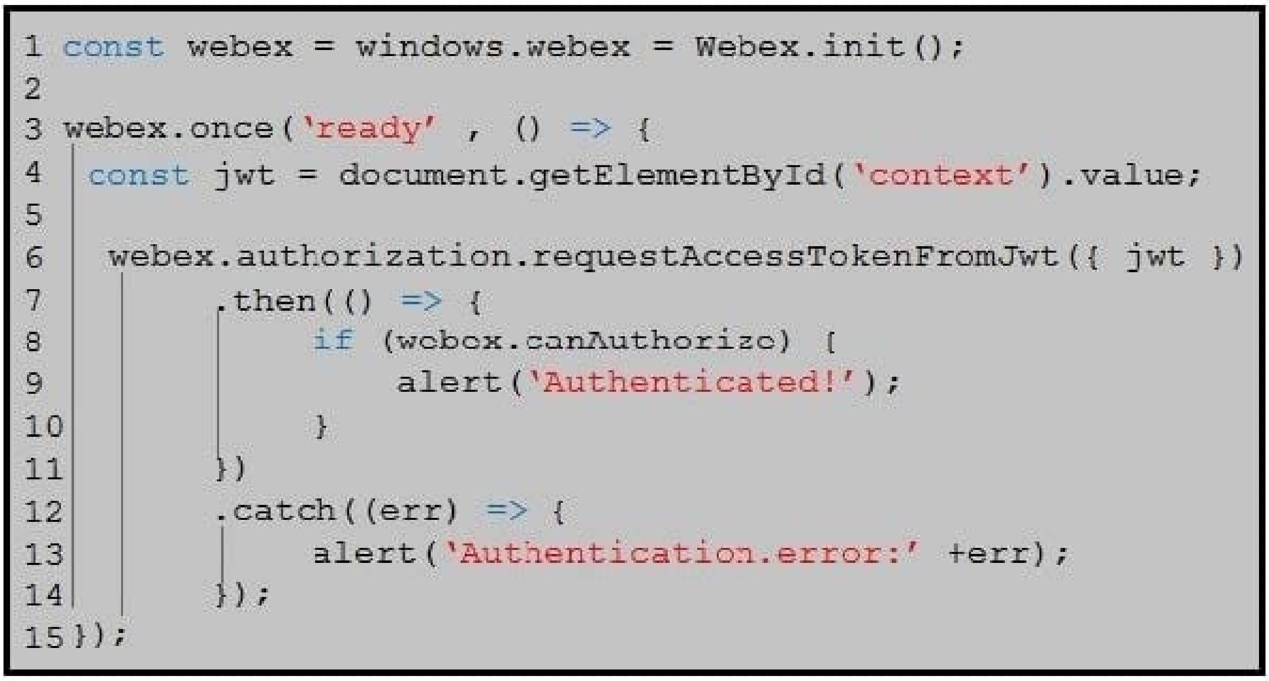
300-920 Q&As

Developing Applications for Cisco Webex and Webex Devices (DEVWBX)

QUESTION 1



```
1 const webex = windows.webex = Webex.init();
2
3 webex.once('ready' , () => {
4   const jwt = document.getElementById('context').value;
5
6    webex.authorization.requestAccessTokenFromJwt({ jwt })
7            .then(() => {
8                  if (webex.canAuthorize) {
9                        alert('Authenticated!');
10                 }
11           })
12           .catch((err) => {
13                 alert('Authentication.error:' +err);
14           });
15 });
```

Refer to the exhibit. On line 4, the script retrieves a context from a DOM element that was generated from a server-side component. How does that server-side component obtain the value for the `context\\' element?

A. by opening a dialog asking the end-user to paste his personal access token

B. by completing an authorization code grant flow using the identifier and secret of an OAuth integration

C. by embedding the access token of a Bot account

D. by creating a guest token using the identifier and secret of a Guest Issuer application

Correct Answer: B

QUESTION 2

```
<xsd:complexType name= "lstRecordingResponse">
     <xsd:complexContent>
          <xsd:extension base= "serv:bodyContentType">
               <xsd:sequence>
                    <xsd:element name= "matchingRecords" type=
"serv:matchingRecordsType" minOccurs="0"/>
                    <xsd:element name= "recording" type= "ep:recordingType" minOccurs=
"0" maxOccurs= "unbounded"/>
...
<xsd:complexType name= "recordingType">
     <xsd:sequence>
          <xsd:element name= "recordingID" type= "xsd:int"/>
          <xsd:element name= "hostWebExID" type= "xsd:string"/>
          <xsd:element name= "name" type= "xsd:string"/>
          <xsd:element name= "createTime" type= "xsd:string"/>
          <xsd:element name= "sreamURL" type= "xsd:string"/>
          <xsd:element name= "fileURL" type= "xsd:string"/>
          <xsd:element name= "password" type= "xsd:string" minOccurs= "0" />
```

Refer to the exhibit. A snippet from the XSD schema of the Webex Meeting XML API `LstRecordingResponse\\' element is listed in the exhibit. Assuming that a variable named `resp\\' exists that contains the XML response from a successful `LstRecording\\' request, which code snippet correctly generates a simple report that lists meeting names and recording file download links?

A.

```
list = (new window.DOMParser()).parseFromString(resp, 'application/json')
for (const item of lst.getElementsByTagNameNS('*', 'matchingRecords')){
document.writeln(
 <p> Meeting Name:${item.getElementsByTagNameNS('*', 'name') [0].textContent}<br>');
document.write(
'Download:${item.getElementsByTagNameNS('*', 'fileURL') [0].textContent}<br>');
```

B.

```
list = (new window.DOMParser()).parseFromString(resp, 'LstRecordingResponse')
for (const item of lst.getElementsByTagNameNS('*', 'matchingRecords')){
document.writeln(
 <p> Meeting Name:${item.getElementsByTagNameNS('*', 'name') [0].textContent}<br>');
document.write(
'Download:${item.getElementsByTagNameNS('*', 'fileURL') [0].textContent}<br>');
```

C.

```
list = (new window.DOMParser()).parseFromString(resp, 'text/xml')
for (const item of lst.getElementsByTagNameNS('*', 'matchingRecords')){
document.writeln(
 <p> Meeting Name:${item.getElementsByTagNameNS('*', 'hostWebExID') [0] }<br>');
document.write(
'Download:${item.getElementsByTagNameNS('*', 'streamURL') [0].textContent}<br>');
```

D.

```
list = (new window.DOMParser()).parseFromString(resp, 'text/xml')
for (const item of lst.getElementsByTagNameNS('*', 'recording')){
document.writeln(
 <p> Meeting Name:${item.getElementsByTagNameNS('*', 'name') [0].textContent}<br>');
document.write(
'Download:${item.getElementsByTagNameNS('*', 'fileURL') [0].textContent}<br>');
```

A. Option A

B. Option B

C. Option C

D. Option D

Correct Answer: A

QUESTION 3

A company wants to adopt Webex Teams as a messaging platform and use REST APIs to automate the creation of teams and rooms. Which sequence of REST API requests is needed to create and populate a new Webex team and create a populated Webex room for the team?

A. POST /teams, POST /memberships, POST /rooms

B. POST /teams, POST /people, POST /rooms

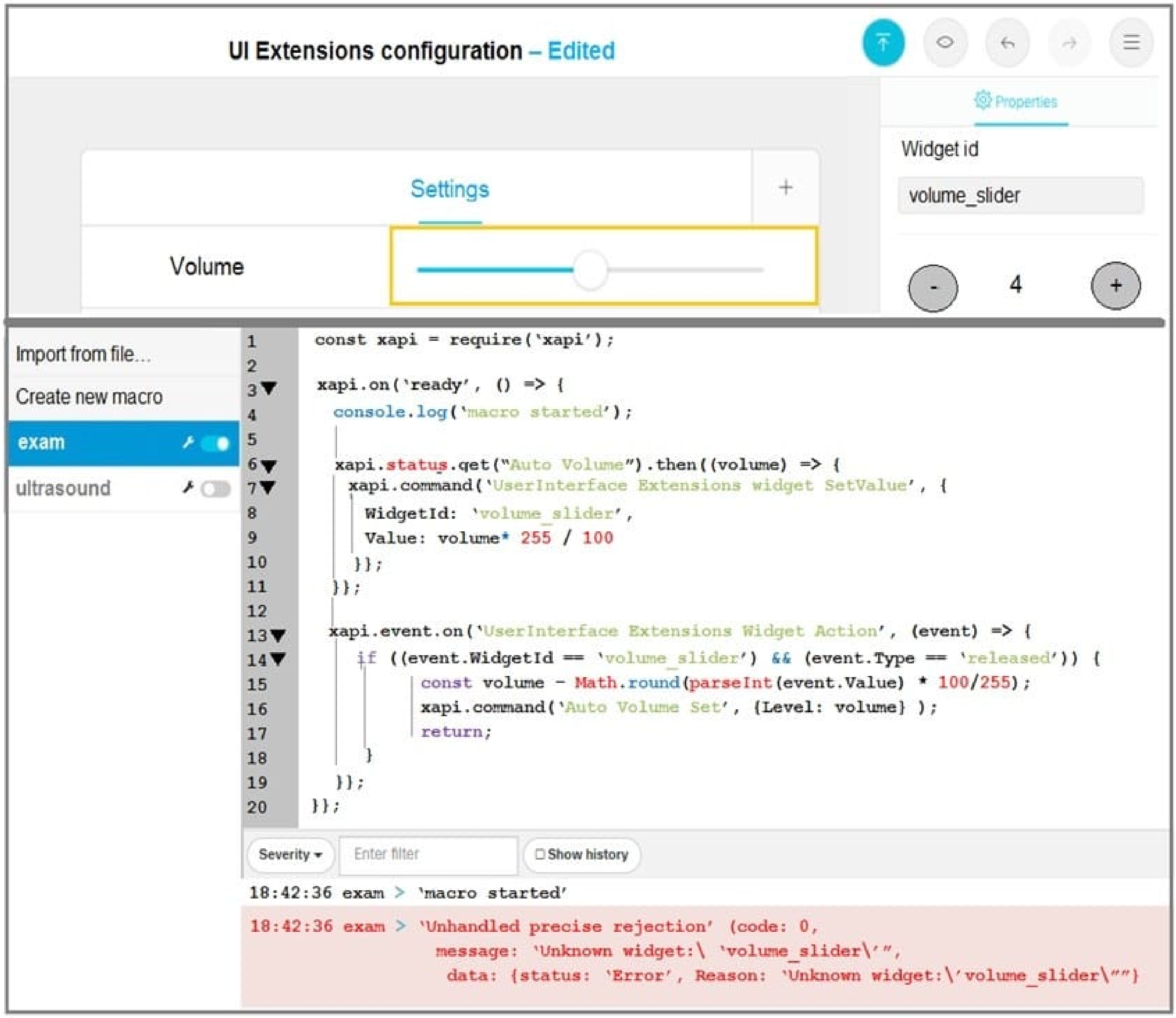
C. POST /teams, POST /team/memberships, POST /rooms

D. POST /teams, POST /team/memberships, POST /rooms, POST /memberships

Correct Answer: B

Reference: https://developer.webex.com/docs/api/basics

QUESTION 4



```
const xapi = require('xapi');

xapi.on('ready', () => {
  console.log('macro started');

  xapi.status.get("Auto Volume").then((volume) => {
    xapi.command('UserInterface Extensions widget SetValue', {
      WidgetId: 'volume_slider',
      Value: volume* 255 / 100
    });
  });

  xapi.event.on('UserInterface Extensions Widget Action', (event) => {
    if ((event.WidgetId == 'volume_slider') && (event.Type == 'released')) {
        const volume - Math.round(parseInt(event.Value) * 100/255);
        xapi.command('Auto Volume Set', {Level: volume} );
        return;
    }
  });
});
```

```
18:42:36 exam > 'macro started'
18:42:36 exam > 'Unhandled precise rejection' (code: 0,
                  message: 'Unknown widget:\ 'volume_slider\'",
                   data: {status: 'Error', Reason: 'Unknown widget:\'volume_slider\""}
```

Refer to the exhibit. A macro and a UI extension (also called In-Room Control) are being developed. What is the reason for the error displayed in the console?

A. Widgets of type "Slider" are not supported on the device.

B. The UI extension was not exported to the device.

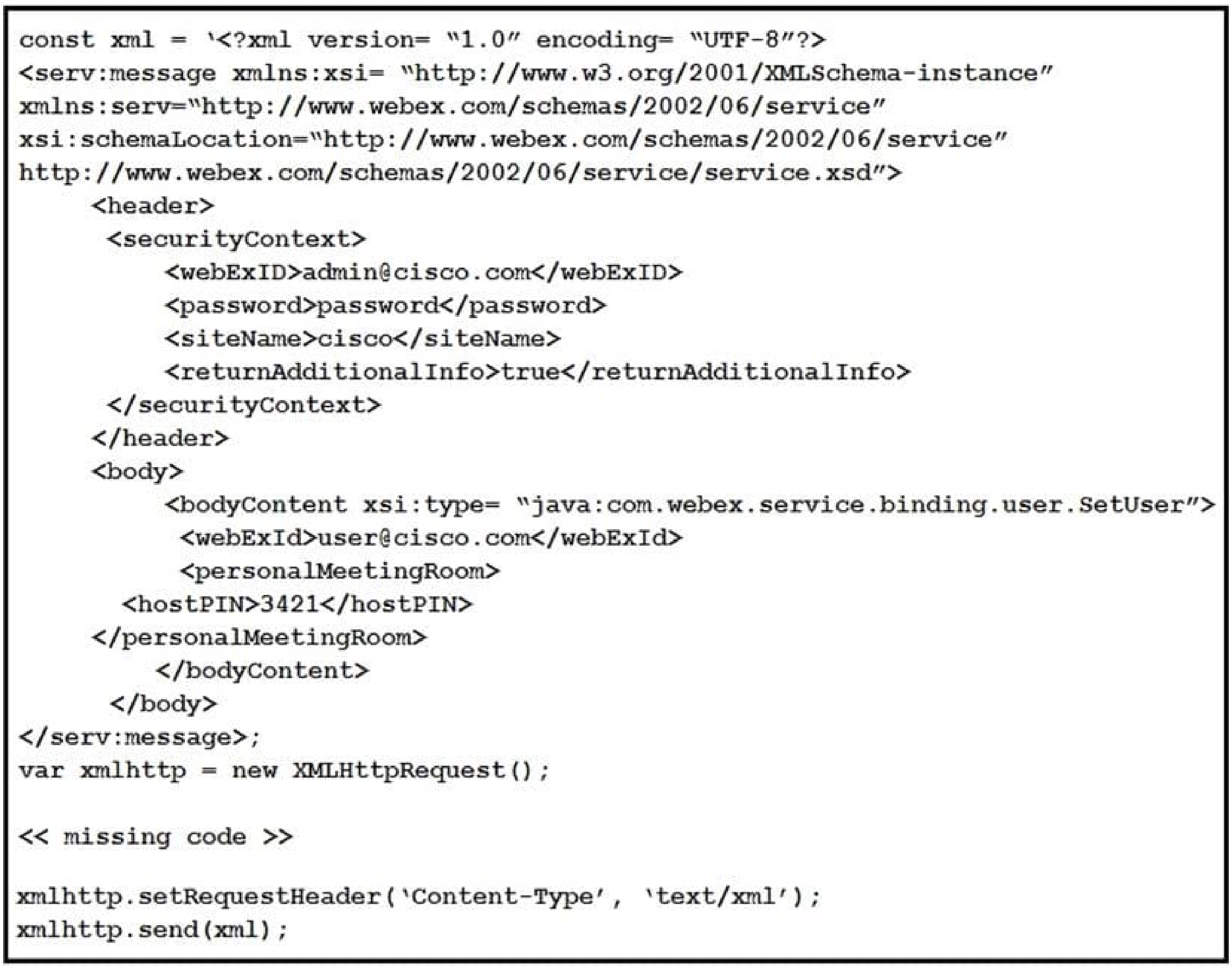
C. The name of the widget in the macro and the UI extension must match.

D. Promises are not supported for this device.

Correct Answer: B

Reference: https://www.cisco.com/c/dam/en/us/td/docs/telepresence/endpoint/ce99/webex-board-administrator-guide-ce99.pdf

QUESTION 5



```
const xml = '<?xml version= "1.0" encoding= "UTF-8"?>
<serv:message xmlns:xsi= "http://www.w3.org/2001/XMLSchema-instance"
xmlns:serv="http://www.webex.com/schemas/2002/06/service"
xsi:schemaLocation="http://www.webex.com/schemas/2002/06/service"
http://www.webex.com/schemas/2002/06/service/service.xsd">
     <header>
      <securityContext>
          <webExID>admin@cisco.com</webExID>
          <password>password</password>
          <siteName>cisco</siteName>
          <returnAdditionalInfo>true</returnAdditionalInfo>
      </securityContext>
     </header>
     <body>
          <bodyContent xsi:type= "java:com.webex.service.binding.user.SetUser">
           <webExId>user@cisco.com</webExId>
           <personalMeetingRoom>
       <hostPIN>3421</hostPIN>
     </personalMeetingRoom>
         </bodyContent>
      </body>
</serv:message>;
var xmlhttp = new XMLHttpRequest();

<< missing code >>

xmlhttp.setRequestHeader('Content-Type', 'text/xml');
xmlhttp.send(xml);
```

Refer to the exhibit. A developer must construct an HTTP Request to use the XML API to set a Personal Meeting Room PIN for a given user. Which code completes the code to create the request?

A. xmlhttp.open("GET", "https://cisco.webex.com/WBXService/XMLService");

B. xmlhttp.open("PATCH", "https://cisco.webex.com/WBXService/XMLService");

C. xmlhttp.open("PUT", "https://cisco.webex.com/WBXService/XMLService");

D. xmlhttp.open("POST", "https://cisco.webex.com/WBXService/XMLService");

Correct Answer: D

The post method can be used for HTTP request that sets up a personal metting room PIN for a user.

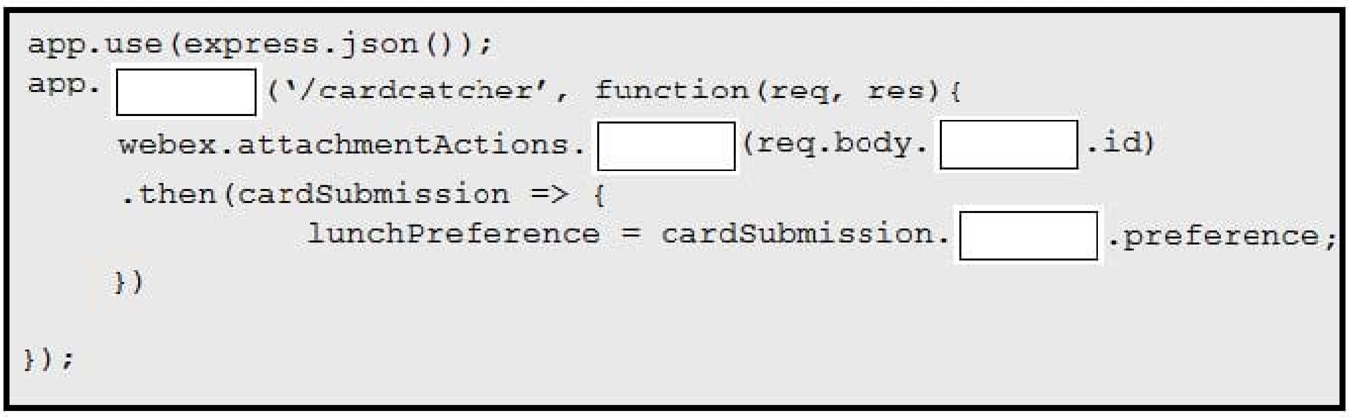
QUESTION 6

DRAG DROP

The Express framework receives a Webex Teams webhook event when someone presses a button in an Adaptive

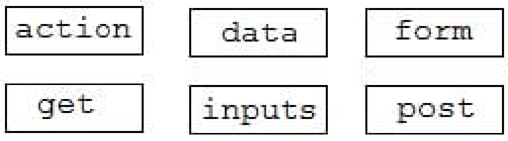
Card. Drag and drop the property names onto the code snippet to access the form field named “preference”. Not all options are used.

Select and Place:

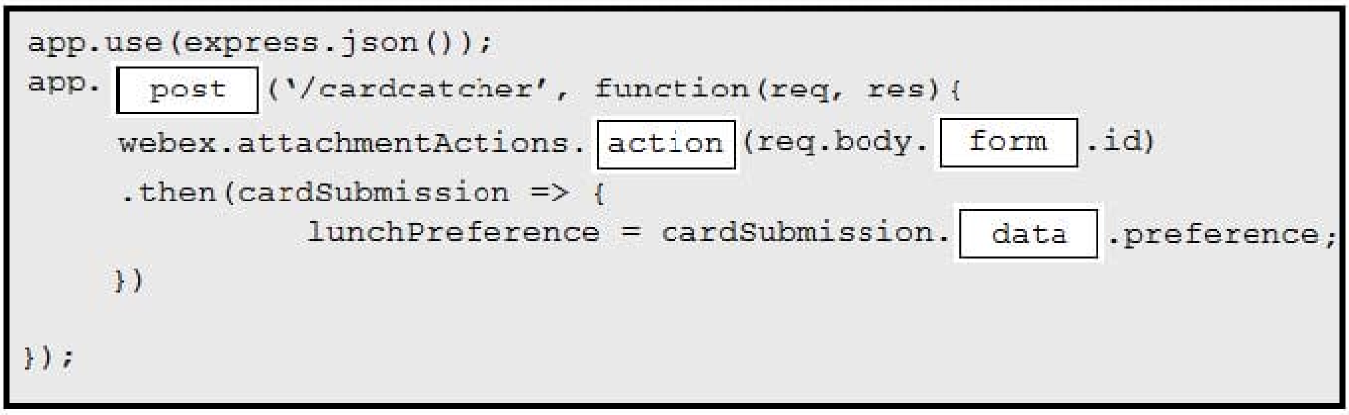


```
app.use(express.json());
app.________('/cardcatcher', function(req, res){
     webex.attachmentActions.________(req.body.________.id)
     .then(cardSubmission => {
               lunchPreference = cardSubmission.________.preference;
     })

});
```

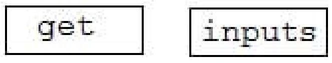


Correct Answer:



```
app.use(express.json());
app.post('/cardcatcher', function(req, res){
     webex.attachmentActions.action(req.body.form.id)
     .then(cardSubmission => {
               lunchPreference = cardSubmission.data.preference;
     })

});
```

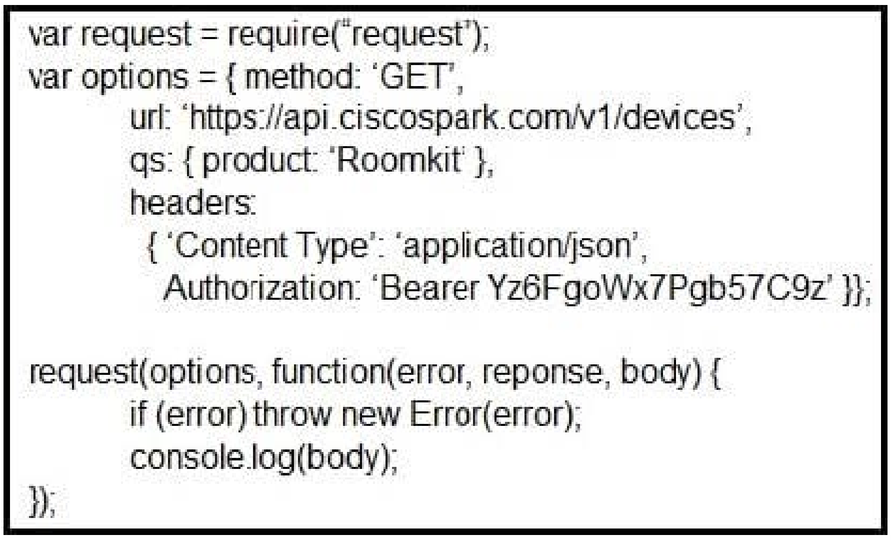


Reference: https://github.com/marchfederico/ciscospark-websocket-events

QUESTION 7

Which REST API request is used to list all the Webex Room Kit devices within a large organization so that a new custom In-Room Control can be deployed on all the devices?

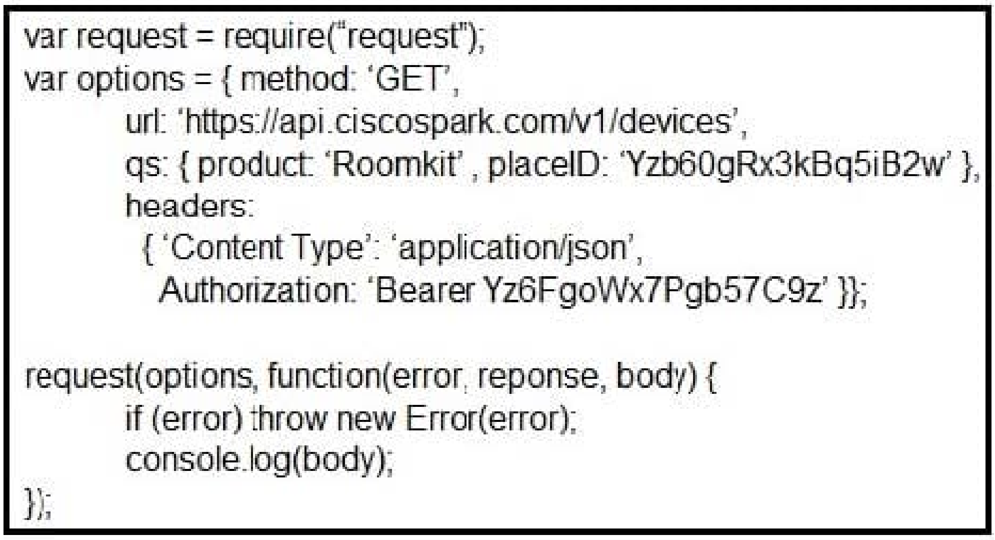
A.



```
var request = require("request");
var options = { method: 'GET',
          url: 'https://api.ciscospark.com/v1/devices',
          qs: { product: 'Roomkit' },
          headers:
            { 'Content Type': 'application/json',
              Authorization: 'Bearer Yz6FgoWx7Pgb57C9z' }};

request(options, function(error, reponse, body) {
          if (error) throw new Error(error);
          console.log(body);
});
```

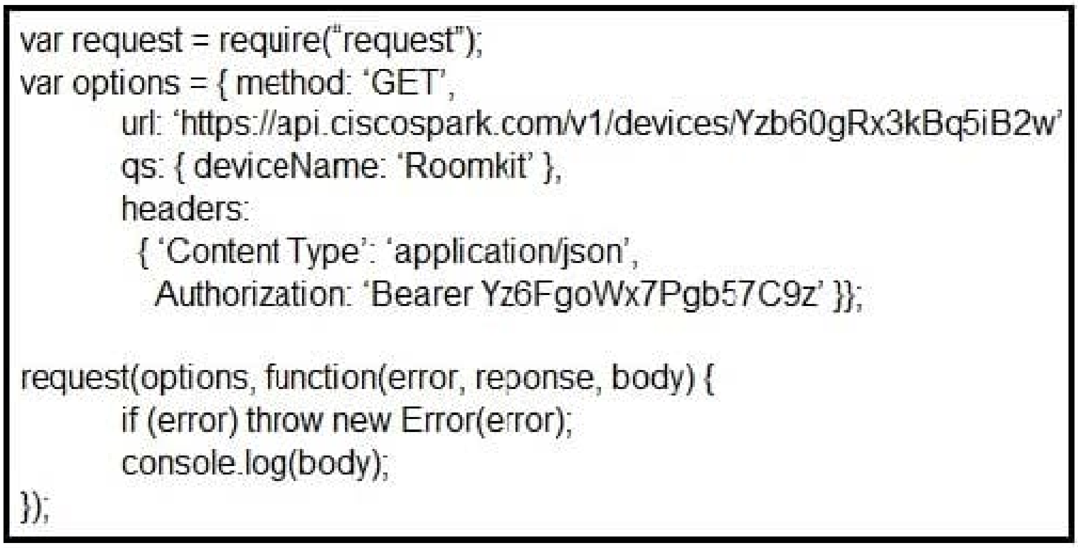
B.



```
var request = require("request");
var options = { method: 'GET',
          url: 'https://api.ciscospark.com/v1/devices',
          qs: { product: 'Roomkit' , placeID: 'Yzb60gRx3kBq5iB2w' },
          headers:
            { 'Content Type': 'application/json',
              Authorization: 'Bearer Yz6FgoWx7Pgb57C9z' }};

request(options, function(error, reponse, body) {
          if (error) throw new Error(error);
          console.log(body);
});
```

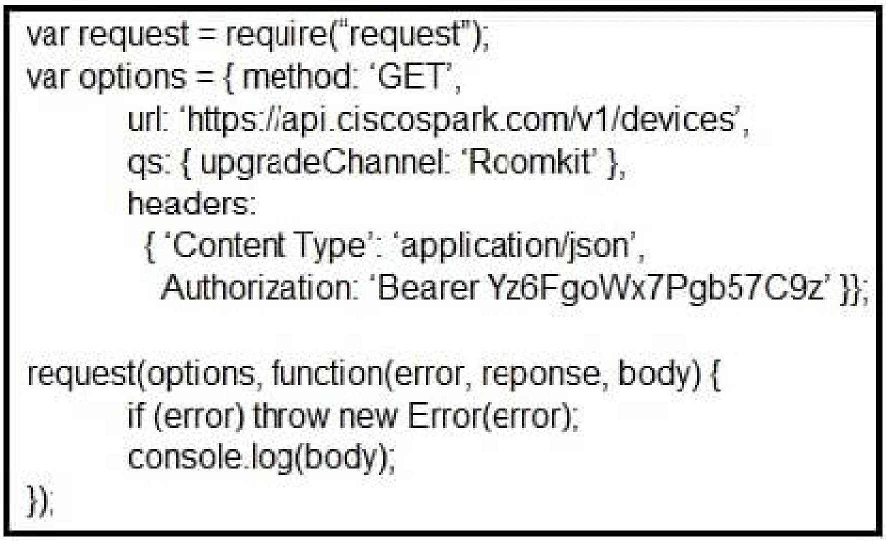
C.



```
var request = require("request");
var options = { method: 'GET',
          url: 'https://api.ciscospark.com/v1/devices/Yzb60gRx3kBq5iB2w'
          qs: { deviceName: 'Roomkit' },
          headers:
           { 'Content Type': 'application/json',
             Authorization: 'Bearer Yz6FgoWx7Pgb57C9z' }};

request(options, function(error, reponse, body) {
          if (error) throw new Error(error);
          console.log(body);
});
```

D.



```
var request = require("request");
var options = { method: 'GET',
          url: 'https://api.ciscospark.com/v1/devices',
          qs: { upgradeChannel: 'Roomkit' },
          headers:
           { 'Content Type': 'application/json',
             Authorization: 'Bearer Yz6FgoWx7Pgb57C9z' }};

request(options, function(error, reponse, body) {
          if (error) throw new Error(error);
          console.log(body);
});
```

A. Option A

B. Option B

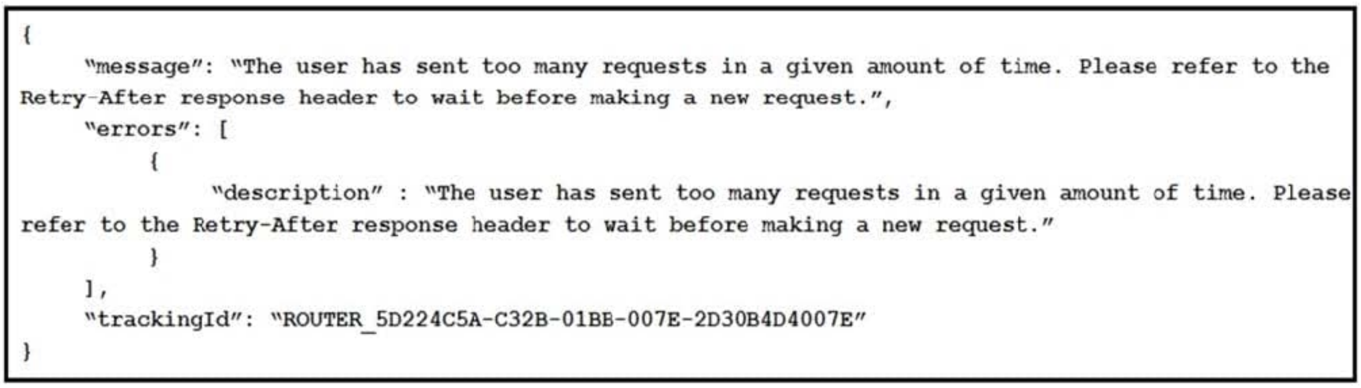
C. Option C

D. Option D

Correct Answer: A

The qs: option is required to list all roomkit devices. Product: `RoomKit\\' is the correct option because it will list all roomkit devices.

QUESTION 8


```
{
     "message": "The user has sent too many requests in a given amount of time. Please refer to the
Retry-After response header to wait before making a new request.",
     "errors": [
          {
               "description" : "The user has sent too many requests in a given amount of time. Please
refer to the Retry-After response header to wait before making a new request."
          }
     ],
     "trackingId": "ROUTER_5D224C5A-C32B-01BB-007E-2D30B4D4007E"
}
```

Refer to the exhibit. What is the Webex Teams REST API HTTP response status code, based on this code snippet?

A. 401

B. 403

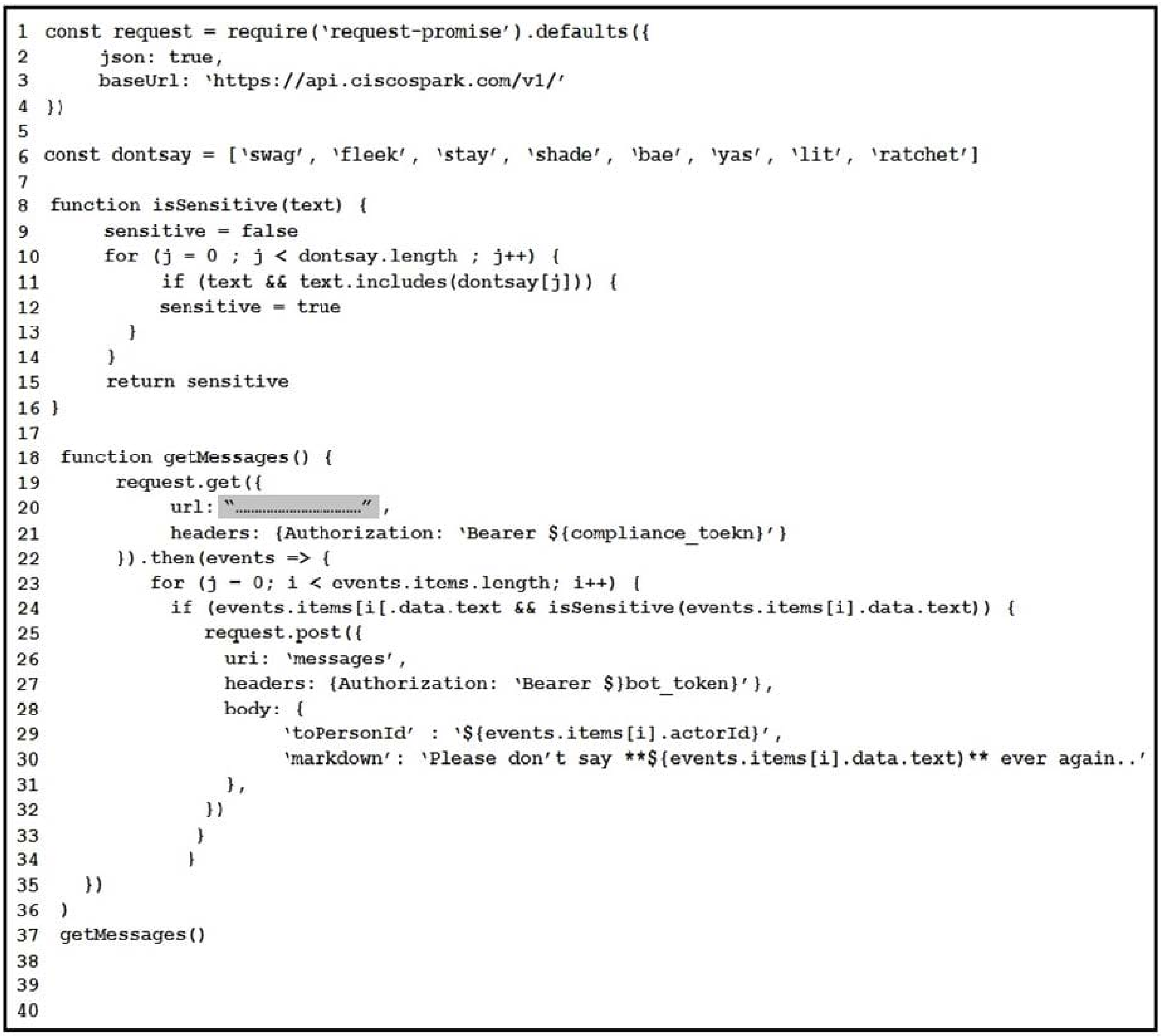
C. 429

D. 501

Correct Answer: C

Reference: https://developer.webex.com/docs/api/v1/messages/get-message-details

QUESTION 9



```
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  )
37  getMessages()
38
39
40
```

Refer to the exhibit. 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?

A. events?resource=people

B. compliance?resource=messages

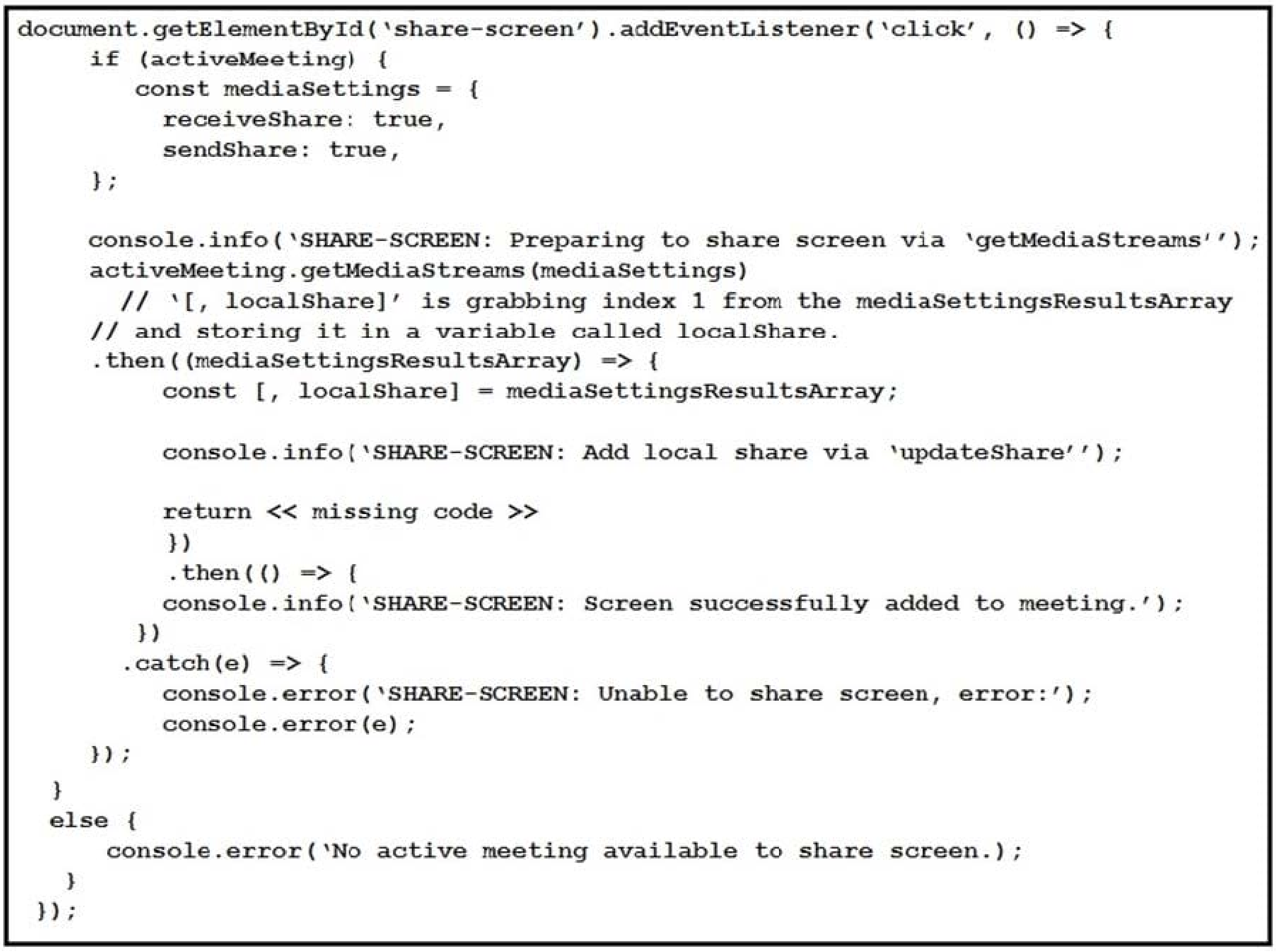
C. events?resource=messages

D. compliance?resource=people

Correct Answer: C

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 10



```
document.getElementById(‘share-screen’).addEventListener(‘click’, () => {
     if (activeMeeting) {
        const mediaSettings = {
          receiveShare: true,
          sendShare: true,
     };

     console.info(‘SHARE-SCREEN: Preparing to share screen via ‘getMediaStreams’’);
     activeMeeting.getMediaStreams(mediaSettings)
       // ‘[, localShare]’ is grabbing index 1 from the mediaSettingsResultsArray
     // and storing it in a variable called localShare.
     .then((mediaSettingsResultsArray) => {
          const [, localShare] = mediaSettingsResultsArray;

          console.info(‘SHARE-SCREEN: Add local share via ‘updateShare’’);

          return << missing code >>
          })
          .then(() => {
          console.info(‘SHARE-SCREEN: Screen successfully added to meeting.’);
        })
       .catch(e) => {
          console.error(‘SHARE-SCREEN: Unable to share screen, error:’);
          console.error(e);
     });
  }
  else {
      console.error(‘No active meeting available to share screen.);
   }
 });
```

Refer to the exhibit.Which code completes the return statement that initiates local screen sharing on the active meeting?

A. activeMeeting.updateShare({ sendShare: true receiveShare: true, stream: null

})

B. activeMeeting.updateShare({ sendShare: true receiveShare: false, stream: remoteShare })

C. activeMeeting.updateShare({ sendShare: true receiveShare: true, stream: localShare })

D. activeMeeting.updateShare({ sendShare: false receiveShare: false, stream: null })

Correct Answer: C

Reference: https://github.com/webex/webex-js-sdk/tree/master/packages/node_modules/%40webex/plugin-meetings

[Latest 300-920 Dumps] [300-920 VCE Dumps] [300-920 Braindumps]