# Function name - fetchAllowedPlanConfig

## Usage and invocation

This function will be invoked to get the list of plans and addons combination for a particular subscription. Only the plans and addons present in the list will be shown to the customer while changing the subscription. You can write your business specific logic in this function to restrict the plan list and addon list for your customers.

## Writing the function

- The function should be written in javascript. This will be executed in nodejs version 6 environment
- We have some default resources that will be passed to this function. You can use this resources to write your business specific use case. These resources format will be similar to the resource output generated from node js client library.
- You need to return an javascript object that matches the JSON schema mentioned below
- You can also use our **chargebee node js client** library(that is already configured with apikey and secret key) for getting additional resources
- Arguments passed to this function are
    - An object containing all the default resources. For this function, these are the resources which we will be pass
        - subscription
            - associated subscription object
        - customer
            - associated customer object
        - plan
            - associated plan
        - addons
            - array of addons present in the subscription
    - Callback function in which you need to pass the output or error message.

    - In case of successful execution, pass your output like this
        - callback(null, output)
    - In case of any errors, pass your output like this
        - callback(error, null)
    - All errors that has occurred can be seen in our app. If any error occurs, we will not show change subscription
- Logger object to log debugging information
    - This will be used only while testing out the function in our testing console (function should be in draft mode)

## Sample code

```
var chargebee = require("chargebee")
exports.fetchAllowedPlanConfig = function({subscription, customer, plan}, callback, logger) {
  chargebee.plan.list().request(function(error,result){
    if(error){
      callback(error, null);
    } else {
      let selectedPlans = result.list.map((p) => p.plan).filter(p => p.charge_model == "per_unit");
      let seats = subscription.cf_seats;
      let output = {};
      output.items = [];
      selectedPlans.forEach(p => {
        let entry = {
          plan_id: p.id
        }
        if(p.charge_model == "per_unit" && seats) {
          entry["meta_data"] = {
            quantity_meta: {
              type: "range",
              min: seats,
```

```
              max: 100,
              step: 1
            }
          }
        }
        output.items.push(entry);
      });
      callback(null, output);
    }
  });
}
```

## Sample output

```
{
        "items": [{
                "plan_id": "plan3",
                "meta_data": {
                        "quantity_meta": {
                                "type": "range",
                                "min": 5,
                                "max": 100,
                                "step": 1
                        }
                }
        }, {
                "plan_id": "basic",
                "meta_data": {
                        "quantity_meta": {
                                "type": "range",
```

```
                        "min": 5,
                        "max": 100,
                        "step": 1
                }
        }
}, {
        "plan_id": "no-trial",
        "meta_data": {
                "quantity_meta": {
                        "type": "range",
                        "min": 5,
                        "max": 100,
                        "step": 1
                }
        }
}, {
        "plan_id": "plan1",
        "meta_data": {
                "quantity_meta": {
                        "type": "range",
                        "min": 5,
                        "max": 100,
                        "step": 1
                }
        }
}, {
        "plan_id": "professional",
        "meta_data": {
                "quantity_meta": {
                        "type": "range",
```

```
                                "min": 5,
                                "max": 100,
                                "step": 1
                        }
                }
        }, {
                "plan_id": "plan2",
                "meta_data": {
                        "quantity_meta": {
                                "type": "range",
                                "min": 5,
                                "max": 100,
                                "step": 1
                        }
                }
        }]
}
```

## Expected JSON schema for this functions output

You can validate your output with the below schema in any one of online json schema validators

```
{
  "type": "object",
  "$schema": "http://json-schema.org/draft-04/schema#",
  "definitions": {
    "range_quantity_meta": {
      "type": "object",
      "additionalProperties": false,
      "properties": {
```

```
        "type": {
          "type": "string",
          "enum": ["range"]
        },
        "min": {
          "type": "number"
        },
        "max": {
          "type": "number"
        },
        "step": {
          "type": "number"
        }
      },
      "required": ["type", "min", "max"]
    },
    "addon": {
      "additionalProperties": false,
      "type": "object",
      "required": ["id"],
      "properties": {
        "id": {
          "type": "string",
          "examples": [
            "sms_credits"
          ]
        },
        "meta_data": {
          "additionalProperties": false,
          "type": "object",
```

```
        "properties": {
          "quantity_meta": {
            "oneOf": [{
              "$ref": "#/definitions/range_quantity_meta"
            }, {
              "$ref": "#/definitions/fixed_quantity_meta"
            }]
          }
        }
      }
    },
    "fixed_quantity_meta": {
      "type": "object",
      "additionalProperties": false,
      "properties": {
        "type": {
          "type": "string",
          "enum": ["fixed"]
        },
        "values": {
          "type": "array",
          "items": {
            "type": "number"
          }
        }
      },
      "required": ["type", "values"]
    }
  },
```

```
  "properties": {
    "items": {
      "type": "array",
      "items": {
        "$id": "/properties/product-item",
        "type": "object",
        "additionalProperties": false,
        "required": ["plan_id"],
        "properties": {
          "plan_id": {
            "type": "string",
            "examples": [
              "professional"
            ]
          },
          "allowed_addons": {
            "type": "array",
            "items": {
              "$ref": "#/definitions/addon"
            }
          },
          "meta_data": {
            "type": "object",
            "additionalProperties": false,
            "properties": {
              "quantity_meta": {
                "oneOf": [{
                  "$ref": "#/definitions/range_quantity_meta"
                }, {
                  "$ref": "#/definitions/fixed_quantity_meta"
```

```
                }]
              }
            }
          },
          "addons": {
            "type": "array",
            "items": {
              "$ref": "#/definitions/addon"
            }
          }
        }
      }
    },
    "allowed_addons": {
      "type": "array",
      "items": {
        "$ref": "#/definitions/addon"
      }
    },
    "use_existing_addons": {
      "type": "boolean",
      "default": false,
      "examples": [
        true
      ]
    }
  },
  "additionalProperties": false
}
```

***use_existing_addons***

- If it is true, addons from the current subscription will be carried over to the changed plan

***allowed_addons***

- List of addons that can be shown to the customer. Customer can add only addons from this list
- If allowed_addons property is attached to the individual item(plan), it will be used rather than the one attached to root object

***addons***

- List of addons that needs to be auto-attached when a customer moves from one plan to another plan.

***meta_data***

- You can control quantity drop down that is shown with this data. Please check this docs for more details