

NAAN MUDHALVAN

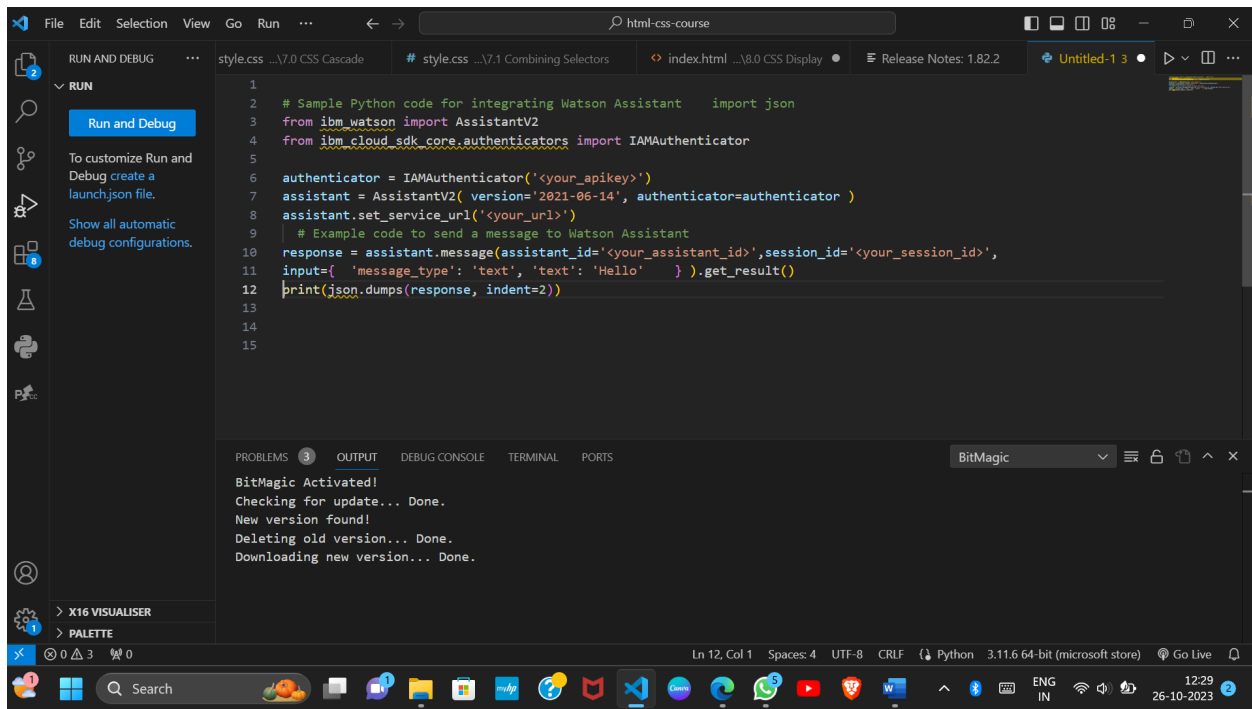
CAD101 Cloud Application Development -Group 1

Project 6: Chatbot Deployment with IBM Cloud Watson Assistant

Phase 4: Development Part 2

1. Integrate Watson Assistant with the application :

- Retrieve the necessary credentials from IBM Cloud for your Watson Assistant service.
- Update the application's code to use the credentials. You will need the API key and URL to authenticate and interact with the Watson Assistant service.



```
1 # Sample Python code for integrating Watson Assistant import json
2 from ibm_watson import AssistantV2
3 from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
4
5 authenticator = IAMAuthenticator('<your_apikey>')
6 assistant = AssistantV2( version='2021-06-14', authenticator=authenticator )
7 assistant.set_service_url('<your_url>')
8
9 # Example code to send a message to Watson Assistant
10 response = assistant.message(assistant_id='<your_assistant_id>', session_id='<your_session_id>',
11 input={ 'message_type': 'text', 'text': 'Hello' } ).get_result()
12 print(json.dumps(response, indent=2))
13
14
15
```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS BitMagic

BitMagic Activated!
Checking for update... Done.
New version found!
Deleting old version... Done.
Downloading new version... Done.

Ln 12, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.6 64-bit (microsoft store) Go Live 12:29 26-10-2023

2. Test the integration :

- Run the application and test the chatbot functionality. Ensure that the application properly sends and receives messages from the Watson Assistant service.
- Test various conversation flows and make sure that the responses are as expected.

3. Deploy the application :

- Choose an appropriate deployment option based on the application's requirements. IBM Cloud provides various options for deploying applications, including Cloud Foundry, Kubernetes, and more.
- Deploy the application to the selected environment and ensure that the necessary resources are provisioned.

4. Monitor the application :

- Set up monitoring for the application to track its performance and usage.
- Monitor the chatbot's interactions to identify any issues or areas for improvement.
- Ensure that the application is functioning correctly and delivering a seamless experience to users.

5. Maintenance and updates :

- Regularly maintain and update the application to incorporate new features and improvements.
- Keep the Watson Assistant service updated with the latest information and conversation flows.
- Monitor user feedback and make necessary adjustments to enhance the chatbot's performance and user experience.

6. Create a Web Application :

- You can use any framework or technology of your choice, such as Node.js, Python, Java, or even simple HTML/CSS/JavaScript.
- For this example, let's assume you're using a basic HTML file with JavaScript to integrate the chatbot.

7. Get IBM Cloud Watson Assistant Credentials :

Log in to your IBM Cloud account.

- Navigate to the Watson Assistant service instance that you created earlier.
- Go to the "Service credentials" section and create new credentials if you haven't already.

8. Integrate Watson Assistant in your Web Application :

Add the Watson Assistant SDK to your HTML file. You can use the following script tag:

```
#html
<script src="https://webchat.global.assistant.watson.appdomain.cloud/LoadWatsonAssistantChat.js"></script>

- Initialize the chatbot with your Watson Assistant credentials and options:
'''javascript
var chatbotIntegration = function() {      var options = {
  integrationID: 'YOUR_INTEGRATION_ID', // Replace with your integration ID
  region: 'YOUR_REGION', // Replace with your region, e.g., ussouth
};
  window.loadWatsonAssistantChat(options).then(function(instance)
{
  instance.render();
});
};
'''
```

9. Test the Integration :

- Open your web application in a browser.
- Ensure that the chatbot is loading correctly and that you can communicate with it.

10. Customize and Train the Chatbot :

- Access your Watson Assistant service instance.
- Create or import a skill to start building your chatbot's responses.
- Train the chatbot using appropriate dialogs, intents, and entities to handle user queries effectively.

11. Deploy the Web Application :

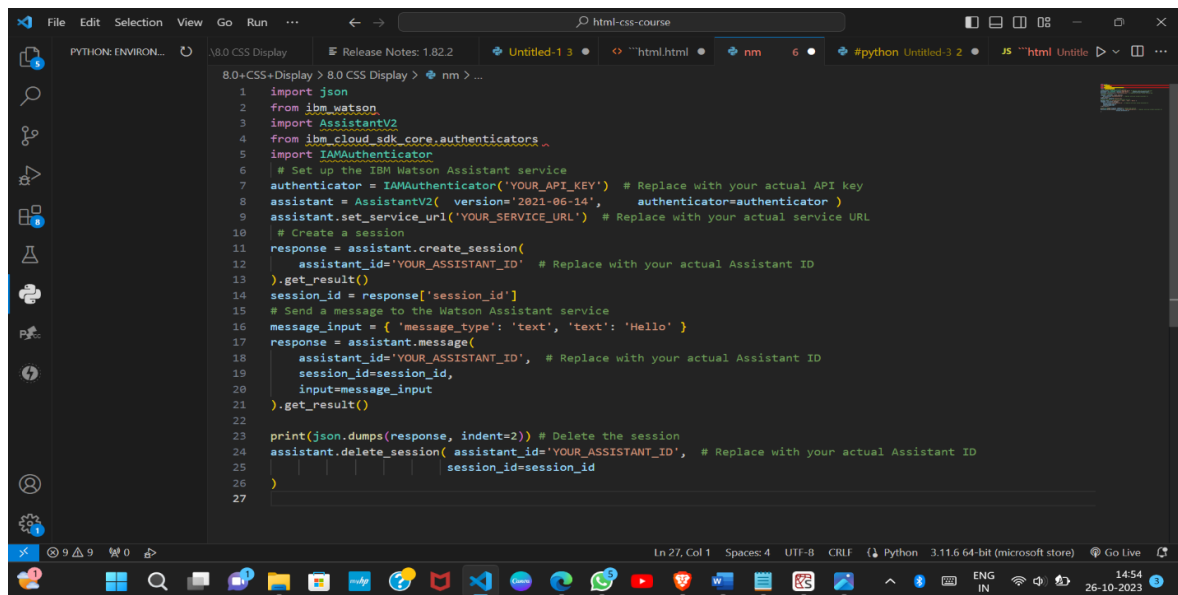
- Once you're satisfied with the integration and the functioning of the chatbot, deploy your web application to a hosting platform of your choice.
- Ensure that the necessary Watson Assistant credentials are securely integrated into your deployment process.

11. Monitor and Improve :

- Monitor user interactions with the chatbot to identify areas for improvement.
- Continuously update and enhance your Watson Assistant skill based on user feedback and real-time data.

11. Scale and Manage :

As your web application grows, make sure to scale your Watson Assistant instance accordingly to handle increased traffic and user interactions effectively.



```

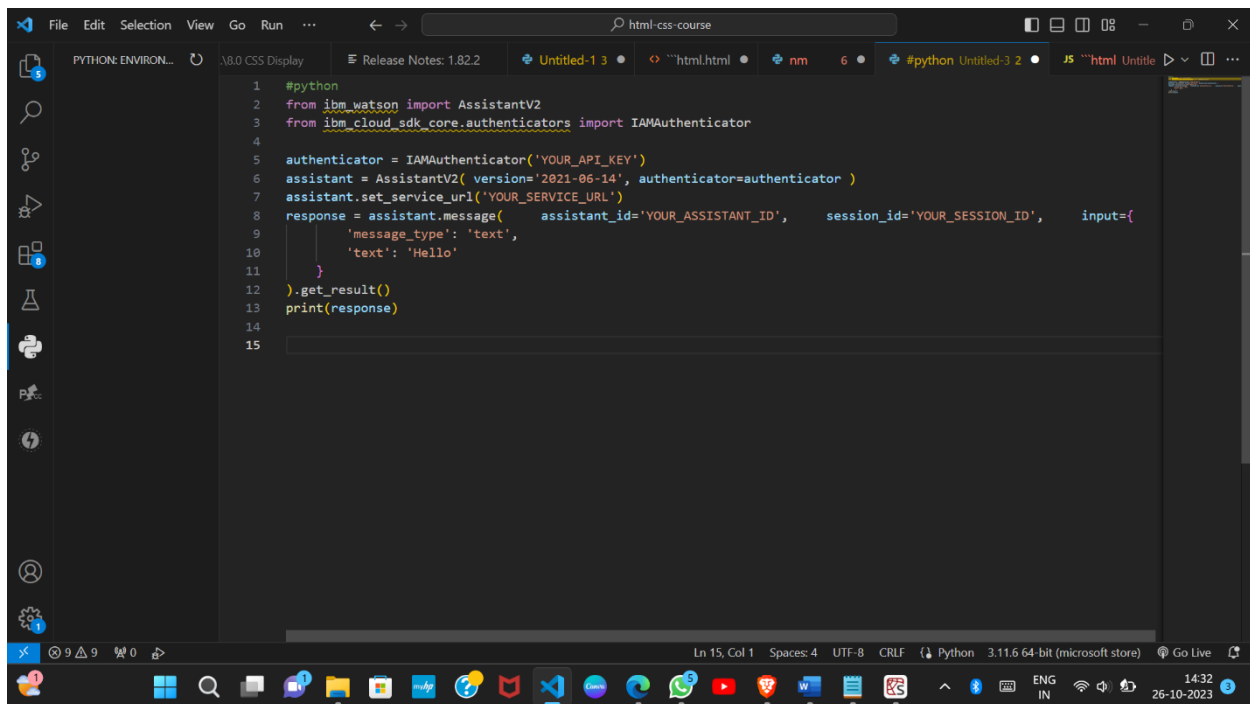
1 import json
2 from ibm_watson import AssistantV2
3 from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
4 # Set up the IBM Watson Assistant service
5 authenticator = IAMAuthenticator('YOUR_API_KEY') # Replace with your actual API key
6 assistant = AssistantV2(version='2021-06-14', authenticator=authenticator)
7 assistant.set_service_url('YOUR_SERVICE_URL') # Replace with your actual service URL
8 # Create a session
9 response = assistant.create_session(
10     assistant_id='YOUR_ASSISTANT_ID' # Replace with your actual Assistant ID
11 ).get_result()
12 session_id = response['session_id']
13 # Send a message to the Watson Assistant service
14 message_input = { 'message_type': 'text', 'text': 'Hello' }
15 response = assistant.message(
16     assistant_id='YOUR_ASSISTANT_ID', # Replace with your actual Assistant ID
17     session_id=session_id,
18     input=message_input
19 ).get_result()
20 print(json.dumps(response, indent=2)) # Delete the session
21 assistant.delete_session(assistant_id='YOUR_ASSISTANT_ID', # Replace with your actual Assistant ID
22     session_id=session_id)
23
24
25
26
27

```

Make sure you have the **ibm_watson** package installed. You can install it using pip.

12. Integrate the Chatbot :

- Use the provided SDKs and APIs to integrate the chatbot into your desired application or platform.
- Here's an example of how to create a basic chatbot using IBM Watson Assistant in Python:

A screenshot of a Visual Studio Code editor window. The editor is open to a file named 'Untitled-3 2' with a Python file icon. The code is a Python script that uses the IBM Watson Assistant SDK. It imports 'AssistantV2' from 'ibm_watson' and 'IAMAuthenticator' from 'ibm_cloud_sdk_core.authenticators'. It then creates an 'IAMAuthenticator' object with a placeholder API key, an 'AssistantV2' object with a version and the authenticator, and sets a service URL. Finally, it sends a message to the assistant with a session ID and an input text 'Hello', and prints the response. The status bar at the bottom shows 'Ln 15, Col 1', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python 3.11.6 64-bit (microsoft store)', and 'Go Live'. The system tray at the bottom right shows the time as 14:32 on 26-10-2023.

```
1 #python
2 from ibm_watson import AssistantV2
3 from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
4
5 authenticator = IAMAuthenticator('YOUR_API_KEY')
6 assistant = AssistantV2( version='2021-06-14', authenticator=authenticator )
7 assistant.set_service_url('YOUR_SERVICE_URL')
8 response = assistant.message(   assistant_id='YOUR_ASSISTANT_ID',   session_id='YOUR_SESSION_ID',   input={
9     'message_type': 'text',
10    'text': 'Hello'
11  }
12 ).get_result()
13 print(response)
14
15
```

Make sure to replace `'YOUR_API_KEY'`, `'YOUR_SERVICE_URL'`, `'YOUR_ASSISTANT_ID'`, and `'YOUR_SESSION_ID'` with your actual values.