Cloud Applications Phase 4: Chatbot Deployment using the Watson assistant in cloud application.

Text chatbot using Watson Assistant

Creating a text chatbot using the Watson assistant in html involves a few steps. Here's a high level overview of the processor:

1. Set Up Watson Assistant:

First, you need to create a Watson Assistant instance on IBM cloud(formerly known as IBM Bluemix).if you haven't already.

Create a new skill within Watson Assistant, which defines the conversational flow and responses of your chatbot.

2. Get API Credentials:

Retrieve the API credentials (API key and URL) for your Watson Assistant instance.

3. HTML and Javascript Setup:

Create an HTML file to host your chatbot interface.

Include the Watson Assistant Javascript SDK in your HTML file by adding the fol,lowing script tag to your HTML's ***<head>*** *section.*

Code:

<script src=https://cdn.jsdelivr.net/npm/ibm-watson/assistant/v1.js></script>

4. Initialize Watson Assistant:

In your Javascript code, initialize the Watson Assistant by providing your API credentials and the assistant ID.

Code:

Const assistant= new AssistantV1({})

5. Create an HTML chat Interface:

Create an HTML structure for the chat interface, which typically consists of an input field and a chat container to display messages.

6. Handle user input:

Add Javascript cocde to handle user input (example: when the user submits a message).

Send the user's message to Watson Assistant for processing using the message API.

Code:

Assistant_id:'YOUR_ASSISTANT_ID', Input: { Message_type:'text', Text: userMessage,}. }).then(response=>{//Handle the response from Watson Assistant and display it in the chat container.

})

.catch(error=>{Console.error(error);

});

7. Display Bot Responses:

Extract the bot's response from the Watson Assistant's API response and display it in the chat container.

8. Styling and Enhancements:

Apply CSS styles to make your chat interface visually appealing. You can add more features like buttons, images and other rich content to your chatbot responses.

9. Testing:

Test your chatbot by interacting with it in the HTML interface.

10.Deployment:

Deploy your HTML page and associated Javascript to a web server or a hosting platform.

coding for Text Chatbot:

```
<!DOCTYPE html>
<html>
<head>
<title>Text Chatbot</title>
<style>
```

```
 body {
    display: flex;
   justify-content: center;
  }

  .chatbot-container {
      width: 500px;
      margin: 0 auto;
      background-color: purple;
      border: 2px solid white;
      border-radius: 7px;
      box-shadow: 0 3px 5px rgba(0, 0, 0, 0.1);
}

  #chatbot {
    background-color: #f5f5f5;
    border: 1px solid #eef1f5;
    box-shadow: 0 2px 6px 0 rgba(0, 0, 0, 0.1);
    border-radius: 4px;
}

  #header {
    background-color: darkslategrey;
    color: #ffffff;
    padding: 20px;
   font-size: 1em;
  font-weight: bold;
}

 message-container {
```

```
        background: #ffffff;
        width: 100%;
        display: flex;
        align-items: center;
}
   #conversation {
        height: 500px;
        overflow-y: auto;
        padding: 20px;
       display: flex;
      flex-direction: column;
}

  @keyframes message-fade-in {
        from {
          opacity: 0;
          transform: translateY(-20px);
  }
    to {
          opacity: 1;
          transform: translateY(0);
  }
}

   .chatbot-message {
          display: flex;
          align-items: flex-start;
          position: relative;
         font-size: 16px;
         line-height: 20px;
```

```
        border-radius: 20px;
        word-wrap: break-word;
        white-space: pre-wrap;
        max-width: 100%;
        padding: 0 15px;
}

  .user-message {
        justify-content: flex-end;
}

  .chatbot-text {
        background-color: white;
        color: #333333;
        font-size: 1.1em;
        padding: 15px;
        border-radius: 5px;
        box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

  #input-form {
        display: flex;
        align-items: center;
        border-top: 1px solid #eef1f5;
}

  #input-field {
        flex: 1;
        height: 60px;
        border: 1px solid #eef1f5;
```

```
    border-radius: 4px;
    padding: 0 10px;
    font-size: 14px;
    transition: border-color 0.3s;
    background: #ffffff;
    color: #333333;
    border: none;
}

 .send-icon {
    margin-right: 10px;
    cursor: pointer;
}

 #input-field:focus {
     border-color: #333333;
     outline: none;
}

 #submit-button {
     background-color: transparent;
     border: none;
}

 p[sentTime]:hover::after {
      content: attr(sentTime);
      position: absolute;
     top: -3px;
     font-size: 14px;
     color: gray;
```

```
}

  .chatbot p[sentTime]:hover::after {
        left: 15px;
}

 .user-message p[sentTime]:hover::after {
      right: 15px;
}

/* width */
   ::-webkit-scrollbar {
      width: 10px;
}

/* Track */
   ::-webkit-scrollbar-track {
      background: #f1f1f1;
}

/* Handle */
   ::-webkit-scrollbar-thumb {
       background: #888;
}

/* Handle on hover */
    ::-webkit-scrollbar-thumb:hover {
         background: #555;
}
</style>
```

```
<script>

  const chatbot = document.getElementById('chatbot');
  const conversation =
document.getElementById('conversation');
 const inputForm = document.getElementById('input-
form');
 const inputField = document.getElementById('input-field');

// Add event listener to input form
  inputForm.addEventListener('submit', function(event) {
  // Prevent form submission
   event.preventDefault();

  // Get user input
   const input = inputField.value;

  // Clear input field
   inputField.value = '';
   const currentTime = new Date().toLocaleTimeString([],
{ hour: '2-digit', minute: "2-digit" });

  // Add user input to conversation
   let message = document.createElement('div');
   message.classList.add('chatbot-message', 'user-message');
   message.innerHTML = `<p class="chatbot-text" s
entTime="${currentTime}">${input}</p>`;
   conversation.appendChild(message);

  // Generate chatbot response
   const response = generateResponse(input);
```

```
  // Add chatbot response to conversation
    message = document.createElement('div');
    message.classList.add('chatbot-message','chatbot');
    message.innerHTML = `<p class="chatbot-text"
sentTime="${currentTime}">${response}</p>`;
    conversation.appendChild(message);
    message.scrollIntoView({behavior: "smooth"});
  });

// Generate chatbot response function
function generateResponse(input) {
    // Add chatbot logic here
     const responses = [
       "Hello, how can I help you today? 😊",

     ];

     // Return a random response
     return responses[Math.floor(Math.random() *
responses.length)];
  }
</script>
</head>
</html>
```

The above mentioned coding is our text chatbot implementation.

S.No	Team Members	Email ID
1	Amirthasri A	Amirthasriarul2324@gmail.com
2	Janani S	jandharsh@gmail.com
3	Harini M	harini662004@gmail.com
4	Srisubhiksha S	srishaxyz@gmail.com