P-AVN-VA GraphQL API GUIDE 1.1.0

P-AVN-VA firmware 1.1.0

Version 1

DRAFT

Contents

Introduction

This document describes the GraphQL API for PlexusAV P-AVN-VA, based on firmware 1.1.0

For the most command tasks, we have added examples in this document. A full list of API commands is available in a separate document or in the schema of the GraphQL Playground or Postman.

NMOS vs PlexusAV API

The Networked Media Open Specifications (NMOS) is a control and management layer used in the IPMX open AV over IP standard. Since both P-AVN-4 and P-AVN-2 are IPMX based, they are NMOS based.

However, NMOS is not easy to implement and uses multiple API's. That's why PlexusAV provides a GraphQL API that is much easier to use.

The PlexusAV Visual Array, our single-point-of-control management platform, also has a GraphQL API.

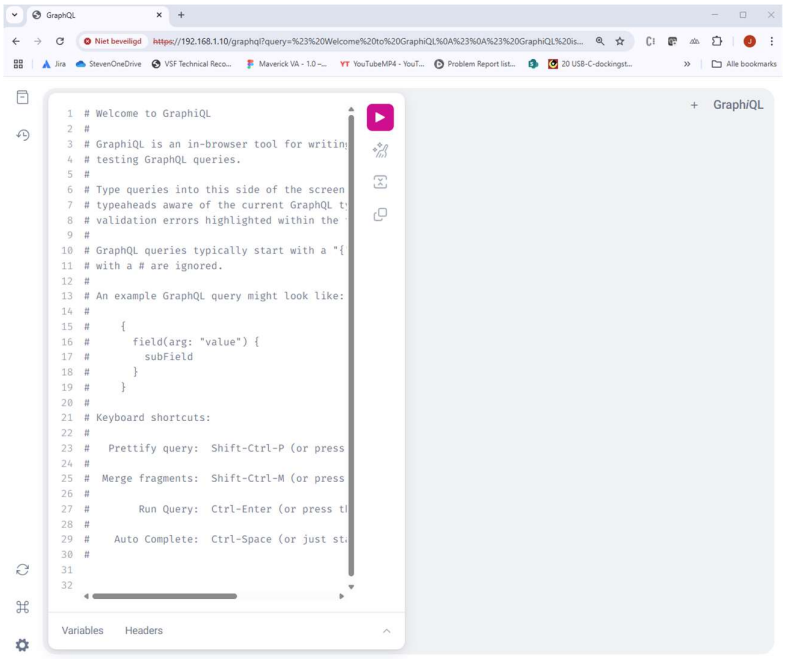
GraphQL API

The GraphQL API that is being used in PlexusAV P-AVN-2/4 and Visual Array has many advantages over the traditional REST API. GraphQL utilizes **a single endpoint** for sending queries or mutations, allowing clients to request **exactly the data they need**.

Simply put there is a single URL for all requests. You can customize the desired request and responses to match your need.

For the P-AVN-VA the URL is: **https://[VisualArray-Address]/graphql**
The IP port being used is the standard https port **443**.

If you enter this URL in the browser, you will see the GraphQL Playground. A place to test API command.

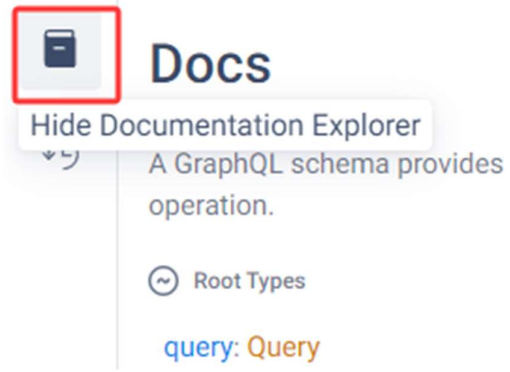


GraphQL uses **Queries** and **Mutations**:

- **Queries**: Used for retrieving data from the server
  (Like GET in REST API)
- **Mutations**: Used for retrieving data from the server.
  (Like POST, PUT and DELETE in REST API)

A GraphQL schema is a crucial part of GraphQL, defining the structure of the data that can be queried. It specifies the types of data available, the relationships between those types, and the operations that can be performed on the data. A GraphQL scheme includes queries and mutations.

You can see the full GraphQL schema by clicking the Show Documentation Explorer button in the GraphQL playground.



The PlexusAV GraphQL API uses the JSON format for a response, which follows the shape of the GraphQL request.

If you want more control and a single point to send commands from to test, we recommend using Postman. From now on, we will use Postman in all the examples. We will also explain how to set up Postman for P-AVN-VA GraphQL commands.

Getting started

API Version Management

This document is based on P-AVN-VA firmware version 1.1. The API commands will not change until we release a 2.0 or higher version.

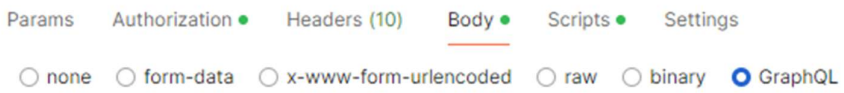
Postman set up

Postman version: Postman-Agent-win64-0.4.37-Setup

Here are the steps needed to send GraphQL requests from Postman to the PlexusAV P-AVN-VA.

1. **Open Postman and create a request**
   - Click on "New" → Select "Request"
   - Choose "POST" as the request type
   - Enter the GraphQL API endpoint (https://[P-AVN-VA IP Address]/graphql)

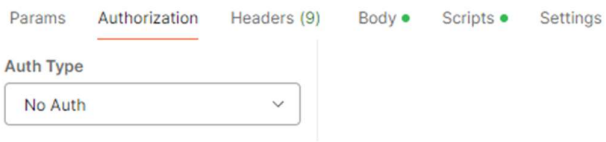
2. **Switch to GraphQL Mode**
   - In the Body tab, select GraphQL



   - You'll see a text box where you can enter your GraphQL query

3. **Write Your GraphQL Query**

4. **Add headers**
   - Go to the Headers tab and add the following:
   - Authorization → No authorization required (it's done in GraphQL)



5. **Send the request**
   - Click Send to execute the query.
   - View the response in the bottom panel.

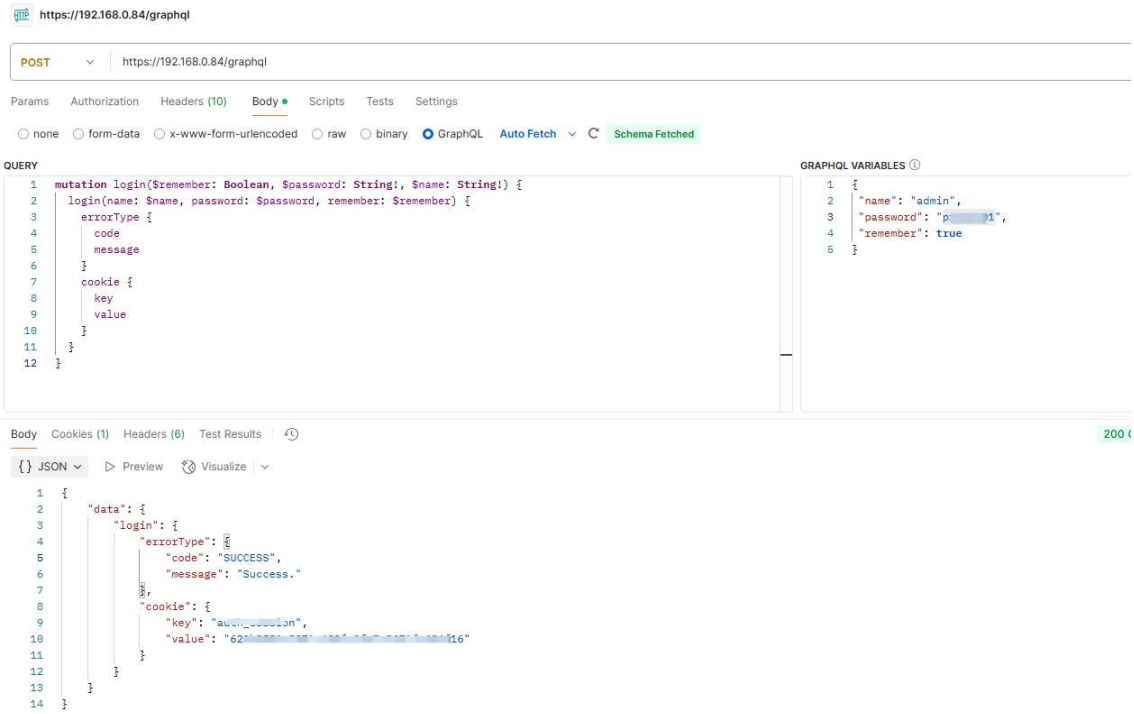
6. **Save the Request**
   - Click Save to store it for future use

7. **Variables and prescripts (optional)**
   - Using variable and prescripts (for getting the SID) in Postman simplifies the usage
   - However, the use of variables and prescripts is beyond the scope of this document

VA Log in

Log into the P-AVN-VA to get a Token for sending other queries and mutations. Instead of entering username and password in each API request, you request a **Token** to get access (saved in a cookie in web browser). The Authentication Key needs to be included in the header with each request. Our web browser does this automatically for us. The Token is valid for 30 minutes.

To get a Token, we do a Mutation with Field “login” and Arguments “name” and “password”. The response data, within the “login” field we list the “Key” and “Value”, these are the values we would like to know. The “Key” value is the token



GRAPHQL VARIABLES

```
{
 "name": "admin",
 "password": "proav101",
 "remember": true
}
```

Code:

```
mutation login($remember: Boolean, $password: String!, $name: String!) {
  login(name: $name, password: $password, remember: $remember) {
    errorType {
      code
      message
    }
    cookie {
      key
      value
    }
  }
}
```

Error handling

With the errorType response, we get feedback if the command was successful:

Error code values:
code: "SUCCESS" - success and message will be blank
code: "FAILURE" - failure and the message will show the detailed errors

First Query: Get System Information

Let's request the P-AVN-VA System Information as a first Query example, show in Dashboard.

GRAPHQL VARIABLES

QUERY for the user to be copied

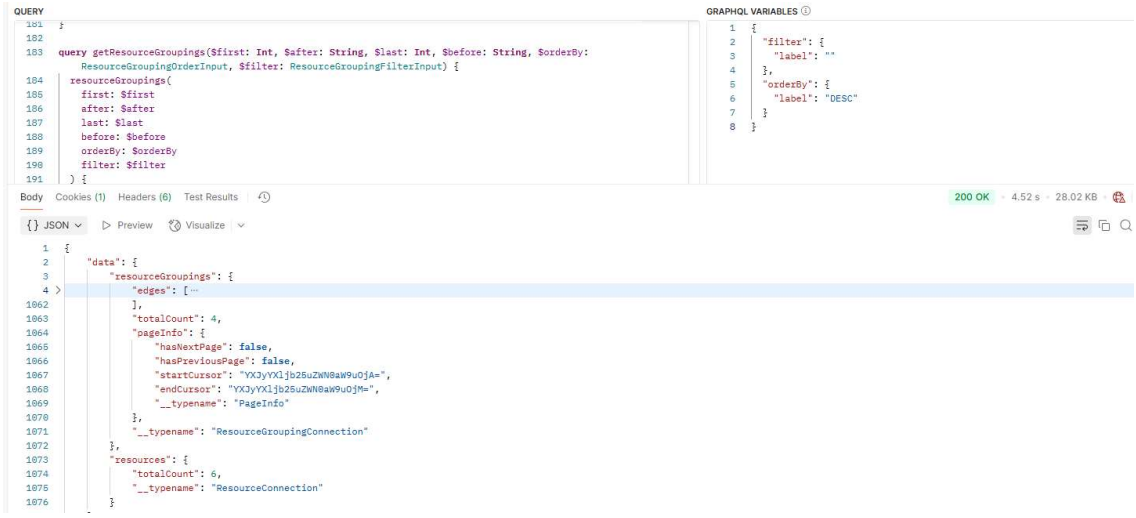
```
query dashboardInfo {
  ipConflicts {
    #Conflicted Device
    nodes {
      id
      ip
      __typename
    }
    __typename
  }
  igmpQueriers {
    #IGMP Querier
    nodes {
      ip
      id
      __typename
    }
    __typename
  }
  getUpdateVersion {
    currentVersion #va version
    __typename
  }
  system {
    #va Bandwidth
    label
    bandwidth {
      egress
      ingress
      __typename
    }
    devices {
      nodes {
        hostname
        uuid
        serialNumber
        networkInterfaces {
            #va Network
          nodes {
            id
            addresses {
              address
              prefixLength
              __typename
            }
            gateway
            status
            ethernet {
              speed
              __typename
            }
            name
            label
            __typename
          }
          __typename
        }
        stats {
          temps     #temperature
          cpuUsage  #cpu
          fanSpeeds #fan speed
          ramUsage {
            #ram
            total
            routes
            others
```

```
          __typename
        }
        __typename
      }
      __typename
    }
    __typename
  }
  __typename
 }
 offlineCount #offline devices count
 activeState {
   #active devices count
   activeCount
   encoderCount
   decoderCount
   ipmxInCount
   ipmxOutCount
   __typename
 }
 destinationIpConflicts {
   #Conflicted Device
   nodes {
     id
     ip
     names
     __typename
   }
   __typename
 }
 device {
   #Uptime
   startTime
   __typename
 }
}
```

Response

Discovering IPMX devices

Use NMOS to discover P-AVN-4/2 and SCG IPMX devices and streams, show in Device List



GRAPHQL VARIABLES

```
{
  "filter": {
```

```
    "label": ""
  },
  "orderBy": {
    "label": "DESC"
  }
}
```

QUERY for the user to be copied

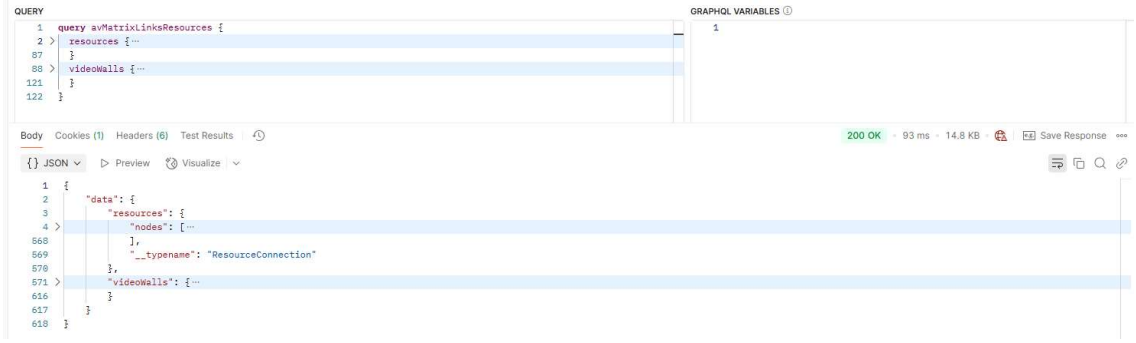
```
fragment resourceFields on Resource {
  outputDeviceType #PC, LAPTOP, TV, PROJECTOR, OTHER_DEVICE
  device {
    deviceType
    deviceNetState
    deviceNmosName
    deviceNmosInterface
    deviceNmosIpAddress
    deviceDanteName
    deviceUidCtl
    deviceOutputCtl
    deviceTestModeCtl
    deviceSelectedProfile
    deviceVersion
    deviceTrafficFlowMode
    deviceIgmpVersion
    __typename
  }
  networks {
    networkInterfaceName
    networkInterfaceMode
    networkInterfaceIpAddress
    networkInterfaceSubnetMask
    networkInterfaceGateway
    networkInterfaceLinkSpeed
    networkInterfaceRxBitrate
    networkInterfaceTxBitrate
    networkInterfaceInst
    __typename
  }
  hdmi {
    hdmiOutputState
    hdmiEdidMode
    hdmiHdr
    hdmiHdrPassThrough
    hdmiInputHdcp
    hdmiInputAudioType
    hdmiInputAudioSampleRate
    hdmiInputAudioChannelNumber
    __typename
  }
  rs232 {
    rs232State
    rs232Speed
    rs232DataBits
    rs232StopBits
    rs232FlowCtl
    rs232Parity
    __typename
  }
  videoStream {
    videoInputSelection
    streamInterface
    streamDstIpAddress
    streamDstPort
    streamState
    streamLinkStatus
    streamInst
    videoStreamOutputState
    videoStreamCodec
    videoStreamChromaFormat
    videoStreamColorDepth
    videoStreamBpp
    videoStreamHdcp
    videoStreamResolution
    videoStreamOutputFormatMode
    videoStreamSwitchingMode
    videoStreamSwitchingDisplay
```

```
  videoStreamColorMode
  videoStreamBitRateMode
  videoStreamOutputManualFrameRate
  videoStreamOutputManualResolution
  videoStreamColorDepthStatus
  videoStreamChromaFormatStatus
  videoInterface
  videoIpInput
  videoStreamInputChromaFormatStatus
  videoStreamInputColorDepthStatus
  __typename
}
audioStream {
  audioInputSelection
  streamInterface
  streamDstIpAddress
  streamDstPort
  streamState
  streamLinkStatus
  streamInst
  audioStreamType
  audioStreamBitDepth
  audioStreamSampleRate
  audioStreamChannelNumber
  __typename
}
dataStream {
  streamDstIpAddress
  streamDstPort
  __typename
}
dante {
  danteInputSampleRate
  danteInputChannels
  __typename
}
secondaryVideoStream {
  streamDstIpAddress
  videoStreamOutputFormatMode
  videoStreamColorMode
  videoStreamChromaFormatStatus
  videoStreamColorDepthStatus
  videoStreamChromaFormat
  videoStreamColorDepth
  videoStreamResolution
  videoStreamOutputManualResolution
  videoStreamOutputManualFrameRate
  videoStreamCodec
  __typename
}
secondaryAudioStream {
  streamDstIpAddress
  __typename
}
group {
  id
  label
  tags {
    nodes {
      id
      label
      color
      __typename
    }
    totalCount
    __typename
  }
  resources {
    nodes {
      id
      __typename
    }
    totalCount
    __typename
  }
  __typename
}
tags {
  nodes {
    id
    label
```

```
      color
      __typename
    }
    totalCount
    __typename
  }
  __typename
}

fragment resourceGroupFields on ResourceGroup {
  resources {
    nodes {
      id
      label
      ...resourceFields
      __typename
    }
    totalCount
    __typename
  }
   tags{
    nodes {
      id
      label
      color
      __typename
    }
    totalCount
    __typename
  }
  __typename
}


query getResourceGroupings($first: Int, $after: String, $last: Int, $before: String, $orderBy: ResourceGroupingOrderInput, $filter:
ResourceGroupingFilterInput) {
  resourceGroupings(
    first: $first
    after: $after
    last: $last
    before: $before
    orderBy: $orderBy
    filter: $filter
  ) {
    edges {
      node {
        id
        label
        ...resourceFields
        ...resourceGroupFields
        __typename
      }
      __typename
    }
    totalCount
    pageInfo {
      hasNextPage
      hasPreviousPage
      startCursor
      endCursor
      __typename
    }
    __typename
  }
  resources {
    totalCount
    __typename
  }
}
```

Discovering IPMX Streams

Let's request the P-AVN-VA stream information, show in AV Matrix.



GRAPHQL VARIABLES

QUERY for the user to be copied

```
query avMatrixLinksResources {
  resources {
    # device list
    nodes {
      id
      rawId
      outputDeviceType
      mainStreamLinkMode
      subStreamLinkMode
      device {
        deviceType
        deviceNetState
        deviceNmosName
        deviceNmosInterface
        deviceNmosIpAddress
        deviceOutputCtl
        deviceUsbIpConnectionStatus
        deviceUsbdeviceProduct
        devicePreviewStatus
        deviceUsbSubscriptionId
        deviceSerialSubscriptionId
        deviceSerialIpConnectionStatus
        __typename
      }
      videoStream {
        videoStreamResolution
        streamLinkStatus
        streamId
        streamSubscriptionId
        streamSubscriptionActive
        __typename
      }
      secondaryVideoStream {
        streamLinkStatus
        streamId
        streamSubscriptionId
        streamSubscriptionActive
        __typename
      }
      audioStream {
        streamLinkStatus
        streamId
        streamSubscriptionId
        streamSubscriptionActive
        __typename
      }
      secondaryAudioStream {
        streamLinkStatus
        streamId
        streamSubscriptionId
        streamSubscriptionActive
        __typename
      }
      networks {
```

```
        networkInterfaceName
        networkInterfaceLinkSpeed
        networkInterfaceRxBitrate
        networkInterfaceTxBitrate
        __typename
      }
      hdmi {
        hdmiOutputState
        __typename
      }
      rs232 {
        rs232CustomerCmdName
        rs232CustomerRawCmd
        rs232CustomerHexMode
        rs232CustomerInst
        __typename
      }
      group {
        id
        __typename
      }
      tags {
        nodes {
          id
          label
          color
          __typename
        }
        __typename
      }
      __typename
    }
    __typename
  }
  videoWalls {
    # video wall list
    nodes {
      id
      label
      videoWallMode
      matrixSize
          devices{
        nodes {
          id
          panelId
          resourceId
          sourceId
          data
          __typename
        }
        __typename
      }
      resource {
        id
        __typename
      }
      tags {
        nodes {
          id
          label
          color
          __typename
        }
        __typename
      }
      __typename
    }
    __typename
  }
}
```

API Command Examples

AV Matrix routing

The AV Matrix routing is for building the stream connection between the senders and receivers in the matrix view. Each plexus sender and receivers provide multiple streams including video, audio, usb, rs232, IR, CEC. When building the stream connection through the API we need to provide sender ID (which sender need to connect with), receiver ID (which receiver need to be connected), stream type (which stream).

Before performing the routing through API. Need to get all the sender IDs and receiver IDs.

1- All routing

Show in AV Matrix.

Variables:

SenderId: encoder id

ReceiverId: decoder id

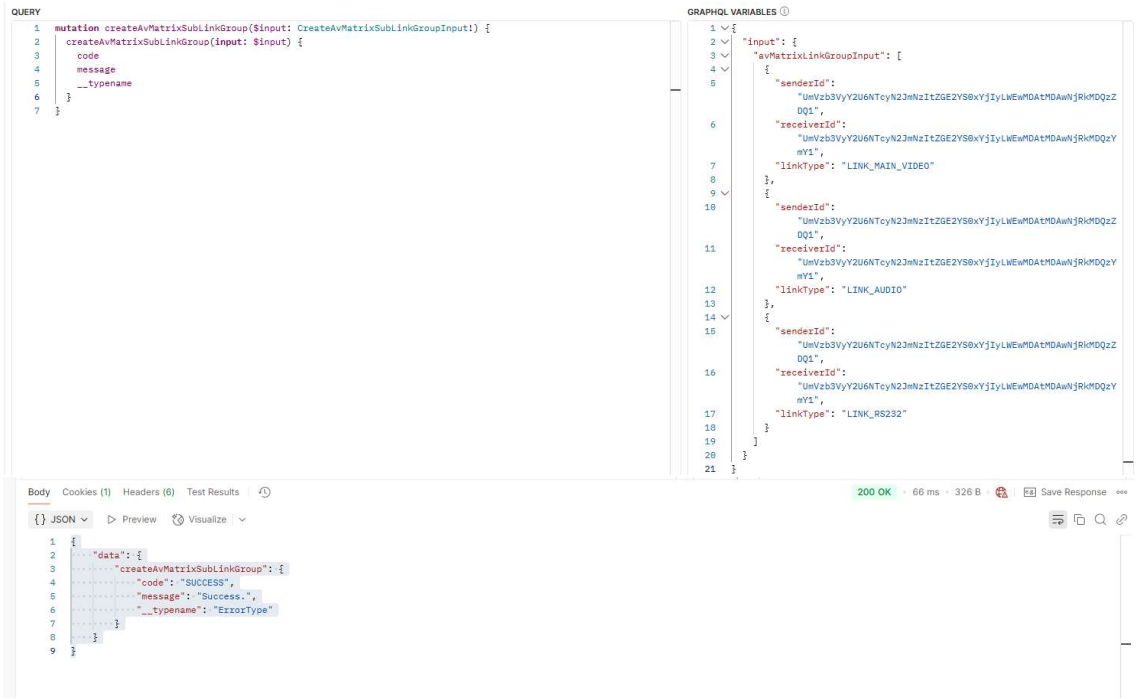
LinkType:

```
'MAIN_LINK_TYPE': JXS Video

'LINK_AUDIO': JXS Audio

'LINK_SUB_VIDEO': H.26x Video

'LINK_SUB_AUDIO': H.26x Audio

'LINK_USB': USB

'LINK_RS232': RS232

'LINK_IR': IR

'LINK_CEC': CEC
```

1.1 When collapse AV Matrix

1.1.1 Create video, audio, rs232 link for all decoders to one encoder.

Click 'All'. In variables:



GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```


QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

1.1.2 Delete video, audio, rs232 link for all decoders to one encoder.

After creating links, click 'All' to delete.

GRAPHQL VARIABLES


```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
```

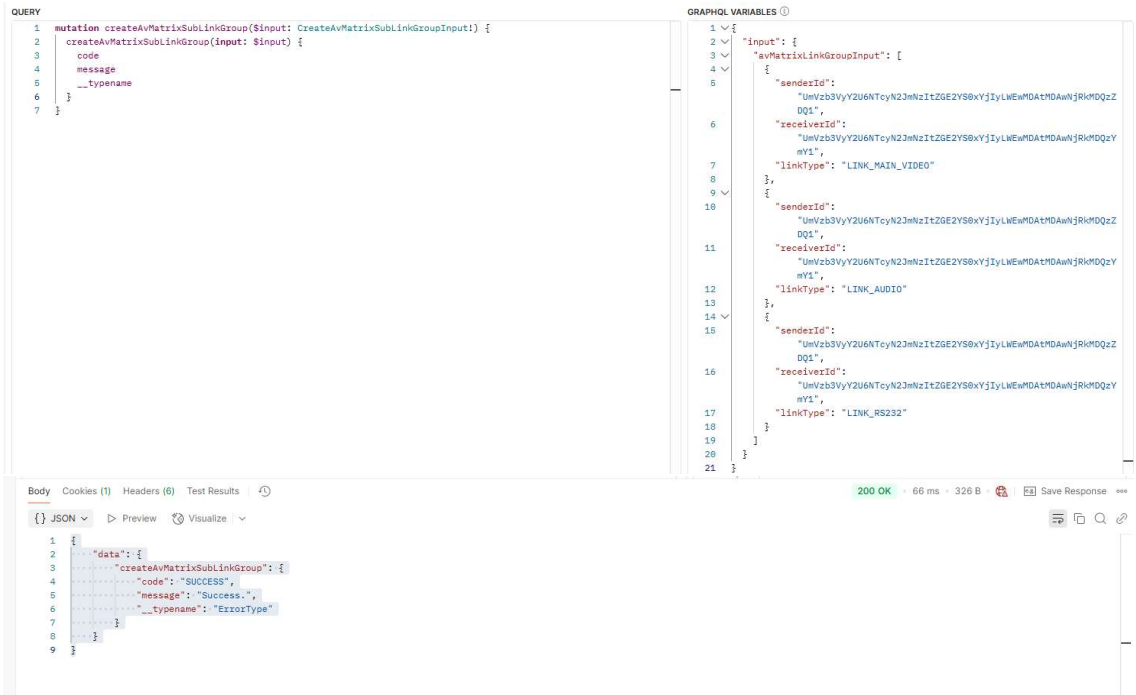
```
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

1.1.3 Create video, audio, rs232 link for one decoder to one encoder.

Click '+'. In variables:



GRAPHQL VARIABLES


```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```


QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

1.1.4 Delete video, audio, rs232 link for one decoder to one encoder.

After creating links, click 🔴 to delete.

GRAPHQL VARIABLES


```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
```

```
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

1.2 When expand AV Matrix

1.2.1 Create links for all decoders to one encoder

Click ‘All’ of Jxs video

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

1.2.2 Delete links for all decoders to one encoder

Click ‘All’ of Jxs video

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

1.2.3 Create link for one decoder to one encoder

Click ○ of the decoder jxs decoder

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

1.2.4 Delete link for one decoder to one encoder

Click ● of the decoder jxs decoder

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

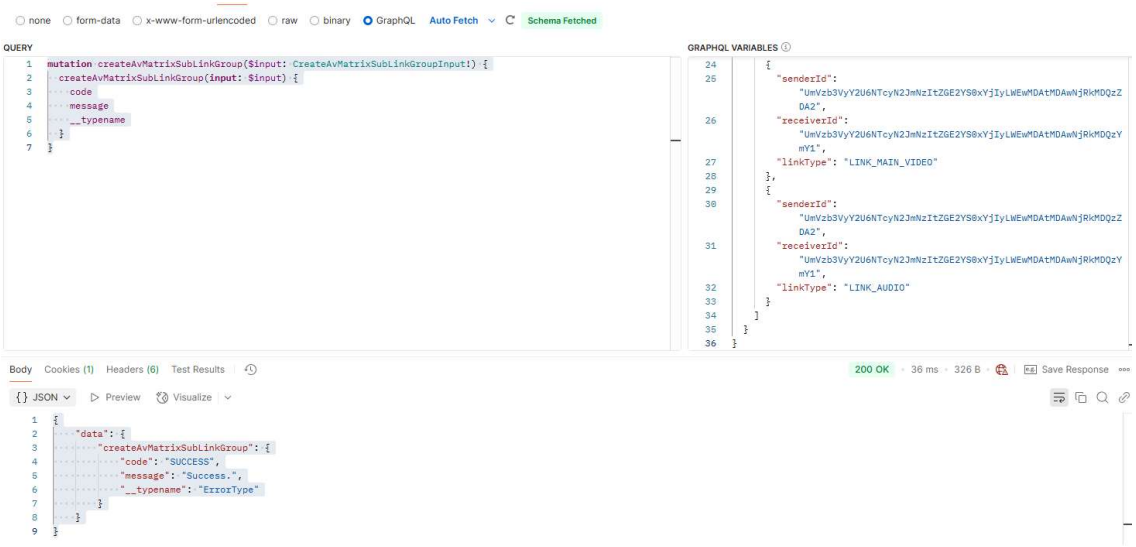
```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

2- AV routing

Create links of type 'MAIN_LINK_TYPE', 'LINK_AUDIO', 'LINK_SUB_VIDEO', 'LINK_SUB_AUDIO'

2.1 Create links for all decoders to one encoder (collapse av matrix)

Click ‘All’



GRAPHQL VARIABLES

```
{
  "input": {
```

```
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_AUDIO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

2.2 Delete links for all decoders to one encoder (collapse av matrix)

Click ‘All’

GRAPHQL VARIABLES

```
{
  "input": {
```

```
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_AUDIO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

2.3 Create links for one decoder to one encoder (collapse av matrix)

Click '+'

GRAPHQL VARIABLES

```
{
  "input": {
```

```
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_AUDIO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

2.4 Delete links for one decoder to one encoder

Click '-' (expand av matrix) or ○ (collapse av matrix)

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_AUDIO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
```

```
}
```

2.5 Create links of type for all decoders to one encoder (expand av matrix)

Click ‘All’ of type

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

2.6 Delete links of type for all decoders to one encoder (expand av matrix)

Click ‘All’ of type

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
```

```
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

2.7 Create links of type for one decoder to one encoder (expand av matrix)

Click ○ of type `"LINK_SUB_VIDEO" of decoder`

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_SUB_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

2.8 Delete links of type for one decoder to one encoder (expand av matrix)

Click ● of type `"LINK_SUB_VIDEO" of decoder`

GRAPHQL VARIABLES

```
{
```

```
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_SUB_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

3- KVM routing

Create links of type ' LINK_RS232'

3.1 Create links for all decoders to one encoder (collapse av matrix)

Click ‘All’

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_USB"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

3.2 Delete links for all decoders to one encoder (expand av matrix)

Click 'All'

GRAPHQL VARIABLES

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

3.3 Create links for one decoder to one encoder (collapse av matrix)

Click '+'

GRAPHQL VARIABLES

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

3.4 Delete links for one decoder to one encoder

Click '-' (expand av matrix) or ⭕ (collapse av matrix)

GRAPHQL VARIABLES

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

3.5 Create links of type for all decoders to one encoder (expand av matrix)

Click 'All' of type

GRAPHQL VARIABLES

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

3.6 Delete links of type for all decoders to one encoder (expand av matrix)

Click 'All' of type

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

3.7 Create links of type for one decoder to one encoder (expand av matrix)

Click ○ of type `"LINK_MAIN_VIDEO"`

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
```

```
        }
      ]
    }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

3.8 Delete links of type for one decoder to one encoder (expand av matrix)

Click ● of type `"LINK_MAIN_VIDEO"`

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

4- Data routing

Create links of type "LINK_MAIN_VIDEO", "LINK_AUDIO", "LINK_USB"

4.1 Create links for all decoders to one encoder (collapse av matrix)

Click ‘All’

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
```

```
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_RS232"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_RS232"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

4.2 Delete links for all decoders to one encoder (collapse av matrix)

Click ‘All’

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_RS232"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_RS232"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_RS232"
      }
    ]
```

```
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

4.3 Create links for one decoder to one encoder (collapse av matrix)

Click '+'

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

4.4 Delete links for one decoder to one encoder (collapse av matrix)

Click '-' (expand av matrix) or ⭕ (collapse av matrix)

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
```

```
    message
    __typename
  }
}
```

4.5 Create links of type for all decoders to one encoder (expand av matrix)

Click ‘All’ of type `"LINK_RS232"`

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_RS232"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_RS232"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

4.6 Delete links of type for all decoders to one encoder (expand av matrix)

Click ‘All’ of type `"LINK_RS232"`

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_RS232"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_RS232"
```

```
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

4.7 Create links of type for one decoder to one encoder (expand av matrix)

Click ○ of type `"LINK_RS232"`

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_RS232"
      }
    ]
  }
}
```

QUERY for the user to be copied

```
mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
  createAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

4.8 Delete links of type for one decoder to one encoder (expand av matrix)

Click ● of type `"LINK_RS232"`

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_RS232"
      }
    ]
```

```
    }
}
```

QUERY for the user to be copied

```
mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
  deleteAvMatrixSubLinkGroup(input: $input) {
    code
    message
    __typename
  }
}
```

Video Wall

1.Video wall list

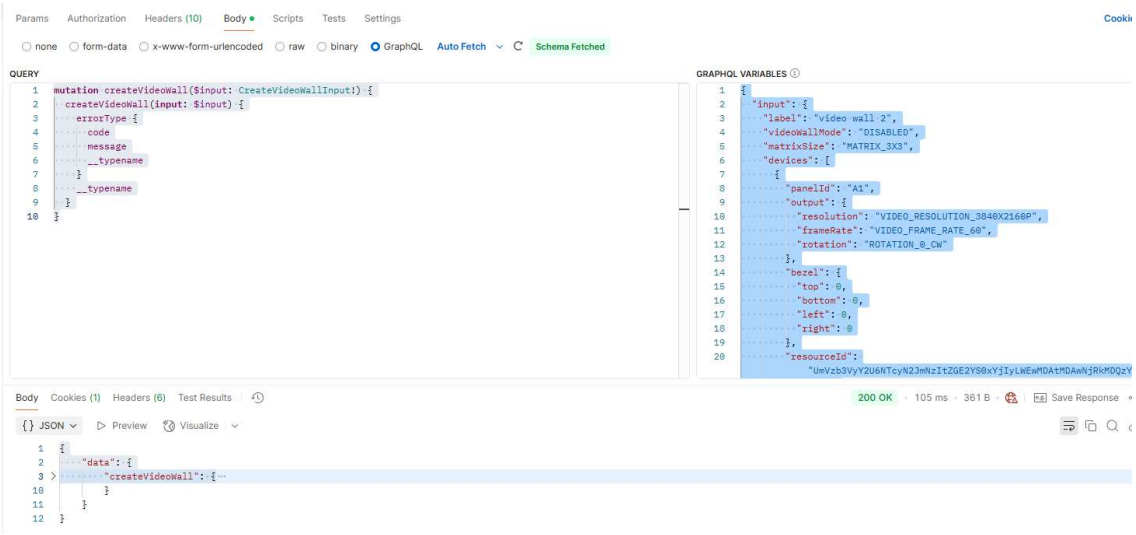
GRAPHQL VARIABLES

QUERY for the user to be copied

```
query videoWalls {
  videoWalls {
    edges {
      node {
        id
        label
        videoWallMode
        matrixSize
        tags {
          nodes {
            id
            label
            color
            __typename
          }
          totalCount
          __typename
        }
        devices {
          nodes {
            id
            panelId
            resourceId
            resource {
              id
              device {
                deviceType
                deviceNetState
                deviceNmosName
                deviceNmosInterface
                deviceNmosIpAddress
                deviceOutputCtl
                __typename
              }
              videoStream {
                videoStreamResolution
                __typename
              }
              networks {
                networkInterfaceName
                networkInterfaceLinkSpeed
                networkInterfaceRxBitrate
                networkInterfaceTxBitrate
                __typename
              }
              hdmi {
                hdmiOutputState
                __typename
              }
              __typename
            }
            sourceId
            output {
              resolution
              frameRate
              rotation
              __typename
            }
            bezel {
              top
              bottom
              left
              right
              __typename
            }
            data
            __typename
          }
          __typename
        }
```

```
        resource {
          id
          device {
            deviceType
            deviceNetState
            deviceNmosName
            deviceNmosInterface
            deviceNmosIpAddress
            deviceOutputCtl
            __typename
          }
          videoStream {
            videoStreamResolution
            __typename
          }
          networks {
            networkInterfaceName
            networkInterfaceLinkSpeed
            networkInterfaceRxBitrate
            networkInterfaceTxBitrate
            __typename
          }
          hdmi {
            hdmiOutputState
            __typename
          }
          __typename
        }
        __typename
      }
      __typename
    }
    totalCount
    pageInfo {
      hasNextPage
      hasPreviousPage
      startCursor
      endCursor
      __typename
    }
    __typename
  }
}
```

2.Create Video wall



GRAPHQL VARIABLES

```
{
  "input": {
```

```
    "label": "video wall 2",
    "videoWallMode": "DISABLED",
    "matrixSize": "MATRIX_3X3",
    "devices": [
      {
        "panelId": "A1",
        "output": {
          "resolution": "VIDEO_RESOLUTION_3840X2160P",
          "frameRate": "VIDEO_FRAME_RATE_60",
          "rotation": "ROTATION_0_CW"
        },
        "bezel": {
          "top": 0,
          "bottom": 0,
          "left": 0,
          "right": 0
        },
        "resourceId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
        "data": true
      }
    ]
  }
}
```


QUERY for the user to be copied

```
mutation createVideoWall($input: CreateVideoWallInput!) {
  createVideoWall(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

3.Update Video wall

GRAPHQL VARIABLES


```
{
```

```
  "input": {
    "label": "video wall 2",
    "videoWallMode": "DISABLED",
    "matrixSize": "MATRIX_2X2",
    "devices": [
      {
        "panelId": "A1",
        "output": {
          "resolution": "VIDEO_RESOLUTION_3840X2160P",
          "frameRate": "VIDEO_FRAME_RATE_60",
          "rotation": "ROTATION_0_CW"
        },
        "bezel": {
          "top": 0,
          "bottom": 0,
          "left": 0,
          "right": 0
        },
        "resourceId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
        "id":
"RGV2aWNlVmlkZW9XYWxsOjFmZmNhMjNjLTdlMWUtNDhkOC1iYmVjLTAzYmY5NzRkMTUyMA==",
        "data": true
      }
    ],
    "id": "VmlkZW9XYWxsOmFiNDdkYThjLWU2YTgtNDU2MC1iZjAwLTBhZDg1ZTcyMTY2Zg=="
  }
}
```

QUERY for the user to be copied

```
mutation updateVideoWall($input: UpdateVideoWallInput!) {
  updateVideoWall(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

4.Delete Video wall

GRAPHQL VARIABLES

```
{
  "id": "VmlkZW9XYWxsOmI4Y2M5ZTc1LTRlZWYtNGQ0OC04MTAxLTJmZjExYjRiMjFkZA=="
}
```

QUERY for the user to be copied

```
mutation deleteVideoWall($id: ID!) {
  deleteVideoWall(id: $id) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

Basic Setting

GRAPHQL VARIABLES

```
{
    "input": {
        "devices": [
            {
                "id":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQ0NzY0",
                "groupId":
"UmVzb3VyY2VHcm91cDowNzJhY2I4Zi0yNDcwLTRmMGYtODY4My1lNTY1OWY2MDFiMzI=",
                "tagIds": [
                    "VGFnOmViYTFkYTBhLTczMGUtNDQ1OC04ZmM3LWJkMmI1YmZmNDI2Zg=="
                ],
                "basicSetting": {
                    "deviceNmosName": "name-1",
                    "deviceDanteName": "name-1",
                    "deviceTrafficFlowMode": true,
                    "deviceControlNic": "CONNECTOR_1",
                    "deviceDanteNic": "CONNECTOR_1",
                    "deviceUsbNic": "CONNECTOR_1"
                },
                "videoStream": {
```

```
                    "streamInterface": "CONNECTOR_1",
                    "streamDstIpAddress": "239.192.1.1",
                    "streamDstPort": 5004
                },
                "audioStream": {
                    "streamInterface": "CONNECTOR_1",
                    "streamDstIpAddress": "239.192.1.2",
                    "streamDstPort": 5004
                },
                "dataStream": {
                    "streamDstIpAddress": "239.192.1.3",
                    "streamDstPort": 5004
                },
                "secondaryVideoStream": {
                    "streamDstIpAddress": "239.192.1.4",
                    "streamDstPort": 5004
                },
                "secondaryAudioStream": {
                    "streamDstIpAddress": "239.192.1.5",
                    "streamDstPort": 5004
                },
                "secondaryDataStream": {
                    "streamDstIpAddress": "239.192.1.6",
                    "streamDstPort": 5004
                },
                "networks": [
                    {
                        "networkInterfaceMode": "STATIC",
                        "networkInterfaceIpAddress": "192.168.0.73",
                        "networkInterfaceSubnetMask": "255.255.254.0",
                        "networkInterfaceGateway": "192.168.1.254",
                        "networkInterfaceInst": ".6.0.6.77.4.71.99"
                    },
                    {
                        "networkInterfaceMode": "STATIC",
                        "networkInterfaceIpAddress": "192.168.0.73",
                        "networkInterfaceSubnetMask": "255.255.254.0",
                        "networkInterfaceGateway": "192.168.1.254",
                        "networkInterfaceInst": ".6.0.6.77.4.71.100"
                    },
                    {
                        "networkInterfaceMode": "STATIC",
                        "networkInterfaceIpAddress": "192.168.0.73",
                        "networkInterfaceSubnetMask": "255.255.254.0",
                        "networkInterfaceGateway": "192.168.1.254",
                        "networkInterfaceInst": ".6.0.6.77.4.71.101"
                    }
                ]
            },
            {
                "id":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi",
                "groupId":
"UmVzb3VyY2VHcm91cDowNzJhY2I4Zi0yNDcwLTRmMGYtODY4My1lNTY1OWY2MDFiMzI=",
                "tagIds": [
                    "VGFnOmViYTFkYTBhLTczMGUtNDQ1OC04ZmM3LWJkMmI1YmZmNDI2Zg=="
                ],
                "basicSetting": {
                    "deviceNmosName": "name-2",
                    "deviceDanteName": "name-2",
```

```
                    "deviceTrafficFlowMode": true,
                    "deviceControlNic": "CONNECTOR_1",
                    "deviceDanteNic": "CONNECTOR_1",
                    "deviceUsbNic": "CONNECTOR_1"
                },
                "videoStream": {
                    "streamInterface": "CONNECTOR_1"
                },
                "audioStream": {
                    "streamInterface": "CONNECTOR_1"
                },
                "dataStream": {},
                "secondaryVideoStream": {},
                "secondaryAudioStream": {},
                "secondaryDataStream": {},
                "networks": [
                    {
                        "networkInterfaceMode": "STATIC",
                        "networkInterfaceIpAddress": "192.168.0.74",
                        "networkInterfaceSubnetMask": "255.255.254.0",
                        "networkInterfaceGateway": "192.168.1.254",
                        "networkInterfaceInst": ".6.0.6.77.6.24.171"
                    },
                    {
                        "networkInterfaceMode": "STATIC",
                        "networkInterfaceIpAddress": "192.168.0.74",
                        "networkInterfaceSubnetMask": "255.255.254.0",
                        "networkInterfaceGateway": "192.168.1.254",
                        "networkInterfaceInst": ".6.0.6.77.6.24.172"
                    },
                    {
                        "networkInterfaceMode": "STATIC",
                        "networkInterfaceIpAddress": "192.168.0.74",
                        "networkInterfaceSubnetMask": "255.255.254.0",
                        "networkInterfaceGateway": "192.168.1.254",
                        "networkInterfaceInst": ".6.0.6.77.6.24.173"
                    }
                ]
            },
            {
                "id":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3",
                "groupId":
"UmVzb3VyY2VHcm91cDowNzJhY2I4Zi0yNDcwLTRmMGYtODY4My1lNTY1OWY2MDFiMzI=",
                "tagIds": [
                    "VGFnOmViYTFkYTBhLTczMGUtNDQ1OC04ZmM3LWJkMmI1YmZmNDI2Zg=="
                ],
                "basicSetting": {
                    "deviceNmosName": "name-3",
                    "deviceDanteName": "name-3",
                    "deviceTrafficFlowMode": true,
                    "deviceControlNic": "CONNECTOR_1",
                    "deviceDanteNic": "CONNECTOR_1",
                    "deviceUsbNic": "CONNECTOR_1"
                },
                "videoStream": {
                    "streamInterface": "CONNECTOR_1"
                },
                "audioStream": {
                    "streamInterface": "CONNECTOR_1"
```

```
                },
                "dataStream": {},
                "secondaryVideoStream": {},
                "secondaryAudioStream": {},
                "secondaryDataStream": {},
                "networks": [
                    {
                        "networkInterfaceMode": "STATIC",
                        "networkInterfaceIpAddress": "192.168.0.75",
                        "networkInterfaceSubnetMask": "255.255.254.0",
                        "networkInterfaceGateway": "192.168.1.254",
                        "networkInterfaceInst": ".6.0.6.77.4.60.55"
                    },
                    {
                        "networkInterfaceMode": "STATIC",
                        "networkInterfaceIpAddress": "192.168.0.75",
                        "networkInterfaceSubnetMask": "255.255.254.0",
                        "networkInterfaceGateway": "192.168.1.254",
                        "networkInterfaceInst": ".6.0.6.77.4.60.56"
                    },
                    {
                        "networkInterfaceMode": "STATIC",
                        "networkInterfaceIpAddress": "192.168.0.75",
                        "networkInterfaceSubnetMask": "255.255.254.0",
                        "networkInterfaceGateway": "192.168.1.254",
                        "networkInterfaceInst": ".6.0.6.77.4.60.57"
                    }
                ]
            }
        ]
    }
}
```


Variables description in devices

```
{
     audioStream : {
        streamDstIpAddress : jsx audio stream ip
        streamDstPort : jsx audio stream port
        streamInterface : flow audio -CONNECTOR_1,CONNECTOR_2,CONNECTOR_3
     },
     basicSetting : {
        deviceControlNic : flow control -CONNECTOR_1,CONNECTOR_2,CONNECTOR_3
        deviceDanteName :
        deviceDanteNic : flow dante -CONNECTOR_1,CONNECTOR_2,CONNECTOR_3
        deviceNmosName :
        deviceTrafficFlowMode : true
        deviceUsbNic : flow usb -CONNECTOR_1,CONNECTOR_2,CONNECTOR_3
     },
     dataStream : {
        streamDstIpAddress :
        streamDstPort :
     },
     groupId : group id
     id : device id
     networks : [
        {  // eth 1
           networkInterfaceGateway : gateway
           networkInterfaceInst : .6.0.6.77.4.71.99,
           networkInterfaceIpAddress : ip
           networkInterfaceMode : eth mode, 'DHCP', 'STATIC'
           networkInterfaceSubnetMask : subnet mask
        },
```

```
        {  // eth 2
           networkInterfaceGateway :
           networkInterfaceInst : .6.0.6.77.4.71.100,
           networkInterfaceIpAddress :
           networkInterfaceMode :
           networkInterfaceSubnetMask :
        },
        {  // sfp
           networkInterfaceGateway :
           networkInterfaceInst : .6.0.6.77.4.71.101,
           networkInterfaceIpAddress :
           networkInterfaceMode :
           networkInterfaceSubnetMask :
        }
     ],
     secondaryAudioStream : {
        streamDstIpAddress :
        streamDstPort :
     },
     secondaryDataStream : {
        streamDstIpAddress :
        streamDstPort :
     },
     secondaryVideoStream : {
        streamDstIpAddress :
        streamDstPort :
     },
     tagIds : tag id list
     videoStream : {
        streamDstIpAddress :
        streamDstPort :
        streamInterface : flow video -CONNECTOR_1,CONNECTOR_2,CONNECTOR_3
     }
  }
```

QUERY for the user to be copied

```
mutation updateDeivces($input: UpdateDevicesInput!){
  updateDeivces(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

Advanced Setting

Variable description in input in updateDeviceList

```
groupId: Group id.
tagIds: Tag ids.
outputDeviceType: The output device type.
("PC"/"LAPTOP"/"TV"/"PROJECTOR"/"OTHER_DEVICE")

networks: eth information. sub-fields:
  networkInterfaceName: The interface name of network.
  networkInterfaceMode: The interface mode of network.  ("DHCP"/"STATIC")
  networkInterfaceIpAddress: The interface ip of network.
  networkInterfaceSubnetMask: The interface subnetmask of network.
  networkInterfaceGateway: The interface gateway of network.
  networkInterfaceInst: The interface instance of network.
```

```
lan: The lan of device.  sub-fields:
  lanInterfaceName: The interface name of the LAN.
  lanInterfaceMode: The interface mode of the LAN.
  lanInterfaceIpAddress: The interface ip of the LAN.
  lanInterfaceSubnetMask: The interface subnetmask of the LAN.
  lanInterfaceGateway: The interface gateway of the LAN.

basicSetting: Basic settings.   sub-fields:
  deviceType: The type of device.("DECODER"/"ENCODER")
  deviceNmosName: The nmos name of device.
  deviceDnsServer: The dns server of device.
  deviceTrafficFlowMode: The traffic flow mode of device.
  deviceControlNic: The control NIC of
device.("CONNECTOR_1"/"CONNECTOR_2"/"CONNECTOR_3")
  deviceDanteName: The dante name of device.
  deviceDanteNic: The dante NIC of
device.("CONNECTOR_1"/"CONNECTOR_2"/"CONNECTOR_3")
  deviceUidCtl: Whether to enable uid Ctl.("ENABLED"/"DISABLED")
  deviceOutputCtl: Whether to enable output Ctl.("ENABLED"/"DISABLED")
  deviceTestModeCtl: Whether to enable test mode Ctl.("ENABLED"/"DISABLED")
  deviceReboot: Whether to reboot device.
  deviceUsbIpAddress: The usb ip of device.
  deviceUsbIpCtl: Whether to enable usb ip Ctl.
  deviceUsbNic: The usb NIC of device.("CONNECTOR_1"/"CONNECTOR_2"/"CONNECTOR_3")
  deviceSerialIpAddress: The usb ip of device.
  deviceSerialIpCtl: Whether to enable usb ip Ctl.
  deviceSerialNic: The usb NIC of
device.("CONNECTOR_1"/"CONNECTOR_2"/"CONNECTOR_3")
  deviceRemoteUpgradeFtpPath: The remote upgrade ftp path of device.
  deviceRemoteApplyProfile: The remote apply profile of device.
  deviceTemporaryProfile: The temporary profile name of device.
  deviceRemoteEdidFtpPath: The remote edid ftp path of device.
  devicePassword: The password of device.
  deviceFactoryReset: Whether to reset device.
  deviceEdidApply: The edid apply of device.
  deviceIgmpVersion: The igmp version of device.

videoStream: jxs video settings. sub-fields:
  streamState: The stream state of device stream.
  streamInterface: The stream interface of device.
  streamDstIpAddress: The stream destination ip of device.
  streamDstPort: The stream destination port of device.
  videoInputSelection: The input selection of
video.("HDMI"/"USB_TYPEC"/"IP"/"GENERATOR")
  videoStreamOutputState: The output state of video.("ENABLED"/"DISABLED")
  videoStreamCodec: The codec of video.("AVC"/"HEVC"/"JPEG_XS_HIGH"/"JPEG_XS_FIP")
  videoStreamColorMode: The color mode of video.("AUTO"/"MANUAL")
  videoStreamChromaFormat: The chroma format of
video.("RGB"/"YUV420"/"YUV422"/"YUV444")
  videoStreamColorDepth: The color depth of
video.("COLOR_DEPTH_8"/"COLOR_DEPTH_10"/"COLOR_DEPTH_12")
  videoStreamBitRateMode: The bitrate mode of
video.("ECONOMY"/"BALANCED"/"HIGH_QUALITY"/"MANUALLY"/"AUTO")
  videoStreamBpp: The bpp of video.
  videoStreamOutputFormatMode: The output format mode of video.("AUTO"/"MANUAL")
  videoStreamOutputManualResolution: The manual resolution of video.  value:
    "VIDEO_RESOLUTION_1280X720P"
    "VIDEO_RESOLUTION_1920X1080P"
    "VIDEO_RESOLUTION_3840X2160P"
```

```
  videoStreamOutputManualFrameRate: The manual frame rate of video.  value:
    "VIDEO_FRAME_RATE_2398"
    "VIDEO_FRAME_RATE_24"
    "VIDEO_FRAME_RATE_25"
    "VIDEO_FRAME_RATE_2997"
    "VIDEO_FRAME_RATE_30"
    "VIDEO_FRAME_RATE_50"
    "VIDEO_FRAME_RATE_5994"
    "VIDEO_FRAME_RATE_60"
    "VIDEO_FRAME_RATE_100"
    "VIDEO_FRAME_RATE_120"
  videoStreamSwitchingMode: The switching mode of video.("SEAMLESS"/"LOW_LATENCY")
  videoStreamSwitchingDisplay: The switching display of
video.("BLACK"/"LAST_FRAME")
  videoHdcpTransmitter: The hdcp transmitter of video.("HDCP_AUTO")
  videoHdcpPort: The hdcp server port of video.
  videoInputHdcpIpAddress: The hdcp connection ip of video.
  videoInputHdcpPort: The hdcp connection port of video.
  videoStreamBitrate: The bitrate of video.
  videoStreamAspectRatioMode: The aspect ratio mode of video.("MAINTAIN"/"STRETCH")
  videoInputHdcpCap: The input hdcp capability of video.("AUTO"/"OFF")
  videoCodecState: The codec state of video.("ENABLED"/"DISABLED")
  videoIpInputStreamType: The codec type of input ip
stream.("MAIN_STREAM"/"SUB_STREAM")

secondaryVideoStream: history.26x video settings. sub-fields:
  the same with videoStream

audioStream: jxs Audio settings. sub-fields:
  streamState: The stream state of device stream.("ENABLED"/"DISABLED")
  streamInterface: The stream interface of
device.("CONNECTOR_1"/"CONNECTOR_2"/"CONNECTOR_3")
  streamDstIpAddress: The stream destination ip of device.
  streamDstPort: The stream destination port of device.
  audioInputSelection: The input selection of audio.("HDMI"/"DANTE"/"ANALOG"/"IP")
  audioStreamBitDepth: The bit depth of
audio.("BIT_DEPTH_16"/"BIT_DEPTH_20"/"BIT_DEPTH_24")
  audioStreamSampleRate: The sample rate of audio.  value:
    "SAMPLE_RATE_32000"
    "SAMPLE_RATE_44100"
    "SAMPLE_RATE_48000"
    "SAMPLE_RATE_88200"
    "SAMPLE_RATE_96000"
    "SAMPLE_RATE_176400"
    "SAMPLE_RATE_192000"
  audioStreamVolume: The volume of audio.
  audioHdcpPort:
  audioInputHdcpIpAddress:
  audioInputHdcpPort:
  audioStreamMute: The mute of audio.("ENABLED"/"DISABLED")
  audioCodecState: Compressed audio.("ENABLED"/"DISABLED")
  audioCodecMode: ("AUDIO_MODE_AUTO"/"AUDIO_MODE_MANUAL")
  audioManualCodecType: ("AUDIO_CODEC_TYPE_PASSTHROUGH"/"AUDIO_CODEC_TYPE_AAC")
  audioManualCodecProfile:   value:
    "AUDIO_PROFILE_AAC_MAIN"    AAC-Main.
    "AUDIO_PROFILE_AAC_LC"      AAC-LC.
    "AUDIO_PROFILE_HE_AAC_V1"   HE-AAC-V1.
    "AUDIO_PROFILE_HE_AAC_V2"   HE-AAC-V2.

  audioManualCodecBitrate:   value:
```

```
    "AUDIO_BITRATE_16K"       Audio Bitrate 16k.
    "AUDIO_BITRATE_20K"       Audio Bitrate 20k.
    "AUDIO_BITRATE_32K"       Audio Bitrate 32k.
    "AUDIO_BITRATE_48K"       Audio Bitrate 48k.
    "AUDIO_BITRATE_56K"       Audio Bitrate 56k.
    "AUDIO_BITRATE_64K"       Audio Bitrate 64k.
    "AUDIO_BITRATE_80K"       Audio Bitrate 80k.
    "AUDIO_BITRATE_96K"       Audio Bitrate 96k.
    "AUDIO_BITRATE_112K"       Audio Bitrate 112k.
    "AUDIO_BITRATE_128K"       Audio Bitrate 128k.
    "AUDIO_BITRATE_160K"       Audio Bitrate 160k.
    "AUDIO_BITRATE_192K"       Audio Bitrate 192k.
    "AUDIO_BITRATE_224K"       Audio Bitrate 224k.
    "AUDIO_BITRATE_256K"       Audio Bitrate 256k.
    "AUDIO_BITRATE_320K"       Audio Bitrate 320k.
    "AUDIO_BITRATE_384K"       Audio Bitrate 384k.
    "AUDIO_BITRATE_448K"       Audio Bitrate 448k.

  audioIpInputStreamType: Input ip stream.("MAIN_STREAM"/"SUB_STREAM")

secondaryAudioStream: history.26x audio settings. sub-fields:
  the same with audioStream

dataStream: Data settings.   sub-fields:
  streamState: The stream state of device stream.("ENABLED"/"DISABLED")
  streamInterface: The stream interface of
device.("CONNECTOR_1"/"CONNECTOR_2"/"CONNECTOR_3")
  streamDstIpAddress: The stream destination ip of device.
  streamDstPort: The stream destination port of device.

secondaryDataStream:    sub-fields:
  the same with dataStream

hdmi: HDMI input.   sub-fields:
  hdmiOutputState: The output state of HDMI.
  hdmiCecStandardCommand: The CEC standard command of HDMI. value:
    "POWER_ON": Power on TV.
    "POWER_OFF": Power off TV.
    "AUDIO_MUTE": Mute audio.
    "AUDIO_UNMUTE": Unmute audio.

  hdmiCecCustomCommand: The CEC custom command of HDMI.
  hdmiEdidMode: The edid mode of HDMI.  value:
    "DEFAULT": Default mode.
    "PREDEFINED": Pre-defined mode.
    "COPY": Copy mode.
    "UPLOAD": Upload mode.

  hdmiHdrPassThrough: Whether HDR passthrough is enabled on
HDMI.("ENABLED"/"DISABLED")
  hdmiOutputTransmitter: The output transmitter of HDMI.  value:
    "HDCP_OFF": HDCP off.
    "HDCP_ON": HDCP on.
    "HDCP_AUTO": HDCP auto.
    "HDCP_1_4": HDCP 1.4.
    "HDCP_2_2": HDCP 2.2.
    "HDCP_2_3": HDCP 2.3.
    "HDCP_2_X_TYPE0": HDCP 2.X type0.
```

"HDCP_2_X_TYPE1": HDCP 2.X type1.
"HDCP_FOLLOW_INPUT": HDCP follow input.
"HDCP_FORCE_HIGHEST": HDCP force highest.
"HDCP_NEVER_AUTHENTICATE": HDCP never authenticate.

hdmiOutputSwitching: The output switching mode of HDMI. value:
"NO_OUTPUT": No Output.
"BLACK_FRAME": Black Frame.
"BACKGROUND_IMAGE": Background Image.

hdmiOutputTimeout: The output timeout period of HDMI.
hdmiMonitTransmitter: The monit transmitter of HDMI. value:
"HDCP_OFF": HDCP off.
"HDCP_ON": HDCP on.
"HDCP_AUTO": HDCP auto.
"HDCP_1_4": HDCP 1.4.
"HDCP_2_2": HDCP 2.2.
"HDCP_2_3": HDCP 2.3.
"HDCP_2_X_TYPE0": HDCP 2.X type0.
"HDCP_2_X_TYPE1": HDCP 2.X type1.
"HDCP_FOLLOW_INPUT": HDCP follow input.
"HDCP_FORCE_HIGHEST": HDCP force highest.
"HDCP_NEVER_AUTHENTICATE": HDCP never authenticate.

hdmiVolume: The volume of HDMI.
hdmiAudioMute: The output audio mute of HDMI.("ENABLED"/"DISABLED")

videoWall: Video wall.
rs232: RS232 input. sub-fields:
rs232State: The state of RS232.("ENABLED"/"DISABLED")
rs232Speed: The serial speed of RS232. value:
"SPEED_1200": 1200 baud.
"SPEED_2400": 2400 baud.
"SPEED_4800": 4800 baud.
"SPEED_9600": 9600 baud.
"SPEED_19200": 19200 baud.
"SPEED_38400": 38400 baud.
"SPEED_57600": 57600 baud.
"SPEED_115200": 115200 baud.
"SPEED_230400": 230400 baud.
"SPEED_460800": 460800 baud.
"SPEED_921600": 921600 baud.

rs232DataBits: The serial data bits of RS232. value:
"BIT_5": 5 bits.
"BIT_6": 6 bits.
"BIT_7": 7 bits.
"BIT_8": 8 bits.

rs232StopBits: The serial stop bits of RS232. value:
"BIT_1": 1 bit.
"BIT_2": 2 bits.

rs232FlowCtl: The serial flow control of RS232. value:
"NONE": None.
"XON_XOFF": XON/XOFF.
"RTS_CTS": RTS/CTS.
"DSR_DTR": DSR/DTR.

```
  rs232Parity: The serial parity of RS232.  value:
    "NONE": None.
    "ODD": Odd.
    "EVEN": Even.
    "MARK": Mark.
    "SPACE": Space.

  rs232HexMode: The serial cmd hex mode of RS232.("ENABLED"/"DISABLED")
  rs232RawCmd: The serial raw cmd of RS232.
  rs232CtrlCmd: The serial control cmd of RS232.
  rs232CustomerRowStatus: The serial customer row status of RS232.  value:
    "ACTIONSTATUS_ACTIVE": Active.
    "STATUS_NOT_IN_SERVICE": Not In Service.
    "STATUS_NOT_READY": Not Ready.
    "ACTION_CREATE_AND_GO": Create And Go.
    "ACTION_CREATE_AND_WAIT": Create And Wait.
    "ACTION_DESTROY": Destroy.

  rs232CustomerCmdName: The serial customer cmd name of RS232.
  rs232CustomerRawCmd: The serial customer raw cmd of RS232.
  rs232CustomerHexMode: The serial customer hex mode of
RS232.("ENABLED"/"DISABLED")
  rs232CustomerInst: The serial customer inst of RS232.

dante: Dante audio input.   sub-fields:
  danteAudioOutputState: The state of dante audio.("ENABLED"/"DISABLED")
  danteAudioSourceSelect: The source selection of dante audio. value:
    "HDMI": HDMI.
    "DANTE": Dante audio.
    "ANALOG": Analog audio.
    "IP": IP.

  danteAudioStreamVolume: The volume of dante audio.
  danteAudioMute: The mute of dante audio.("ENABLED"/"DISABLED")

analog: Analog audio input.   sub-fields:
  analogAudioState: The state of analog audio.("ENABLED"/"DISABLED")
  analogAudioSourceSelection: The source selection of analog audio. value:
    "HDMI": HDMI.
    "DANTE": Dante audio.
    "ANALOG": Analog audio.
    "IP": IP.
  analogAudioVolume: The volume of analog audio.
  analogAudioMute: The mute of analog audio.("ENABLED"/"DISABLED")
  analogAudioDirection: The direction of analog audio. value:
    "INPUT": Analog Audio input.
    "OUTPUT": Analog Audio output.

mainStreamLinkMode: main(JXS) stream link mode.  value:
  "LINK_SEPARATE": Link audio and video separately.
  "LINK_BOTH": Link both main audio and video.
subStreamLinkMode: sub(H26X) stream link mode.  value:
  "LINK_SEPARATE": Link audio and video separately.
  "LINK_BOTH": Link both main audio and video.
```

1- Encoder setting

GRAPHQL VARIABLES


```
{
  "input": {
    "ids": [
      "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQ0NzY0"
    ],
    "outputDeviceType": "PC",
    "basicSetting": {
      "deviceEdidApply":
".1.1.49.48.56.48.112.54.48.32.76.80.67.77.32.56.99.104.32.72.68.82"
    },
    "videoStream": {
      "videoStreamCodec": "JPEG_XS_HIGH",
      "videoStreamChromaFormat": "RGB",
      "videoStreamColorDepth": "COLOR_DEPTH_8",
      "videoStreamOutputFormatMode": "MANUAL",
      "videoStreamOutputManualFrameRate": "VIDEO_FRAME_RATE_60",
      "videoStreamOutputManualResolution": "VIDEO_RESOLUTION_3840X2160P",
      "videoStreamColorMode": "MANUAL",
      "videoStreamAspectRatioMode": "MAINTAIN",
      "videoStreamBitrate": 400,
      "videoInputSelection": "HDMI",
      "videoStreamBitRateMode": "MANUALLY",
      "videoInputHdcpCap": "AUTO"
    },
    "audioStream": {
      "audioInputSelection": "HDMI",
      "audioStreamVolume": 100,
      "audioStreamMute": "DISABLED"
    },
    "hdmi": {
      "hdmiEdidMode": "PREDEFINED",
      "hdmiMonitTransmitter": "HDCP_FORCE_HIGHEST"
    },
    "analog": {
      "analogAudioDirection": "INPUT"
    },
    "dante": {
      "danteAudioOutputState": "ENABLED",
      "danteAudioSourceSelect": "HDMI",
      "danteAudioStreamVolume": 151,
      "danteAudioMute": "DISABLED"
    },
    "rs232": {},
    "secondaryVideoStream": {
      "videoStreamOutputManualFrameRate": "VIDEO_FRAME_RATE_60",
      "videoStreamOutputManualResolution": "VIDEO_RESOLUTION_3840X2160P",
      "videoStreamAspectRatioMode": "MAINTAIN",
      "videoStreamOutputFormatMode": "MANUAL",
      "videoCodecState": "ENABLED",
      "videoStreamCodec": "AVC",
      "videoStreamBitRateMode": "MANUALLY",
      "videoStreamBitrate": 10,
      "videoStreamColorMode": "MANUAL",
      "videoStreamChromaFormat": "YUV420"
    },
```

```
    "secondaryAudioStream": {
      "audioCodecState": "ENABLED",
      "audioCodecMode": "AUDIO_MODE_MANUAL",
      "audioManualCodecType": "AUDIO_CODEC_TYPE_AAC",
      "audioManualCodecProfile": "AUDIO_PROFILE_AAC_MAIN",
      "audioManualCodecBitrate": "AUDIO_BITRATE_256K"
    }
  }
}
```

QUERY for the user to be copied

```
mutation updateDeviceList($input: UpdateDeviceListInput!) {
  updateDeviceList(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

2- Decoder setting

GRAPHQL VARIABLES

```
{
   "input" : {
      "analog" : {
         "analogAudioDirection" : "OUTPUT",
         "analogAudioMute" : "DISABLED",
         "analogAudioSourceSelection" : "IP",
         "analogAudioState" : "ENABLED",
         "analogAudioVolume" : 100
      },
      "audioStream" : {
         "audioInputSelection" : "IP"
      },
      "basicSetting" : {
         "deviceIgmpVersion" : "V2"
      },
      "dante" : {
         "danteAudioMute" : "DISABLED",
         "danteAudioOutputState" : "ENABLED",
         "danteAudioSourceSelect" : "IP",
         "danteAudioStreamVolume" : 100
      },
      "hdmi" : {
         "hdmiAudioMute" : "DISABLED",
         "hdmiOutputSwitching" : "BLACK_FRAME",
         "hdmiOutputTimeout" : 10,
         "hdmiOutputTransmitter" : "HDCP_FORCE_HIGHEST",
         "hdmiVolume" : 24
      },
      "ids" : [ "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3" ],
      "rs232" : {
         "rs232DataBits" : "BIT_8",
         "rs232FlowCtl" : "NONE",
         "rs232Parity" : "NONE",
```

```
            "rs232Speed" : "SPEED_115200",
            "rs232StopBits" : "BIT_1"
        },
        "secondaryAudioStream" : {},
        "secondaryVideoStream" : {},
        "videoStream" : {
            "videoInputSelection" : "IP",
            "videoIpInputStreamType" : "MAIN_STREAM",
            "videoStreamAspectRatioMode" : "MAINTAIN",
            "videoStreamChromaFormat" : "RGB",
            "videoStreamColorDepth" : "COLOR_DEPTH_8",
            "videoStreamColorMode" : "MANUAL",
            "videoStreamOutputFormatMode" : "MANUAL",
            "videoStreamOutputManualFrameRate" : "VIDEO_FRAME_RATE_60",
            "videoStreamOutputManualResolution" : "VIDEO_RESOLUTION_3840X2160P",
            "videoStreamSwitchingDisplay" : "BLACK",
            "videoStreamSwitchingMode" : "LOW_LATENCY"
        }
    }
}
```


QUERY for the user to be copied

```
mutation updateDeviceList($input: UpdateDeviceListInput!){
  updateDeviceList(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

Maintain Device

1- FW Update

GRAPHQL VARIABLES


```
{
  "input": {
    "ids": [
      "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYzM3"
    ]
  }
}
```

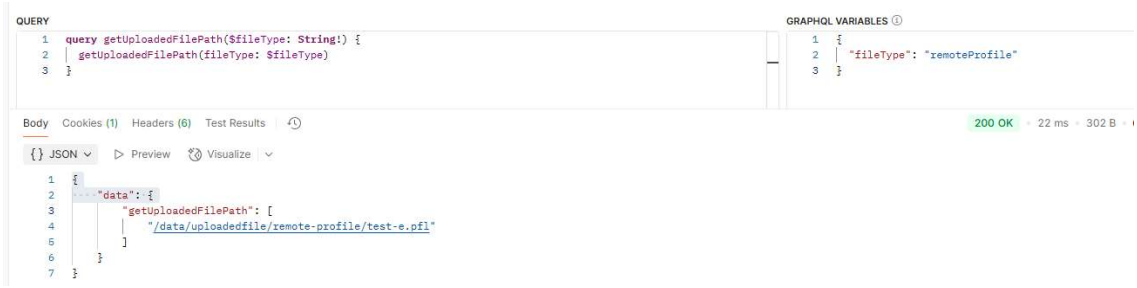

QUERY for the user to be copied

```
mutation startUpgradingRemoteDevices($input: ResourceRemoteUpgradeInput!) {
  startUpgradingRemoteDevices(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
```

}

2- Profile

1.Get profile list



GRAPHQL VARIABLES

```
{
  "fileType": "remoteProfile"
}
```

QUERY for the user to be copied

```
query getUploadedFilePath($fileType: String!) {
  getUploadedFilePath(fileType: $fileType)
}
```

2.Select a profile (get filename from the path)and apply

GRAPHQL VARIABLES

```
{
  "input": {
    "ids": [
      "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQ0NzY0"
    ],
    "profileName": "test-e.pfl"
  }
}
```

QUERY for the user to be copied

```
mutation applyRemoteProfile($input: ResourceRemoteProfileInput!) {
  applyRemoteProfile(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

3- Factory Default

GRAPHQL VARIABLES

```
{
    "input": {
        "ids": [
            "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi"
```

```
        ],
        "basicSetting": {
            "deviceFactoryReset": "ENABLED"
        }
    }
}
```

QUERY for the user to be copied

```
mutation updateDeviceList($input: UpdateDeviceListInput!) {
  updateDeviceList(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

4- Reboot

GRAPHQL VARIABLES

```
{
    "input": {
        "ids": [
            "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi"
        ],
        "basicSetting": {
            "deviceReboot": true
        }
    }
}
```

QUERY for the user to be copied

```
mutation updateDeviceList($input: UpdateDeviceListInput!) {
  updateDeviceList(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

5- Change Encode/Decode Mode

GRAPHQL VARIABLES

```
{
    "input": {
        "ids": [
            "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi"
        ],
```

```
        "basicSetting": {
            "deviceType": "DECODER"
        }
    }
}
```

QUERY for the user to be copied

```
mutation updateDeviceList($input: UpdateDeviceListInput!) {
  updateDeviceList(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

6- Test Mode ON/OFF

GRAPHQL VARIABLES

```
{
    "input": {
        "ids": [
            "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi"
        ],
        "basicSetting": {
            "deviceTestModeCtl": "ENABLED"
        }
    }
}
```

QUERY for the user to be copied

```
mutation updateDeviceList($input: UpdateDeviceListInput!) {
  updateDeviceList(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

Start / Stop Streaming

`deviceOutputCtl: "ENABLED" / "DISABLED"`

GRAPHQL VARIABLES

```
{
  "input": {
    "ids": [
      "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi"
```

```
    ],
    "basicSetting": {
      "deviceOutputCtl": "DISABLED"
    }
  }
}
```

QUERY for the user to be copied

```
mutation updateDeviceList($input: UpdateDeviceListInput!) {
  updateDeviceList(input: $input) {
 errorType  {
      code
      message
      __typename
    }
    __typename
  }
}
```

Unit ID

```
"deviceUidCtl": "ENABLED" / "DISABLED"
```

GRAPHQL VARIABLES

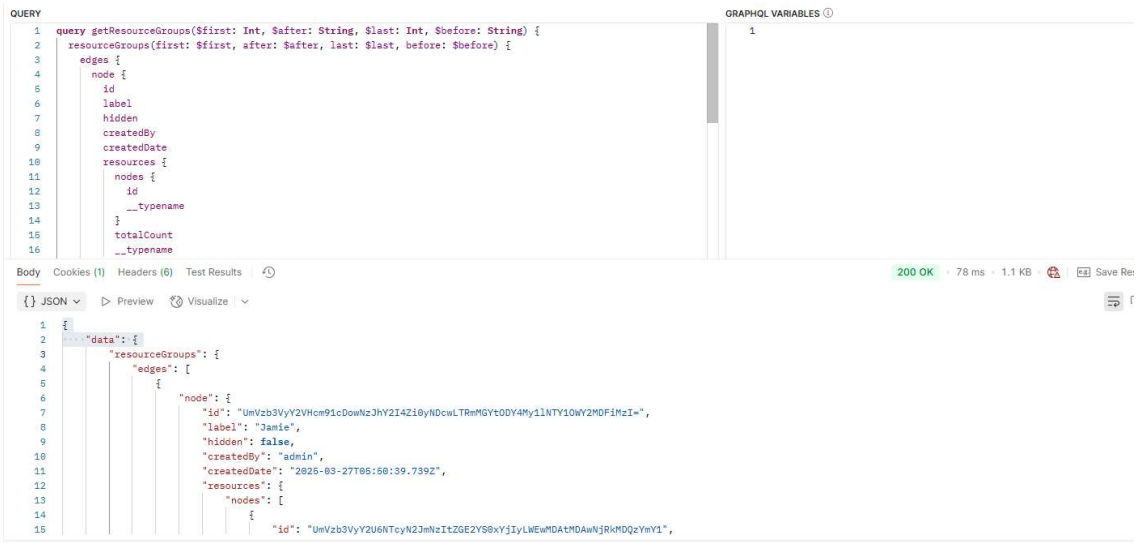
```
{
  "input": {
    "ids": [
      "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDYxOGFi"
    ],
    "basicSetting": {
      "deviceUidCtl": "ENABLED"
    }
  }
}
```

QUERY for the user to be copied

```
mutation updateDeviceList($input: UpdateDeviceListInput!) {
  updateDeviceList(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

Group

1.Group List



GRAPHQL VARIABLES

QUERY for the user to be copied

```
query getResourceGroups($first: Int, $after: String, $last: Int, $before: String) {
  resourceGroups(first: $first, after: $after, last: $last, before: $before) {
    edges {
      node {
        id
        label
        hidden
        createdBy
        createdDate
        resources {
          nodes {
            id
            __typename
          }
          totalCount
          __typename
        }
        tags {
          nodes {
            id
            label
            color
            __typename
          }
          __typename
        }
        __typename
      }
      __typename
    }
    totalCount
    pageInfo {
      hasNextPage
      hasPreviousPage
      startCursor
      endCursor
      __typename
    }
    __typename
  }
}
```

2.Create Group

GRAPHQL VARIABLES

```
{
  "input": {
    "label": "test",
    "hidden": false,
    "tagIds": [],
    "resourceIds": [
      "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQ0NzY0"
    ],
    "createdBy": "admin"
  }
}
```

QUERY for the user to be copied

```
mutation createResourceGroup($input: CreateResourceGroupInput!) {
  createResourceGroup(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

3.Update Group

GRAPHQL VARIABLES

```
{
  "input": {
    "id": "UmVzb3VyY2VHcm91cDplNGY5NDdkNi04MzRkLTQwNDItOWE5OC05YWU3MWYyNDRkZjc=",
    "label": "test2",
    "tagIds": []
  }
}
```

QUERY for the user to be copied

```
mutation updateResourceGroup($input: UpdateResourceGroupInput!) {
  updateResourceGroup(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

4.Delete Group

GRAPHQL VARIABLES

```
{
  "id": "UmVzb3VyY2VHcm91cDplNGY5NDdkNi04MzRkLTQwNDItOWE5OC05YWU3MWYyNDRkZjc="
}
```

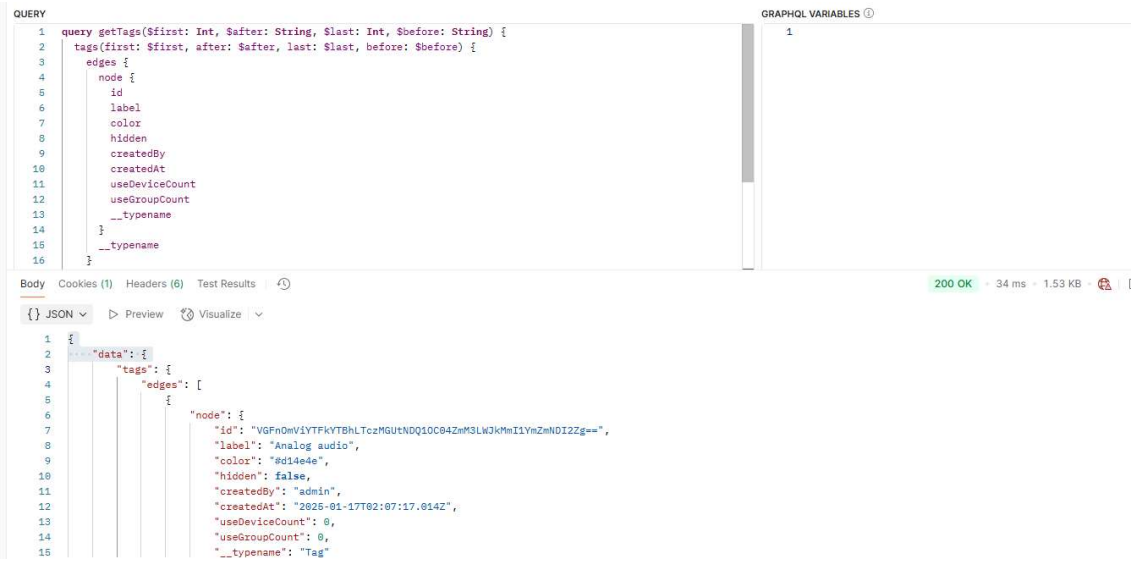
QUERY for the user to be copied

```
mutation deleteResourceGroup($id: ID!) {
  deleteResourceGroup(id: $id) {
    errorType {
      code
      message
```

```
      __typename
    }
    __typename
  }
}
```

Tag

1.Tag List



GRAPHQL VARIABLES

QUERY for the user to be copied

```
query getTags($first: Int, $after: String, $last: Int, $before: String) {
  tags(first: $first, after: $after, last: $last, before: $before) {
    edges {
      node {
        id
        label
        color
        hidden
        createdBy
        createdAt
        useDeviceCount
        useGroupCount
        __typename
      }
      __typename
    }
    totalCount
    pageInfo {
      hasNextPage
      hasPreviousPage
      startCursor
      endCursor
      __typename
    }
    __typename
```

```
  }
}
```

2.Create Tag

GRAPHQL VARIABLES

```
{
  "input": {
    "label": "warn",
    "hidden": false,
    "color": "#d14e4e",
    "createdBy": "admin"
  }
}
```

QUERY for the user to be copied

```
mutation createTag($input: CreateTagInput!) {
  createTag(input: $input) {
    tag {
      id
      label
      color
      __typename
    }
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

3.Update Tag

GRAPHQL VARIABLES

```
{
  "input": {
    "id": "VGFnOjU4NzA1ZjM1LTAxZWQtNGM1NC1hMjNmLWU2Njg0ZjVkNDk1MA==",
    "color": "rgba(189, 16, 224, 1)",
    "label": "warn-red"
  }
}
```

QUERY for the user to be copied

```
mutation updateTag($input: UpdateTagInput!) {
  updateTag(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

4.Delete Tag

GRAPHQL VARIABLES

```
{
  "id": "VGFnOjU4NzA1ZjM1LTAxZWQtNGM1NC1hMjNmLWU2Njg0ZjVkNDk1MA=="
}
```

QUERY for the user to be copied

```
mutation deleteTag($id: ID!) {
  deleteTag(id: $id) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}
```

Additional information

Thumbnails