

300-835^{Q&As}

Automating Cisco Collaboration Solutions (CLAUTO)

Pass Cisco 300-835 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.leads4pass.com/300-835.html>

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Cisco
Official Exam Center

-  **Instant Download** After Purchase
-  **100% Money Back** Guarantee
-  **365 Days** Free Update
-  **800,000+** Satisfied Customers



QUESTION 1

What is a benefit of using Python virtual environments?

- A. It isolates dependencies of every project from the system and each other.
- B. It allows Python to differentiate between package versions.
- C. It frees the developer from installing the project dependencies.
- D. It puts dependent packages in a common site-packages directory.

Correct Answer: A

QUESTION 2

Which two capabilities can be implemented in a custom application using the Cisco Unified IP Phone Services API? (Choose two.)

- A. Authenticate the phone to the network.
- B. Display corporate directory information.
- C. Play multicast messages.
- D. Upgrade phone firmware.
- E. Create new phone devices.

Correct Answer: CE

QUESTION 3

DRAG DROP



Refer to the exhibit. A developer has implemented ChatOps to a Webex Teams space as described in the exhibit. The Python script that pushes the notifications to the Teams space is shown. Drag and drop the code to complete the script. Not all options are used.

Select and Place:

Answer Area

```
import requests, json, os

header = { [ ] : [ ] % os.environ.get("BOT_TOKEN"),
          "Content-Type": "application/json"}

message = "_The bot says: \n> I am a **developer** too!"

payload = {"roomId": os.environ.get("SPACE_ID"),
          "markdown": message}

res = requests.request([ ], url="https://api.ciscospark.com/v1/messages",
                      headers=header, data=[ ], verify=True)

if res.[ ] == 200:
    print("your message was successfully posted to Webex Teams")
else:
    print('failed with status code: %d' %res.status_code)

    if res.status_code == 404:
        print("please check the bot is in the space you're attempting to post to...")
    elif res.status_code == 400:
        print("please check the identifier of the space you're attempting to post to...")
    elif res.status_code == 401:
        print("please check if the access token is correct...")
```

json.dumps(payload)	"PUT"	status_code	payload
"Authentication"	"Authorization"	"Bearer"	status
"Bearer %s"	json(payload)	"POST"	code

Correct Answer:

Answer Area

```
import requests, json, os

header = { "Authentication" : "Bearer" % os.environ.get("BOT_TOKEN"),
          "Content-Type": "application/json"}

message = "_The bot says:\n> I am a **developer** too!"

payload = {"roomId": os.environ.get("SPACE_ID"),
          "markdown": message}

res = requests.request("POST", url="https://api.ciscospark.com/v1/messages",
                      headers=header, data=json.dumps(payload), verify=True)

if res.status_code == 200:
    print("your message was successfully posted to Webex Teams")
else:
    print('failed with status code: %d' %res.status_code)

    if res.status_code == 404:
        print("please check the bot is in the space you're attempting to post to...")
    elif res.status_code == 400:
        print("please check the identifier of the space you're attempting to post to...")
    elif res.status_code == 401:
        print("please check if the access token is correct...")
```

	"PUT"	payload
	"Authorization"	status
"Bearer %s"	json(payload)	code

QUESTION 4

```
1 POST /WBXService/XMLService/LstsummaryUser HTTP/1.1
2 Content-Type: application/xml
3 Host: api.webex.com
4 Content-Length: 974
5
6 <?xml version="1.0" encoding="UTF-8"?>2 <serv:message xmlns:serv="http://www.webex.com/schemas/2002/06/service"
7   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
8   <header>
9     <securityContext>
10      <siteName>apidemoeu</siteName>
11      <webExID>janedoe@example.com</webExID>
12      <sessionTicket>AAABb5LTcGcAABUYA0gAKEgyU</sessionTicket>
13    </securityContext>
14    <action>java.com.webex.service.binding.meeting.LstsummaryUser</action>
15  </header>
16  <body>
17    <method>java.com.webex.service.binding.meeting.LstsummaryUser</method>
18    <bodyContent xsi:type="java.com.webex.service.binding.meeting.LstsummaryUser">
19      <webExId>{WEBEXID}</webExId>
20    </bodyContent>
21  </body>
22 </serv:message>
```

Refer to the exhibit. A Webex Meetings XML API HTTP request message with several invalid portions is shown. Which reference points to a line in the exhibit that correctly indicates that this is a LstsummaryUser request?

- A. line 17
- B. line 1
- C. line 18
- D. line 14

Correct Answer: B

QUESTION 5

DRAG DROP

Drag and drop the code snippets into the correct order to create a Python Webex Teams REST API script that performs this sequence of tasks: create a Space, add a user to the Space, post the word "test" in the Space. Assume that the Python "requests" module has already been imported.

Select and Place:

Answer Area

Step 1

Step 2

Step 3

Step 4

Step 5

```
payload = { 'roomId: resp1.json()['id'], 'text': 'test' }  
resp2 = requests.post(url + '/messages', headers=headers, json=payload)
```

```
if resp2.status_code == 200:  
print('Success!')
```

```
headers = { 'Content-type': 'application/json',  
'Authorization': 'Bearer YTQzMzBiZmItNDJkNCO643-417974-ad72cae0e10f' }  
url = 'https://api.ciscospark.com/v1'
```

```
payload = { 'title': 'test' }  
resp1 = requests.post(url + '/rooms', headers=headers, json=payload)
```

```
payload = { 'roomId: resp1.json()['id'], 'personEmail': 'test@test.com' }  
requests.post(url + '/memberships', headers=headers, json=payload)
```

Correct Answer:

Answer Area

```
headers = { 'Content-type': 'application/json',  
'Authorization': 'Bearer YTQzMzBiZmItNDJkNCO643-417974-ad72cae0e10f' }  
url = 'https://api.ciscospark.com/v1'
```

```
payload = { 'roomId': resp1.json()['id'], 'personEmail': 'test@test.com' }  
requests.post(url + '/memberships', headers=headers, json=payload)
```

```
payload = { 'title': 'test' }  
resp1 = requests.post(url + '/rooms', headers=headers, json=payload)
```

```
payload = { 'roomId': resp1.json()['id'], 'text': 'test' }  
resp2 = requests.post(url + '/messages', headers=headers, json=payload)
```

```
if resp2.status_code == 200:  
print('Success!')
```