Question 1

Which two statements describe valid limitations for Webex Teams bot accounts in a group space? (Choose two.)

Options:

A- A bot account cannot send a message without an SSL certificate.

B- A bot account cannot hear a message unless the bot is @ mentioned.

C- A bot account cannot send a message without a valid access token.

D- A bot account cannot respond to another bot.

E- A bot account cannot send a message with an @mention.

Answer:

B, C

Explanation:

https://developer.webex.com/docs/bots

Question 2

Which XML snippet does an admin use with a Webex Meetings XML API request to retrieve info on multiple users?

Options:

A- <bodyContent xsi:type="java:com.webex.service.binding.user.GetUser">
<webExId>.*</webExId>
</bodyContent>

B- <bodyContent xsi:type="java:com.webex.service.binding.user.LstsummaryUser">
<webExId>.*</webExId>
</bodyContent>

C- <bodyContent xsi:type="java:com.webex.service.binding.user.GetUser">
</bodyContent>

D- <bodyContent xsi:type="java:com.webex.service.binding.user.LstsummaryUser">
</bodyContent>

Answer:

A

Explanation:

https://developer.cisco.com/docs/webex-xml-api-reference-guide/#!getuser

Question 3

Question Type: **MultipleChoice**

Refer to the exhibit.

```
1  const request = require('request-promise').defaults({
2      json: true,
3      baseUrl: 'https://api.ciscospark.com/v1/'
4  })
5
6  const dontsay = ['swag', 'fleek', 'stay', 'shade', 'bae', 'yas', 'lit', 'ratchet']
7
8   function isSensitive(text) {
9       sensitive = false
10      for (j = 0 ; j < dontsay.length ; j++) {
11          if (text && text.includes(dontsay[j])) {
12          sensitive = true
13        }
14      }
15      return sensitive
16  }
17
18   function getMessages() {
19       request.get({
20           url: "..............................." ,
21           headers: {Authorization: 'Bearer ${compliance_toekn}'}
22       }).then(events => {
23          for (j = 0; i < events.items.length; i++) {
24            if (events.items[i[.data.text && isSensitive(events.items[i].data.text)) {
25                request.post({
26                  uri: 'messages',
27                  headers: {Authorization: 'Bearer $}bot_token}'},
28                  body: {
29                        'toPersonId' : '${events.items[i].actorId}',
30                        'markdown': 'Please don't say **${events.items[i].data.text)** ever again..'
31                  },
32                })
33              }
34            }
35      })
36  )
```

A company uses Webex Teams extensively for communications involving customers, and want to enforce a consistent messaging policy. Which code completes line 20 to send a notification when noncompliant messages are detected?

Options:

A- events?resource=people

B- compliance?resource=messages

C- events?resource=messages

D- compliance?resource=people

Answer:

C

Explanation:

The function is getmessages. Then a request is made so the get comes in action and does events? Resource=messages to get the data for the events.

Question 4

Which code adds a Space Widget in an HTML script that uses the CSS Webex CDN?

Options:

A- <link rel= "stylesheet" href= "https://code.s4d.io/widget-recents/production/main.css">

B- <script src= "stylesheet" href= "https://code.s4d.io/widget-space/production/main.css">

C- <link rel= "stylesheet" href= "https://code.s4d.io/widget-recents/production/bundle.js">

D- <link rel= "stylesheet" href= "https://code.s4d.io/widget-space/production/main.css">

Answer:

D

Explanation:

https://developer.webex.com/docs/widgets

Question 5

Which user role implicitly has permission in the Events API to retrieve "message created" events from other users in the same organization?

Options:

A- Compliance Officer

B- User and Device Administrator

C- Full Administrator

D- User Administrator

Answer:

A

Explanation:

https://developer.webex.com/docs/api/guides/compliance

Question 6

Refer to the exhibit.

```
xapi.on ('ready', () => {
    console.log ("connection successful");

    // Start a call

    *_____*


            .then ((call) => {
                console.log ('Started call with status: ${call.status}, id: ${call.CallId}');
```

A.
```
xapi.command ('Dial', {Endpoint: 'jdoe@example.com'})
```

B.
```
xapi.command ('Call', { SIP: 'jdoe@example.com'})
```

C.
```
xapi.command ('SIP', {Number: 'jdoe@example.com'})
```

D.
```
xapi.command ('Dial', {Number: 'jdoe@example.com'})
```

Options:

A- Option A

B- Option B

C- Option C

D- Option D

Answer:

D

Explanation:

https://github.com/CiscoDevNet/labs-xapi/pull/12/ files/40c9fc38e115f4dec4709836501a165008caeaba