# Channel Devices Server

Microservice to manage devices of the channels.

**Note**: This Service uses a Json file named “config.json” as its configuration file.

## Technologies

* Back: NodeJS
* Database: MongoDB
* Messaging: Kafka

## Basic Response Model

* All responses are JSON objects
* All responses must have at least these two parameters
  + status: “success” or “failed”
  + message: String
* All responses must include response code (200 for success, non 200 for failures)

## Part 1: Authorization

Use Authorization Client library to provide access to Authorization server.

In this document, wherever mentioned “Authorization” refers to Authorization client library.

## Part 2: Channel Devices APIs

/ownership/bind

Bind a device to a channel

### Parameters

* user\_id: String
* bid: String
* chid: String
* dev\_id: String

### Returns

* <Basic Response>

### Steps

1. Call Authorization.authorize(“channel\_devices/ownership/bind”, {user\_id, bid, chid, dev\_id}) to check authorization
2. Create a new record in ownership table with bid, chid and dev\_id
3. Publish event on Kafka
   1. Topic: “channel\_devices\_device\_bound”
   2. Content:
      1. user\_id
      2. bid
      3. chid
      4. dev\_id
      5. created\_at
4. Return Success

/ownership/bindAll

Bind a group of devices to a channel

### Parameters

* user\_id: String
* bid: String
* chid: String
* dev\_ids: String Array

### Returns

* <Basic Response>

### Steps

1. Call Authorization.authorize(“channel\_ devices/ownership/bind”, {user\_id, bid, chid, dev\_ids}) to check authorization
2. For each dev\_id in dev\_ids list create a new record in ownership table with dev\_id, bid and chid
3. Publish event on Kafka
   1. Topic: “channel\_devices\_device\_group\_bound”
   2. Content:
      1. user\_id
      2. bid
      3. chid
      4. dev\_ids
      5. created\_at
4. Return Success

/ownership/unbind

Unbind a device from a channel

### Parameters

* user\_id: String
* bid: String
* chid: String
* dev\_id: String

### Returns

* <Basic Response>

### Steps

1. Call Authorization.authorize(“channel\_devices/ownership/unbind”, {user\_id, bid, chid, dev\_id}) to check authorization
2. Delete a record in ownership table with bid, chid and dev\_id
3. Publish event on Kafka
   1. Topic: “channel\_devices\_device\_unbound”
   2. Content:
      1. user\_id
      2. bid
      3. chid
      4. dev\_id
      5. created\_at
4. Return Success

/ownership/unbindAll

Unbind a group of devices from a channel

### Parameters

* user\_id: String
* bid: String
* chid: String
* dev\_ids: String Array

### Returns

* <Basic Response>

### Steps

1. Call Authorization.authorize(“channel\_devices/ownership/unbind”, {user\_id, bid, chid, dev\_ids}) to check authorization
2. For each dev\_id in dev\_ids list remove a record in ownership table with dev\_id, bid and chid
3. Publish event on Kafka
   1. Topic: “channel\_devices\_device\_group\_unbound”
   2. Content:
      1. user\_id
      2. bid
      3. chid
      4. dev\_ids
      5. created\_at
4. Return Success

/ownership/get

Get bid and chid of a device

### Parameters

* user\_id: String
* dev\_id: String

### Returns

* <Basic Response>

### Steps

1. Call Authorization.authorize(“channel\_devices/ownership/get”, {user\_id, dev\_id}) to check authorization
2. Get a record with same dev\_id from ownership table
3. Return record or Error if not found

/ownership/list

List devices of a channel.

### Parameters

* user\_id: String
* bid: String
* chid: String

### Returns

* <Basic Response>

### Steps

1. Call Authorization.authorize(“channel\_devices/ownership/list”, {user\_id, bid, chid}) to check authorization
2. Get all records with same bid and chid from ownership table
3. Return records