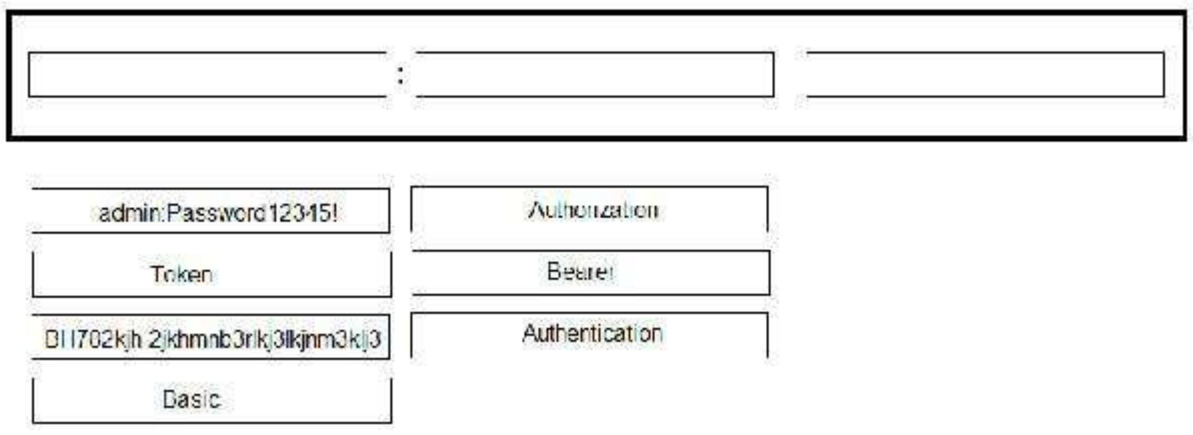
- **Vendor:** Cisco
- **Exam Code:** 300-920
- **Exam Name:** Developing Applications for Cisco Webex and Webex Devices (DEVWBX)
- **New Updated Questions from Braindump2go (Updated in April/2020)**

Exam A

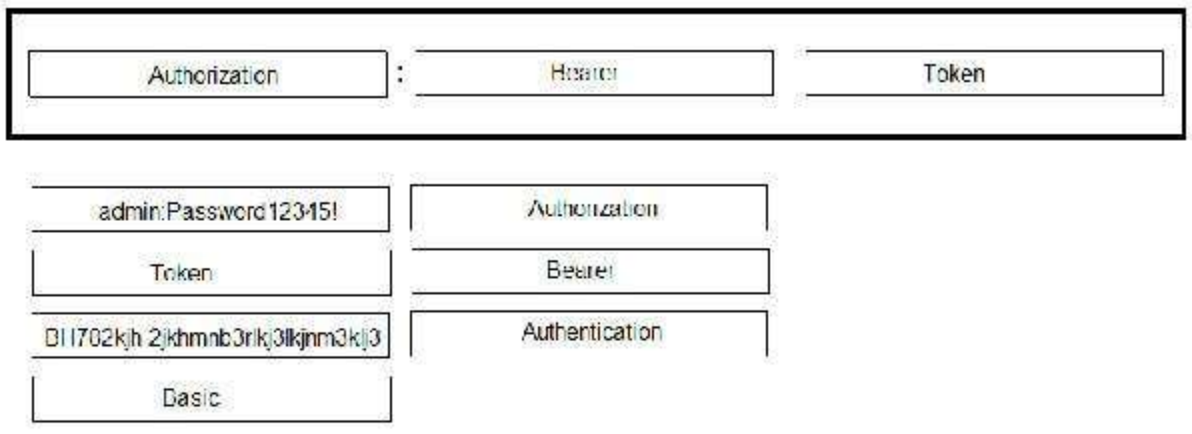
QUESTION 1
DRAG DROP

Drag and drop the components to create the authentication header used for the Webex Teams API. Not all options are used.

Select and Place:



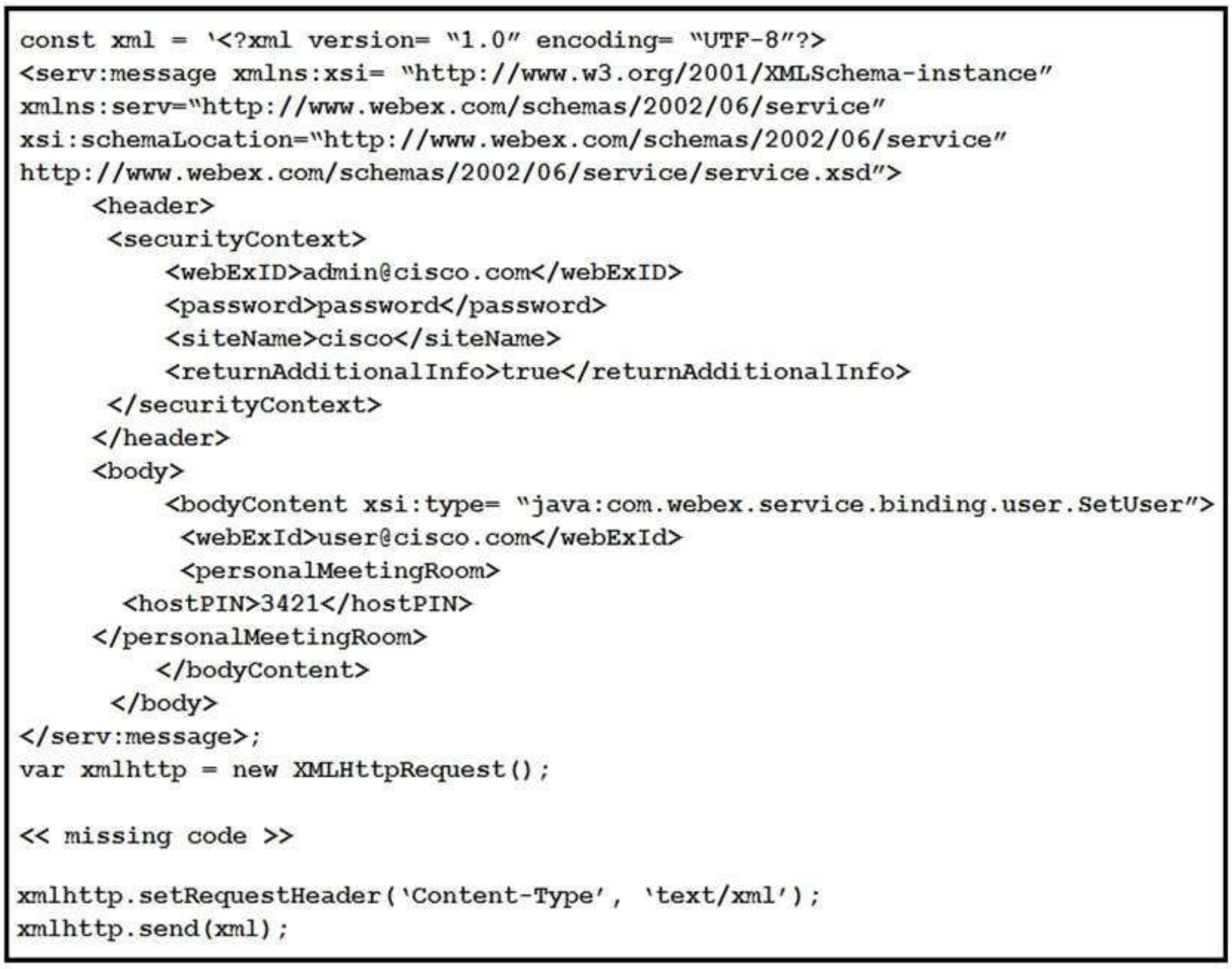
Correct Answer:



Explanation

Explanation/Reference:
Reference: https://developer.webex.com/docs/guest-issuer

QUESTION 2



```
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
Explanation

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

QUESTION 3
Which expression is a valid Webex Teams webhook filter?

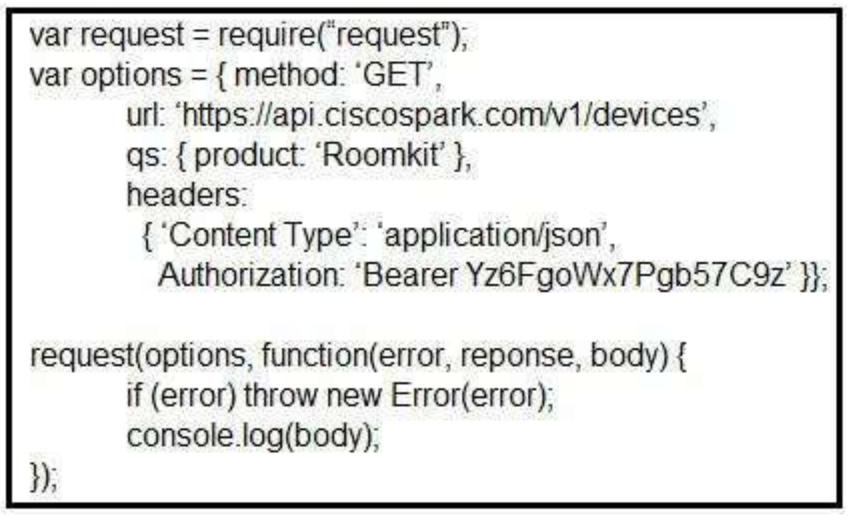
A. personEmail=person@example.com+roomId=abc123
B. personEmail=person@example.com-roomId=abc123
C. personEmail=person@example.com&roomId=abc123
D. personEmail=person@example.com,roomId=abc123

Correct Answer: C
Explanation

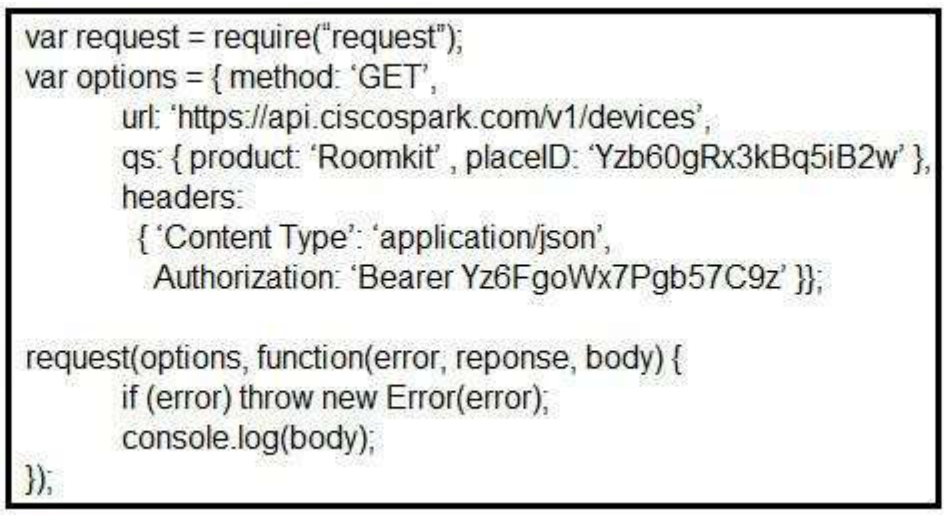
Explanation/Reference:
Explanation:
You can also use more than one filter in a webhook. To use multiple filters, combine them with the “&” symbol. For example, to create a webhook that only sends notifications when a specific person performs an action in a specific room, such as sending a message or creating a membership, combine the personEmail and roomId filters.

Reference: https://developer.webex.com/docs/api/guides/webhooks

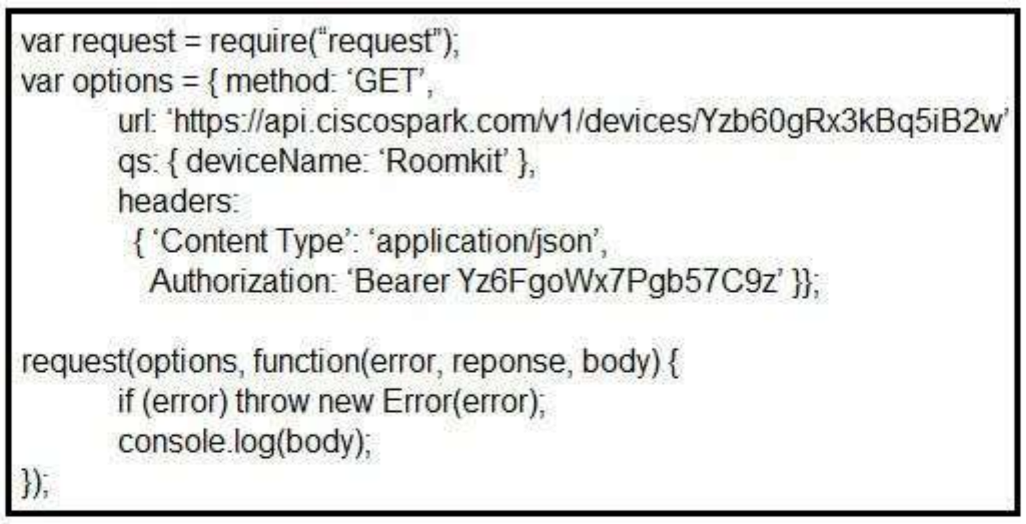
QUESTION 4
Which REST API request is used to list all the Webex Room Kit devices within a large organization so that a new custom In-Room Control can be deployed on all the devices?

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);
});
```

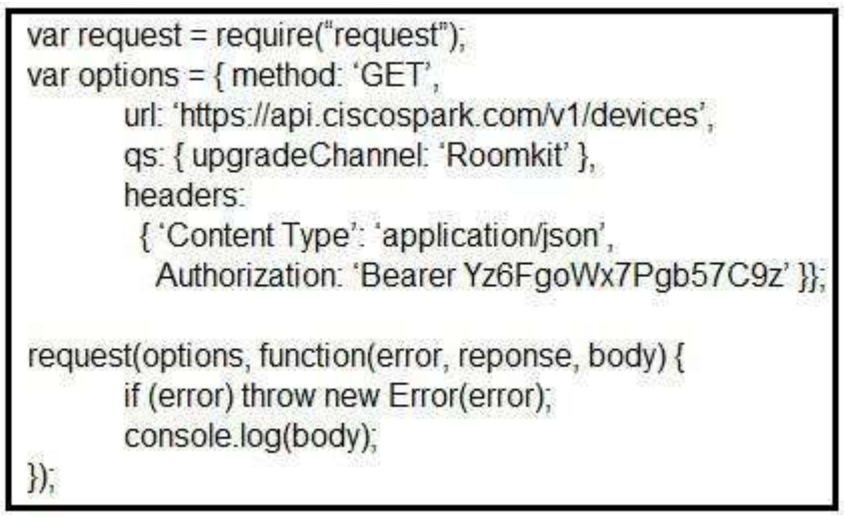
B.


```
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);
});
```

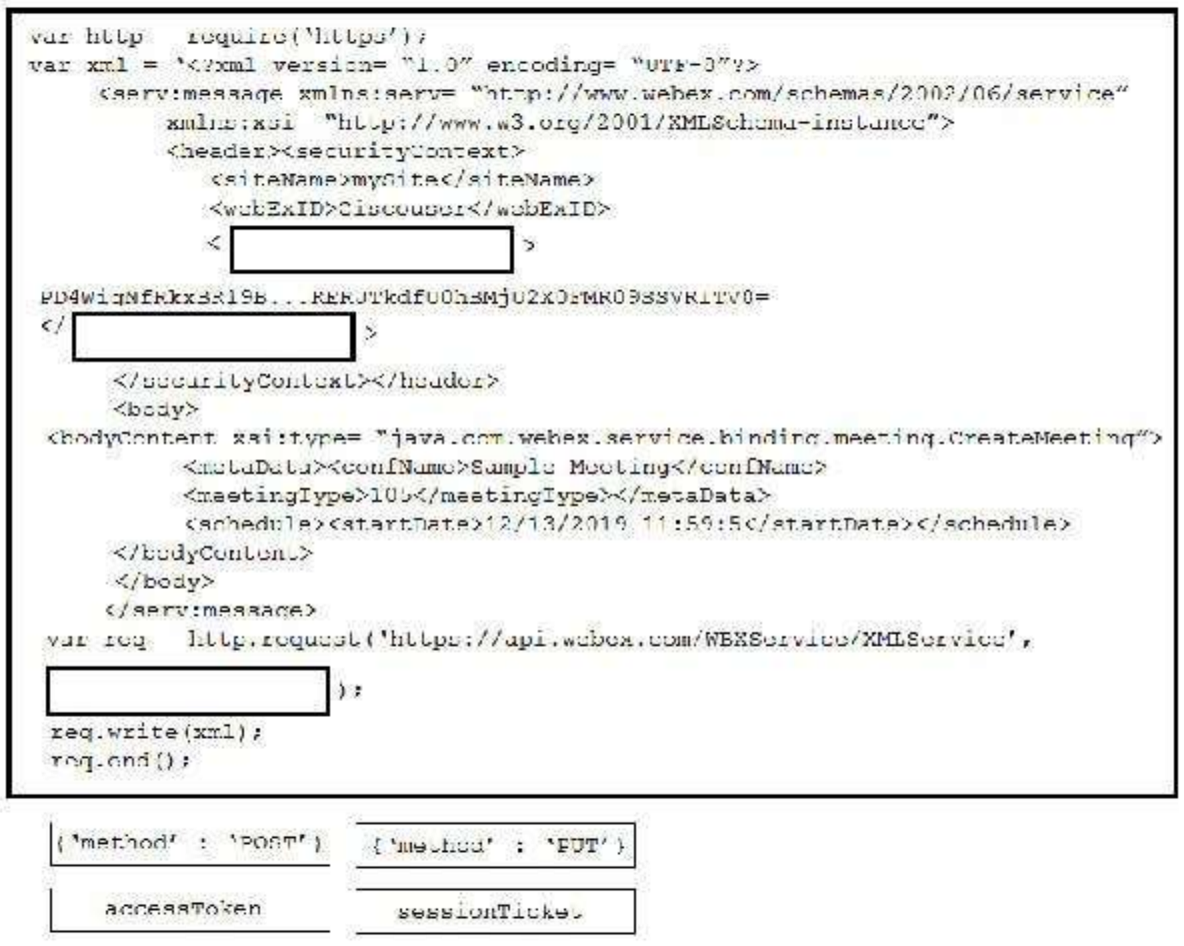
Correct Answer: A
Explanation

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

QUESTION 5
DRAG DROP

Drag and drop the code onto the snippet to construct the JavaScript to create a new meeting with the Webex Meetings XML API. Options can be used more than once.

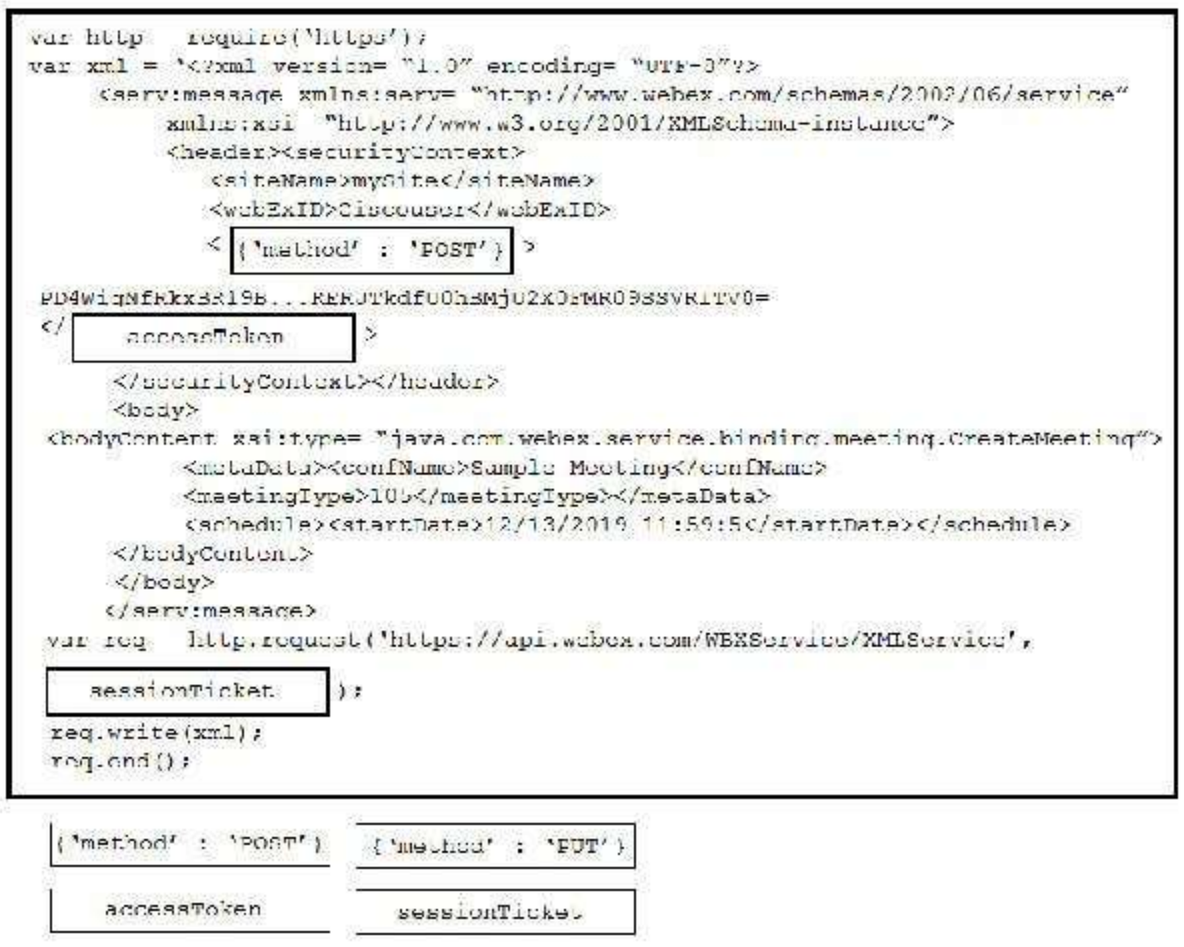
Select and Place:



```
var http   require('https');
var xml = '<?xml version= "1.0" encoding= "UTF-8"?>
     <serv:message xmlns:serv= "http://www.webex.com/schemas/2002/06/service"
          xmlns:xsi  "http://www.w3.org/2001/XMLSchema-instance">
          <header><securityContext>
             <siteName>mySite</siteName>
             <webExID>Ciscouser</webExID>
             <                    >
 PD4WiqNfRkx3R19B...RERJTkdfU0h3Mju2X0rMR09SSVRITV0=
 </                    >
        </securityContext></header>
        <body>
 <bodyContent xsi:type= "java.com.webex.service.binding.meeting.CreateMeeting">
             <metaData><confName>Sample Meeting</confName>
             <meetingType>105</meetingType></metaData>
             <schedule><startDate>12/13/2019 11:59:5</startDate></schedule>
        </bodyContent>
        </body>
     </serv:message>
 var req   http.request('https://api.webex.com/WBXService/XMLService',
                      );
 req.write(xml);
 req.end();
```

('method' : 'POST')	{'method' : 'PUT'}
accessToken	sessionTicket

Correct Answer:



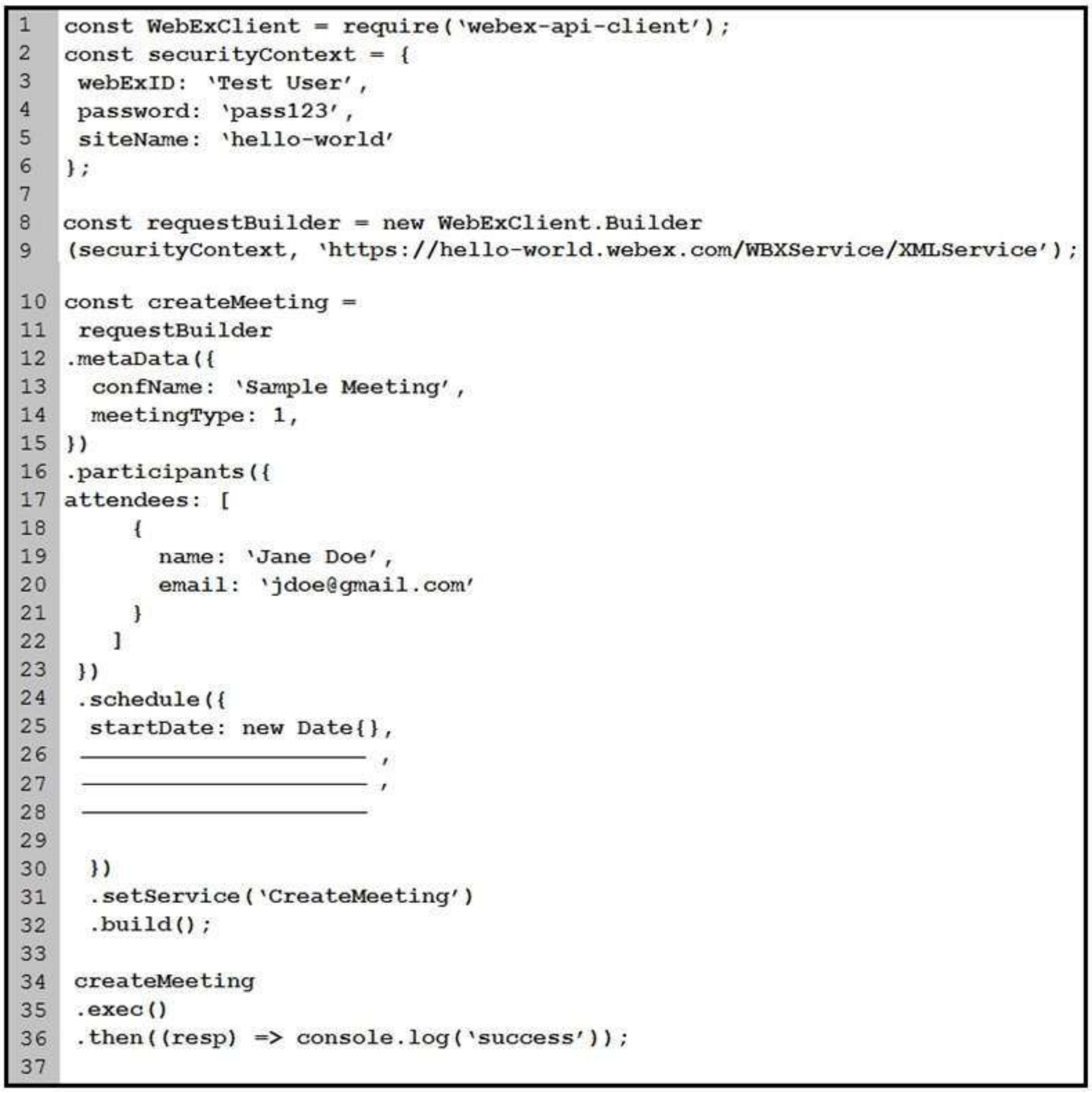
```
var http   require('https');
var xml = '<?xml version= "1.0" encoding= "UTF-8"?>
     <serv:message xmlns:serv= "http://www.webex.com/schemas/2002/06/service"
          xmlns:xsi  "http://www.w3.org/2001/XMLSchema-instance">
          <header><securityContext>
             <siteName>mySite</siteName>
             <webExID>Ciscouser</webExID>
             < {'method' : 'POST'} >
 PD4WiqNfRkxBR19B...RERJTkdfU0hBMjU2XDFMR09SSVRITV0=
 </   accessToken   >
       </securityContext></header>
       <body>
 <bodyContent xsi:type= "java.com.webex.service.binding.meeting.CreateMeeting">
            <metaData><confName>Sample Meeting</confName>
            <meetingType>105</meetingType></metaData>
            <schedule><startDate>12/13/2019 11:59:5</startDate></schedule>
       </bodyContent>
       </body>
      </serv:message>
 var req   http.request('https://api.webex.com/WBXService/XMLService',
   sessionTicket   );
 req.write(xml);
 req.end();
```

```
{'method' : 'POST'}     {'method' : 'PUT'}

    accessToken            sessionTicket
```

Explanation

Explanation/Reference:

QUESTION 6



```
1  const WebExClient = require('webex-api-client');
2  const securityContext = {
3   webExID: 'Test User',
4   password: 'pass123',
5   siteName: 'hello-world'
6  };
7
8  const requestBuilder = new WebExClient.Builder
9  (securityContext, 'https://hello-world.webex.com/WBXService/XMLService');

10 const createMeeting =
11  requestBuilder
12 .metaData({
13   confName: 'Sample Meeting',
14   meetingType: 1,
15 })
16 .participants({
17 attendees: [
18      {
19        name: 'Jane Doe',
20        email: 'jdoe@gmail.com'
21      }
22    ]
23  })
24  .schedule({
25   startDate: new Date{},
26   ______________________ ,
27   ______________________ ,
28   ______________________
29
30   })
31   .setService('CreateMeeting')
32   .build();
33
34  createMeeting
35  .exec()
36  .then((resp) => console.log('success'));
37
```

Refer to the exhibit. Which code for blank lines 26, 27, and 28 gives invitees 900 seconds before the scheduled time to join the meeting, sets the meeting to last for 30 minutes, and sets the meeting timezone to Pacific US?

A.
```
26 openMeeting: 900
27 stopTime: 30,
28  timeZone: 4
```

B.
```
26  openTime: 900
27  duration: 30,
28  timeZoneID: 4
```

C.
```
26  joinTime: 900
27  meetingLength: 30,
28  timezone: 5
```

D.
```
26  joinTime: 900
27  endTime: 30,
28  timezoneID: pt_us
```

Correct Answer: B
Explanation

Explanation/Reference:
Reference: https://github.com/cisco-ie/webex-api-client

QUESTION 7
Which two items are needed to give a Webex user the ability to archive all Webex Teams messages for an organization? (Choose two.)

A. Give the user "Read-only administrator privileges" in the Webex Control Hub.
B. Create an Integration app with all "spark_compliance" read scopes enabled.
C. Use the Webex Meetings XML API "SetUser" to update the user's "<roSiteAdmin>" value to "TRUE".
D. Configure the user as a "Compliance Officer" in the Webex Control Hub.

E. Create a Bot app with all “webex_compliance” read scopes enabled.

Correct Answer: AD
Explanation

Explanation/Reference:
Reference: https://www.cisco.com/c/dam/en/us/td/docs/voice_ip_comm/cloudCollaboration/spark/esp/Webex-Teams-Security-Frequently-Asked-Questions.pdf

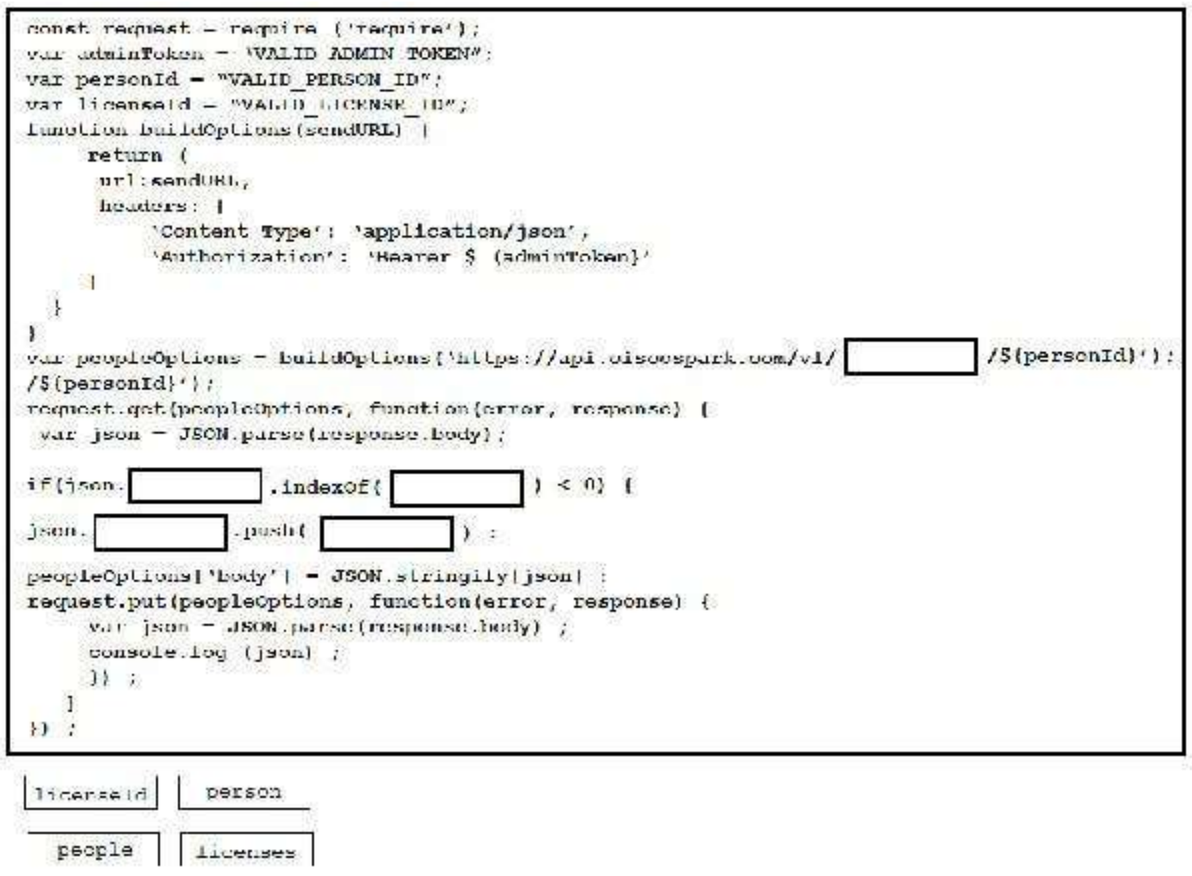
QUESTION 8
DRAG DROP

Drag and drop the code to complete the JavaScript snippet so that it:

- retrieves the details of an individual user
- checks what licenses they have already
- updates their account with a new license

Options can be used more than once.

Select and Place:

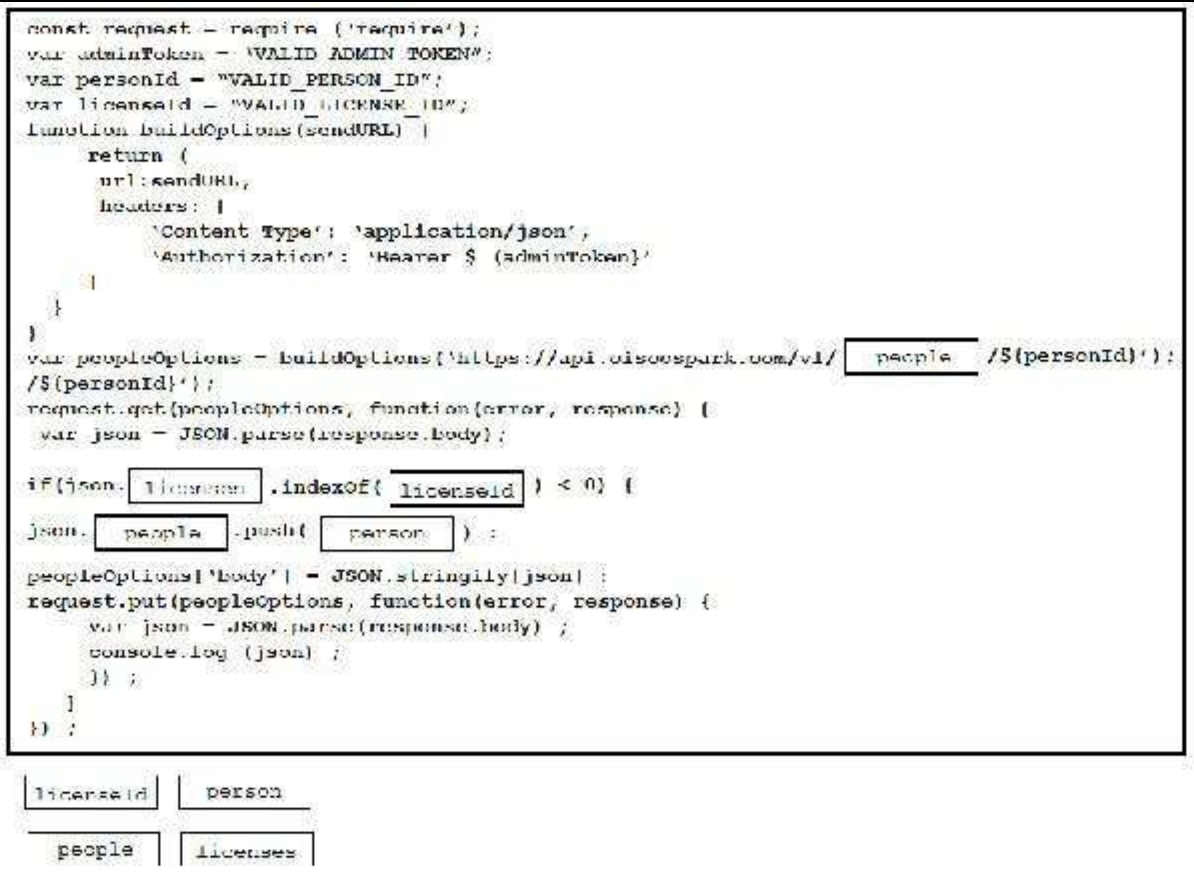


```
const request = require ('require');
var adminToken = 'VALID ADMIN TOKEN";
var personId = "VALID_PERSON_ID";
var licenseId = "VALID_LICENSE_ID";
function buildOptions(sendURL) {
      return (
       url:sendURL,
       headers: {
            'Content Type': 'application/json',
            'Authorization': 'Bearer $ (adminToken}'
       }
   }
}
var peopleOptions = buildOptions('https://api.ciscospark.com/v1/ ______ /${personId}');
/${personId}');
request.get(peopleOptions, function(error, response) {
 var json = JSON.parse(response.body);

if(json. ______ .indexOf( ______ ) < 0) {

json. ______ .push( ______ ) ;

peopleOptions['body'] = JSON.stringify(json) ;
request.put(peopleOptions, function(error, response) {
      var json = JSON.parse(response.body) ;
      console.log (json) ;
      }) ;
    ]
}) ;
```

licenseId   person

people   licenses

Correct Answer:



```
const request = require ('require');
var adminToken = "VALID ADMIN TOKEN";
var personId = "VALID_PERSON_ID";
var licenseId = "VALID_LICENSE_ID";
function buildOptions(sendURL) {
      return (
       url:sendURL,
       headers: {
           'Content Type': 'application/json',
           'Authorization': 'Bearer $ (adminToken}'
      }
   }
}
var peopleOptions = buildOptions('https://api.ciscospark.com/v1/ [ people ] /$(personId)');
/${personId}');
request.get(peopleOptions, function(error, response) {
 var json = JSON.parse(response.body);

if(json.[ licenses ].indexOf( [ licenseId ] ) < 0) {

json.[ people ].push( [ person ] ) ;

peopleOptions['body'] = JSON.stringify(json) ;
request.put(peopleOptions, function(error, response) {
      var json = JSON.parse(response.body) ;
      console.log (json) ;
      }) ;
    ]
}) ;
```

```
[ licenseId ]  [ person ]

[ people ]     [ licenses ]
```

Explanation

Explanation/Reference:

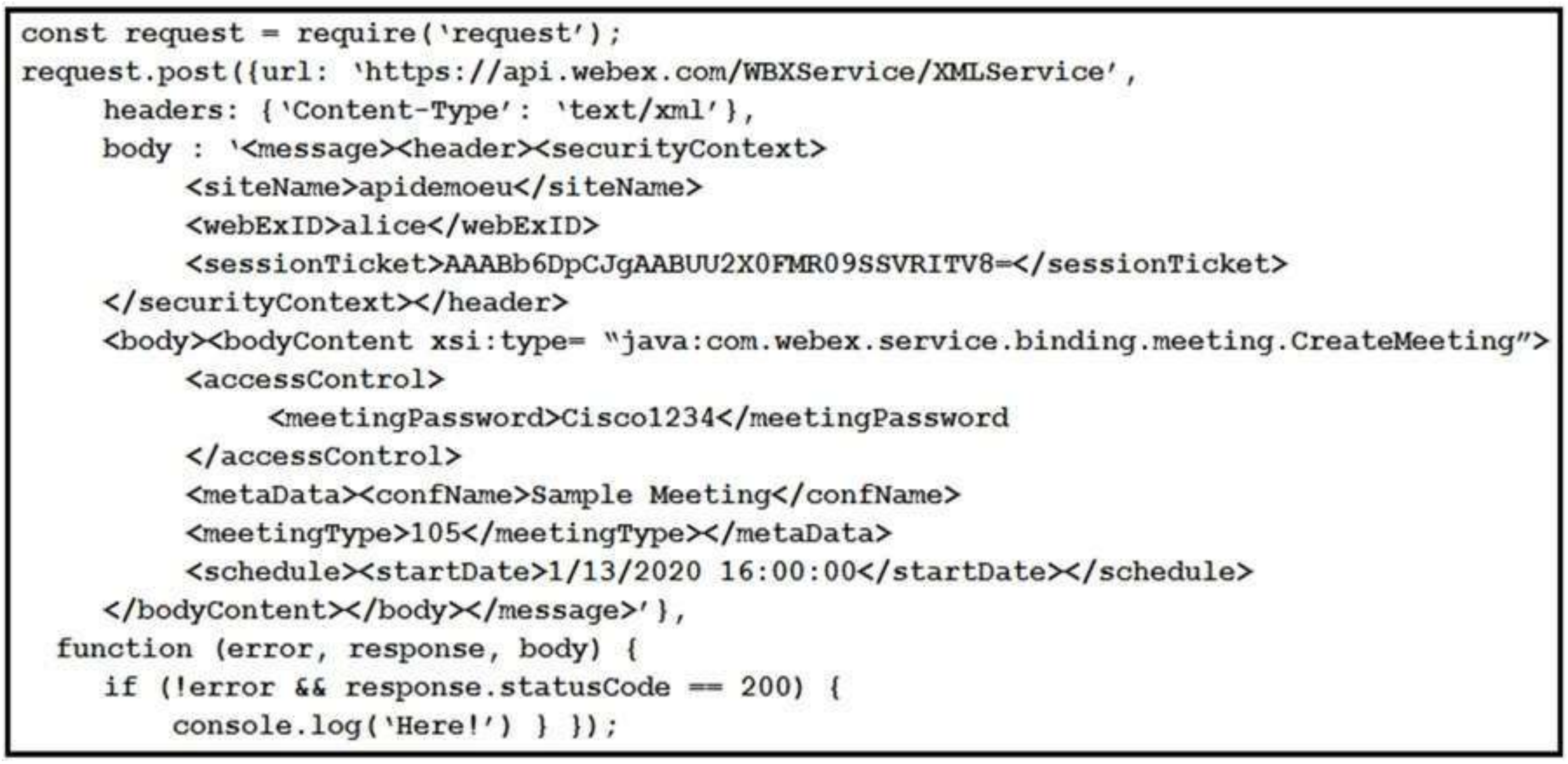
QUESTION 9
What happens if a meeting is in progress when a DelMeeting request is sent in the Webex Meetings XML API?

A. The meeting host is notified and prompted to allow the meeting to be deleted.
B. The DelMeeting request drops all call-in users and deletes the meeting.
C. The DelMeeting request waits until the meeting is completed and then deletes the meeting.
D. The DelMeeting request results in an error.

Correct Answer: A
Explanation

Explanation/Reference:
Reference: https://pdfslide.net/documents/webex-we.html (p.216)

QUESTION 10



```
const request = require('request');
request.post({url: 'https://api.webex.com/WBXService/XMLService',
     headers: {'Content-Type': 'text/xml'},
     body : '<message><header><securityContext>
          <siteName>apidemoeu</siteName>
          <webExID>alice</webExID>
          <sessionTicket>AAABb6DpCJgAABUU2X0FMR09SSVRITV8=</sessionTicket>
     </securityContext></header>
     <body><bodyContent xsi:type= "java:com.webex.service.binding.meeting.CreateMeeting">
          <accessControl>
               <meetingPassword>Cisco1234</meetingPassword
          </accessControl>
          <metaData><confName>Sample Meeting</confName>
          <meetingType>105</meetingType></metaData>
          <schedule><startDate>1/13/2020 16:00:00</startDate></schedule>
     </bodyContent></body></message>'},
   function (error, response, body) {
     if (!error && response.statusCode == 200) {
         console.log('Here!') } });
```

Refer to the exhibit. The Node.js script shown uses the Webex Meetings XML API to print "Here!" to the console. Which statement is a correct observation about the results of the script?

A. The <meetingPassword> was not complex enough.
B. The <sessionTicket> credential was expired.
C. The WebexMeetings XML API service processed the request.
D. The meeting was created successfully.

Correct Answer: C
Explanation

Explanation/Reference:

Explanation:
The password, although not that good, has a capital letter and numbers. Therefore, it is okay. SessionTicket credential is not expired because the error function doesn't check that. We are not sure if the meeting was created successfully however, there is no wrong in the code, therefore, webexmeetings XML API service has processed the request.

QUESTION 11

```
const webex = require('webex/env');
const BOT_ID = 'Y2ABCDEFG3VzL1BFT1BMRS9mNWTy1jGIyZi1m';
// ...
let notification = {
     "id": "Y21zY29zcGFyazovL3VzL1dFQkwMQ50DgxNjA5NDk3",
     "name": "notifier",
     "resource": "messages",
     "event": "created",
     "filter": "roomId=Y21zy29zcGFyazovL3VzL1JyjU4LtkxNDctZjE0YmIwYzRkMTU0",
     "orgId": "OTZhYmMyYWEtM2RjYy0xMWUUzNDgxOWNkYz1h",
     "createdBy": "Y21zY29zcGFyazovL3VzLOFQUExOTdjODUwM2YyNjZhYmY2NmMSOTIIYzFm",
     "ownedBy": "creator",
     "status": "active",
     "actorId": "Y21zY29zcGFyazovL3VzL1BFT18MRS9mNWIyijGIyZilmOWMONDdmMjkwNDY",
     "data": {
          "id": "Y21zY29zcGFyazovL3VzLZlYjAtMzRkNC0xMWVhLTgwZmMtYjFlNmRiMjk0YjM5",
          "roomId": "Y21zY29zcGFyazovL3VzLiJPT00vYmJjZWzYjU4LtkxNDctZjE0YmIwYzRkMTU0",
          "personId": "Y21zY29zcGFyazovL3VzL1BFTijctOGIyZilmOWMONDdnMjkwNDY",
          "personEmail": "alice@example.com",
          "created": "2019-10-18T14:26:26.000Z"
     }};
```

Refer to the exhibit. A webhook has been created so that an application is notified when users mention a bot in a Webex Teams space. The exhibit shows an example of a notification received by the application. Which code snippet correctly processes the JSON payload using the Webex Node.js SDK in order to print out messages that mention the bot?

A.
```
if ((notification.name == 'notifier') &&
   (notification.event == 'created')) {
   if (notification.resource == 'messages') {
      webex.messages.get(notification.data.id).then(
      msg => console.log(msg.text));
}}
```

B.
```
if    ((notification.resource == 'messages')&&
      (notification.event == 'created')) {
      webex.messages.get(notification.data).then(
      msg => console.log(msg.text));
   }
```

C.
```
if ((notification.resource == 'messages') &&
    (notification.event == 'created')) {
    if (notification.data.personId !== BOT_ID) {
      webex.messages.get(notification.data.id).then(
      msg => console.log(msg.text));
}}
```

D.
```
if ((notification.resource == 'notifier') &&
    (notification.event == 'created')) {
    if (notification.data.personId !== BOT_ID) {
      webex.messages.get(notification.data.id).then(
      msg => console.log(msg.text));
}}
```

Correct Answer: B
Explanation

Explanation/Referen

ce: