

P-AVN-VA GraphQL API GUIDE 1.1.0

P-AVN-VA firmware 1.1.0

Version 1

DRAFT

Contents

Introduction.....	4
NMOS vs PlexusAV API.....	5
GraphQL API	6
Getting started.....	8
API Version Management	8
Postman set up	9
VA Log in	10
Error handling.....	11
First Query: Get System Information.....	12
Discovering IPMX devices	13
Discovering IPMX Streams	17
API Command Examples	19
AV Matrix routing	19
1- All routing	20
2- AV routing	26
3- KVM routing	32
4- Data routing.....	35
Video Wall.....	41
1.Video wall list	41
2.Create Video wall	42
3.Update Video wall	43
4.Delete Video wall	44
Basic Setting	45
Variables description in devices	48
Advanced Setting	49
Variable description in input in updateDeviceList	49
1- Encoder setting.....	55
2- Decoder setting	56
Maintain Device.....	57
1- FW Update.....	57
2- Profile.....	58
3- Factory Default	58
4- Reboot.....	59

5- Change Encode/Decode Mode	59
6- Test Mode ON/OFF	60
Start / Stop Streaming.....	60
Unit ID.....	61
Group	62
1.Group List	62
2.Create Group	63
3.Update Group	63
4.Delete Group	63
Tag	64
1.Tag List.....	64
2.Create Tag.....	65
3.Update Tag.....	65
4.Delete Tag.....	66
Additional information.....	66
Thumbnails	66

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

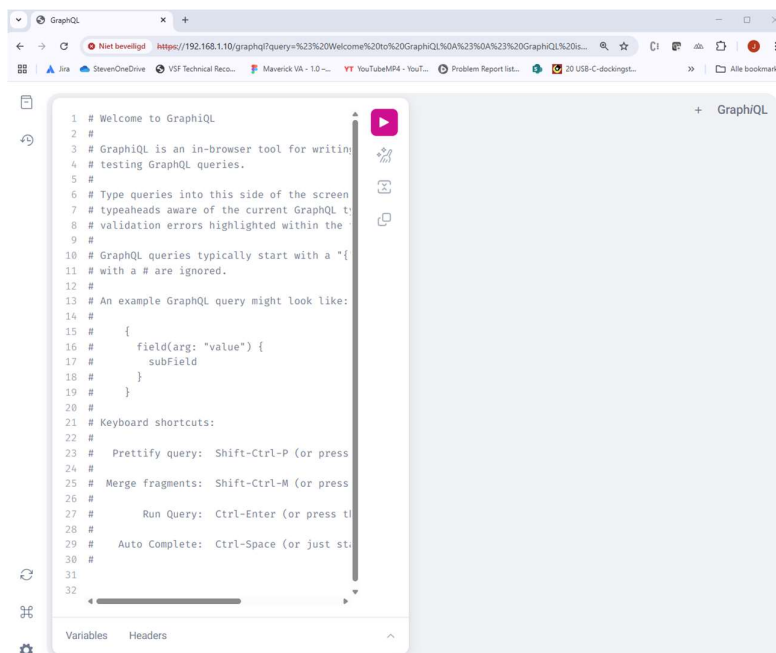
The GraphQL API that is being used in PlexusAV P-AVN-2/4 and Visual Array has many advantages over the traditional REST API. GraphQL utilizes **a single endpoint** for sending queries or mutations, allowing clients to request **exactly the data they need**.

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](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.

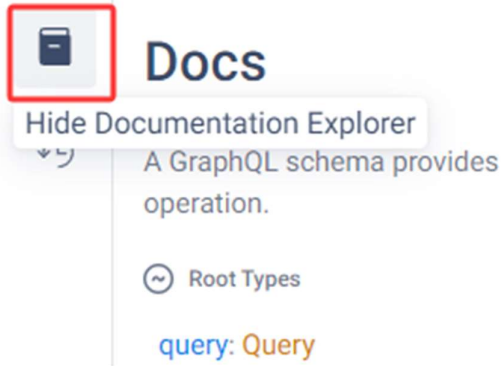


GraphQL uses **Queries** and **Mutations**:

- **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 schema 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

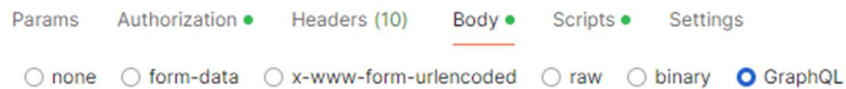
Here are the steps needed to send GraphQL requests from [Postman](#) to the PlexusAV P-AVN-VA.

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](https://[P-AVN-VA IP Address]/graphql))

2. Switch to GraphQL Mode

- In the Body tab, select GraphQL

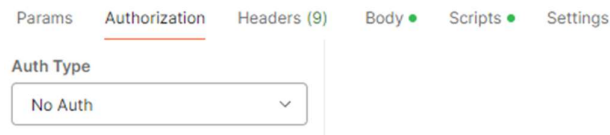


- You'll see a text box where you can enter your GraphQL query

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

The screenshot shows a GraphQL client interface with the following components:

- URL:** `https://192.168.0.84/graphql`
- Method:** `POST`
- Body Tab:** Selected, showing the GraphQL mutation:

```
1 mutation login($remember: Boolean!, $password: String!, $name: String!) {
2   login(name: $name, password: $password, remember: $remember) {
3     errorType {
4       code
5       message
6     }
7     cookie {
8       key
9       value
10    }
11  }
12 }
```
- GraphQL VARIABLES Tab:** Shows the variables for the mutation:

```
1 {
2   "name": "admin",
3   "password": "proav101",
4   "remember": true
5 }
```
- Body Tab:** Shows the JSON response:

```
1 {
2   "data": {
3     "login": {
4       "errorType": {
5         "code": "SUCCESS",
6         "message": "Success."
7       },
8       "cookie": {
9         "key": "auth_session",
10        "value": "62b1b88a0e08b1000f4d0e0e0e0e0e0e"
11      }
12    }
13  }
14 }
```

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
}
}
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

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

1000

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

```

        "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
}

```

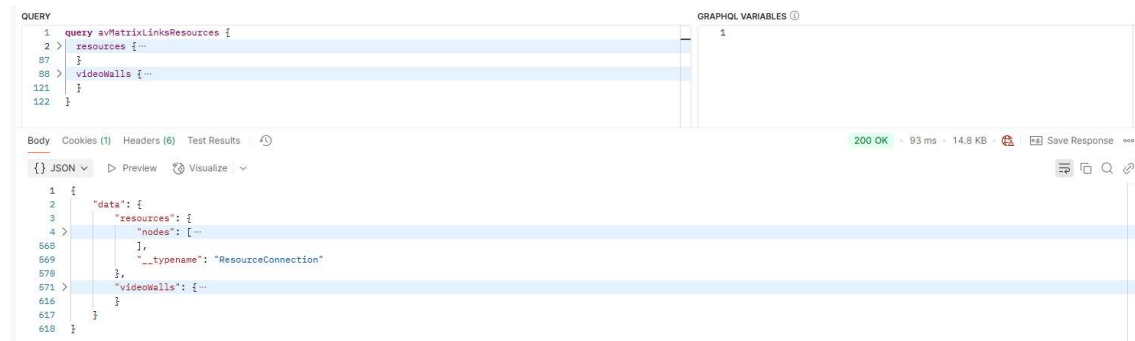
```

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
}
}
__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:

```
1 mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
2   createAvMatrixSubLinkGroup(input: $input) {
3     code
4     message
5     __typename
6   }
7 }

GRAPHQL VARIABLES
1 {
2   "input": {
3     "avMatrixLinkGroupInput": [
4       {
5         "senderId":
6           "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS8xYjYyLWwMDAtMDAwNjRkMDQzZDQ1",
7         "receiverId":
8           "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS8xYjYyLWwMDAtMDAwNjRkMDQzYmY1",
9         "linkType": "LINK_MAIN_VIDEO"
10      },
11      {
12        "senderId":
13          "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS8xYjYyLWwMDAtMDAwNjRkMDQzZDQ1",
14        "receiverId":
15          "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS8xYjYyLWwMDAtMDAwNjRkMDQzYmY1",
16        "linkType": "LINK_AUDIO"
17      },
18      {
19        "senderId":
20          "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS8xYjYyLWwMDAtMDAwNjRkMDQzZDQ1",
21        "receiverId":
22          "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS8xYjYyLWwMDAtMDAwNjRkMDQzYmY1",
23        "linkType": "LINK_RS232"
24      }
25    ]
26  }
27 }
```

GRAPHQL VARIABLES

```

{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "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.

The screenshot shows a GraphQL IDE interface with three main sections: QUERY, GRAPHQL VARIABLES, and the response body.

QUERY:

```

1 mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
2   deleteAvMatrixSubLinkGroup(input: $input) {
3     code
4     message
5     __typename
6   }
7 }

```

GRAPHQL VARIABLES:

```

1 {
2   "input": {
3     "avMatrixLinkGroupInput": [
4       {
5         "senderId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
6         "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
7         "linkType": "LINK_MAIN_VIDEO"
8       },
9       {
10        "senderId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
11        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
12        "linkType": "LINK_AUDIO"
13      }
14     ]
15   }
16 }

```

Response Body (JSON):

```

1 {
2   "data": {
3     "deleteAvMatrixSubLinkGroup": {
4       "code": "SUCCESS",
5       "message": "Success.",
6       "typename": "ErrorType"
7     }
8   }
9 }

```

GRAPHQL VARIABLES

```

{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",

```

```

      "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
      "linkType": "LINK_MAIN_VIDEO"
    },
    {
      "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
      "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
      "linkType": "LINK_AUDIO"
    },
    {
      "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
      "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
      "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:

The screenshot displays a GraphQL IDE interface with three main sections: QUERY, GRAPHQL VARIABLES, and the response body.

QUERY:

```

1 mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
2   createAvMatrixSubLinkGroup(input: $input) {
3     code
4     message
5     __typename
6   }
7 }

```

GRAPHQL VARIABLES:

```

1 {
2   "input": {
3     "avMatrixLinkGroupInput": [
4       {
5         "senderId":
6         "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
7         "receiverId":
8         "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
9         "linkType": "LINK_MAIN_VIDEO"
10      },
11      {
12        "senderId":
13        "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
14        "receiverId":
15        "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
16        "linkType": "LINK_AUDIO"
17      },
18      {
19        "senderId":
20        "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
21        "receiverId":
22        "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
23        "linkType": "LINK_RS232"
24      }
25    ]
26  }
27 }

```

Body (JSON):

```

1 {
2   "data": {
3     "createAvMatrixSubLinkGroup": {
4       "code": "SUCCESS",
5       "message": "Success.",
6       ".__typename": "ErrorType"
7     }
8   }
9 }

```

At the bottom, the status bar shows "200 OK", "66 ms", and "326 B".

GRAPHQL VARIABLES

```
{
  "input": {
    "avMatrixLinkGroupInput": [
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "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.



The screenshot shows the GraphQL Playground interface. On the left, the 'QUERY' tab is active, displaying a mutation query to delete a link group. On the right, the 'GRAPHQL VARIABLES' tab is active, showing the variables for the query. The query is as follows:

```
1 mutation deleteAvMatrixSubLinkGroup($input: DeleteAvMatrixSubLinkGroupInput!) {
2   deleteAvMatrixSubLinkGroup(input: $input) {
3     code
4     message
5     __typename
6   }
7 }
```

The variables are as follows:

```
1 {
2   "input": {
3     "avMatrixLinkGroupInput": [
4       {
5         "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
6         "receiverId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
7         "linkType": "LINK_MAIN_VIDEO"
8       },
9       {
10        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
11        "receiverId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
12        "linkType": "LINK_AUDIO"
13      },
14    ]
15  }
16 }
```

At the bottom, the 'Body' tab shows the JSON response, which indicates a successful deletion:

```
1 {
2   "data": {
3     "deleteAvMatrixSubLinkGroup": {
4       "code": "SUCCESS",
5       "message": "Success.",
6       "typename": "ErrorType"
7     }
8   }
9 }
```

GRAPHQL VARIABLES

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

```

```

        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_MAIN_VIDEO"
    },
    {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "linkType": "LINK_AUDIO"
    },
    {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "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'

The screenshot shows the GraphQL Playground interface. At the top, there are tabs for 'none', 'form-data', 'x-www-form-urlencoded', 'raw', 'binary', 'GraphQL' (selected), 'Auto Fetch', and 'Schema Fetched'. The 'QUERY' tab is active, showing a mutation query:

```
1 mutation createAvMatrixSubLinkGroup($input: CreateAvMatrixSubLinkGroupInput!) {
2   createAvMatrixSubLinkGroup(input: $input) {
3     code
4     message
5     __typename
6   }
7 }
```

 The 'GRAPHQL VARIABLES' tab is also active, showing variables:

```
24 {
25   "senderId":
26     "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
27   "receiverId":
28     "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
29   "linkType": "LINK_MAIN_VIDEO"
30 }
31 {
32   "senderId":
33     "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzZDQ1",
34   "receiverId":
35     "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWEwMDAtMDAwNjRkMDQzYmY1",
36   "linkType": "LINK_AUDIO"
37 }
```

 The 'Body' tab is active, showing the JSON response:

```
1 {
2   "data": {
3     "createAvMatrixSubLinkGroup": {
4       "code": "SUCCESS",
5       "message": "Success.",
6       "__typename": "ErrorType"
7     }
8   }
9 }
```

 At the bottom right, there is a status bar showing '200 OK', '36 ms', '326 B', and a 'Save Response' button.

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":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDYxOGFi",
      "linkType": "LINK_MAIN_VIDEO"
    },
    {
      "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDA2",
      "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
      "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDA2",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",
        "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDA2",
          "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",
          "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_MAIN_VIDEO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_AUDIO"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",
        "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",
        "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_RS232"
    },
    {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_RS232"
    },
    {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",
        "linkType": "LINK_RS232"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDYxOGFi",
        "linkType": "LINK_RS232"
      },
      {
        "senderId": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",
        "receiverId":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYmY1",
        "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",  
        "receiverId":  
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",  
        "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": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzZDQ1",  
        "receiverId":  
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",  
        "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
  }
}
totalCount
pageInfo {
  hasNextPage
  hasPreviousPage
  startCursor
  endCursor
  __typename
}
__typename
}
}

```

2.Create Video wall

Params Authorization Headers (10) Body Scripts Tests Settings Cookie

☐ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☒ GraphQL Auto Fetch ☐ Schema Fetched

QUERY

```

1 mutation createVideoWall($input: CreateVideoWallInput!) {
2   createVideoWall(input: $input) {
3     ... errorType {
4       ... code
5       ... message
6       ... __typename
7     }
8     ... __typename
9   }
10 }

```

GRAPHQL VARIABLES

```

1 {
2   "input": {
3     "label": "Video wall 2",
4     "videoWallMode": "DISABLED",
5     "matrixSize": "MATRIX_3X3",
6     "devices": [
7       {
8         "panelId": "A1",
9         "output": {
10          "resolution": "VIDEO_RESOLUTION_3840X2160P",
11          "frameRate": "VIDEO_FRAME_RATE_60",
12          "rotation": "ROTATION_0_CN"
13        },
14        "bezel": {
15          "top": 0,
16          "bottom": 0,
17          "left": 0,
18          "right": 0
19        },
20        "resourceId":
21          "UnVzb3VyY2U6NTcyN2MzIit2GE2YS0xYjIyLWwMDAtMDAwjRkMDQzY

```

Body Cookies (1) Headers (6) Test Results Visualize

{ } JSON Preview Visualize

```

1 {
2   "data": {
3     "createVideoWall": {
4       "errorType": {
5         "code": "UNKNOWN_ERROR",
6         "message": "Unknown error",
7         "__typename": "ErrorType"
8       },
9       "__typename": "CreateVideoWallResponse"
10     }
11   }
12 }

```

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":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWwMDAwMDAwNjRkMDQzYmY1",
    "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

The screenshot shows a GraphQL IDE interface with a query editor on the left and a variables editor on the right. The query is a mutation to update a video wall. The variables are an object containing the input for the mutation.

QUERY

```

1 mutation updateVideoWall($input: UpdateVideoWallInput!) {
2   updateVideoWall(input: $input) {
3     errorType {
4       code
5       message
6       __typename
7     }
8     __typename
9   }
10 }

```

GRAPHQL VARIABLES

```

1 {
2   "input": {
3     "label": "video wall 2",
4     "videoWallMode": "DISABLED",
5     "matrixSize": "MATRIX_2X2",
6     "devices": [
7       {
8         "panelId": "A1",
9         "output": {
10          "resolution": "VIDEO_RESOLUTION_3840X2160P",
11          "frameRate": "VIDEO_FRAME_RATE_60",
12          "rotation": "ROTATION_0_CW"
13        },
14        "bezel": {
15          "top": 0,
16          "bottom": 0,
17          "left": 0,
18          "right": 0
19        },
20        "resourceId":
          "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWwMDAwMDAwNjRkMDQzYmY1"
21      }
22    ]
23  }
24 }

```

The response is shown in the bottom panel, indicating a successful update.

```

1 {
2   "data": {
3     "updateVideoWall": {
4       "errorType": {
5         "code": null,
6         "message": null,
7         "__typename": null
8       },
9       "__typename": null
10     }
11   }
12 }

```

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":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQzYzM3",
      "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

The screenshot shows a GraphQL IDE interface with two main panels: 'QUERY' on the left and 'GRAPHQL VARIABLES' on the right. Below these panels is a status bar and a JSON viewer.

QUERY:

```

1 mutation deleteVideoWall($id: ID!) {
2   deleteVideoWall(id: $id) {
3     errorType {
4       code
5       message
6       __typename
7     }
8     __typename
9   }
10 }

```

GRAPHQL VARIABLES:

```

1 {
2   "id":
3     "VmlkZW9XYWxsOmFiNDdkYThjLWU2YTgtNDU2MC1iZjAwLTBhZDg1ZTcyMTY2Zg=="
4 }

```

Status Bar: 200 OK · 50 ms · 435 B · Save Response

JSON Viewer:

```

1 {
2   "data": {
3     "deleteVideoWall": {
4       "errorType": {
5         "code": "NOT_FOUND",
6         "message": "Video wall not found",
7         "__typename": "ErrorType"
8       }
9     }
10  }
11 }
12 }

```

GRAPHQL VARIABLES

```
{
  "id": "VmlkZW9XYWxsOmI4Y2M5ZTclLTRlZWYtNGQ0OC04MTAxLTJmZjExYjRiMjFkZA=="
}
```

QUERY for the user to be copied

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

Basic Setting

The screenshot shows a GraphQL IDE interface. On the left, the 'QUERY' tab is active, displaying a mutation query: `mutation updateDeivces($input: UpdateDeivcesInput!) { updateDeivces(input: $input) { errorType { code message __typename } } }`. On the right, the 'GRAPHQL VARIABLES' tab is active, showing a JSON object for the variables: `{ "input": { "devices": [ { "id": "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQ0NzY0", "tagIds": [], "basicSetting": { "deviceNmosName": "E-DV1-91", "deviceDanteName": "E-DV1-91" }, "videoStream": { "streamDstIpAddress": "239.0.0.91", "streamDstPort": 5084 } } ] } }`. Below these panels, the 'Body' tab shows the JSON response: `{ "data": { "updateDeivces": { "errorType": { "code": "SUCCESS", "message": "Success.", "typename": "ErrorType" }, "typename": "ResourceDevicePayload" } } }`. The status bar at the bottom indicates a successful response with a 200 OK status, 110 ms execution time, and 364 B response size.

GRAPHQL VARIABLES

```
{
  "input": {
    "devices": [
      {
        "id":
"UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQ0NzY0",
        "groupId":
"UmVzb3VyY2VHcm91cDowNzJhY2I4Zi0yNDcwLTRmMGYtODY4My1lNTY1OWY2MDFiMzI=",
        "tagIds": [
          "VGFnOmViYTFkYTBlLTczMGUtNDQ1OC04ZmM3LWJkMmI1YmZmNDI2Zg=="
        ],
        "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":
    "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDYxOGFi",
    "groupId":
    "UmVzb3VyY2VHcm91cDowNzJhY2I4Zi0yNDcwLTRmMGYtODY4Myl1NTY1OWY2MDFiMzI=",
    "tagIds": [
        "VGFnOmViYTFkYTBlLTczMGUtNDQ1OC04ZmM3LWJkMmI1YmZmNDI2Zg=="
    ],
    "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":
    "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWwMDAtMDAwNjRkMDQzYzM3",
    "groupId":
    "UmVzb3VyY2VHcm91cDowNzJhY2I4Zi0yNDcwLTRmMGYtODY4My1lNTY1OWY2MDFiMzI=",
    "tagIds": [
        "VGFmOmViYTFkYTBhLTczMGUtNDQ1OC04ZmM3LWJkMmI1YmZmNDI2Zg=="
    ],
    "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.
  deviceIcmpVersion: The icmp 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": [
      "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQ0NzY0"
    ],
    "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"
    },
    "hdm": {
      "hdmEdidMode": "PREDEFINED",
      "hdmMonitTransmitter": "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

```

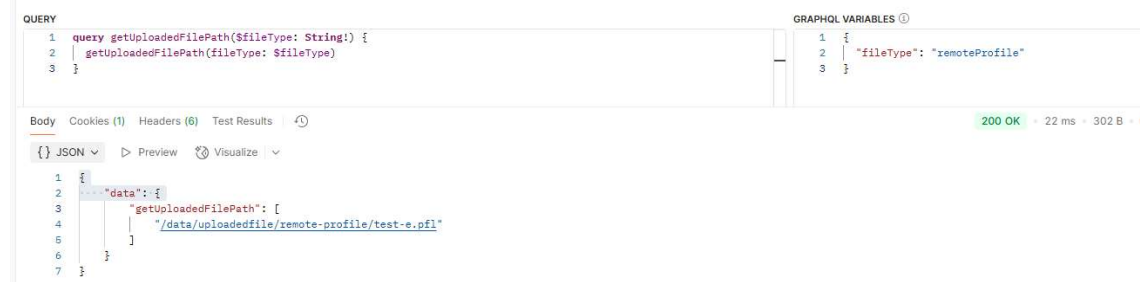
mutation startUpgradingRemoteDevices($input: ResourceRemoteUpgradeInput!) {
  startUpgradingRemoteDevices(input: $input) {
    errorType {
      code
      message
      __typename
    }
    __typename
  }
}

```

```
}
```

2- Profile

1. Get profile list



GRAPHQL VARIABLES

```
{  
  "filePath": "remoteProfile"  
}
```

QUERY for the user to be copied

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

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

GRAPHQL VARIABLES

```
{  
  "input": {  
    "ids": [  
      "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDQ0NzY0"  
    ],  
    "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": [  
      "UmVzb3VyY2U6NTcyN2JmNzItZGE2YS0xYjIyLWUwMDAtMDAwNjRkMDYxOGFi"  
    ]  
  }  
}
```

```

    ],
    "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

QUERY

```
1 query getResourceGroups($first: Int, $after: String, $last: Int, $before: String) {
2   resourceGroups(first: $first, after: $after, last: $last, before: $before) {
3     edges {
4       node {
5         id
6         label
7         hidden
8         createdBy
9         createdAt
10        resources {
11          nodes {
12            id
13            __typename
14          }
15          totalCount
16          __typename

```

GRAPHQL VARIABLES

1

Body Cookies (1) Headers (6) Test Results

200 OK • 78 ms • 1.1 KB Save Re

{ } JSON Preview Visualize

```
1 {
2   "data": {
3     "resourceGroups": {
4       "edges": [
5         {
6           "node": {
7             "id": "UnVzb3VyY2VHcm91cDowNzJhY2I4Zi8yNDcwLTZhMGVtODY4My1lNTY1OWY2MDFiMzI=",
8             "label": "Jamie",
9             "hidden": false,
10            "createdBy": "admin",
11            "createdAt": "2026-03-27T05:50:39.739Z",
12            "resources": {
13              "nodes": [
14                {
15                  "id": "UnVzb3VyY2U6NTcyNzJhNzItZGE2YS8xYjIyLWUwMDAtMDAwNjRkMDQzMmY1",

```

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
        createdAt
        resources {
          nodes {
            id
            __typename
          }
          totalCount
          __typename
        }
      }
    }
    tags {
      nodes {
        id
        label
        color
        __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

The screenshot shows a GraphQL IDE interface. The top section displays a query and its variables. The query is:

```

1 query getTags($first: Int, $after: String, $last: Int, $before: String) {
2   tags(first: $first, after: $after, last: $last, before: $before) {
3     edges {
4       node {
5         id
6         label
7         color
8         hidden
9         createdBy
10        createdAt
11        useDeviceCount
12        useGroupCount
13        __typename
14      }
15      __typename
16    }
17  }
18 }

```

The right panel shows the GraphQL variables:

```

1 {
2   first: 10
3   after: ""
4   last: 10
5   before: ""
6 }

```

The bottom section shows the JSON response:

```

1 {
2   "data": {
3     "tags": {
4       "edges": [
5         {
6           "node": {
7             "id": "VGFnOmVlYTFkYTBlTzMGUzNDQ1OC84ZmM3LWJkMmI1YmZmNDI2Zg==",
8             "label": "Analog audio",
9             "color": "#d14e4e",
10            "hidden": false,
11            "createdBy": "admin",
12            "createdAt": "2025-01-17T02:07:17.014Z",
13            "useDeviceCount": 0,
14            "useGroupCount": 0,
15            "__typename": "Tag"
16          }
17        }
18      ]
19    }
20  }
21 }

```

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