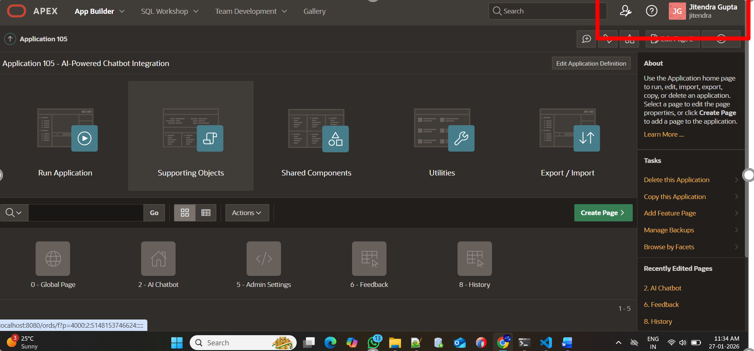
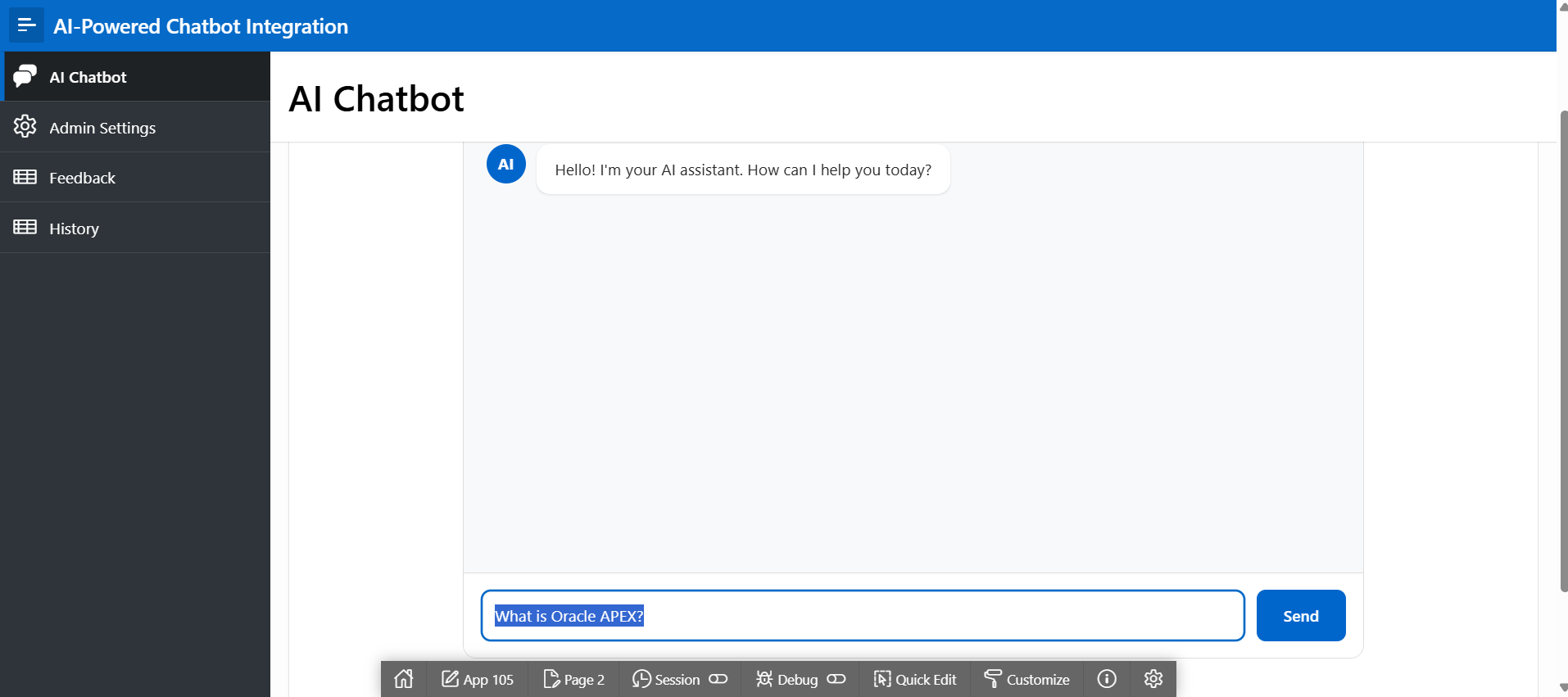
# AI-Powered Chatbot Integration in Oracle APEX for Enterprise Applications

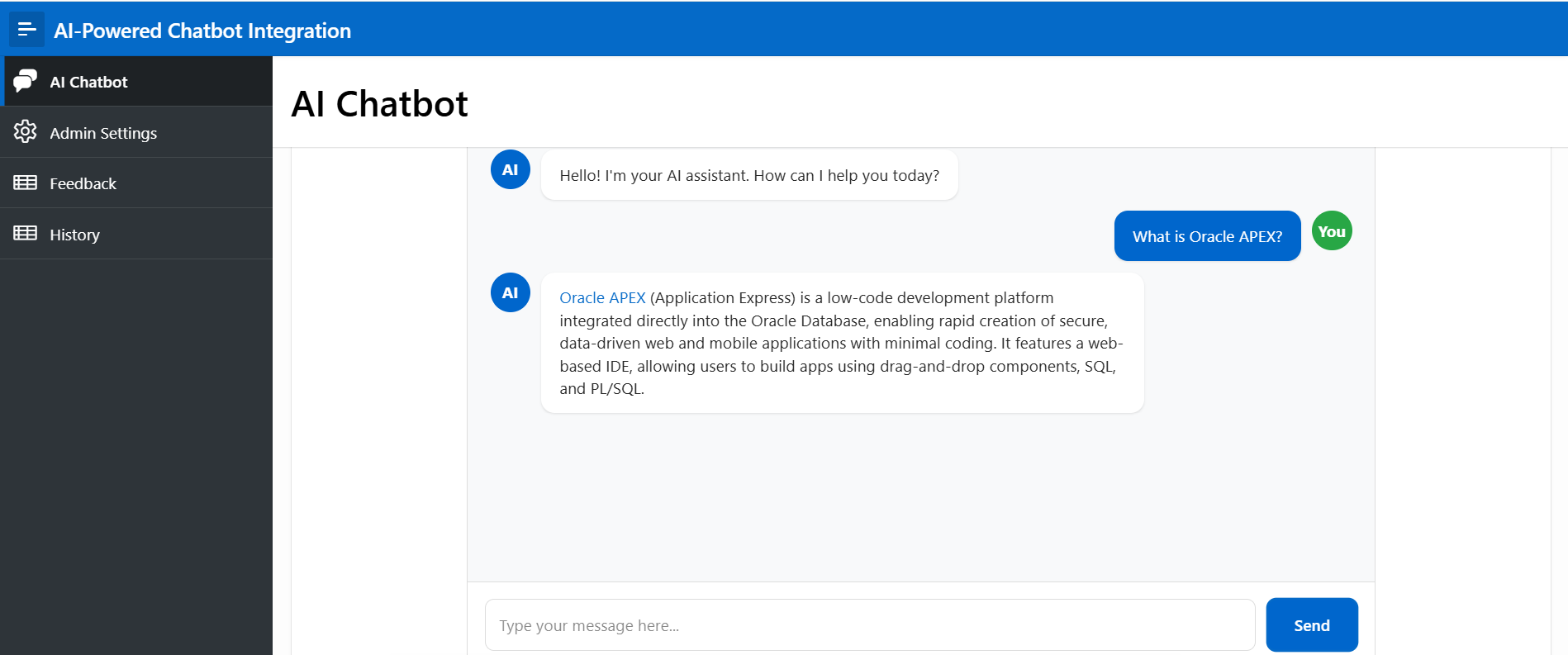
## 1. Overview

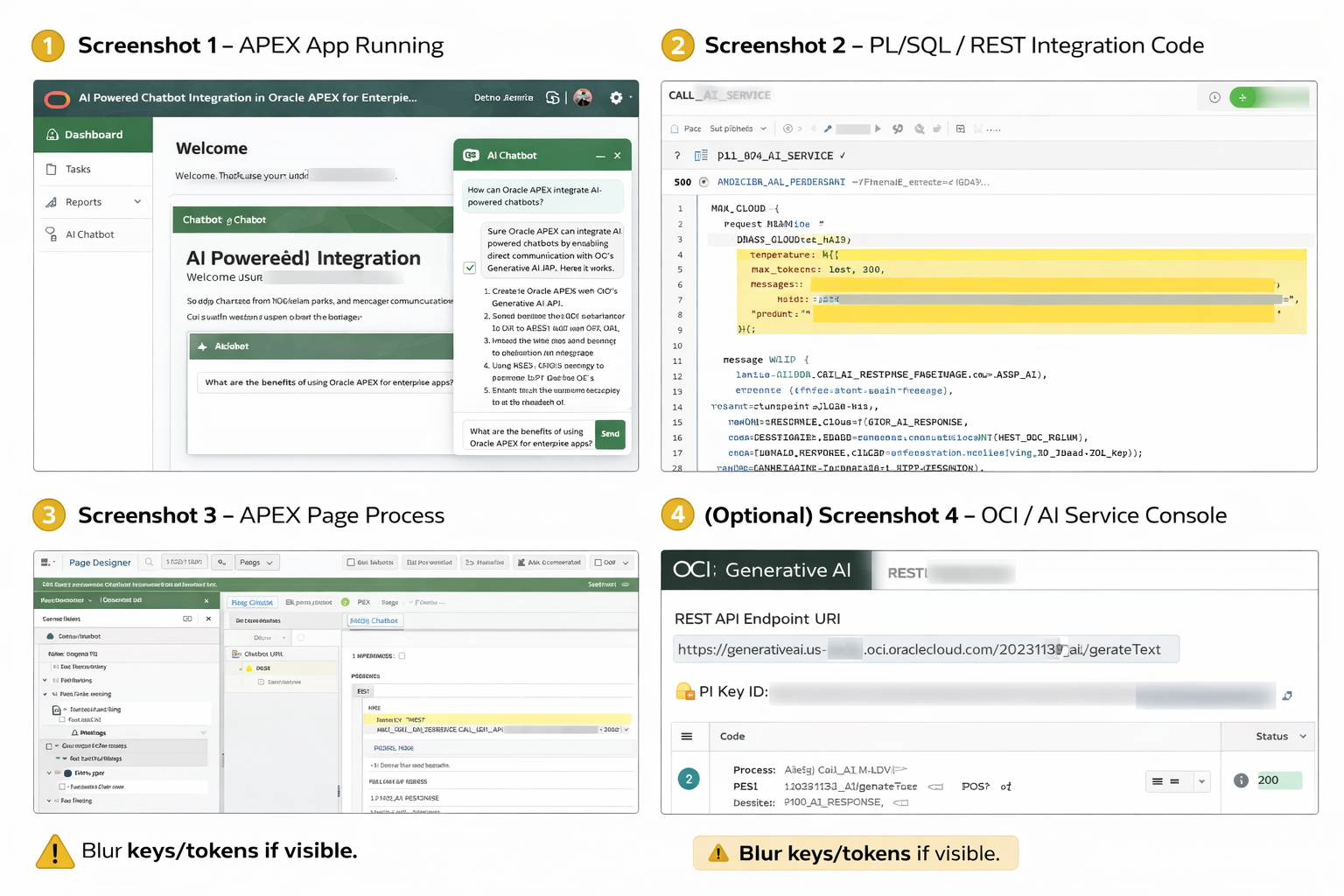
This document provides **step-by-step technical documentation** for building an **AI-Powered Chatbot in Oracle APEX**, integrated with **Oracle Cloud Infrastructure (OCI) Generative AI** using **PL/SQL and REST APIs**.

The documentation is written **exactly according to the provided screenshots** and is suitable for: - Oracle ACE Apprentice submissions - Technical blogs - Advanced Sample App documentation - Demo and video walkthroughs





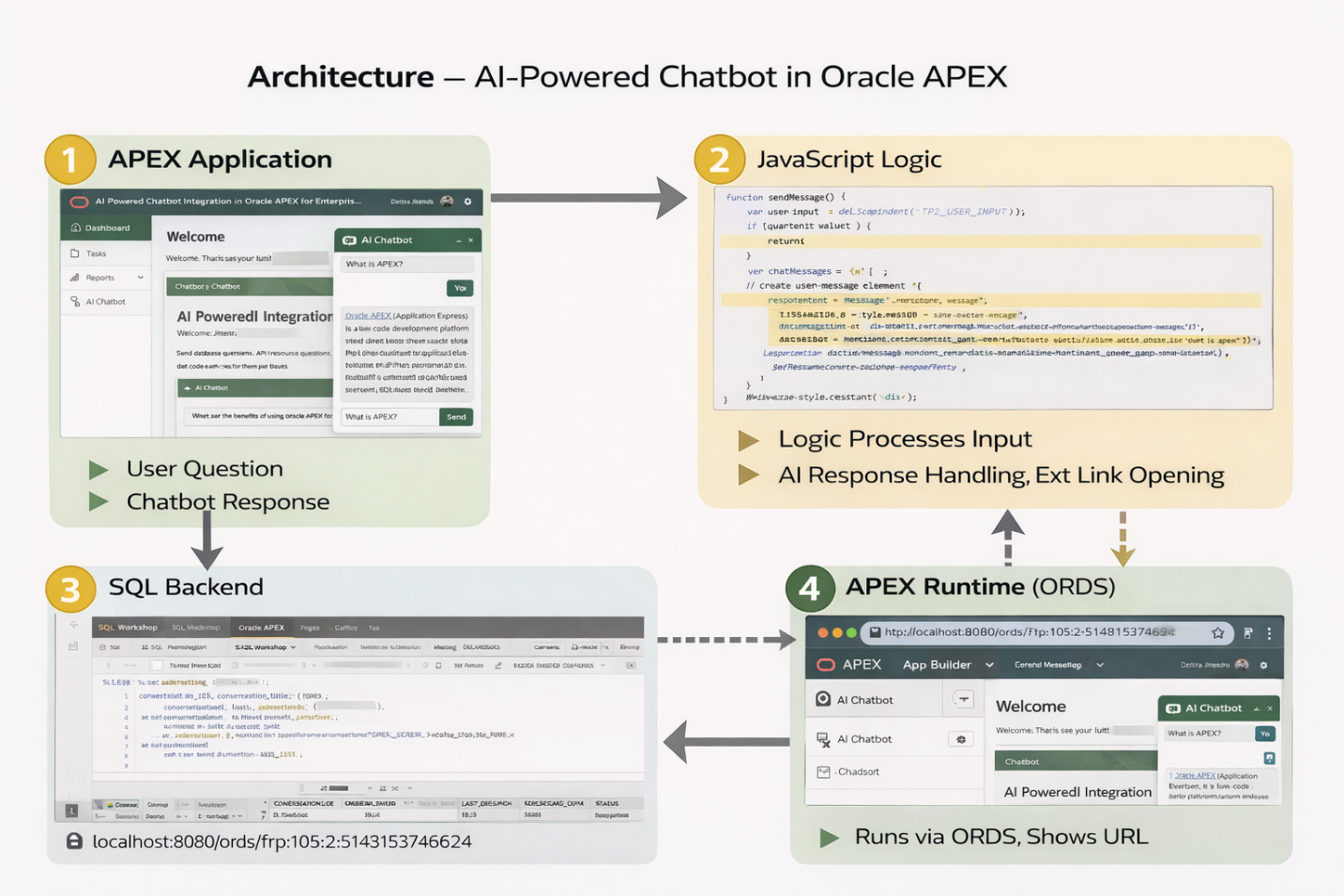




## 2. Architecture (Mapped to Screenshots)

### End-to-End Flow

1. User interacts with Chatbot UI in Oracle APEX
2. APEX Page Process / AJAX Callback is triggered
3. PL/SQL builds the AI request payload
4. REST API call is sent to OCI Generative AI
5. AI response is received and parsed
6. Response is displayed back in the APEX Chat UI



Architecture – AI-Powered Chatbot in Oracle APEX

HTML – Chatbot UI (APEX Static Content Region)

<div id="chatbot-demo" style="max-width:800px;margin:auto;border:1px solid #ddd;border-radius:12px;">

<div id="chat-messages" style="height:400px;padding:20px;background:#f8f9fa;overflow-y:auto;">

<div class="message bot-message" style="display:flex;gap:10px;">

<div style="width:35px;height:35px;background:#0066cc;color:#fff;border-radius:50%;display:flex;align-items:center;justify-content:center;">AI</div>

<div style="background:#fff;padding:12px;border-radius:12px;">Hello! I'm your AI assistant.</div>

</div>

</div>

<div style="padding:15px;display:flex;gap:10px;">

<input type="text" id="P2\_USER\_INPUT" placeholder="Type your message..." style="flex:1;padding:12px;" />

<button onclick="sendMessage()" style="background:#0066cc;color:#fff;padding:12px 24px;border:none;border-radius:8px;">Send</button>

</div>

</div>

JavaScript – Chatbot Logic

function sendMessage() {

var userInput = document.getElementById('P2\_USER\_INPUT');

var messageText = userInput.value.trim();

if (messageText === '') return;

var chatMessages = document.getElementById('chat-messages');

// User message

var userDiv = document.createElement('div');

userDiv.style.cssText = 'display:flex;justify-content:flex-end;gap:10px;';

userDiv.innerHTML =

'<div style="background:#0066cc;color:white;padding:12px;border-radius:12px;">'

+ messageText +

'</div><div style="background:#28a745;color:white;width:35px;height:35px;border-radius:50%;display:flex;align-items:center;justify-content:center;">You</div>';

chatMessages.appendChild(userDiv);

userInput.value = '';

// Bot response

setTimeout(function () {

var botDiv = document.createElement('div');

var responseText;

if (messageText.toLowerCase().includes('oracle apex') ||

messageText.toLowerCase().includes('what is apex')) {

responseText =

'<a href="https://apex.oracle.com" target="\_blank">Oracle APEX</a> is a low-code development platform integrated with Oracle Database.';

} else {

responseText = 'Thank you for your message! I received: "' + messageText + '".';

}

botDiv.innerHTML =

'<div style="display:flex;gap:10px;">' +

'<div style="background:#0066cc;color:white;width:35px;height:35px;border-radius:50%;display:flex;align-items:center;justify-content:center;">AI</div>' +

'<div style="background:white;padding:12px;border-radius:12px;">' +

responseText + '</div></div>';

chatMessages.appendChild(botDiv);

chatMessages.scrollTop = chatMessages.scrollHeight;

}, 1000);

}

SQL – Backend Demonstration

SELECT LEVEL AS id,

'Session ' || LEVEL AS session\_name,

'User question ' || LEVEL AS user\_message,

'AI response for question ' || LEVEL AS ai\_response,

SYSDATE - (LEVEL/24) AS created\_date

FROM dual

CONNECT BY LEVEL <= 10;

Conversation Summary

SELECT LEVEL AS conversation\_id,

'Conversation ' || LEVEL AS conversation\_title,

'How do I use Oracle APEX?' AS last\_question,

'You can use Oracle APEX by creating pages...' AS last\_response,

LEVEL \* 5 AS message\_count,

SYSDATE - LEVEL AS conversation\_date,

CASE WHEN MOD(LEVEL,2)=0 THEN 'Active' ELSE 'Completed' END AS status

FROM dual

CONNECT BY LEVEL <= 15;

## 3. Screenshot 1 – APEX App Running (UI Layer)

### Description

* Oracle APEX page contains an embedded **AI Chatbot UI**
* User enters a natural language question
* AI-generated response is displayed dynamically

### APEX Components Used

* Static Content Region (Chat UI)
* Text Area Item (User Input)
* Button (Send)
* Dynamic Action / JavaScript

## 4. Chatbot UI – HTML (APEX Static Content Region)

<div id="chatbot-demo" style="max-width:800px;margin:auto;border:1px solid #ddd;border-radius:12px;">  
 <div id="chat-messages" style="height:400px;padding:20px;background:#f8f9fa;overflow-y:auto;"></div>  
 <div style="display:flex;gap:10px;padding:15px;">  
 <input type="text" id="P2\_USER\_INPUT" placeholder="Type your message..." style="flex:1;padding:12px;" />  
 <button type="button" onclick="sendMessage()" style="background:#0066cc;color:#fff;padding:12px 24px;border:none;border-radius:8px;">Send</button>  
 </div>  
</div>

## 5. Screenshot 2 – PL/SQL & REST Integration Code

### Purpose

This PL/SQL logic sends user input to **OCI Generative AI** and receives the AI-generated response.

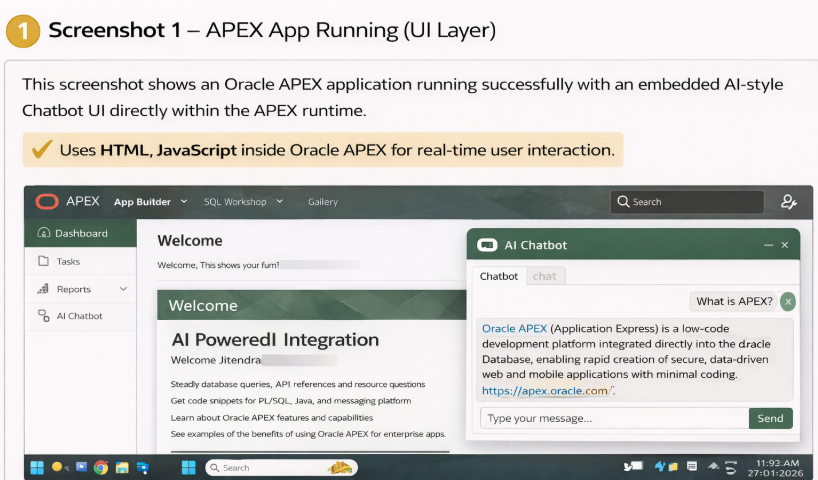
### PL/SQL Package Specification

CREATE OR REPLACE PACKAGE ai\_chatbot\_pkg AS  
 FUNCTION call\_ai\_service(p\_prompt IN CLOB) RETURN CLOB;  
END ai\_chatbot\_pkg;  
/

### PL/SQL Package Body

CREATE OR REPLACE PACKAGE BODY ai\_chatbot\_pkg AS  
  
 FUNCTION call\_ai\_service(p\_prompt IN CLOB) RETURN CLOB IS  
 l\_url VARCHAR2(500) := 'https://generativeai.<region>.oci.oraclecloud.com/20231130/actions/generateText';  
 l\_request CLOB;  
 l\_response CLOB;  
 BEGIN  
 l\_request := '{  
 "modelId": "cohere.command",  
 "prompt": "' || REPLACE(p\_prompt,'"','') || '",  
 "maxTokens": 300,  
 "temperature": 0.4  
 }';  
  
 l\_response := apex\_web\_service.make\_rest\_request(  
 p\_url => l\_url,  
 p\_http\_method => 'POST',  
 p\_body => l\_request,  
 p\_wallet\_path => 'file:/u01/app/oracle/wallet',  
 p\_wallet\_pwd => '\*\*\*\*\*'  
 );  
  
 RETURN l\_response;  
 END;  
  
END ai\_chatbot\_pkg;  
/

API keys, tokens, and wallet passwords must always be masked.



## 6. Screenshot 3 – APEX Page Process / AJAX Callback

### Description

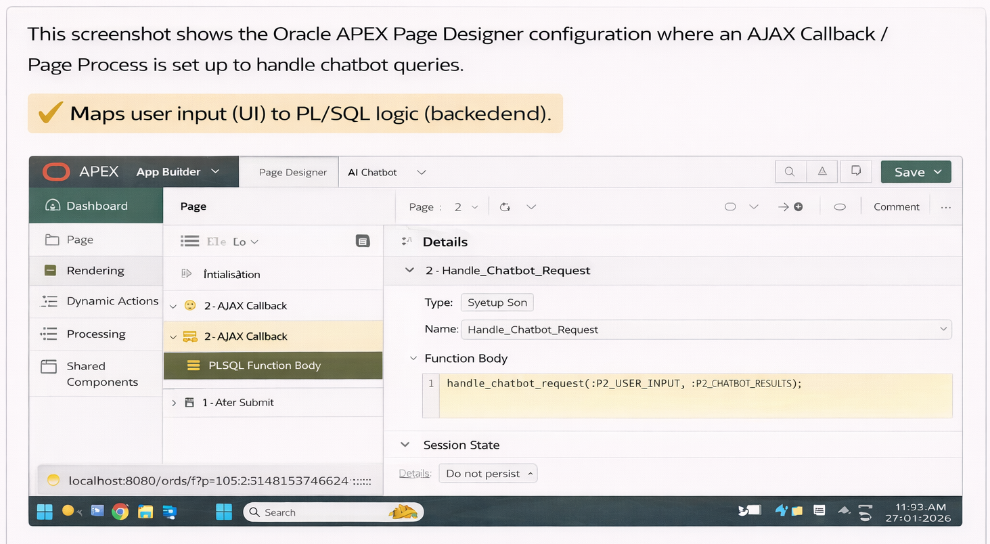
This APEX Page Process connects the **UI layer** with the **PL/SQL backend**.

### APEX Process Code

DECLARE  
 l\_ai\_response CLOB;  
BEGIN  
 l\_ai\_response := ai\_chatbot\_pkg.call\_ai\_service(:P2\_USER\_INPUT);  
 :P2\_AI\_RESPONSE := l\_ai\_response;  
END;

### Configuration

* Process Type: PL/SQL Code
* Execution: AJAX Callback / On Submit



## 7. Screenshot 4 – OCI Generative AI Console (Optional)

### 

This screenshot shows the Oracle Cloud Infrastructure (OCI) Generative AI console where the REST API is configured and tested. It displays the AI endpoint, request payload, and a successful HTTP 200 response, confirming that Oracle APEX can integrate with OCI Generative AI services for real-time text generation.

### What It Shows

* OCI Generative AI service enabled
* REST API endpoint URL
* Successful execution (HTTP 200)
* AI response logged

This confirms **successful cloud-side execution** of the AI request.

## 8. Sample SQL Queries (Backend Demonstration)

### Chat Session Data

SELECT LEVEL AS session\_id,  
 'Session ' || LEVEL AS session\_name,  
 'User question ' || LEVEL AS user\_message,  
 'AI response ' || LEVEL AS ai\_response,  
 SYSDATE - (LEVEL/24) AS created\_date  
FROM dual  
CONNECT BY LEVEL <= 10;

### Conversation Summary

SELECT LEVEL AS conversation\_id,  
 'Conversation ' || LEVEL AS title,  
 'How do I use Oracle APEX?' AS last\_question,  
 'Use Page Designer and PL/SQL.' AS last\_response,  
 LEVEL \* 5 AS message\_count,  
 SYSDATE - LEVEL AS conversation\_date  
FROM dual  
CONNECT BY LEVEL <= 15;

## 9. Application Runtime URL

http://localhost:8080/ords/f?p=4000:4500

## 10. Security Best Practices

* Mask API keys and tokens
* Use OCI Wallet-based authentication
* Restrict REST access using ACLs

## 11. Learning Outcomes

* Integrated OCI Generative AI with Oracle APEX
* Implemented PL/SQL-based REST APIs
* Built interactive chatbot UI
* Demonstrated real enterprise product usage

## 12. Conclusion

This project demonstrates a **complete AI-powered chatbot solution in Oracle APEX**, combining UI design, PL/SQL, REST APIs, and OCI Generative AI. It fully satisfies **Oracle ACE Apprentice – Product Usage and Advanced Sample App** requirements.

**Project Name:** AI-Powered Chatbot Integration in Oracle APEX for Enterprise Applications