Resources

Alta Instances

Resource	Description
GET /alta/alta_instances	**Action** Retrieves a list of Alta Virtual Encoder instances **Query Parameters** By default, this endpoint will return a list of instance IDs, each of which can be passed to other endpoints to get more infoabout an instance. However, if one or more of the following query parameters is provided and set to `true`, this endpoint will instead return a list of objects, each of which has additional info about an instance.

settings	Set this query paremeter to `true` to include the currently configured settings for each instance in the response.
status	Set this query paremeter to `true` to include the current status for each instance in the response.

Example Request:

GET http://username:password@localhost/alta/alta_instances

Example Response:

```
{
  "alta_instances": [
    "EEG_BR_3G46Z6VLSV8B01JZ",
    "EEG_BR_24RNNSDZ1J020IFJ"
  ]
}
```

Resource	Description

Resource	Description
POST /alta/alta_instances	Action Create and configure a new Alta Virtual Encoder instance JSON Parameters

multicastinterface	IP address of the local interface to subscribe to media multicast through and send them out. Note device startup will fail if this server does not have a NIC using the specified IP address. In 2022-7 usage, set the primary NIC IP here, and use secondarymulticastinterface for the secondary NIC IP.
icapco	Company credential for logging into EEG iCap (omit if not connecting to iCap)
icapuser	Encoder username credential for logging into EEG iCap
icappass	Password credential for logging into EEG iCap
audiomix	Comma-separated list of channels to be included in the mono mix down sent to the captioner. Recommended to list one, two, or six (5.1) channels
td1	Destination address for an external telnet caption encoder device that wants to receive cloned S1 caption data matching the iCap input to this channel (typically CTRL+A format)
tu1	Username for external S1 telnet clone device (if authentication is required)
tp1	Password for external S1 telnet clone device (if authentication is required)
td2	Destination address for a second external telnet device that wants to receive cloned caption data matching the iCap input to the tlang2 channel
tu2	Username for second external telnet clone device (if authentication is required)
tp2	Password for second externa telnet clone device (if authentication is required)
tlang2	Number 1-6 for the language/service number that second telnet clone should copy

Resource	Description
	commands for (based on iCap access code service number)
telnetinput	Local listen address to accept input caption data from non-iCap users or devices (insecure).
testcc	Write looping test closed caption messages into the output data. This feature will turn itself off for the remainder of the run if/when a live iCap captioner connects and begins writing.
warnlevel	Logging detail level
dvb_text_cfg	List of ISO 639 language codes and teletext magazine/page numbers, only needed for output of DVB Text. The string should be a comma separated list where each entry has a language code, a slash, and a page number, like "eng/801,spa/802".
name	Custom name for identifying this instance
icapaudiopid	PIDs in the input program to use for the iCap audio reference sent to the captioner. PIDs are associated with iCap language services in the order listed - i.e. the first PID listed is sent to S1 access codes, the second is sent to S2 access codes, etc.
icapvideopid	PID in the input program to use for the iCap video reference sent to the captioner
dvb_bitmap_cfg	List of ISO 639 language codes and bitmap subtitle composition/ancillary page id's, only needed for output of DVB Bitmap. The string should be a comma separated list where each entry has a language code, a slash, and an id number, like "eng/2,spa/4".
udp_buf_size	Delay (in milliseconds) required to buffer output for a smooth packet stream. An aggressive minimum setting would be 50 ms in low latency mode or with all I-frame compressions - the default 800 ms. You may need to set this parameter higher for streams with GOPs longer than 2 seconds.

Resource	Description

low_latency_mode	Disables some re-multiplexing features of the transport stream processor to reduce the delay from input to output. In low-latency mode, captions cannot be inserted in atsc_user_data, since this requires at least one GOP/AU of delay to handle picture re-ordering
scte35pids	A list of PIDS to insert SCTE-35 data on

Query Parameters

turn_on	For backwards compatibiity with previous versions of this API, the newly created instance will be immediately turned on by default. Set this query parameter to false to just create a new instance and save the settings without also turning the instance on.

Example Request:

POST http://username:password@localhost/alta/alta_instances?turn_on=false

JSON body:

```
{
  "output": "239.120.200.011:5000",
  "primary": "udp://239.120.100.015:5000",
  "source": "udp://239.120.100.014:5000",
  "warnlevel": 1
}
```

Example Response:

```
{
  "instanceID": "EEG_BR_3G46Z6VLSV8B01JZ"
}
```

Start Process

Resource	Description
POST /alta/alta_instances/<**instanceID**>/turn_on	Action Turns on a previously created Alta instance URL parameters **instanceID** — The instanceID for the Alta instance to turn on

Example Request:

POST http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/turn_on

Example Response:

```
{}
```

Stop Process

Resource	Description
POST /alta/alta_instances/<**instanceID**>/turn_off	Action Turns off a previously created Alta instance URL parameters **instanceID** — The instanceID for the Alta instance to turn off

Example Request:

POST http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/turn_off

Example Response:

```
{}
```

Toggle Process

Resource	Description
POST /alta/alta_instances/<**instanceID**>/toggle_on_off	Action Turns on a previously created Alta instance if it is currently off, or turns off a previously created Alta instance if it is currently on URL parameters **instanceID** — The instanceID for the Alta instance to toggle

Example Request:

POST http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/toggle_on_off

Example Response:

```
{}
```

Settings

Resource	Description
GET /alta/alta_instances/<**instanceID**>/settings	Action Retrieves Alta instance configuration/settings URL parameters **instanceID** — The instanceID for the Alta instance to retrieve

Example Request:

GET http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/settings

Example Response:

```
{
  "settings": {
    "primary": "udp://239.120.100.101:5000",
    "source": "rtp://224.0.101.140:5000",
    "output": "239.120.200.101:5000"
  }
}
```

Resource	Description
PUT /alta/alta_instances/<**instanceID**>/settings	**Action** Modifies Alta instance configuration/settings. Note that this operation is only supported for instances that are off. **JSON Parameters** The allowed JSON fields are the same as those for the above `POST /alta/alta_instances` endpoint for creating a new instance. Note that the settings provided to this endpoint will completely overwrite the previous settings (e.g., any settings that are absent from this request but were provided previously will be removed). **URL parameters** **instanceID** — The instanceID for the Alta instance to modify

Example Request:

PUT http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/settings

JSON body:

```
{
  "primary": "udp://239.120.100.101:5000",
  "source": "rtp://224.0.101.140:5000",
  "output": "239.120.200.101:5000"
}
```

Example Response:

```
{
  "settings": {
    "primary": "udp://239.120.100.101:5000",
    "source": "rtp://224.0.101.140:5000",
    "output": "239.120.200.101:5000"
  }
}
```


Status

Resource	Description
GET /alta/alta_instances/<**instanceID**>/status	Action Retrieves status of an Alta instance, including a list of log files that can be downloaded from the logging endpoints URL parameters **instanceID** — The instanceID for the Alta instance to retrieve

Example Request:

GET http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/status

Example Response:


```
{
  "health_indicators": {
    "primary": {
      "captions_present": true,
      "empty_streams": "10C0",
      "input_present": true,
      "pcr_present": true,
      "pmt_present": true,
      "transport_rate": 20000000.0,
      "transport_rate_locked": true,
```

```
      "transport_rate_locked_since": "2016-Jun-24 18:09:10"
    },
    "source": {
      "captions_present": true,
      "input_present": true,
      "pcr_present": true,
      "pmt_present": true,
      "transport_rate": 19961759.0,
      "transport_rate_locked": true,
      "transport_rate_locked_since": "2016-Jun-24 18:10:11"
    }
  },
  "log_list": [
    "20180830_14.log",
    "20180828_13.log",
    "20180823_12.log",
    "20180821_11.log",
    "20180820_10.log",
    "20180820_9.log",
    "20180820_8.log",
    "20180818_7.log",
    "20180815_6.log",
    "20180815_5.log",
    "20180815_4.log",
    "20180815_3.log",
    "20180815_2.log",
    "20180815_1.log",
    "20180815_0.log"
  ],
  "state": "RUNNING",
  "status": [
    "18:09:10:  Initiating primary input: RTP listen on port 5000\n",
    "18:09:10:  Initiating source input: RTP listen on port 60000\n",
    "18:09:10:  Receive: reset RTP counter to 13047\n",
    "18:09:10:  Target pace in bytes per millisecond set to 0\n",
    "18:09:10:  Initiating video send on port 8034\n",
    "18:09:10:  Added destination 10.0.0.8:5000 to session\n",
```

```
    "18:09:10:  Target pace in bytes per millisecond set to 2500\n",
    "18:09:11:  Output GOP ready, preloading\n",
    "18:09:11:  Output GOP ready, preloading\n",
    "18:09:11:  Output GOP ready, preloading\n",
    "18:09:11:  Output GOP ready, preloading\n",
    "18:09:12:  Output GOP ready, preloading\n",
    "18:09:13:  Receive: RTP received: 6956, lost: 0, recovered by FEC: 0\n",
    "18:09:19:  Receive: RTP received: 16956, lost: 0, recovered by FEC: 0\n",
  ]
}
```


Errors

Resource	Description
GET /alta/alta_instances/<**instanceID**>/errors	Action Retrieves errors status/counts of an Alta instance URL parameters **instanceID** — The instanceID for the Alta instance to retrieve

Example Request:

GET http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/errors

Example Response:


```
{ "ts_sync_loss": { "name": "ts_sync_loss", "present": "false", "count": "0", "last_detected": "",
"last_detected_pcr_time": "-1", "last_good_state_pcr_time": "-1", "message": "" },
"sync_byte_error": { "name": "sync_byte_error", "present": "false", "count": "0", "last_detected":
"", "last_detected_pcr_time": "-1", "last_good_state_pcr_time": "-1", "message": "" },
"PAT_error": { "name": "PAT_error", "present": "false", "count": "0", "last_detected": "",
"last_detected_pcr_time": "-1", "last_good_state_pcr_time": "72933.21240014814", "message": "" },
"continuity_count_error": { "name": "continuity_count_error", "present": "false", "count": "0",
"last_detected": "", "last_detected_pcr_time": "-1", "last_good_state_pcr_time": "-1", "message":
"" }, "PMT_error": { "name": "PMT_error", "present": "false", "count": "533", "last_detected":
"09-22-2020 13:20:59", "last_detected_pcr_time": "72845.70340699999", "last_good_state_pcr_time":
"72933.06240422222", "message": "PMT has not been present for 0.510041 seconds" }, "PID_error": {
"name": "PID_error", "present": "false", "count": "0", "last_detected": "",
"last_detected_pcr_time": "-1", "last_good_state_pcr_time": "-1", "message": "" },
"transport_error": { "name": "transport_error", "present": "false", "count": "0", "last_detected":
```

```
"", "last_detected_pcr_time": "-1", "last_good_state_pcr_time": "-1", "message": "" } , "RTP
packets lost": { "name": "RTP packets lost", "present": "false", "count": "0", "last_detected":
"", "message": "" }, "New SSRC count": { "name": "New SSRC count", "present": "false", "count":
"2", "last_detected": "09-21-2020 18:06:27", "message": "Current SSRC: 590080 (0x90100)" }, "RTP
packets out of order": { "name": "RTP packets out of order", "present": "false", "count": "0",
"last_detected": "", "message": "" }, "RTP Sequence Number Rebase": { "name": "RTP Sequence Number
Rebase", "present": "false", "count": "1", "last_detected": "09-21-2020 18:06:27", "message":
"Reset RTP counter to 24670" }, "RTP Parse Errors": { "name": "RTP Parse Errors", "present":
"false", "count": "0", "last_detected": "", "message": "" }, "Output underflow gaps": { "name":
"Output underflow gaps", "present": "false", "count": "0", "last_detected": "", "message": "" },
"Output jitter gaps": { "name": "Output jitter gaps", "present": "false", "count": "174",
"last_detected": "09-22-2020 13:21:45", "message": "Output IPAT of 89.8 ms" } }
```


Logs

Resource	Description
GET /alta/alta_instances/<**instanceID**>/logs	**Action** Retrieves the current or most recent log file associated with an Alta instance. The current log for a running instance will contain events logged only since midnight local time on the current day. For older events, see the status endpoint for a list of available files and specify a specific filename in the URI **URL parameters** **instanceID** — The instanceID for the Alta channel that generated the log
GET /alta/alta_instances/<**instanceID**>/logs/<**logfilename**>	**Action** Retrieves a specific log file related to this Alta channel. Use the Status endpoint first to retrieve a list of log filenames that exist for this Alta channel. **URL parameters** **instanceID** — The instanceID for the Alta channel that generated the logs **logfilename** — The filename for a specific log file. Typically this looks like YYYYMMDD_xyz.log, for a list of available files use the

Resource	Description
	Status endpoint for this Alta channel first.

Example Request:

GET http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/logs/20180820_9.log

Example Response iCap Alta Encoder:

```
20:31:32: <info> Startup: EEG iCap Alta Encoder Version 2.47

20:31:32: <info> Using Plain UDP output

20:31:33: <info> Initiating primary input: RTP listen on multicast addr 224.1.1.1 port 5000, on
interface 10.0.0.7

20:31:33: <debug> Granted socket receive buffer size of 12582912

20:31:33: <info> Target pace in bytes per millisecond set to 0. Estimate of bytes per GOP chunk
set to 0

20:31:33: <info> Target low water buffer mark set to 1000ms, 0 bytes at current rate

20:31:33: <info> Initiating video send on port 8210

20:31:33: <info> Target low water buffer mark set to 800ms, 0 bytes at current rate

20:31:33: <info> Added plain UDP destination 224.1.1.3:5000

20:31:33: <info> Added destination 224.1.1.3:5000 to session

20:31:33: <trace> iCap: Created RTP session on port 6685

20:31:33: <warning> Not starting iCap networking, no company credential provided

20:31:33: <trace> iCap Driver: Starting the event loop thread

20:31:41: <debug> Receive: RTP packet out-of-order or in startup probation, dropped

20:31:41: <info> Receive: reset RTP counter to 1

20:31:41: <debug> Receive: using RTP seq number 1

20:31:41: <info> Receive: RTP received: 3, lost: 0, recovered by FEC: 0

20:31:41: <info> Target pace in bytes per millisecond set to 1875.01. Estimate of bytes per GOP
chunk set to 937507

20:31:41: <warning> Could not promote output pace thread to real-time scheduling priority -
suggest running as root

20:31:44: <debug> Starting output stream with initial load of 2552.66 KB and chunk avg of
1.01826e+06

20:31:44: <debug> Destination write: send more bytes than slots, count of 1

20:31:44: <debug> Caption data: 1425 142D 1470 4545 4720 416C 7461 2054 6573 7420 4361 7074 696F
6E73 1425 142D 1470 4D50 4547 2D54 5320 416C 7461 1425 142D 1470 5254 5020 496E 7075 742F 4F75
7470 7574 ( % - pEEG Alta Test Captions % - pMPEG-TS Alta % - pRTP Input/Output)
```

Delete Instance

Resource	Description
DELETE /alta/alta_instances/<**instanceID**>	Action Deletes a previously created Alta instance. Note that this operation is only supported for instances that are off. **WARNING**: Use this endpoint with care. This will completely scrub the record of an Alta instance and cannot be undone. URL parameters **instanceID** — The instanceID for the Alta instance to delete

Example Request:

```
DELETE
http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ
```

Example Response:

```
{}
```

Network GPI's, Presets + Cue Point Triggering

Resource	Description
GET /system/gpi/settings	Action Retrieves the current GPI settings including 'protocol', 'ip' and 'port'.
GET /system/gpi/triggers/<**protocol**>	Action Retrieves a list of the pre-saved GPI triggers for the given protocol. URL parameters

Resource	Description
	protocol — Network GPI protocol (i.e. 'JL Cooper eBox')
POST /system/gpi/triggers/<**protocol**>/<**gpi_num**>/trigger	Action Triggers a pre-saved GPI trigger preset with property 'gpi_num' for the given 'protocol'. URL parameters **protocol** — Network GPI protocol (i.e. 'JL Cooper eBox') **gpi_num** — GPI number of the preset to trigger
POST /alta/instances/<**instanceID**>/cue-point	Action Triggers a SCTE35 cue-in or cue-out point for the specified instanceID JSON parameters **action** — 'SCTE_35_SPLICE_INSERT' or 'SCTE_35_RETURN_TO_NETWORK' **spliceEventId** — SCTE35 event ID (must be integer and > 0) **duration** — Break duration in 90kHz clock cycles (i.e. 5400000 = 60secs)
POST /alta/instances/<**instanceID**>/trigger_preset	Action Triggers a GPI preset for the specified instanceID JSON parameters **type** — Trigger type ('lexi', 'scte104' **function** — Lexi functions: 'speaker_change', 'turn_on', 'turn_off'; SCTE104 functions: 'start_normal', 'start_immediate', 'end_normal', 'end_immediate', 'cancel'

Resource	Description			
	parameters	Lexi speaker change: 'delay_comp'	Lexi on/off: 'instance_id' (Lexi instance ID)	SCTE104 start normal: 'program_id', 'break_duratio (1/10sec), 'preroll_time' (1/1000sec)

Example Request:

GET http://username:password@localhost/system/gpi/triggers/JL Cooper eBox

Example Response iCap Alta Encoder:


```
[
  {
    "alta_channel": "EEG_BR_P0BX1U3EL77MKLGJ",
    "gpi_num": 1,
    "gpi_pin": 2,
    "preset": {
      "function": "start_normal",
      "parameters": {
        "break_duration": "0",
        "preroll_time": "4000",
        "program_id": "4"
      },
      "type": "scte104"
    }
  },
  {
    "alta_channel": "EEG_BR_P0BX1U3EL77MKLGJ",
    "gpi_num": 2,
    "gpi_pin": 3,
    "preset": {
      "function": "speaker_change",
      "parameters": {
        "delay_comp": "0.8"
      },
```

```
      "type": "lexi"
    }
  },
  {
    "alta_channel": "EEG_BR_WWO0T2H4Q2B9GSBC",
    "gpi_num": 3,
    "gpi_pin": 4,
    "preset": {
      "function": "start_immediate",
      "parameters": {
        "break_duration": "0",
        "program_id": "2"
      },
      "type": "scte104"
    }
  },
  {
    "alta_channel": "EEG_BR_WWO0T2H4Q2B9GSBC",
    "gpi_num": 4,
    "gpi_pin": 5,
    "preset": {
      "function": "end_normal",
      "parameters": {
        "preroll_time": "5",
        "program_id": "4"
      },
      "type": "scte104"
    }
  },
  {
    "alta_channel": "EEG_BR_WWO0T2H4Q2B9GSBC",
    "gpi_num": 5,
    "gpi_pin": 6,
    "preset": {
      "function": "end_immediate",
      "parameters": {
        "program_id": "6"
```

```
      },
      "type": "scte104"
    }
  },
  {
    "alta_channel": "EEG_BR_WWO0T2H4Q2B9GSBC",
    "gpi_num": 6,
    "gpi_pin": 7,
    "preset": {
      "function": "cancel",
      "parameters": {
        "program_id": "7"
      },
      "type": "scte104"
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 7,
    "gpi_pin": 8,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 8,
    "gpi_pin": 9,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
```

```
  },
  {
    "alta_channel": "",
    "gpi_num": 9,
    "gpi_pin": 10,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 10,
    "gpi_pin": 11,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 11,
    "gpi_pin": 12,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 12,
```

```
    "gpi_pin": 13,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 13,
    "gpi_pin": 14,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 14,
    "gpi_pin": 15,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 15,
    "gpi_pin": 16,
    "preset": {
      "function": "",
      "parameters": {
```

```
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 16,
    "gpi_pin": 17,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 17,
    "gpi_pin": 18,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 18,
    "gpi_pin": 19,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
```

```
  {
    "alta_channel": "",
    "gpi_num": 19,
    "gpi_pin": 20,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 20,
    "gpi_pin": 21,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 21,
    "gpi_pin": 22,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 22,
    "gpi_pin": 23,
```

```
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 23,
    "gpi_pin": 24,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  },
  {
    "alta_channel": "",
    "gpi_num": 24,
    "gpi_pin": 25,
    "preset": {
      "function": "",
      "parameters": {
      },
      "type": ""
    }
  }
]
```


Example Request:

POST http://username:password@localhost/system/gpi/triggers/JL Cooper eBox/3/trigger

Example Response iCap Alta Encoder:


```
{
```

```
  "msg": "GPI 3 preset executed successfully",
  "success": true
}
```

Utils/Helpers

Resource	Description
POST /alta/alta_instances/get_id/<**name**>	Action Gets the bridgeID of an instance from the instance's name property. Returns a 404 error if name is not found. URL parameters **name** — The name of the Alta instance

Example Request:

POST http://username:password@localhost/alta/alta_instances/get_id/AltaInstance1

Example Response:

```
{ "instanceID": "EEG_BR_3G46Z6VLSV8B01JZ" }
```

Deprecated Endpoints

Resource	Description
POST /alta/alta_instances/<**instanceID**>/terminate	Action Legacy endpoint that will both turn off and delete an Alta instance endpoint. URL parameters **instanceID** — The instanceID for the Alta instance to terminate

Example Request:

POST http://username:password@localhost/alta/alta_instances/EEG_BR_3G46Z6VLSV8B01JZ/terminate

Example Response:

```
{}
```

Instance State Conditions

State	Description
RUNNING	The instance is on
STOPPED with code: n	The instance has encountered an unrecoverable error
TERMINATED	The instance is off

HTTP Error Codes

Error Code	Possible Cause
400 Bad Request	Missing or malformed JSON
404 Not Found	Referenced instanceID no longer exists
422 Unprocessable Entity	The request is well-formed but semantically incorrect (e.g., trying to turn off an instance that is already off)
500 Internal Server Error	Server logic issues
503 Service Unavailable	Server connectivity issues

Last Updated: 1/26/2023