DATE')]
Detected Intent: BookFlight (score: 0.39)
Current Context: {'intent': 'BookFlight', 'slots': {'departure': 'Paris', 'date': 'next Friday'}}
----------------------------
ChatBot: To book your flight, please provide your destination.

Customer:  Newyork to Paris

--- Debug Information ---
Tokens: [' ', 'Newyork', 'to', 'Paris']
Entities: [('Paris', 'GPE')]
Detected Intent: BookFlight (score: 0.45)
Current Context: {'intent': None, 'slots': {}}
----------------------------
ChatBot: Great! Here are the details I got:
Departure: Paris
Destination: Paris
Date: next Friday
I will now proceed to book your flight.

Customer:
```