VCC Live®
INTEGRATION

Developers

DATABASE API

WEBHOOK

SCRIPT SDK

CLICK 2 CALL

WEBSITE EMBEDDING

3RD PARTY INTEGRATIONS

Database API

Leverage VCC Live from any ERP and CRM system. Add new records, get statistics and much more.

Introduction

By using VCC Live Database API, you can send requests towards VCC to get or set any specific data in VCC Live database. In this way you can gather project settings or user information and also add, remove or modify records to keep your contact list synchronized with your custom database. There are several other possibilities which are described in the following chapters.

From now on we refer to VCC Live Database API as DB API. This documentation describes DB API v2. Any previous API version is obsolete.

The API

The Database API is based on REST and JSON, which are currently the most popular API technologies. By using these technologies any system can easily communicate with VCC Live's database through HTTP GET, PUT, POST and DELETE messages. Request and response body are both serialized using JSON.

For more information on REST and JSON:

- http://en.wikipedia.org/wiki/Representational_state_transfer
- http://en.wikipedia.org/wiki/JSON

The API location

Database API resources consist of the following parts:

- Protocol definition: https://
- Domain: [customer].asp.virtual-call-center.eu
- Parameters: /v2/[resource]/

So a complete resource specification looks like this: https://[customer].asp.virtual-call-center.eu/v2/[resource]/

All variable names are wrapped in brackets, and should be replaced by the appropriate value. All of our customers have their own customer id, so if you are already a customer you will have one. In this document we use customer id "mycc" as an example.

Let's see a sample resource:

- https://mycc.asp.virtual-call-center.eu/v2/projects/123

This resource represents information about mycc company's project 123.

A description of the available resources is detailed in the following chapters.

Authentication and Security

A lot of sensitive information can be accessed using Database API, so it is essential to configure it properly.

- The Database API can only be reached from pre-defined IP addresses and IP ranges.
- The authentication process is based on HTTP Basic Authentication.

Warning: Please set the IP address definitions as restrictive as you can, and keep your password safe at all times.

Setting up the API

Before you can access the API, you must register your IP address(es) and generate a password. To prepare API for use, please follow these steps:

1. Open the VCC Live client software and log in as an admin user (e.g. supervisor)
2. Select VCC Live / Tools / Call Center settings menu
3. Select the Database API tab
4. Type in the IP address or IP range or domain from where you want to access the API
5. Create a password by pressing the "New token" button
6. Press the "Save" button to save modifications
7. You can use the genarated URL example, which should look similar to the following: https://mycc:157f...de62c@mycc.asp.virtual-call-center.eu/v2/projects

For developing and testing you should use your desktop's public IP address.

Authentication process

The Database API uses HTTP Basic Authentication. Both the username and password should be set in every HTTP request. Credential information can be added to either the URL or the request header.

Here is an example using credentials in URL:

- https://[customer]:[password]@[customer].asp.virtual-call-center.eu/v2/[resource]

For more information on HTTP Basic Authentication:

- http://en.wikipedia.org/wiki/Basic_access_authentication

Testing API

You can easily check whether the API works correctly using any web browser. Before testing API settings, please check if your public IP address has been already set. To test the API:

- Either copy and paste your URL example (specified above) OR type the following resource in the browser's URL field, then press Enter: https://: @.asp.virtual-call-center.eu/v2/projects
- Your web browser will send an HTTP GET request to Database API
- You should then automatically receive a list of VCC projects in JSON format

If there are any errors, please check HTTP Response Codes.

Request and Response

Each Database API request has a unique resource (or URL) and one of four HTTP methods, which is displayed in every 'Request' section. There are four supported HTTP methods: GET, PUT, POST, DELETE. POST and PUT requests should contain a JSON-encoded HTTP body.

For each Resource you can find the following summary table:

Request	
Method	GET | PUT | POST | DELETE
Resource	http://[customer].asp.virtual-call-center.eu/v2/resource
Options	param1, param2
Body	JSON object
Response	
Body	JSON object

The complete URL contains a resource and relevant optional parameters, e.g.:
http://[customer].asp.virtual-call-center.eu/v2/resource?param1=x&param2=y

All resource parameters and options are described in the relevant section.

Any successful Database API request should result in HTTP response code 200 (OK). Any other response code may indicate a communication or standard HTTP problem. For an overview of all error codes (used by VCC Live) please check HTTP Response Codes.

If there is neither communication nor HTTP problem, the Database API returns with a JSON encoded response representing an associative array, which has two keys: errors and response. If the errors key doesn't have any value, then you should use the response key's value in your code as a successful response.

Below is an example of a successful response:

```
HTTP/1.1 200 OK
Server nginx is not blacklisted
Server: nginx
Date: Fri, 06 Feb 2015 11:36:41 GMT
Content-Type: text/plain; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Allow: GET
Etag: "19dd91e51d3e63d4d8010bc5fbff26cf"
X-Resource: ProjectsResource

{
    "response": [
        {
            "projectid": 2,
            "name": "Sales - Budapest",
            "status": "active",
            "folderid": 1,
            "container": false,
            "folder": "Sales"
        }
    ],
    "errors": []
}
```

HTTP Response Codes

Code	Description	Troubleshooting
200	OK	
401	Authorization Required	Check credentials
403	Forbidden	
404	Not found	
405	Method not Allowed	
417	Expectation Failed	Check resource
500	Internal Error	
600	Partial Error	

Examples

The following three examples, which are written in PHP language, demonstrate how simple the Database API is to use. Please use PHP 5.3.0 or a later version to run the example codes. Before you start looking at the codes you may need to refresh your knowledge of PHP, JSON and HTTP.

Hello World Example

Let's see how to list, for example, users. Save the following code as 'vcc-db-api.php' onto your web server, then open it, change the credential information as required (see: Authentication and Security, and finally save it.

```
<?php
error_reporting(0);

// settings
$customer = 'CUSTOMER';
$password = 'PASSWORD';

// build Database API URL
$url = sprintf('https://%s:%s@%s.asp.virtual-call-center.eu', $customer, $password, $custome
r);

// build resource
$resource = $url . '/v2/users';

// send HTTP GET request to API and obtain the result
$http_response = file_get_contents($resource);

if ($http_response === false) {

    // there is an error
    $last_error = error_get_last();
    echo "Connection error:n" . $last_error['message'] . "nn";
    exit(1);
}

// $http_response_header global var contains HTTP response code
list(, $http_response_code) = explode(' ', $http_response_header[0]);

// check if HTTP response is OK
if ($http_response_code != 200) {

    // there is an error
    echo 'Database API error code: ' . $http_response_code . "nn";
    exit(1);
}

echo "Output in JSON format:n" . $http_response . "nn";

// convert json structure into php array
$vcc_users = json_decode($http_response, true);

// print users array
echo "Output as PHP array:n";
var_export($vcc_users);
echo "nn";
```

Type the URL into any browser (e.g.: http://your.url/vcc-db-api.php), press Enter, and check the output on your screen, which should look similar to the following:

```
{
    "response": [
        {
            "status": "active",
            "name": "Admin",
            "username": "admin",
            "extension": "4",
            "userid": 4,
            "teams_name": "default_team",
            "group_name": "admin"
        },
        {
            "status": "active",
            "name": "Supervisor",
            "username": "supervisor",
            "extension": "5",
            "userid": 5,
            "teams_name": "default_team",
            "group_name": "supervisor"
        },
        {
            "status": "active",
            "name": "mozmill operator",
            "username": "mozmill_operator",
            "extension": "104",
            "userid": 9,
            "teams_name": "default_team",
            "group_name": "operator"
        }
    ],
    "errors": []
}
```

Using http://jsonlint.com/ you can convert JSON texts into a readable format, and also validate them. The example code should hopefully be easy to understand, but one section may need some explanation. When you call file_get_contents() function it returns with the HTTP response body and also sets the $http_response_header global variable. It should be checked whether the HTTP response code is 200 or not. If it is not, please check HTTP Response Codes for possible errors.

Note: Do not use this code in production, as the file_get_contents() function is not able to send HTTP POST, PUT and DELETE requests. The above code is only for demonstration.

Advanced Example

The following example uses PHP cURL module, and thus it can send all four supported HTTP request methods (GET, PUT, POST and DELETE). The example consists of a library and the example code.

```php
<?php
useVCCAPIv2RestClient;
require_once ('../RestClient.php');

$customer = 'bdevel';
$password = 'PASSWORD';
$apiClient = new RestClient($customer, $password);

if (!$apiClient->get('/v2/projects')) {
    echo "Error(1):n";
    echo 'Status code: ' . $apiClient->response_code . "n";
    echo $apiClient->error . "nn";
    exit(1);
}

$projects = $apiClient->result['response'];
var_export($projects);
$selected_projectid = $projects[0]['projectid'];

if (!$apiClient->get('/v2/projects/' . $selected_projectid)) {
    echo 'Error(2): ' . $apiClient->response_code . "n";
    exit(2);
}

$project = $apiClient->result;
var_export($project);
```

```php
<?php
namespaceVCCAPIv2;
useInvalidArgumentException;
/**
 * Rest API client
 */
class RestClient

{
    const DOMAIN_PATTERN = 'https://%s.asp.virtual-call-center.eu';
    public $http_info;

    public $result;

    public $response_code;

    public $connect_timeout = 3;

    public $exec_timeout = 60;

    public $error;

    public
    function __construct($customer, $password, $options = array())
    {
        $this->url = sprintf(self::DOMAIN_PATTERN, $customer);
        $this->username = $customer;
        $this->password = $password;
        $this->options = $options;
    }

    public

    function get($resource)
    {
        return $this->call('GET', $resource);
    }

    public

    function post($resource, $data)
    {
        return $this->call('POST', $resource, $data);
    }

    public

    function put($resource, $data)
    {
        return $this->call('PUT', $resource, $data);
    }

    public

    function delete($resource)
    {
        return $this->call('DELETE', $resource);
    }

    protected
    function call($method, $resource, $data = null)
    {
        unset($this->http_info);
        unset($this->result);
        unset($this->response_code);
        $ch = curl_init();
        curl_setopt($ch, CURLOPT_URL, $this->url . $resource);
        curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
        if (isset($this->username) and isset($this->password)) {
            curl_setopt($ch, CURLOPT_USERPWD, $this->username . ':' . $this->password);
            curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
        }

        if (isset($this->options['accept_invalid_ssl']) and $this->options['accept_invalid_s
sl']) {
```

```
            curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
            curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false);
        }

        curl_setopt($ch, CURLOPT_CONNECTTIMEOUT, $this->connect_timeout);
        curl_setopt($ch, CURLOPT_TIMEOUT, $this->exec_timeout);
        if ($data) {
            curl_setopt($ch, CURLOPT_HTTPHEADER, array(
                'Content-type: application/json'
            ));
            $data = json_encode($data);
        }

        switch ($method) {
        case 'GET':
            break;

        case 'PUT':
            curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'PUT');
            curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
            curl_setopt($ch, CURLOPT_HTTPHEADER, array(
                'X-HTTP-Method-Override: PUT'
            ));
            break;

        case 'POST':
            curl_setopt($ch, CURLOPT_POST, true);
            curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
            curl_setopt($ch, CURLOPT_HTTPHEADER, array(
                'X-HTTP-Method-Override: POST'
            ));
            break;

        case 'DELETE':
            curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'DELETE');
            curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
            curl_setopt($ch, CURLOPT_HTTPHEADER, array(
                'X-HTTP-Method-Override: DELETE'
            ));
            break;

        case 'OPTIONS':
            curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'OPTIONS');
            curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
            curl_setopt($ch, CURLOPT_HTTPHEADER, array(
                'X-HTTP-Method-Override: OPTIONS'
            ));
            break;

        default:
            throw new InvalidArgumentException('HTTP method invalid: ' . $method);
        }
        set_time_limit($this->exec_timeout + 1);
        $json_result = curl_exec($ch);
        $this->result = json_decode($json_result, true);
        $this->http_info = curl_getinfo($ch);
        $this->error = curl_error($ch);
        $this->response_code = $this->http_info['http_code'];
        curl_close($ch);
        if ($this->response_code == '200') {
            return true;
        }

        return false;
    }
}
```

Web Callback Example

This section describes how to implement web callback technology into your website using the Database API.

Web callbacks allow any website visitor to request a callback through your website by entering their phone number and some relevant information in a form. To implement this feature, you need to perform the following steps:

1. Create a new project in VCC Live or use an existing one, and make a note of the project ID. Add your required fields below the phone number and name fields, which are automatically created in every new project. Enable the Database API described above.
2. You need to create a simple information request form on your own website, which visitors can fill in and send to your server by pressing the Submit button.
3. You also need a server side script, which receives form data and sends the given phone number and other relevant information to VCC Live's database by calling the Database API
4. The given number is stored on your VCC Live project database and an agent can manually (or by using our predictive dialer) dial the number immediately.

Note: To increase form filling conversion rates, ask for as little information as possible. By doing this you can reduce the time it takes for visitors to complete the form, and thus more visitors will be willing to fill it in.

Warning: You should also bear in mind that many records can be inserted in a short time by harmful spam robots, so ensure that the form is filled by a genuine visitor by using, for example, captcha technology. You should also check the phone number's prefix to avoid calling expensive private or international numbers.

For more information on Web Callback:

- http://en.wikipedia.org/wiki/Web_callback

Below is the HTML form for retrieving a telephone number:

Add record

```
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="Content-tpye" content="text/html; charset=utf-8" />
      <title>Teszt form</title>
   </head>
   <body>
      Request call back
      <form method="post" action="webcallback.php">
         Name: <input type="text" name="fullname" value="" /><br />
         Your number*: <input type="text" name="phone" value="" /> (e.g.: +4211234567)<br /
>
         <input type="submit" value="Submit" />
      </form>
   </body>
</html>
```

Below is a server-side PHP script, which receives and forwards form data to VCC Live's Database API. This example code uses RestClient.php, which can be found in the 'Advanced Example' section above.

```
<?php
useVCCAPIv2RestClient;

if (empty($_POST['fullname']) or empty($_POST['phone'])) {
    echo 'All fields required';
}

require_once ('../../RestClient.php');

$customer = 'CUSTOMER';
$password = 'PASSWORD';
$apiClient = new RestClient($customer, $password);
$data = array(
    'name' = -- > $_POST['fullname'],
    'phone1' => $_POST['phone'],
);
$projectId = 3;
$resource = sprintf('/v2/projects/%s/records', $projectId);
$result = $apiClient->post($resource, $data);
echo $result ? 'ok' : 'error';
```

Reference

Please find a comprehensive list of resources below. Further information about each resource can be found by clicking on the links.

Description	Method	Resource
Project settings		
List projects	GET	/v2/projects
Get project parameters	GET	/v2/projects/[projectid]
Cloning projects	PUT	/v2/projects/clone/[projectid]
Quota settings		
Get quota limits	GET	/v2/projects/[projectid]/quotas
Modify quota limits	PUT	/v2/projects/[projectid]/quotas
Database structure		
Add field and values	POST	/v2/projects/[projectid]/fields
Add new value(s) to a field	POST	/v2/projects/[projectid]/fields/[fieldid]/values
Modify values of a field	PUT	/v2/projects/[projectid]/fields/[fieldid]/values/[valueid]
Delete a value	DELETE	/v2/projects/[projectid]/fields/[fieldid]/values/[valueid]
Data/records		
List records	GET	/v2/projects/[projectid]/records
Get detailed record	GET	/v2/projects/[projectid]/records/[numberid]
Modify record	PUT	/v2/projects/[projectid]/records/[numberid]
Modify contact	PUT	/v2/projects/[projectid]/records/[numberid]/contacts/[contact]
Add record	POST	/v2/projects/[projectid]/records
Add disposition	POST	/v2/projects/[projectid]/records/[numberid]/dispositions
Batch modification		
Batch records modification	PUT	/v2/projects/[projectid]/records
Batch dispositions modification	PUT	/v2/projects/[projectid]/records/dispositions
User resources		
List users	GET	/v2/users

Description	Method	Resource
Add new user	PUT	/v2/users
List teams	GET	/v2/teams
List all roles	GET	/v2/roles
Statistic resources		
Get CDR log	GET	/v2/cdrs/[year]/[month]
Get CDR log for a specified uuid	GET	/v2/cdr/[year]/[month]/[uuid]
Get voicefile for a specified CDR	GET	/v2/cdr/[year]/[month]/[day]/[uuid]/voicefile
Get Mobile CDR log	GET	/v2/mcdrs/[year]/[month]/[day]
Get Mobile CDR log for a specified uuid	GET	/v2/mcdr/[year]/[month]/[uuid]
Get voicefile for a specified mobile CDR	GET	/v2/mcdr/[year]/[month]/[day]/[uuid]/voicefile
Get inbound statistics	GET	/v2/statistics/inbound
Get call statistics	GET	/v2/statistics/outbound
User state log	GET	/v2/statistics/userstate
Number of available agents in queue	GET	/v2/queues/[queueid]/availableusers
Other resources		
Add new value to Robinson list	POST	/v2/robinson/default_out

Project

List projects

List all folders and projects, whether they are active or not.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects
Options	N/A
Body	N/A
Response	
Body	Project array, encoded in JSON

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).

Response

Response object

Name	Type	Comment
container	boolean	Always false. Deprecated.
folder	string	Folder name.
folderid	integer	Unique folder identifier.
name	string	Project name.
projectid	integer	Unique project identifier.
status	string	Project status. Possible values: - active - inactive

Example

Request

URL: https://mycc.asp.virtual-call-center.eu/v2/projects

Response body

```
{
    "response": [
        {
            "projectid": 1,
            "name": "Sales – Budapest",
            "status": "active",
            "folderid": 1,
            "container": false,
            "folder": "Sales"
        },
        {
            "projectid": 2,
            "name": "Sales – Berlin",
            "status": "active",
            "folderid": 1,
            "container": false,
            "folder": "Sales"
        },
        {
            "projectid": 3,
            "name": "Script test",
            "status": "active",
            "folderid": 0,
            "container": false,
            "folder": "0"
        },
        {
            "projectid": 4,
            "name": "Inbound CC",
            "status": "active",
            "folderid": 2,
            "container": false,
            "folder": "Inbound"
        }
    ],
    "errors": []
}
```

Get project parameters

Get detailed information about a given project.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]
Options	N/A
Body	N/A
Response	
Body	Detailed information about the given project, encoded in JSON

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
projectid	integer	yes	Unique project identifier.

Response

Response object

Name	Type	Comment
dispositions	array of objects	Array of dispositions associated with the project.
fields	array of objects	Array of fields associated with the project.

Field object

Name	Type	Comment
customised	string	Defines whether this field should appear on an agent's manual dialing list. Possible values: - yes - no
fieldid	integer	Unique field identifier in VCC's database.
indexed	string	Defines whether records can be searched for on VCC Live's user interface using this field. Possible values: - yes - no
label	string	Field label. It appears on an agent's manual dialing list.
name	string	Field name.
type	string	Field type. Possible values: - text - simple (e.g.: combo) - multiple (e.g.: checkboxes)
values	array of objects	Field values.

Value object

Name	Type	Comment
commission	integer	Agent's commission, as defined by supervisor.
description	string	Comments, as provided by supervisor.
export_value	string	Export value.
fieldid	integer	Field identifier which this value is connected to.
label	string	Value label.
name	string	Value name.
price	integer	Call center's commission, as defined by supervisor.
valueid	integer	Value identifier in VCC's database.

Disposition object

Name	Type	Comment
assessment	string	Disposition assessment: - success: call recipient reached - ordered: call recipient reached and call goal achieved - failed: call recipient not reached
commission	integer	Call centre's commission, as defined by supervisor.
description	string	Disposition comment, as provided by supervisor.
id	integer	Disposition identifier in VCC's database.
label	string	Disposition export value, as provided by supervisor.
mode	string	Disposition status. Possible values: - active - inactive - deleted
name	string	Disposition name.
price	integer	Agent's commission, as defined by supervisor.
recall	integer	The default call-back time, in seconds, as defined by the supervisor for the given disposition. It is set only if the status is 'recall' or 'shared_recall'.
status	string	Disposition type, specifying the actual status of the record. Possible values: - recall (callback) - shared_recall (shared callback) - busy - limit_exceeded (channel limit exceeded) - unavailable - discard - temporary_not_available - discard_failed - dropped - quota (quota limit exceeded) - answering_machine - machine - robinson - finished

Example

Request

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/134

Response body

```
{
    "response": {
        "fields": [
            {
                "fieldid": 1,
                "name": "name",
                "type": "text",
                "indexed": "yes",
                "customised": "yes",
                "label": "name"
            },
            {
                "fieldid": 2,
                "name": "phone1",
                "type": "text",
                "indexed": "yes",
                "customised": "yes",
                "label": "phone1"
            },
            {
                "fieldid": 3,
                "name": "contract",
                "type": "multiple",
                "indexed": "yes",
                "customised": "no",
                "label": "Contract type",
                "values": [
                    [
                        {
                            "label": "Investment",
                            "name": "Investment",
                            "valueid": 1,
                            "fieldid": 3,
                            "export_value": "Credit",
                            "description": "Account",
                            "commission": 0,
                            "price": 0
                        }
                    ]
                ]
            },
            {
                "fieldid": 4,
                "name": "pay",
                "type": "simple",
                "indexed": "no",
                "customised": "no",
                "label": "Pay type",
                "values": [
                    [
                        {
                            "label": "Cash",
                            "name": "Cash",
                            "valueid": 2,
                            "fieldid": 4,
                            "export_value": "pay_cash",
                            "description": "paid by cash",
                            "commission": 10,
                            "price": 100
                        },
                        {
                            "label": "Bank",
                            "name": "Bank",
                            "valueid": 3,
                            "fieldid": 4,
                            "export_value": "pay_bank",
                            "description": "paid via transfer",
                            "commission": 20,
                            "price": 200
                        },
                        {
                            "label": "Credit card",
                            "name": "Credit card",
                            "valueid": 4,
                            "fieldid": 4,
                            "export_value": "pay_credit",
                            "description": "paid by credit card",
                            "commission": 30,
                            "price": 300
                        }
                    ]
                ]
            },
            {
                "fieldid": 5,
                "name": "adress",
                "type": "text",
                "indexed": "no",
                "customised": "no",
                "label": "Adress"
            }
        ],
        "dispositions": [
            {
                "name": "Callback",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "recall",
                "price": 0,
                "commission": 0,
                "id": 1,
                "label": "Callback",
                "mode": "active"
            },
            {
                "name": "Shared callback",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "shared_recall",
                "price": 0,
                "commission": 0,
                "id": 2,
                "label": "Shared callback",
                "mode": "active"
            },
            {
                "name": "Busy",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "busy",
                "price": 0,
                "commission": 0,
                "id": 3,
                "label": "Busy",
                "mode": "active"
            },
            {
                "name": "Limit exceeded",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "limit_exceeded",
                "price": 0,
                "commission": 0,
                "id": 4,
                "label": "Limit exceeded",
                "mode": "active"
            },
            {
                "name": "Unavailable",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "unavailable",
                "price": 0,
                "commission": 0,
                "id": 5,
                "label": "Unavailable",
                "mode": "active"
            },
            {
                "name": "Discard",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "discard",
                "price": 0,
                "commission": 0,
                "id": 6,
                "label": "Discard",
                "mode": "active"
            },
            {
                "name": "Temporary not available",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "temporary_not_available",
                "price": 0,
                "commission": 0,
                "id": 7,
                "label": "Temporary not available",
                "mode": "active"
            },
            {
                "name": "Discard (wrong number)",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "discard_failed",
                "price": 0,
                "commission": 0,
                "id": 8,
                "label": "Discard (wrong number)",
                "mode": "active"
            },
            {
                "name": "Dropped",
                "assesment": "failed",
                "description": "",
                "recall": 86400,
                "status": "dropped",
                "price": 0,
                "commission": 0,
                "id": 9,
                "label": "Dropped",
                "mode": "active"
            },
            {
                "name": "Quota",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "quota",
                "price": 0,
                "commission": 0,
                "id": 10,
                "label": "Quota",
                "mode": "active"
            },
            {
                "name": "Answering machine",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "answering_machine",
                "price": 0,
                "commission": 0,
                "id": 11,
                "label": "Answering machine",
                "mode": "active"
            },
            {
                "name": "Machine",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "machine",
                "price": 0,
                "commission": 0,
                "id": 12,
                "label": "Machine",
                "mode": "active"
            },
            {
                "name": "Robinson list",
                "assesment": "failed",
                "description": "",
                "recall": 0,
                "status": "robinson",
                "price": 0,
                "commission": 0,
                "id": 13,
                "label": "Robinson list",
                "mode": "inactive"
            },
            {
                "name": "Rejected",
                "assesment": "success",
                "description": "The customer is not interested, not live.",
                "recall": 0,
                "status": "finished",
                "price": 0,
                "commission": 0,
                "id": 15,
                "label": "disp_rejected",
                "mode": "active"
            },
            {
                "name": "Possible",
                "assesment": "success",
                "description": "The customer is interested in, but should be recalled later
. Please add the reason in description!",
                "recall": 41400,
                "status": "recall",
                "price": 0,
                "commission": 0,
                "id": 16,
                "label": "Possible",
                "mode": "active"
            },
            {
                "name": "Pay",
                "assesment": "ordered",
                "description": "This disposition is used when the customer paid.",
                "recall": 0,
                "status": "finished",
                "price": 500,
                "commission": 50,
                "id": 14,
                "label": "Pay",
                "mode": "active"
            }
        ]
    },
    "errors": []
}
```

Cloning projects

Copying a specified project's settings to a new project.

Request	
Method	PUT
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/clone/[projectid]
Options	N/A
Body	Name of the new project, encoded in JSON
Response	
Body	Result encoded in JSON

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
projectid	integer	yes	Identifier of the project to be cloned in VCC's database.

Request object

Name	Type	Mandatory	Comment
name	string	yes	Name of the new project.

Response

See the examples.

Example

Request

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/clone/17

Request body

```
{
    "name": "Customer Service UK"
}
```

Response body

```
{
    "response": 33,
    "errors": []
}
```

Possible errors

HTTP Code	Message	Description
417	error.required	No name entered in [name] key.
417	error.projectname_already_exists	The name added already exists in VCC's database.
417	error.wrong_projectname_format	The name added contains non-supported characters.
417	error.bad_arguments	The HTTP Body can't be interpreted eg: non-valid JSON format.
500	error.clone_project_error	Unsuccessful cloning, the project was not created due to an error.

Get quota limits

Get quota limits of a given project.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/quotas
Options	N/A
Body	N/A
Response	
Body	Detailed information about the given project's quota limits, encoded in JSON

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
projectid	integer	yes	Unique project identifier.

Response

Response object

Name	Type	Comment
cell	array of objects	Array of cell quotas.
edge	array of objects	Array of edge quotas.

Cell type quota object

Name	Type	Comment
children	array of objects	Recursive. Array of Cell type quota object.
container	boolean	Specifies whether the given element contains further elements (more quota values). Possible values: - true - false
label	object	Cell label object.
quotaid	integer	Unique identifier of the quota field. Optional. It only appears on the deepest level.
valueid	integer	Identifier of the quota field's value in VCC's database.

Cell label object

Name	Type	Comment
act_value	integer	The actual quota value. Optional. It only appears on the deepest level.
all	object	Always null. Deprecated.
diff_value	integer	The difference between the value and the actual value. Optional. It only appears on the deepest level.
export_value	string	The export name of the quota fields's value.
value	integer	The quota value set by the supervisor. Optional. It only appears on the deepest level.
value_name	string	The name of the quota fields's value.

Edge type quota object

Name	Type	Comment
children	array of objects	Array of child objects.
container	boolean	Specifies whether the given element contains further elements (more quota values). Possible values: - true - false
fieldid	integer	Identifier of the quota field.
label	string	The name of the quota fields's value.
projectid	integer	Unique project identifier.

Edge child object

Name	Type	Comment
children	object	Always null. Deprecated.
container	boolean	Always false. Deprecated.
label	object	Edge label object.
quotaid	integer	Identifier of the specific qouta in VCC's database.
valueid	integer	Identifier of the quota field's value in VCC's database.

Edge label object

Name	Type	Comment
act_value	integer	The actual quota value.
diff_value	integer	The difference between the value and the actual value.
name	string	The name of the quota fields's value.
value	integer	The quota value set by the supervisor.

Example

Request URL: https://mycc.asp.virtual-call-center.eu/v2/projects/134/quotas

Response body

```
{
    "response": {
        "cell": [
            {
                "valueid": 7,
                "label": {
                    "value_name": "male",
                    "export_value": ""
                },
                "container": true,
                "children": [
                    {
                        "valueid": 6,
                        "label": {
                            "value_name": "elementary",
                            "export_value": "",
                            "value": 150,
                            "act_value": 0,
                            "diff_value": 150,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 3
                    },
                    {
                        "valueid": 10,
                        "label": {
                            "value_name": "high",
                            "export_value": "",
                            "value": 150,
                            "act_value": 0,
                            "diff_value": 150,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 4
                    },
                    {
                        "valueid": 11,
                        "label": {
                            "value_name": "college",
                            "export_value": "",
                            "value": 200,
                            "act_value": 0,
                            "diff_value": 200,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 5
                    },
                    {
                        "valueid": 12,
                        "label": {
                            "value_name": "university",
                            "export_value": "",
                            "value": 200,
                            "act_value": 0,
                            "diff_value": 200,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 6
                    },
                    {
                        "valueid": 13,
                        "label": {
                            "value_name": "na",
                            "export_value": "",
                            "value": 0,
                            "act_value": 0,
                            "diff_value": 0,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 7
                    }
                ]
            },
            {
                "valueid": 9,
                "label": {
                    "value_name": "female",
                    "export_value": ""
                },
                "container": true,
                "children": [
                    {
                        "valueid": 6,
                        "label": {
                            "value_name": "elementary",
                            "export_value": "",
                            "value": 150,
                            "act_value": 0,
                            "diff_value": 150,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 8
                    },
                    {
                        "valueid": 10,
                        "label": {
                            "value_name": "high",
                            "export_value": "",
                            "value": 150,
                            "act_value": 0,
                            "diff_value": 150,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 9
                    },
                    {
                        "valueid": 11,
                        "label": {
                            "value_name": "college",
                            "export_value": "",
                            "value": 200,
                            "act_value": 0,
                            "diff_value": 200,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 10
                    },
                    {
                        "valueid": 12,
                        "label": {
                            "value_name": "university",
                            "export_value": "",
                            "value": 200,
                            "act_value": 0,
                            "diff_value": 200,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 11
                    },
                    {
                        "valueid": 13,
                        "label": {
                            "value_name": "na",
                            "export_value": "",
                            "value": 0,
                            "act_value": 0,
                            "diff_value": 0,
                            "all": null
                        },
                        "container": false,
                        "children": null,
                        "quotaid": 12
                    }
                ]
            }
        ],
        "edge": [
            {
                "projectid": 45,
                "fieldid": 6,
                "container": true,
                "label": {
                    "name": "q_country"
                },
                "children": [
                    {
                        "quotaid": 1,
                        "valueid": 5,
                        "label": {
                            "name": "Hungary",
                            "value": 1000,
                            "act_value": 0,
                            "diff_value": 1000
                        },
                        "container": false,
                        "children": null
                    },
                    {
                        "quotaid": 2,
                        "valueid": 8,
                        "label": {
                            "name": "Germany",
                            "value": 2000,
                            "act_value": 0,
                            "diff_value": 2000
                        },
                        "container": false,
                        "children": null
                    }
                ]
            }
        ]
    },
    "errors": []
}
```

Modify quota limits

Set quota limits of a given project.

Request	
Method	PUT
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/quotas
Options	N/A
Body	Quota limits to be modified, encoded in JSON
Response	
Body	Result of modification, encoded in JSON

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
projectid	integer	yes	Unique project identifier in VCC's database.

Response object

Name	Type	Mandatory	Comment
quotas	array of objects	yes	Array of quota objects.

Quota object

Name	Type	Mandatory	Comment
quotaid	integer	yes	Quota identifier.
value	integer	yes	Value of the quota to be set.

Response

See the examples.

Example

Request URL: https://mycc.asp.virtual-call-center.eu/v2/projects/134/quotas

Request body

```
[
    {
        "quotaid": 2,
        "value": 15
    },
    {
        "quotaid": 4,
        "value": 20
    }
]
```

Response body if success

```
{
    "errors": [],
    "response": {
    "0": true,
    "1": true
    }
}
```

Response body if partially succesful

```
{
    "errors": [
        {
            "errorcode": 417,
            "index": 1,
            "errormessage": "error.missing_arguments"
        },
        {
            "errorcode": 417,
            "index": 3,
            "errormessage": "error.missing_arguments"
        }
    ],
    "response": {
        "0": true,
        "2": true
    }
}
```

Database

Add field and values

Add a field and their relevant values to a project.

Request	
Method	POST
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/fields
Options	N/A
Body	Field and relevant values, encoded in JSON.
Response	
Body	New fieldID(s), encoded in JSON.

Request

Resource parameters

Name	**Type**	**Mandatory**	**Comment**
customer	string	yes	Your call centre's unique identifier (subdomain).
projectid	integer	yes	Unique project identifier.

Request object

Name	Type	Mandatory	Comment
customised	string	no	Defines whether field is shown in agent's list. Possible values: - yes - no
indexed	string	no	Records can be searched using this field. Possible values: - yes - no
label	string	no	Field label.
name	string	yes	Field's unique identifier (may contain only lower-case english and underscore characters).
quota	string	no	Defines quota type if needed. Possible values: - none - edge - cell
type	string	yes	Field type. 'Values' object needs to be set if type is 'simple' or 'multiple'. Possible values: - text - simple - multiple
values	array of objects	no	Array of field values.

Value object

Name	Type	Mandatory	Comment
commission	integer	no	Agent's commission.
description	string	no	Simple description.
exportvalue	string	no	Value's export name.
name	string	yes	Value name.
price	integer	no	Call centre's commission.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/10/fields

Request body

```
{
    "name": "education",
    "label": "education attainment",
    "type": "simple",
    "indexed": "yes",
    "customised": "yes",
    "quota": "none",
    "values": [
        {
            "name": "high school",
            "exportvalue": "HS",
            "description": "high school graduate",
            "commission": "100",
            "price": "200"
        },
        {
            "name": "university",
            "exportvalue": "U",
            "description": "university degree",
            "commission": "10",
            "price": "20"
        }
    ]
}
```

Response body - on success

```
{
    "errors": [],
    "response": 11
}
```

Response body - on error

```
{
    "errors": [
        {
            "errorcode": 417,
            "index": 0,
            "errormessage": "error.name_must_be_between",
            "property": "name"
        }
    ],
    "response": false
}
```

Possible errors

HTTPCode	Message	Description
417	error.adding_field_was_unsuccessful	Unsuccessful addition, the field was not created due to an error.
417	error.field_name_already_exists	The field already exists in VCC's database.
417	error.logged_in_agents	
417	error.name_must_be_between	The name is too short or long.
417	error.value_name_already_exists	The value already exists in VCC's database.
417	error.wrong_field_name_format	The name contains one or more non-supported characters eg: special character.

Add new value(s) to a field

Values can be added to a project's field.

Request	
Method	POST
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/fields/[fieldid]/values
Options	N/A
Body	Array of value objects, encoded in JSON.
Response	
Body	New valueID(s), encoded in JSON.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
fieldid	integer	yes	Field identifier.
projectid	integer	yes	Unique project identifier.

Request object

Name	Type	Mandatory	Comment
	array	yes	Array of value objects.

Value object

Name	Type	Mandatory	Comment
comission	string	no	Agent's commission.
description	string	no	Simple description.
exportvalue	string	no	Value's export name.
name	string	yes	Value name.
price	string	no	Call centre's commission.

Example

Request

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/10/fields/3/value

Request body

```
[
    {
        "name": "high school",
        "exportvalue": "HS",
        "description": "high school graduate",
        "commission": "100",
        "price": "200"
    },
    {
        "name": "university",
        "exportvalue": "U",
        "description": "university degree",
        "commission": "10",
        "price": "20"
    }
]
```

Response body - on success

```
{
    "errors": [],
    "response": [
        6,
        7
    ]
}
```

Response body - on error

```
{
    "errors": [
        {
            "errorcode": 417,
            "index": 1,
            "errormessage": "error.name_must_be_between",
            "property": "name"
        },
        {
            "errorcode": 417,
            "index": 3,
            "errormessage": "error.name_must_be_between",
            "property": "name"
        },
        {
            "errorcode": 417,
            "index": 6,
            "errormessage": "error.value_name_already_exists",
            "property": "name"
        }
    ],
    "response": {
        "0": 8,
        "2": 9,
        "4": 10,
        "5": 11
    }
}
```

Possible errors

HTTP Code	Message	Description
417	error.adding_values_was_unsuccessful	Unsuccessful addition, the values were not created due to an error.
417	error.name_must_be_between	The name is too short or long.
417	error.value_name_already_exists	The value already exists in VCC's database.
417	error.wrong_value_format	Missing value on a mandatory key.
417	error.wrong_values_format	The values sent are not correctly defined as an array or object.

Modify value(s) of a field

Values of a field in a project can be modified.

Request	
Method	PUT
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/fields/[fieldid]/values/[valueid]
Options	N/A
Body	Customised data, encoded in JSON.
Response	
Body	Shows whether the request has been successful or not.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
fieldid	integer	yes	Unique field identifier.
projectid	integer	yes	Unique project identifier.
valueid	integer	yes	Unique value identifier.

Request object

Name	Type	Mandatory	Comment
comission	string	no	Agent's commission.
description	string	no	Simple description.
exportvalue	string	no	Value's export name.
name	string	yes	Value name.
price	string	no	Call centre's commission.

Example

Request

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/251/fields/11/values/13

Request body

```
{
    "name": "Subscription1",
    "description": "subscription for internet",
    "export_value": "E002",
    "price": "1000",
    "commission": "2000"
}
```

Response body - on success

```
{
    "errors": [],
    "response": true
}
```

Possible errors

HTTP Code	Message	Description
417	error.name_must_be_between	The name is too short or long.
417	error.value_name_already_exists	The name already exists in VCC's database.
417	error.wrong_value_format	Mandatory parameter (name) is missing.
417	error.wrong_values_format	The format of data sent is not an array or object.

Delete a value

Delete a field's value.

Request	
Method	DELETE
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/fields/[fieldid]/values/[valueid]
Options	N/A
Body	N/A
Response	
Body	Result of the request.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
fieldid	integer	yes	Field identifier.
projectid	integer	yes	Unique project identifier.
valueid	integer	yes	Value identifier.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/251/fields/11/values/13

Response body - on success

```
{
    "errors": [],
    "response": true
}
```

Response body - on error

```
{
    "errors": [
        {
            "errorcode": 417,
            "index": 0,
            "errormessage": "error.deleteing_value_was_unsuccessful"
        }
    ],
    "response": false
}
```

List records

List records of a given project.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/records
Options	start, num, fieldname, value
Body	N/A
Response	
Body	Array of records filtered by options, encoded in JSON.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
projectid	integer	yes	Unique project identifier.

Options

Name	Type	Comment
fieldname	string	Result set can be filtered by fieldname and value parameters. Fieldname should be an indexed field, and can only be used in conjunction with the 'value' parameter.
num	integer	Specifies the maximum number of records to be returned.
start	integer	Specifies the offset of the first record.
value	string	Can only be used in conjunction with the 'fieldname' parameter.

Response

Response object

Name	Type	Comment
id	integer	Unique record identifier.
name	srting	Value of the record's 'name' field (usually the customer's name).

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/134/records?fieldname=age&value=36

```
{
    "response": {
        "rows": [
            {
                "id": 1,
                "name": "Stephen Green"
            },
            {
                "id": 2,
                "name": "Jack Wallis"
            },
            {
                "id": 3,
                "name": "Elizabeth Shawn"
            }
        ],
        "totalCount": 3
    },
    "errors": []
}
```

Get detailed record information

Retrieve customised data and relevant information, such as disposition and CDR, of a given record.

Important: DO NOT USE THIS METHOD FOR SYNCHRONISING YOUR OWN DATABASE WITH VCC'S DATABASE! Please use the Webhook instead.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/records/[numberid]
Options	N/A
Body	N/A
Response	
Body	A record's custom data, dispositions and CDRs, encoded in JSON.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
numberid	integer	yes	Unique record identifier.
projectid	integer	yes	Unique project identifier.

Response

Response object

Name	Type	Comment
cdrs	array of objects	Contains all Call Detail Record (CDR) information.
contacts	object	Contains all contact data related to the client that has been previously imported by a supervisor, or set by an agent.
data	object	Contains all user-specified data that has been previously imported by a supervisor, or set by an agent. If the value type is 'simple' or 'multiple', valueids are returned as a result.
events	array of objects	Contains all dispositions that have been set by the system, supervisor or agent.

Contact object

Name	Type	Comment
email	string	Email of the contact.
name	string	Name of the contact.
phone	string	Phone number of the contact.
title	string	Title of the contact e.g.: supervisor, Head of Customer Service.

Event object

Name	Type	Comment
attempted_connection	integer	Number of dial attempts by VCC Live's predictive or power dialer.

Name	Type	Comment
client_search	string	Whether the record has been specifically searched for and selected by the agent. Possible values: - yes - no
commission	integer	Agent's commission value for the record.
comp	object	If an agent sets an interval-based call-back, then the scheduled call-back details recorded in the system are displayed here.
create_date	string	Time the disposition is created, in YYYY-MM-DD hh-mm-ss format.
description	string	Comments on the disposition. Possible values: - recorded by agent or supervisor - updateimport (system) - predictivedialer (system)
dispositionid	integer	Disposition ID in the database.
next_calldate	string	Time the record is next due to be called, in YYYY-MM-DD hh-mm-ss format.
phone	string	The specific telephone number within the telephone numbers belonging to the record, that will be called by the system.
recycled_as_new	boolean	Whether the supervisor has recycled the number as a new record. Possible values: - true - false
price	integer	Call centre's commission value for the record.
shared_call	boolean	Whether the disposition is a shared call-back type or not. Possible values: - true - false
state	string	The agent's status when setting the disposition e.g.: CALL, AFTERWORK.
type	string	Type of the dispositon's modification. Possible values: - dispositon - recycle - user_change

Name	Type	Comment
userid	integer	User's identifier to whom the call is connected (-2 supervisor, -1 system, otherwise agent).
uuid	string	The call's universally unique identifier, as set by the system.

Comp object

Name	Type	Comment
from	string	From what time the system should call the number (hh-mm-ss format).
period	integer	What time interval there should be between attempted calls by the system (in minutes).
to	string	Until what time the system should call the number (hh-mm-ss format).

CDR object

Name	Type	Comment
beforequeuetime	integer	Time spent before the call is placed in a queue (e.g. time spent in IVR), in seconds.
billing_ts	string	Time the conversation begins, in YYYY-MM-DD hh-mm-ss format.
billingtime	integer	Length of complete call, in seconds (including ivr, queue and conversation time, but excluding ringtime).
destination	string	Called phone number.
dispositionid	integer	Unique disposition identifier in the database set for the call.
dispositionreach	integer	Disposition summary. Possible values: 1: not reached 2: reached 3: successful
dispositionstatus	integer	Disposition type. See: status key
holdtime	integer	Duration/length of hold, in seconds.
numberid	integer	Unique record identifier in the database.
projectid	integer	Unique project identifier in the database.
queuetime	integer	If the call is placed in a queue, then the time spent in the queue, in seconds.
ringtime	integer	Duration/length of ringing, in seconds.
source	string	Caller's phone number.
start_ts	string	Time the call is initiated, in YYYY-MM-DD hh-mm-ss format.
talktime	integer	Time during the call in which talking takes place, in seconds.
userid	string	Unique user identifier (-1 system, otherwise agent).
uuid	string	Unique call identifier.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/134/records/19847

Response body

```
{
    "response": {
        "data": {
            "name": "VCC Live",
            "phone2": "3619997400",
            "address": "8-14. Nagyenyed street, Budapest"
        },
        "contacts": {
            "2": {
                "title": "Head of Customer Service",
                "name": "Peter Green",
                "phone": "36203399877",
                "email": "peter.green@vcc.live"
            }
        },
        "events": [
            {
                "type": "disposition",
                "dispositionid": 3,
                "next_calldate": "2015-07-07T09:10:47+02:00",
                "phone": "36203399877",
                "attempted_connections": null,
                "shared_call": null,
                "state": "AFTERWORK",
                "client_search": "no",
                "uuid": "0b4b4302-2477-11e5-8803-f9fe10f1f802",
                "comp": {
                    "from": "00:00",
                    "to": "00:00",
                    "period": "0"
                },
                "price": 0,
                "commission": 0,
                "create_date": "2015-07-07 09:10:47",
                "userid": 6,
                "description": "client is busy!"
            }
        ],
        "cdrs": {
            "rows": [
                {
                    "uuid": "e78685dc-516a-11e5-863b-edb07dc0e690",
                    "source": "3619997400",
                    "destination": "36203399877",
                    "userid": 6,
                    "numberid": 2,
                    "start_ts": "2015-09-02 14:05:29",
                    "billing_ts": "2015-09-02 14:05:38",
                    "ringtime": 9,
                    "billingtime": 36,
                    "talktime": 36,
                    "queuetime": 0,
                    "beforequeuetime": 0,
                    "dispositionid": 3,
                    "dispositionreach": 1,
                    "dispositionstatus": 3,
                    "projectid": 3,
                    "holdtime": 0
                }
            ],
            "totalCount": 1
        }
    },
    "errors": []
}
```

Modify record

Modify a record's customised data.

If the value field (that can handle only one value) contains text, then:

- it finds the field value based on the text input, and sets it (based on the id)
- if it does not find the field value, then it automatically creates a new field value

If the value field (that can handle more than one values) contains text, then:

- it attempts to divide the text separated by the "|" character
- if it doesn't find any "|" character, then it handles it as one value

The followings apply to each separated values:

- it finds the field value based on the text input, and sets it (based on the id)
- if it does not find the field value, then it automatically creates a new field value

Request	
Method	PUT
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/records/[numberid]
Options	next_contact
Body	Customised data, encoded in JSON.
Response	
Body	Results whether the request is succeeded or not.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
numberid	integer	yes	Unique record identifier.
projectid	integer	yes	Unique project identifier.

Options

Name	Type	Comment
next_contact	string	Contains the telephone number the system should call next. Possible values: - phone1, phone2...phone9

Response

Response object

Name	Type	Comment
response	boolean	States whether request is successful or not. Possible values: - true - false

Example

This request sets 'phone1' field's value and sets it as next callable number.

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/134/records/19873

Request body

```
{
    "phone1": "3619997400",
    "address": "1 Main street",
    "city": "Budapest",
    "likes": "sports|movies"
}
```

Response body - on success

```
{
    "errors": [],
    "response": true
}
```

Response body - on error

```
{
    "errors": [
        {
            "errorcode": 400,
            "errormessage": "400 Bad request",
            "index": 0
        }
    ],
    "response": false
}
```

Modify contact

Modify a record's contacts.

Request	
Method	PUT
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/records/[numberid]/contacts/[contact]
Options	NA
Body	Customised data, encoded in JSON.
Response	
Body	Results whether the request is succeeded or not.

Request

Resource parameters

Name	Type	Mandatory	Comment
contacts	integer	yes	Contact identifier.
customer	string	yes	Your call centre's unique identifier (subdomain).
numberid	integer	yes	Unique record identifier.
projectid	integer	yes	Unique project identifier.

Response

Response object

Name	Type	Comment
response	boolean	States whether request is successful or not. Possible values: - true - false

Example

This request sets contact informations to the second contact of the client.

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/134/records/19873/contacts/2

Request body

```
{
    "title": "Supervisor",
    "name": "Peter Green",
    "phone": "3619997400",
    "email": "peter.green@gmail.com"
}
```

Response body - on success

```
{
    "errors": [],
    "response": true
}
```

Response body - on error

```
{
    "response": false,
    "errors": [
        {
            "index": 1,
            "errorcode": 417,
            "errormessage": "error.wrong_email_format"
        }
    ]
}
```

Possible errors

HTTP Code	Message	Description
417	error.contact_not_exists	The contacts is not exists.
417	error.wrong_email_format	The email format is not valid.

Add record

Add a new record to a project's database.

If the value field (that can handle only one value) contains text, then:

- it finds the field value based on the text input, and sets it (based on the id)
- if it does not find the field value, then it automatically creates a new field value

If the value field (that can handle more than one values) contains text, then:

- it attempts to divide the text separated by the "|" character
- if it doesn't find any "|" character, then it handles it as one value

The followings apply to each separated values:

- it finds the field value based on the text input, and sets it (based on the id)
- if it does not find the field value, then it automatically creates a new field value

Request	
Method	POST
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/records
Options	N/A
Body	Customised data, encoded in JSON.
Response	
Body	States whether the request is successful or not.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
numberid	integer	yes	Unique record identifier.
projectid	integer	yes	Unique project identifier.

Request object

Name	Type	Mandatory	Comment
contacts	object	no	Contains all contact data related to the client.
disposition	object	no	Disposition for the given record to be set.
form	object	yes	The record's customised data, such as name, telephone number, address, city, likes, etc. can be set via the form key. If simple or multiple values are set, use 'valueid'.
phone_field	srting	no	Which telephone number should be called next. Possible values: phone1, phone2...phone9
premium	object	no	Agent and call centre commissions.

Contacts object

Name	Type	Mandatory	Comment
email	string	no	Email of the contact.
name	string	no	Name of the contact.
phone	string	yes	Phone number of the contact.
title	string	no	Title of the contact e.g.: supervisor, Head of Customer Service.

Premium object

Name	Type	Mandatory	Comment
commission	string	no	Agent's commission value connected to the record.
price	string	no	Call centre commission value connected to the record.

Disposition object

Name	Type	Mandatory	Comment
dispositionid	integer	yes	Unique disposition identifier.
next_calldate	string	no	If call-back or shared call-back type dispositions are set, this is the next time the record is due to be called, in YYYY-MM-DD hh-mm-ss format.
userid	string	yes	Unique user identifier.

Response

Response object

Name	Type	Comment
response	array	Unique 'numberid'(s) of the inserted record(s).

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/134/records

Request body

```
{
    "phone_field": "phone4",
    "form": {
        "name": "Peter Green",
        "city": "Budapest",
        "likes": "sports|movies"
    },
    "contacts": {
        "1": {
            "title": "Supervisor",
            "name": "Peter Green",
            "phone": "3619997400",
            "email": "peter.green@email.com"
        },
        "4": {
            "phone": "3619996400"
        }
    },
    "premium": {
        "price": "100",
        "commission": "50"
    },
    "disposition": {
        "dispositionid": "1",
        "userid": "7",
        "next_calldate": "2015-07-20 10:26:55"
    }
}
```

Response body

```
{
    "errors": [],
    "response": [
        6
    ]
}
```

Add disposition

Add a new disposition to a specified record.

Request	
Method	POST
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/records/[numberid]/dispositions
Options	N/A
Body	Disposition object, encoded in JSON.
Response	
Body	Results whether the request is succeeded or not.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
numberid	integer	yes	Unique record identifier.
projectid	integer	yes	Unique project identifier.

Request object

Name	Type	Mandatory	Comment
avoid_in_call	bool	no	Ignores all records being handled by agents (prework, in call, afterwork).
description	string	no	Description added by the supervisor.
dispositionid	string	yes	Unique disposition identifier.
next_calldate	string	no	If call-back or shared call-back type dispositions are set, this is the next time the record is due to be called, in YYYY-MM-DD hh-mm-ss format.
next_contactid	string	no	Defines which phone field should be called next time (e.g.: next_contactid:2, that means phone2 will be called)
userid	string	yes	User's unique identifier, to whom the disposition is linked.

Response

Response body is empty if it is successful.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/134/records/8377/dispositions

Request body

```
{
    "dispositionid": "1",
    "userid": "17",
    "next_calldate": "2017-01-17 11:00:00",
    "next_contactid": "2",
    "description": "University degree",
    "avoid_in_call": true
}
```

Response body

```
{
    "errors": [],
    "response": true
}
```

Batch records modification

Modify multiple records in batch mode in a project.

If the value field (that can handle only one value) contains text, then:

- it finds the field value based on the text input, and sets it (based on the id)
- if it does not find the field value, then it automatically creates a new field value

If the value field (that can handle more than one values) contains text, then:

- it attempts to divide the text separated by the "|" character
- if it doesn't find any "|" character, then it handles it as one value

The followings apply to each separated values:

- it finds the field value based on the text input, and sets it (based on the id)
- if it does not find the field value, then it automatically creates a new field value

Request	
Method	PUT
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/records
Options	N/A
Body	Customised data, encoded in JSON.
Response	
Body	Array of partial results by elements.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
projectid	integer	yes	Unique project identifier.

Request object

Name	Type	Mandatory	Comment
default	object	no	Key value pairs that should be set for every record, specified by an 'element' object.
elements	array of objects	yes	Array of element objects to be set.

Element object

Name	Type	Mandatory	Comment
form	object	yes	Custom data fields to be set as key value pairs.
numberid	string	yes/no	Unique record identifier specifying a record to be set. Either 'search' or 'numberid' can be set.
search	object	yes/no	Specifies a record to which key value pairs should be applied to. Either 'search' or 'numberid' can be set.

Response

Array of partial results by elements.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/22/records

Request body

```
{
    "elements": [
        {
            "form": {
                "name": "Peter Green",
                "city": "Budapest",
                "likes": "sports|movies"
            },
            "search": {
                "name": "Green"
            }
        },
        {
            "form": {
                "name": "Thomas",
                "address": "London"
            },
            "numberid": "4"
        }
    ],
    "default": {
        "address": "Budapest",
        "age": "30"
    }
}
```

Response body

```
{
    "errors": [],
    "response": {
        "0": true,
        "1": true
    }
}
```

Response body if partially succesful

```
{
    "errors": [
        {
            "errorcode": 417,
            "errormessage": "error.missing_numberid_or_search",
            "index": 1
        }
    ],
    "response": {
        "0": true,
        "2": true
    }
}
```

Batch dispositions modification

Modify multiple dispositions in batch mode within a project.

Request	
Method	PUT
Resource	https://[customer].asp.virtual-call-center.eu/v2/projects/[projectid]/records/dispositions
Options	N/A
Body	Disposition objects, encoded in JSON.
Response	
Body	Array of partial results by elements.

Request

Resource parameters

Name	**Type**	**Mandatory**	**Comment**
customer	string	yes	Your call centre's unique identifier (subdomain).
projectid	integer	yes	Unique project identifier.

Request object

Name	Type	Mandatory	Comment
avoid_in_call	bool	no	Ignores all records being handled by agents (prework, in call, afterwork).
disposition	object	no	Key value pairs that should be set for every record, specified by an 'element' object.
elements	array of objects	yes	Array of disposition objects to be set.

Element object

Name	Type	Mandatory	Comment
description	string	no	Description added by the supervisor.
dispositionid	string	no	Unique disposition identifier.
next_calldate	string	no	Defines callback time in YYYY-MM-DD HH:MM:SS format.
numberid	string	yes/no	Unique record identifier specifying a record to be set. Either 'search' or 'numberid' can be set.
search	object	yes/no	Specifies a record to which key value pairs should be applied to. Either 'search' or 'numberid' can be set. Possible values: - numberid: Unique record identifier. - pref_userid: User identifier to whom the disposition is linked. - act_status: Disposition identifier which the disposition is linked. If more than one 'search' object is added, the search method will be 'OR'. If more than one search criteria is defined within the 'search' object, the search method will be 'AND'.
userid	string	no	User identifier to whom the disposition is linked.

Disposition object

Name	Type	Mandatory	Comment
description	string	no	Description added by the supervisor.
dispositionid	integer	no	Unique disposition identifier.
next_calldate	string	no	Defines callback time in YYYY-MM-DD HH:MM:SS format.
userid	integer	no	User identifier to whom the disposition is linked.

Response

Array of partial results by elements.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/projects/17/records/dispositions

Request body

```
{
    "elements": [
        {
            "numberid": "1",
            "dispositionid": "1",
            "userid": "22",
            "next_calldate": "2017-10-10 11:00:00",
            "description": "High school graduate"
        },
        {
            "search": {
                "name": "Green"
            }
        }
    ],
    "disposition": {
        "dispositionid": "2",
        "userid": "7",
        "next_calldate": "2017-09-10 12:00:00",
        "description": "University degree"
    },
    "avoid_in_call": true
}
```

Response body

```
{
    "errors": [],
    "response": {
        "0": true,
        "1": true
    }
}
```

Response body if partially successful

```
{
    "errors": [
        {
            "errorcode": 417,
            "errormessage": "error.missing_numberid_or_search",
            "index": 1
        }
    ],
    "response": {
        "0": true,
        "2": true
    }
}
```

Add new value to Robinson list

Request	
Method	POST
Resource	https://[customer].asp.virtual-call-center.eu/v2/robinson/default_out
Options	N/A
Body	Array of objects, encoded in JSON.
Response	
Body	Results whether the request is succeeded or not.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/robinson/default_out

Request body

```
[
    {
        "phone": "3619997400"
    },
    {
        "phone": "3619996400",
        "expire": "2017-01-01"
    }
]
```

Response body - on success

```
{
    "response": true,
    "errors": []
}
```

Response body - on error

```
{
    "response": true,
    "errors": [
        {
            "errorcode": 600,
            "errormessage": "error.invalid_phone",
            "property": "phone",
            "index": 0
        }
    ]
}
```

Statistics

Get CDR log

Retrieve CDR log list, filtered according to given options.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/cdrs/[year]/[month]/[day]
Options	projectid, start, num
Body	N/A
Response	
Body	Array of CDR objects, encoded in JSON.

Request

Resource parameters

Name	**Type**	**Mandatory**	**Comment**
customer	string	yes	Your call centre's unique identifier (subdomain).
year	integer	yes	Year.
month	integer	yes	Month within the requested year. Possible values: - 01-12
day	integer	no	Day within the requested month. Possible values: - 01-31

Options

Name	**Type**	**Mandatory**	**Comment**
num	integer	no	Specifies the maximum number of CDRs to be returned. Possible values: - [0-9]+
projectid	integer	no	Narrows down the scope of data to be searched in a given project.
start	integer	no	Specifies the offset of the first CDR. Possible values: - [0-9]+

Response

Response object

Name	Type	Comment
rows	array of object	Array of CDR elements.
totalCount	integer	Number of CDRs returned.

CDR object

Name	Type	Comment
afterwork	integer	Duration/length of afterwork, in seconds.
beforequeuetime	integer	Time spent before the call is placed in a queue (e.g. time spent in IVR), in seconds.
billing_ts	string	Time the conversation begins, in YYYY-MM-DD hh-mm-ss format.
billingtime	integer	Length of complete call, in seconds (including ivr, queue and conversation time, but excluding ringtime).
destination	string	Called phone number.
dispositionid	integer	Unique disposition identifier in the database set for the call.
dispositionreach	integer	Disposition summary. Possible values: - 1: not reached - 2: reached - 3: successful
dispositionstatus	integer	Disposition type. See: status key.
direction	string	Call direction: - inbound - outbound
holdtime	integer	Duration/length of hold, in seconds.
numberid	integer	Unique record identifier in the database.
prework	integer	Duration/length of prework, in seconds.
projectid	integer	Unique project identifier in the database.
queuetime	integer	If the call is placed in a queue, then the time spent in the queue, in seconds.

Name	Type	Comment
ringtime	integer	Duration/length of ringing, in seconds.
source	string	Caller's phone number.
start_ts	string	Time the call is initiated, in YYYY-MM-DD hh-mm-ss format.
talktime	integer	Time during the call in which talking takes place, in seconds.
userid	integer	Unique user identifier (-1 system, otherwise agent).
uuid	string	Unique call identifier.

Example

List the first CDR in November, 2014.

Request

https://mycc.asp.virtual-call-center.eu/v2/cdrs/2014/11?start=0&num=1

Response body

```
{
    "response": {
        "rows": [
            {
                "uuid": "95bd5f46-44e0-11e5-a6da-6547f8762750",
                "source": "3619980106",
                "destination": "3619996400",
                "userid": 6,
                "numberid": 11,
                "start_ts": "2015-08-17 15:05:08",
                "billing_ts": "2015-08-17 15:05:17",
                "ringtime": 9,
                "billingtime": 3,
                "talktime": 3,
                "queuetime": 0,
                "beforequeuetime": 0,
                "dispositionid": 3,
                "dispositionreach": 1,
                "dispositionstatus": 3,
                "projectid": 3,
                "holdtime": 0,
                "afterwork": 9,
                "prework": 0,
                "direction": "outbound"
            }
        ],
        "totalCount": 1
    },
    "errors": []
}
```

Get CDR log for a specified uuid

Retrieve specified uuid CDR log.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/cdr/[year]/[month]/[uuid]
Options	
Body	N/A
Response	
Body	CDR array, encoded in JSON.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
year	integer	yes	Year.
month	integer	yes	Month within the requested year. Possible values: - 01-12
uuid	string	yes	Unique call identifier.

Response

Name	Type	Comment
afterwork	integer	Duration/length of afterwork, in seconds.
beforequeuetime	integer	Time spent before the call is placed in a queue (e.g. time spent in IVR), in seconds.
billing_ts	string	Time the conversation begins, in YYYY-MM-DD hh-mm-ss format.
billingtime	integer	Length of complete call, in seconds (including ivr, queue and conversation time, but excluding ringtime).
destination	string	Called phone number.
direction	string	Call direction: - inbound - outbound
dispositionid	integer	Unique disposition identifier in the database set for the call.
dispositionreach	integer	Disposition summary. Possible values: - 1: not reached - 2: reached - 3: successful
dispositionstatus	integer	Disposition type. See: status key.
holdtime	integer	Duration/length of hold, in seconds.

Name	Type	Comment
numberid	integer	Unique record identifier in the database.
prework	integer	Duration/length of prework, in seconds.
projectid	integer	Unique project identifier in the database.
queuetime	integer	If the call is placed in a queue, then the time spent in the queue, in seconds.
ringtime	integer	Duration/length of ringing, in seconds.
source	string	Caller's phone number.
start_ts	string	Time the call is initiated, in YYYY-MM-DD hh-mm-ss format.
talktime	integer	Time during the call in which talking takes place, in seconds.
userid	integer	Unique user identifier (-1 system, otherwise agent).
uuid	string	Unique call identifier.

Example

List the CDR with the following uuid: bbf5825e-85de-4ec1-8625-e6eaf5c96b1e

Request

https://mycc.asp.virtual-call-center.eu/v2/cdr/2015/01/bbf5825e-85de-4ec1-8625-e6eaf5c96b1e

Response body

```
{
    "response": {
        "uuid": "95bd5f46-44e0-11e5-a6da-6547f8762750",
        "source": "3619980106",
        "destination": "36203399877",
        "userid": 6,
        "numberid": 11,
        "start_ts": "2015-08-17 15:05:08",
        "billing_ts": "2015-08-17 15:05:17",
        "ringtime": 9,
        "billingtime": 3,
        "talktime": 3,
        "queuetime": 0,
        "beforequeuetime": 0,
        "dispositionid": 3,
        "dispositionreach": 1,
        "dispositionstatus": 3,
        "projectid": 3,
        "holdtime": 0,
        "afterwork": 9,
        "prework": 0,
        "direction": "outbound"
    },
    "errors": []
}
```

Get voicefile for a specified CDR

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/cdr/[year]/[month]/[day]/[uuid]/voicefile
Options	
Body	N/A
Response	
Body	The current voicefile.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
year	integer	yes	Year.
month	integer	yes	Month within the requested year. Possible values: - 01-12
day	integer	yes	Day within the requested month. Possible values: - 01-31
uuid	string	yes	Unique call identifier.

Response

The requested voicefile directly downloaded.

NOTE: The system answers "429 Too many request" until the previous process is completed.

Example

Get the voicefile with the following uuid: bbf5825e-85de-4ec1-8625-e6eaf5c96b1e

Request

https://mycc.asp.virtual-call-center.eu/v2/cdr/2015/01/27/bbf5825e-85de-4ec1-8625-e6eaf5c96b1e/voicefile

Get mobile CDR log

Retrieve mobile CDR list, filtered according to given options.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/mcdrs/[year]/[month]/[day]
Options	projectid, start, num
Body	N/A
Response	
Body	Array of Mobile CDR objects, encoded in JSON.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
year	integer	yes	Year.
month	integer	yes	Month within the requested year. Possible values: - 01-12
day	integer	no	Day within the requested month. Possible values: - 01-31

Options

Name	Type	Mandatory	Comment
num	integer	no	Specifies the maximum number of CDRs to be returned. Possible values: - [0-9]+
projectid	integer	no	Narrows down the scope of data to be searched in a given project.
start	integer	no	Specifies the offset of the first CDR. Possible values: - [0-9]+

Response

Response object

Name	Type	Comment
rows	array of object	Array of Mobile CDR elements.
totalCount	integer	Number of Mobile CDRs returned.

CDR object

Name	Type	Comment
destination	string	Called phone number.
direction	string	Call direction: - inbound - outbound
dispositionid	integer	Unique disposition identifier in the database set for the call.
dispositionreach	integer	Disposition summary. Possible values: - 1: not reached - 2: reached - 3: successful
dispositionstatus	integer	Disposition type. See: status key.
numberid	integer	Unique record identifier in the database.
projectid	integer	Unique project identifier in the database.
source	string	Caller's phone number.
start_ts	string	Time the call is initiated, in YYYY-MM-DD hh-mm-ss format.
talktime	integer	Time during the call in which talking takes place, in seconds.
userid	integer	Unique user identifier.
uuid	string	Unique call identifier.

Example

List the first Mobile CDR in June, 2016.

Request

https://mycc.asp.virtual-call-center.eu/v2/mcdrs/2016/06?start=0&num=1

Response body

```
{
    "response": {
        "rows": [
            {
                "uuid": "29a16a57-2801-43a5-b04c-65eeaa40de78",
                "source": "36201233456",
                "destination": "3619996400",
                "userid": 81,
                "numberid": 503,
                "start_ts": "2016-06-02 15:43:45",
                "talktime": 40,
                "dispositionid": 15,
                "dispositionreach": 2,
                "dispositionstatus": 5,
                "projectid": 2,
                "direction": "outbound"
            }
        ],
        "totalCount": 1
    },
    "errors": [

    ]
}
```

Get mobile CDR log for a specified uuid

Retrieve specified uuid Mobile CDR log.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/mcdr/[year]/[month]/[uuid]
Options	
Body	N/A
Response	
Body	Mobile CDR array, encoded in JSON.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
year	integer	yes	Year.
month	integer	yes	Month within the requested year. Possible values: - 01-12
uuid	string	yes	Unique call identifier.

Response

Name	Type	Comment
destination	string	Called phone number.
dispositionid	integer	Unique disposition identifier in the database set for the call.
dispositionreach	integer	Disposition summary. Possible values: - 1: not reached - 2: reached - 3: successful
dispositionstatus	integer	Disposition type. See: status key.
direction	string	Call direction: - inbound - outbound
numberid	integer	Unique record identifier in the database.
projectid	integer	Unique project identifier in the database.
source	string	Caller's phone number.
start_ts	string	Time the call is initiated, in YYYY-MM-DD hh-mm-ss format.
talktime	integer	Time during the call in which talking takes place, in seconds.
userid	integer	Unique user identifier (-1 system, otherwise agent).
uuid	string	Unique call identifier.

Example

List the Mobile CDR with the following uuid: 29a16a57-2801-43a5-b04c-65eeaa40de78

Request

https://mycc.asp.virtual-call-center.eu/v2/mcdr/2016/06/29a16a57-2801-43a5-b04c-65eeaa40de78

Response body

```
{
    "response": {
            "uuid": "29a16a57-2801-43a5-b04c-65eeaa40de78",
            "source": "36201233456",
            "destination": "3619996400",
            "userid": 81,
            "numberid": 503,
            "start_ts": "2016-06-02 15:43:45",
            "talktime": 40,
            "dispositionid": 15,
            "dispositionreach": 2,
            "dispositionstatus": 5,
            "projectid": 2,
            "direction": "outbound"
      },
      "errors": [

      ]
}
```

Get voicefile for a specified mobile CDR

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/mcdr/[year]/[month]/[day]/[uuid]/voicefile
Options	
Body	N/A
Response	
Body	The current mobile voicefile.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
year	integer	yes	Year.
month	integer	yes	Month within the requested year. Possible values: - 01-12
day	integer	yes	Day within the requested month. Possible values: - 01-31
uuid	string	yes	Unique mobile call identifier.

Response

The requested voicefile directly downloaded.

NOTE: The system answers "429 Too many request" until the previous process is completed.

Example

Get the mobile voicefile with the following uuid: d7fb40cc-6b72-4cd1-b7e8-bfcbca6a5453

Request

https://mycc.asp.virtual-call-center.eu/v2/mcdr/2015/01/27/d7fb40cc-6b72-4cd1-b7e8-bfcbca6a5453/voicefile

Get inbound statistics

Retrieve inbound statistics, filtered according to given options.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/statistics/inbound
Options	from, to, dimensions, teams, projects, manual_sla, of
Body	N/A
Response	
Body	Requested inbound statistic records in tabulator-tabbed, CSV or JSON format.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).

Options

Name	Type	Mandatory	Comment
dimensions	string	yes	Dimensions. If there are several items, they should be separated with commas (e.g: dimensions=project,queue,date). Possible values: - project - queue - date
from	string	yes	First date of the requested time period to be searched, in yyyymmdd format (e.g: from=20120301).
to	string	yes	Last date of the requested time period to be searched, in yyyymmdd format (e.g: from=20120331).
manual_sla	integer	no	A default SLA (30 seconds) is used in some statistics values eg: 'answered_after' or 'answered_before'. This value can be overwritten with a custom value eg: manual_sla=40.
of	string	no	Output format. Possible values: - tab - cvs - json (default)
projects	integer	no	From which specific project(s) should data be retrieved. If there are several items, they should be separated with commas. (e.g: projects=53,64).
teams	integer	no	Which team, or teams, should the data relate to. If there are several items, they should be separated with commas. (e.g: teams=3,4).

Response

Response object

Name	Type	Comment
abort_before_queue	integer	Number of calls discarded by caller, before being placed in a queue.
abort_case_queue_time	integer	Average time spent in queue before the caller discards the call.
abort_in_queue_after	integer	Number of calls discarded by caller, after being placed in a queue and after the predefined SLA value is reached.
abort_in_queue_before	integer	Number of calls discarded by caller, after being placed in a queue and before the predefined SLA value is reached.
afterwork	integer	Average time spent in 'afterwork' status.
answered_after	integer	Number of calls answered by agents after the predefined SLA value is reached.
answered_before	integer	Number of calls answered by the agents before the predefined SLA value is reached.
date	integer	Date in yyyymmdd format.
projectid	integer	Project identifier.
queue_time	integer	Time spent in queue before being answered by an agent.
queue_time_without_talktime	integer	Average time before calls are discarded.
queueid	integer	Queue identifier.
redirect_before_queue	integer	Number of calls redirected before entering a queue.
redirect_after_queue	integer	Number of calls redirected after leaving a queue.
talk_time	integer	Time spent talking with an agent.
time_before_queue	integer	Average time spent before calls are placed in a queue.

Example

List inbound statistics between 01.01.2015. and 01.20.2015. based on the 'project' dimension.

Request

https://mycc.asp.virtual-call-center.eu/v2/statistics/inbound?from=20150101&to=20150120&dimensions=project

Response body

```
{
    "response": [
        {
            "projectid": 422,
            "date": "",
            "queueid": "",
            "answered_after": 0,
            "answered_before": 3,
            "queue_time": 27,
            "redirect_after_queue": 0,
            "abort_in_queue_after": 0,
            "talk_time": 15,
            "abort_in_queue_before": 3,
            "abort_before_queue": 7,
            "afterwork": 516,
            "abort_case_queue_time": 3,
            "redirect_before_queue": 0,
            "queue_time_without_talktime": 7,
            "time_before_queue": 7
        },
        {
            "projectid": "",
            "date": 20120620,
            "queueid": "",
            "answered_after": 0,
            "answered_before": 0,
            "queue_time": 4,
            "redirect_after_queue": 0,
            "abort_in_queue_after": 0,
            "talk_time": 0,
            "abort_in_queue_before": 2,
            "abort_before_queue": 7,
            "afterwork": 0,
            "abort_case_queue_time": 2,
            "redirect_before_queue": 0,
            "queue_time_without_talktime": 4,
            "time_before_queue": 7
        },
        {
            "projectid": "",
            "date": "",
            "queueid": 595,
            "answered_after": 0,
            "answered_before": 0,
            "queue_time": 4,
            "redirect_after_queue": 0,
            "abort_in_queue_after": 0,
            "talk_time": 0,
            "abort_in_queue_before": 2,
            "abort_before_queue": 7,
            "afterwork": 0,
            "abort_case_queue_time": 2,
            "redirect_before_queue": 0,
            "queue_time_without_talktime": 4,
            "time_before_queue": 7
        }
    ],
    "errors": []
}
```

Possible errors

HTTP Code	Message	Description
417	error.argument_format_error	One of the parameters is incorrect.
417	error.argument_missing_or_empty	Missing parameter in the URL.
417	error.maximum_date_interval_reached	More than the maximum search period (1 month) has been requested.

Get call statistics

Retrieve call statistics, filtered according to given options.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/statistics/outbound
Options	from, to, dimensions, teams, projects, users, of
Body	N/A
Response	
Body	Requested call statistic records in tabulator-tabbed, CSV or JSON format.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).

Options

Name	Type	Mandatory	Comment
dimensions	string	yes	Dimensions. If there are several items, they should be separated with commas (e.g: dimensions=project,user). Possible values: - project - user - date
from	string	yes	First date of the requested time period to be searched, in yyyymmdd format (e.g: from=20120301).
of	string	no	Output format. Possible values: - tab - cvs - json (default)
projects	integer	no	From which specific project(s) should data be retrieved. If there are several items, they should be separated with commas. (e.g: projects=53,64).
teams	integer	no	Which team, or teams, should the data relate to. If there are several items, they should be separated with commas. (e.g: teams=3,4).
to	string	yes	Last date of the requested time period to be searched, in yyyymmdd format (e.g: from=20120331).
users	integer	no	From which specific user(s) should data be retrieved. If there are several items, they should be separated with commas. (e.g: users=53,64).

Response

Response object

Name	Type	Comment
AFTERWORK	integer	Agent time spent in 'afterwork' status, in seconds.
AVAILABLE	integer	Agent time spent in 'available' status, in seconds.
AUX	integer	Agent time spent in 'aux' status, in seconds.

Name	Type	Comment
CALL	integer	Agent time spent in 'call' status, in seconds.
date	integer	Date in yyyymmdd format.
HOLD	integer	Agent time spent in 'hold' status, in seconds.
inbound	integer	Total duration of inbound calls, in seconds.
inbound_count	integer	Total number of inbound calls.
OFFLINE	integer	Agent time spent in 'offline' status, in seconds.
outbound	integer	Total duration of outbound calls, in seconds.
outbound_count	integer	Total number of outbound calls.
PREWORK	integer	Agent time spent in 'prework' status, in seconds.
projectid	integer	Project identifier.
reached	integer	Number of 'reached' disposition calls.
RINGING	integer	Agent time spent in 'ringing' status, in seconds.
successful	integer	Number of 'successful' disposition calls.
total_fee	integer	Total cost of calls.
UNAVAILABLE	integer	Agent time spent in 'unavailable' status, in seconds.
unreached	integer	Number of 'unreached' disposition calls.
userid	integer	User identifier.
WAITING4CALL	integer	Agent time spent in 'waiting4call' status, in seconds.
WAITING4RECALL	integer	Agent time spent in 'waiting4recall' status, in seconds.
*	integer	Agent time spent in custom break (AUX) codes, in second.

Example

List call statistics of projectid:15 between 01.01.2015. and 02.01.2015. based on the 'project' dimension.

https://mycc.asp.virtual-call-center.eu/v2/statistics/outbound?from=20150101&to=20150102&dimensions=project&projects=15

Response body

```
{
    "response": [
        {
            "projectid": "4",
            "UNAVAILABLE": "",
            "AVAILABLE": 304,
            "CALL": 37,
            "AUX": "",
            "PREWORK": "",
            "RINGING": 31,
            "WAITING4CALL": 3,
            "WAITING4RECALL": "",
            "AFTERWORK": 327,
            "OFFLINE": "",
            "HOLD": "",
            "inbound": 37,
            "outbound": "",
            "inbound_count": 3,
            "outbound_count": 0,
            "unreached": 4,
            "reached": "",
            "successful": "",
            "technical": "",
            "total_fee": 0
        }
    ],
    "errors": []
}
```

User state log

Retrieve user state log, filtered according to given options.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/statistics/userstate
Options	from, to, projects, users, of
Body	N/A
Response	
Body	Requested user states in tabulator-tabbed, CSV or JSON format.

Request

Resource parameters

Options

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
from	string	yes	First date of the requested time period to be searched, in yyyymmdd format (e.g: from=20120301).
of	string	no	Output format. Possible values: - json (default) - tab - csv
projects	integer	no	From which specific project(s) should data be retrieved. If there are several items, they should be separated with commas. (e.g: projects=53,64).
to	string	yes	Last date of the requested time period to be searched, in yyyymmdd format (e.g: from=20120331).
users	integer	no	Which user(s) data should be retrieved. If there are several items, they should be separated with commas. (e.g: users=13,14,15).

Response

Response object

Name	Type	Comment
auxid	integer	Auxiliary codes connected with breaks from work.
duration	integer	Time spent in 'state'.
name	integer	Name of the user.
numberid	integer	Record's identifier for the caller.
prevstate	string	Previous status.
prevtime	integer	Timestamp when entering 'prevstate', in 'YYYY-MM-DD hh:mm:ss' format.
projectid	integer	Project identifier.
state	string	Actual status. Possible values: - OFFLINE - UNAVAILABLE - AVAILABLE - PREWORK - WAITING4CALL - WAITING4RECALL - RINGING - CALL - HOLD - AFTERWORK - AUX
time	integer	Timestamp when entering 'state', in 'YYYY-MM-DD hh:mm:ss' format.
userid	integer	User identifier.
username	integer	User's username.

Example

List the states of Robert Green (userid=6) on 20.02.2013.

https://mycc.asp.virtual-call-center.eu/v2/statistics/userstate?from=20130220&to=20130220&users=6

Response body

```
{
    "response": [
        {
            "userid": 6,
            "prevstate": "OFFLINE",
            "state": "UNAVAILABLE",
            "prevtime": "2013-02-20 15:40:06",
            "time": "2013-02-20 15:40:06",
            "projectid": 0,
            "numberid": 0,
            "auxid": null,
            "duration": 0,
            "username": "green_op",
            "name": "Robert Green"
        },
        {
            "userid": 6,
            "prevstate": "UNAVAILABLE",
            "state": "AVAILABLE",
            "prevtime": "2013-02-20 15:40:06",
            "time": "2013-02-20 15:40:11",
            "projectid": 0,
            "numberid": 0,
            "auxid": null,
            "duration": 5.75,
            "username": "green_op",
            "name": "Robert Green"
        },
        {
            "userid": 6,
            "prevstate": "AVAILABLE",
            "state": "AUX",
            "prevtime": "2013-02-20 15:40:11",
            "time": "2013-02-20 15:40:19",
            "projectid": 1,
            "numberid": 0,
            "auxid": null,
            "duration": 7.44,
            "username": "green_op",
            "name": "Robert Green"
        },
        {
            "userid": 6,
            "prevstate": "AUX",
            "state": "AVAILABLE",
            "prevtime": "2013-02-20 15:40:19",
            "time": "2013-02-20 15:41:05",
            "projectid": 1,
            "numberid": 0,
            "auxid": "lunch",
            "duration": 46.4,
            "username": "green_op",
            "name": "Robert Green"
        },
        {
            "userid": 6,
            "prevstate": "AVAILABLE",
            "state": "WAITING4CALL",
            "prevtime": "2013-02-20 15:41:05",
            "time": "2013-02-20 15:41:12",
            "projectid": 1,
            "numberid": 0,
            "auxid": null,
            "duration": 6.96,
            "username": "green_op",
            "name": "Robert Green"
        },
        {
            "userid": 6,
            "prevstate": "WAITING4CALL",
            "state": "RINGING",
            "prevtime": "2013-02-20 15:41:12",
            "time": "2013-02-20 15:41:17",
            "projectid": 1,
            "numberid": 2,
            "auxid": null,
            "duration": 4.57,
            "username": "green_op",
            "name": "Robert Green"
        },
        {
            "userid": 6,
            "prevstate": "RINGING",
            "state": "AFTERWORK",
            "prevtime": "2013-02-20 15:41:17",
            "time": "2013-02-20 15:41:25",
            "projectid": 1,
            "numberid": 2,
            "auxid": null,
            "duration": 7.88,
            "username": "green_op",
            "name": "Robert Green"
        }
    ],
    "errors": []
}
```

Number of available agents in a queue

Get number of online and available agents in a specific queue.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/queues/[queueid]/availableusers
Options	N/A
Body	N/A
Response	
Body	Number of online and available agents, encoded in JSON.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
queueid	integer	yes	Queue identifier.

Response

Response object

Name	Type	Comment
available	integer	Number of available agents in a specific queue.
online	integer	Number of online agents in a specific queue.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/queues/44/availableusers

```
{
    "response": {
        "online": 0,
        "available": 0
    },
    "errors": []
}
```

Users

List users

List all users in VCC's database.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/users
Options	N/A
Body	N/A
Response	
Body	Array of users, encoded in JSON

Request

Resource parameters

Name	**Type**	**Mandatory**	**Comment**
customer	string	yes	Your call centre's unique identifier (subdomain).

Response

Response object

Name	Type	Comment
extension	string	The user's extension.
group_name	string	User's group, as defined by the supervisor.
groupid	integer	User's groupid.
hourly	integer	Agent's hourly wage.
name	string	Name of the user.
status	string	User status, defining whether the user can log in or not. Possibe values: - active - inactive
teams_name	string	Name of the user's team, as defined by the supervisor.
userid	integer	The user's unique identifier in VCC's database.
username	string	The user's username.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/users

```
{
    "response": [
        {
            "status": "active",
            "name": "Peter Green",
            "username": "peter",
            "extension": "4",
            "userid": 4,
            "teams_name": "default_team",
            "groupid": 2,
            "group_name": "admin"
        },
        {
            "status": "active",
            "name": "Michaela Cooper",
            "username": "michaela_supervisor",
            "extension": "5",
            "userid": 5,
            "teams_name": "default_team, Google_external_team",
            "groupid": 3,
            "group_name": "supervisor"
        },
        {
            "status": "active",
            "name": "Alexandra",
            "username": "g_alexandra",
            "extension": "5",
            "userid": 5,
            "hourly": 1200,
            "teams_name": "default_team, Google_external_team",
            "groupid": 4,
            "group_name": "operator"
        }
    ],
    "errors": []
}
```

Add new user

Add a new user to VCC's database.

Request	
Method	PUT
Resource	https://[customer].asp.virtual-call-center.eu/v2/users
Options	N/A
Body	New user's details, encoded in JSON
Response	
Body	Result encoded in JSON

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).

Request object

Name	Type	Mandatory	Comment
is_agent	boolean	yes	Defines the user's type. Possible values: - true - false
name	string	yes	User's name.
password	string	yes	The user's password, created within the parameters of the previously set password policy. The data added is encrypted.
role_id	integer	yes	Defines the user's right.
username	string	yes	The user's username.

Response

See examples below.

Example

Request

URL: https://mycc.asp.virtual-call-center.eu/v2/users

Request body

```
{
    "name": "Joseph Landry",
    "username": "joseph",
    "password": "LKU76.GFTR#23",
    "is_agent": true,
    "role_id": 4
}
```

Response body

```
{
    "response": 33,
    "errors": []
}
```

Possible errors

HTTP Code	Message	Description
417	error.name_must_be_between	The name is too short or long.
417	error.non_exist	The specified role does not exist in VCC's database.
417	error.password_policy_min_chars_error	The password added is too short (based on the password policy previously set in VCC).
417	error.password_policy_min_number_chars_error	The password added contains less than the minimum number of numbers required (based on the password policy previously set in VCC).
417	error.password_policy_min_special_chars_error	The password added contains less than the minimum number of special characters required (based on the password policy previously set in VCC).

HTTP Code	Message	Description
417	error.password_policy_min_uppercase_chars_error	The password added contains less than the minimum number of uppercase letters required (based on the password policy previously set in VCC).
417	error.required	Missing value on a mandatory key.
417	error.reserved_username_prefix	The username starts with a reserved username prefix (vcc_).
417	error.reserved	The added role is reserved for VCC.
417	error.role_mismatch	The value on the [is_agent] or the [role] key is not supported.
417	error.username_already_exists	The username already exists in VCC's database.
417	error.username_must_be_between	The username is too short or long.
417	error.wrong_value_format	The value format on the [is_agent] or the [role] key is not supported.
417	error.wrong_name_format	The name contains one or more non-supported characters eg: special character.
417	error.wrong_username_format	The username contains one or more non-supported characters eg: special character.
417	error.wrong_username_only_consists_small_letters	The username contains one or more uppercase letters.

List teams

List all teams which currently exist in VCC.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/teams
Options	N/A
Body	N/A
Response	
Body	Array of teams, encoded in JSON

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).

Response

Response object

Name	Type	Comment
name	string	Team name.
teamid	integer	The team's unique identifier in VCC's database.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/teams

```
{
    "response": [
        {
            "teamid": 1,
            "name": "default_team"
        },
        {
            "teamid": 2,
            "name": "inhouse_cc"
        }
    ],
    "errors": []
}
```

List all roles

List all roles which currently exist in VCC.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/roles
Options	N/A
Body	N/A
Response	
Body	Array of roles, encoded in JSON

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).

Response

Response object

Name	Type	Comment
id	integer	The role's unique identifier in VCC's database.
is_agent	boolean	Determine the role type. Possible values: - true - false
name	string	Role name.

Example

URL: https://mycc.asp.virtual-call-center.eu/v2/roles

```
{
    "response": [
        {
            "id": 2,
            "name": "admin",
            "is_agent": false
        },
        {
            "id": 3,
            "name": "supervisor",
            "is_agent": false
        },
        {
            "id": 4,
            "name": "operator",
            "is_agent": true
        },
        {
            "id": 5,
            "name": "visitor",
            "is_agent": false
        }
    ],
    "errors": []
}
```

Archiver

Get the year and month when no voicefiles were archived

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/[files]/
Options	
Body	N/A
Response	
Body	Array of years and months.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
files	string	yes	Possible values: - voicefiles - mobilevoicefiles

Response

Response object

Array of Month objects.

Month object

Name	Type	Comment
year	string	Year
month	string	Month

Example

List all months when no voicefiles were archived yet.

Request

https://mycc.asp.virtual-call-center.eu/v2/archiver/voicefiles/

Response body

```
{
    "response": [
    {
        "year": "2016",
        "month": "01"
    },
    {
        "year": "2016",
        "month": "02"
    },
    {
        "year": "2016",
        "month": "03"
    },
    {
        "year": "2016",
        "month": "04"
    },
    {
        "year": "2016",
        "month": "05"
    },
    {
        "year": "2016",
        "month": "06"
    },
    {
        "year": "2016",
        "month": "07"
    },
    {
        "year": "2016",
        "month": "08"
    },
    {
        "year": "2016",
        "month": "09"
    }
    ],
    "errors": []
}
```

List all downloadable voice files in a given month

This resource provides a maximum of 10 elements in its response.

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/[files]/[year]/[month]?lastUuid=[uuid]
Options	
Body	N/A
Response	
Body	Array of years and months.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
files	string	yes	Possible values: - voicefiles - mobilevoicefiles
year	integer	yes	Year
month	integer	yes	Month within the requested year. Possible values: - 01-12
uuid	string	no	Unique call identifier (Universally Unique ID). - Uuid needs to be empty at first request. - Any following requests the previous response's last uuid needs to be set.

Response

Response object

Array of UUID, year, month and name objects.

Name	Type	Comment
UUID	string	Unique call identifier (Universally Unique ID).
year	string	Year
month	string	Month
name	string	Name of the voicefile

Example

Request a 10 element list of downloadable files from 2016.12.

Request

https://mycc.asp.virtual-call-center.eu/v2/voicefiles/2016/12

Response body

```
{
    "response": [
    {
        "uuid": "f1340322-bc5b-11e6-bca8-57db46f903da",
        "year": "2016",
        "month": "12",
        "name": "2016-12-07-10-02_Peter_Green_36159996400_36201234567_.mp4"
    },
    {
        "uuid": "512ffeb4-c113-11e6-b0d0-1b480c92fadc",
        "year": "2016",
        "month": "12",
        "name": "2016-12-13-10-05_Peter_Green_3619997400_36301234567_Busy.mp4"
    },
    {
        "uuid": "ddb213fc-bc5b-11e6-bc7f-57db46f203da",
        "year": "2016",
        "month": "12",
        "name": "2016-12-07-10-02_Peter_Green_3619997400_36201234567_.mp4"
    }
    ],
    "errors": []
}
```

Set a successfully downloaded voicefile as archived

Every downloaded voicefile needs to be checked using MD5 checksum method. The MD5 checksum needs to be sent to the server to set a voicefile as archived.

Important: If this step is skipped or invalid checksum is provided, the given voicefile will be downloaded again and again during the archiving cycle.

Request	
Method	PUT
Resource	https://[customer].asp.virtual-call-center.eu/v2/[files]/[yyyy]/[mm]/[uuid]
Options	
Body	Customised data, encoded in JSON.
Response	
Body	Shows whether the request has been successful or not.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
files	string	yes	Possible values: - voicefiles - mobilevoicefiles
year	integer	yes	Year
month	integer	yes	Month within the requested year. Possible values: - 01-12
uuid	string	yes	Unique call identifier.

Request object

Name	Type	Mandatory	Comment
checksum	string	yes	CheckSum number

Response

Response object

Name	Type	Comment
response	boolean	States whether request is successful or not. Possible values: - true - false

Example

Set the voice file with the following uuid as archived.

Request

URL: https://mycc.asp.virtual-call-center.eu/v2/voicefiles/2015/07/0cb7c880-3770-11e5-b81a-cb51ae8ad242

Request body

```
{
    "checkSum": "65d86872be001e0c988e689fbd433c85"
}
```

Response body

```
{
    "response": true,
    "errors": []
}
```

Download voice file

Request	
Method	GET
Resource	https://[customer].asp.virtual-call-center.eu/v2/[files]/[yyyy]/[mm]/[uuid]
Options	
Body	
Response	
Body	The current voice file.

Request

Resource parameters

Name	Type	Mandatory	Comment
customer	string	yes	Your call centre's unique identifier (subdomain).
files	string	yes	Possible values: - voicefiles - mobilevoicefiles
year	integer	yes	Year
month	integer	yes	Month within the requested year. Possible values: - 01-12
uuid	string	no	Unique call identifier (Universally Unique ID). - Uuid needs to be empty at first request. - Any following requests the previous response's last uuid needs to be set.

Response

The requested voicefile itself.

Example

Download voice file with the given uuid.

Request

https://mycc.asp.virtual-call-center.eu/v2/archiver/voicefile/2016/12/95bd5f46-44e0-11e5-a6da-6547f8762750

Response

The voicefile itself as a binary file is saved.

Webhook

Former name: Callback API. Get real-time automated information when an event occurs.

Overview

About Webhooks

Webhooks (formerly Callback API) push information via HTTP(S) requests from the VCC Live system to your server when specific events occur.

Webhook Events

Event	Description
Call Disposition (Async)	A record is allocated a disposition.
IVR (Sync)	An IVR interaction.
Mobile Dispositon (Async)	A record is allocated a disposition via VCC Live App.
Payment Transaction (Async)	A successful or unsuccessful payment occurs.
Project Login (Sync)	A user logs in.
User Created (Async)	A user is created.
User Modified (Async)	A user's details are modified.

Processing Webhook Requests

Any pogramming language supporting JSON (eg. PHP, Java, Ruby, C, Javascript, etc.) can handle Webhook requests. Webhook requests are HTTP(S) messages, so a webserver required.

A PHP script sample that saves Webhook requests in a temp file:

```
<?php

$body = file_get_contents('php://input');

// If you enable encrpytion you shoud enable next few lines
// $cipherMethod = 'aes-256-ctr';
// $secretKey = 'very secret key';
// $iv = hex2bin($_GET['iv']);
// $body = openssl_decrypt($body, $cipherMethod, $secretKey, 0, $iv);

file_put_contents('/tmp/webhook.log', $body.PHP_EOL.PHP_EOL, FILE_APPEND);
```

Firewall Settings

source	destination(1)	port	protocol
data center IP range	local IP	443/TCP	HTTPS

(1) Use an IP address in your firewall configuration instead of domain names.

Data Center IP Ranges

hosting	IP range
hu1	194.38.106.64/26
hu2	193.68.62.192/26
ke1	62.12.118.64/27
au1	108.61.213.28/32

Example of Using hu1 Hosting

source	destination	port
194.38.106.64/26	local IP	443/TCP

Handling Responses

Your server needs to response to requests with an appropriate HTTP response code. A code may differ, depending on whether the request is successful or fails to be proceeded.

Successful Requests

Use HTTP response code '200 OK' to indicate a successful job delivery and stop Webhook attempting to forward the request again.

Failed Requests

Use an appropriate 4xx or 5xx HTTP response code depending on the type of the error. The Webhook attempts to deliver the failed request until it succeeds or reaches the limit.

Note: The Webhook attempts to deliver the failed limit 15 times before dropping it, using an increasing delay between attempts (1, 2, 4, 8, etc minutes). Unsuccessful requests are stored in an error queue.

Securing Requests

You can take a few extra steps to prevent malicious developers accessing your requests.

Use HTTPS

Use (https://) instead of (http://) in your URL to ensure a more secure communication channel.

Set Up Your Firewall

Use the required firewall settings. See Firewall Settings paragraph.

Use a Token or Secure Key

In the URL, add a secret key to all requests received via a Webhook, for example: https://your-url/resource?secret-key=xxxxxxxxxxxxxx. When your server receives a request, but the authorization fails, send a response back with the HTTP response code *'401 Unauthorized'*.

Enable HTTP Body Encryption

You can enable encryption in the HTTP body. Many cipher methods are available.

For decryption, you need:

- the **HTTP body** (base64 encoded if encryption is enabled)
- the selected **cipher method**
- the **secret key**
- the **iv** (if you enabled random initialization vector)

Tip: An example decryption code is available in the Processing Webhook Requests section.

Note: If you enable encryption, you must set up a secret key for encryption and decryption.

Use Random Initialization Vector

Random initialization vector is a commonly used technique. To use this technique, use the "iv" URL parameter. See To Use Dynamic URLs section.

Example URL: https://your-url/resource?iv=${iv}

Note: If you disable it, an empty initialization vector is used instead.

Setting Up Webhook

You can set up global and project-specific webhooks via the VCC Live Desk interface.

To Set up a Project-Specific Webhook

1. Select a project from the project list, then select the **Webhook** tab.
2. Press

   ➕

   to add new.
3. Select an option from the **Event** drop-down list. The event you select will trigger the webhook.

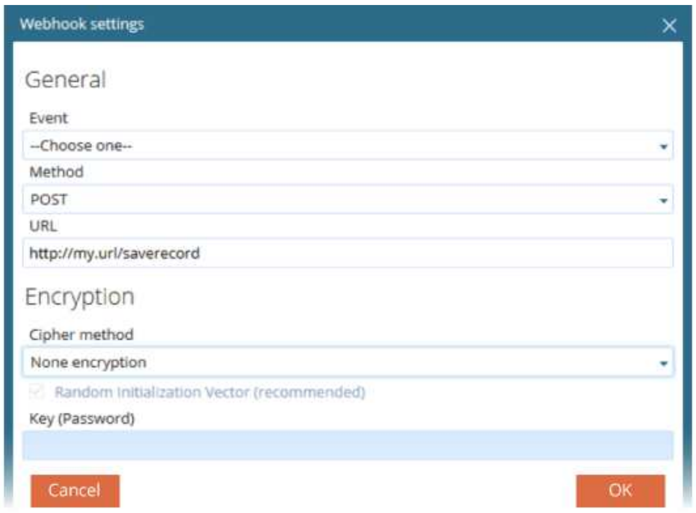
4. Select an option from the **Method** drop-down list.
5. Enter an address in the **URL** text field.
6. Select an encryption method, then enter a password.
7. Press **OK** to save changes.

Tip: you can add more than one setting in a project to utilize more than one URL.

Note: we recommend using HTTPS instead of HTTP.

To Set Up a Global Webhook

1. From the **VCC Live menu**, select **Contact Center** > **Global settings**, then select the **Webhook** tab.
2. Follow the instructions from step 2 in the **To Set up a Project-Specific Webhook** section.



Event: --Choose one--
Method: POST
URL: http://my.url/saverecord
Cipher method: None encryption
Random Initialization Vector (recommended)
Key (Password)

To Use Dynamic URLs

You can create dynamic URLs using parameters. Required format is ${varname}. URL parameters:

Parameter	Description
projectid	Project identifier.
numberid	Record identifier.
iv	Random initialization vector in hexadecimal format, used for encryption.

Example: http://my.url/saverecord?projectid=${projectid}&numberid=${numberid}&iv=${iv}

To Test a Webhook Request

RequestBin helps you debug HTTP requests by providing you an URL that collects requests.

1. Visit http://requestb.in.
2. Press **Create a RequestBin**.
3. Copy the URL to your clipboard.
4. Paste the URL into URL textbox in the Webhook settings.
5. Make a phone call and set a disposition.
6. Check your bin at http://requestb.in to analyse the request.

Note: The last 20 requests are stored for 48 hours.

Async Webhook Requests

When one or more events occur, VCC Live can send the events' details in the background to your server via asynchronous Webhook requests. A task with all relevant request details is created and is put into a queue.

Note: An asynchronous request lets the VCC Live Desk application continue to work without interruption while your request is being handled. This means that you can keep your system up-to-date without needing to make the application unresponsive.

Handling Responses

Read about responses in the Handling Responses section.

Note: Job congestion happens if requests are not stored and/or processed as quickly as events are generated in the VCC Live system.

Note: The first attempt of delivering a job usually takes a few seconds only, but may take up to a few minutes.

To Test Sync Webhook Requests

Read about the process in the To Test a Webhook Request section.

Call Disposition (Async)

When a disposition is set, the following object is sent to your site, encoded in JSON.

Request Object

Name	Type	Comment
agent_description	string	Comment added by the agent.
client_data	object	Custom data object, including contacts.
closed	integer	Shows that the record is closed or the PPD dialler is going to call it later. Possible values: - 1 - 0
create_time	string	Time of disposition setting in yyyy-mm-dd hh-mm-ss format.
destination	string	Called party's phone number.
direction	string	Direction of the call that is terminated with the disposition: - in - out
disposition	object	Disposition object.
numberid	integer	Record identifier.
projectid	int	Project identifier.
source	string	Calling party's phone number.
sum_attempted_calls	integer	Total number of call attempts.
teamids	array of integers	Array of teamids.
timestamp	string	Time of disposition setting in ISO 8601 format.
userid	integer	Agent identifier.
uuid	string	Call identifier.

Disposition object

Name	Type	Comment
assesment	string	Disposition assessment: - success: call recipient reached - ordered: call recipient reached and call goal achieved - failed: call recipient not reached

Name	Type	Comment
callback	integer	Disposition Webhook attribute. Possible values: - 1 - 0
comission_collector	integer	Disposition comission collection possibility attribute. Possible values: - 0 - 1
comment_mode	string	Defines the possibility for agents to write comments. Possible values: - allow - force - disable
commission	integer	Call centre's commission, as defined by supervisor.
default	string	Disposition system attribute. Possible values: - yes - no
description	string	Disposition comment, as provided by supervisor.
hide_history	integer	Disposition hide possibility attribute. Possbile values: - 0 - 1
id	integer	Disposition identifier in VCC's database.
instant	string	Disposition auto-save possibility attribute. Possible values: - yes - no
label	string	Disposition export value, as provided by supervisor.
mode	string	Disposition status. Possible values: - active - inactive - deleted
name	string	Disposition name.
phone_id	integer	Phone's identifier.
price	integer	Agen's commission, as defined by supervisor.

Name	Type	Comment
quota	string	Disposition quota attribute. Possible values: - yes - no
recall	integer	The default call-back time, in seconds, as defined by the supervisor for the given disposition. It is set only if the status is 'recall' or 'shared_recall'.
status	string	Disposition type, specifying the actual status of the record. Possible values: see on the link (status key).
try_before_reach	integer	Number of call attempts before contact is reached.

Sample Request Body

```
{
    "disposition": {
        "name": "Successful",
        "assesment": "ordered",
        "status": "finished",
        "price": 0,
        "commission": 0,
        "description": "",
        "label": "",
        "recall": 0,
        "id": 14,
        "mode": "active",
        "quota": "no",
        "instant": "no",
        "default": "no",
        "commission_collector": 0,
        "hide_history": 0,
        "comment_mode": "allow",
        "callback": 1,
        "mobileEnabled": 0
    },
    "agent_description": "test",
    "sum_attempted_calls": 1,
    "direction": "out",
    "source": "3617777777",
    "destination": "1234",
    "create_time": "2015-11-12 10:32:26",
    "uuid": "a7587a1a-2581-4dfe-b9eb-6a8ddb7caf0e",
    "projectid": 4,
    "numberid": 25,
    "userid": 26,
    "teamids": [
        1
    ],
    "client_data": {
        "email": "elemer.erdosi@vcc.live",
        "scenario": [
            {
                "valueid": 3,
                "fieldid": 7,
                "label": "Success",
                "export_value": "",
                "description": "1"
            }
        ],
        "name": "Elemer Erdosi",
        "amount": "100000",
        "pci_currency": [
            {
                "valueid": 8,
                "fieldid": 3,
                "label": "huf",
                "export_value": null,
                "description": ""
            }
        ],
        "phone1": "36301234567",
        "termekek": [
            {
                "valueid": 12,
                "fieldid": 10,
                "label": "Ticket",
                "export_value": "100000",
                "description": ""
            }
        ],
        "contacts": {
            "1": {
                "name": null,
                "phone": "36301234567",
                "email": null,
                "title": null
            }
        }
    },
    "timestamp": "2015-11-12T10:32:26+01:00"
}
```

Mobile Disposition (Async)

When a disposition is set via VCC Live App, the following object is sent to your site, encoded in JSON.

Request Object

Name	Type	Comment
agent_description	string	Comment added by the agent.
client_data	object	Custom data object, including contacts.
create_time	string	Time of disposition setting in yyyy-mm-dd hh-mm-ss format.
description	string	Comment added by the supervisor.
disposition	object	Disposition object.
numberid	integer	Record identifier.
projectid	int	Project identifier.
teamids	array of integers	Array of teamids.
timestamp	string	Time of disposition setting in ISO 8601 format.
userid	integer	Agent identifier.
uuid	string	Call identifier.

Disposition object

Name	Type	Comment
assesment	string	Disposition assessment: - success: call recipient reached - ordered: call recipient reached and call goal achieved - failed: call recipient not reached
callback	integer	Disposition Webhook attribute. Possible values: - 1 - 0
comission_collector	integer	Disposition comission collection possibility attribute. Possible values: - 0 - 1
comment_mode	string	Defines the possibility for agents to write comments. Possible values: - allow - force - disable
commission	integer	Call centre's commission, as defined by supervisor.

Name	Type	Comment
default	string	Disposition system attribute. Possible values: - yes - no
description	string	Disposition comment, as provided by supervisor.
hide_history	integer	Disposition hide possibility attribute. Possbile values: - 0 - 1
id	integer	Disposition identifier in VCC's database.
instant	string	Disposition auto-save possibility attribute. Possible values: - yes - no
label	string	Disposition export value, as provided by supervisor.
mobileEnabled	integer	Disposition mobile Application attribute. Possible values: - 1 - 0
mode	string	Disposition status. Possible values: - active - inactive - deleted
name	string	Disposition name.
price	integer	Agent's commission, as defined by supervisor.
quota	string	Disposition quota attribute. Possible values: - yes - no
recall	integer	The default call-back time, in seconds, as defined by the supervisor for the given disposition. It is set only if the status is 'recall' or'shared_recall'.
status	string	Disposition type, specifying the actual status of the record. Possible values: see on the link (status key).

Sample Request Body

```
{
 "disposition": {
  "name": "customer satisfied",
  "assesment": "failed",
  "status": "ordered",
  "price": 10,
  "commission": 110,
  "description": "",
  "label": "",
  "recall": 0,
  "id": 1,
  "mode": "active",
  "quota": "no",
  "instant": "no",
  "default": "yes",
  "commission_collector": 0,
  "hide_history": 0,
  "comment_mode": "allow",
  "callback": 1,
  "mobileEnabled": 1
 },
  "agent_description": "successfull order",
  "description": "",
  "create_time": "2016-06-07 12:47:20",
  "uuid": "ab22cbc8-ebe8-440c-b276-e50b72862897",
  "projectid": 2,
  "numberid": 503,
  "userid": 81,
  "teamids": [
  2,
  1
  ],
  "client_data": {
   "name": "Peter Green",
   "phone1": "3619996400",
   "contacts": {
   "1": {
   "name": null,
   "phone": "3619996400",
   "email": null,
   "title": null
   }
  }
 },
  "timestamp": "2016-06-07T12:47:20+02:00"
}
```

Payment Transaction (Async)

When either a successful or an unsuccessful payment occurs, the following object is sent to your site, encoded in JSON.

Request Object

Name	Type	Comment
client_data	object	Custom data object, including contacts.
in_call	boolean	True if the payment happened during the phone call. Payment can happen after the call if the payment process takes longer than the conversation.
numberid	integer	Record identifier.
payment	object	Payment object.
projectid	int	Project identifier.
teamids	array of integers	Array of teamids.
timestamp	string	Time of disposition setting, in ISO 8601 format.
userid	integer	Agent identifier.
uuid	string	Call identifier.

Payment Object

Name	Type	Comment
amount	number	Amount of payment.
card_expiration	sting	Expiration date (MMYY).
card_number_length	number	Card number's lenght.
card_number_trailing	string	Card number's last 4 characters.
currency	string	Currency of payment in lowercase ISO 4217 format.
payment_gw	string	The payment gateway's identifier.
payment_type	string	Type of the payment. Currently credit card is the only supported value.
status	string	Status of the payment: - succesful - unsuccesful
status_message	string	Status message if there is any.
token	string	Unique payment identifier generated by VCC. This token is used for recurring payment.
transactionid	string	A unique number, that represents the payment.
vposid	string	Virtual POS identifier.

Sample Request Body

```
{
    "payment": {
        "payment_type": "creditcard",
        "card_number_trailing": "1234",
        "card_number_length": 16,
        "card_expiration": "1218",
        "payment_gw": "testgw",
        "transactionid": "431fba930c344aa0b6d64458824036d1",
        "amount": 100000,
        "currency": "huf",
        "status": "successful",
        "status_message": "cde.success",
        "vposid": null,
        "token": null,

    },
    "in_call": true,
    "uuid": "a7587a1a-2581-4dfe-b9eb-6a8ddb7caf0e",
    "projectid": 4,
    "numberid": 25,
    "userid": 26,
    "teamids": [
        1
    ],
    "client_data": {
        "email": "elemer.erdosi@vcc.live",
        "scenario": [
            {
                "valueid": 3,
                "fieldid": 7,
                "label": "Success",
                "export_value": "",
                "description": "1"
            }
        ],
        "name": "Elemer Erdosi",
        "amount": "100000",
        "pci_currency": [
            {
                "valueid": 8,
                "fieldid": 3,
                "label": "huf",
                "export_value": null,
                "description": ""
            }
        ],
        "phone1": "36301234567",
        "termekek": [
            {
                "valueid": 12,
                "fieldid": 10,
                "label": "Ticket",
                "export_value": "100000",
                "description": ""
            }
        ],
        "contacts": {
            "1": {
                "name": null,
                "phone": "36304739238",
                "email": null,
                "title": null
            }
        }
    },
    "timestamp": "2015-11-12T10:31:48+01:00"
}
```

Sync Webhook Requests

When an event occurs that you expect an immediate response for, VCC Live can send the events' details to your server via synchronous Webhook requests.

> **Note:** A synchronous request blocks the VCC Live Desk application while it's waiting for a response. This means that in order to keep your system up-to-date, the application may become unresponsive until it gets a result.

Handling Responses

Read about responses in the Handling Responses section.

> **Warning:** To avoid usability and timeout issues, an immediate response should be sent to the VCC Live system, as it is blocked until it receives a respond.

To Test Sync Webhook Requests

Read about the process in the To Test a Webhook Request section.

IVR (Sync)

IVR Webhook (formerly known as IVR API) helps you create an agent-free, highly-customized, interactive environment between your customer and company, based on your CRM or ERP system. Callers can leverage IVR menus using their phone keypads as DTMF codes. They are then transferred as HTTP(S) requests to your ERP or CRM system, which needs to be replied to based on your business logic.

Typical uses:

- caller authentication using caller number and/or PIN
- personalised interactive voice menu
- balance confirmation read-back (TTS)
- utilities consumption measurements

Variables

You can use system and user variables in the URL and request body.

System Variables

variable name	description
customer	Customer identifier
projectid	Project identifier
source	Caller phone number
destination	Called phone number
uuid	Call's unique identifier
vccsys_shortid	Call's short identifier

User Variables

icon	IVR action's name	description
	Get digits and playback	Any number typed in during a phone call. Customers can provide you with data, for example a customer identificaton number, using a keypad. You can also add a recorded voice file before the data is inputted. The data is assigned a variable that you can use in other processes.
HTTP	Transfering data via HTTP	You can refer to variables and forward the data to a specific system via APIs.
x=	Set variable	You can set variables.
	Query data	You can assign field values to variables.

Requests

A variable can be used in a HTTP(S) URL and body. You can refer to a variable as **${variable_name}**.

Note: HTTP **Body content** can only be used if **HTTP method** is set to POST or PUT.

HTTP method is GET/DELETE

URL: https://example.com?key1=${variable1}&key2=${variable2}&key3=string
Body content: <empty>

HTTP method is POST/PUT

Required format of the URL and **Body content** depends on the **Content type** you select.

Content type is form

URL: https://example.com
Body content: key1=${variable1}&key2=${variable2}&key3=string

Content type is JSON

URL: https://example.com
Body content:

```
{
   "key1": "${variable1}",
   "key2": "${variable2}",
   "key3": "string"
}
```

Response

An appropriate response should be sent by your system to set one or more variables, which may affect the IVR process, making the process dynamic, for example, a value can be read back to the customer. The response can define new variables and also override existing ones, except system variables. If successful, the HTTP response code should be 200 OK.

Response Requirements and Limits

- Connection timeout: 5 sec
- Response timeout: 5 sec
- Response key/value limit: max. 100
- Body content content size limit: 65536 byte
- Value length limit: 256 characters
- Key name: lowercase alphanumerics (a-z, 0-9 and low_dash, first character must be a letter or low_dash)

Content type is application/JSON or text/JSON

> Note: A JSON response can contain up to three-level depth objects (key/value pair). Values can be BOOLEAN, FLOAT, INT and STRING data.

Body content:

```
{
   "key1": "value",
   "key2": 123,
   ...
   "key100":"value"
}
```

Content type is text/x-ini

Body content:

```
key1=value1
key2=123
key3=true
...
key100=value100
```

Content type is text/plain (or anything else)

Body content: format should be a single number or string, that will be stored in the variable previously set in the **Set variable** text field.

Setting Up

VCC Live IVR Webhook can be set up by adding the **IVR Webhook** process to your inbound processes in **VCC Live Desk** .

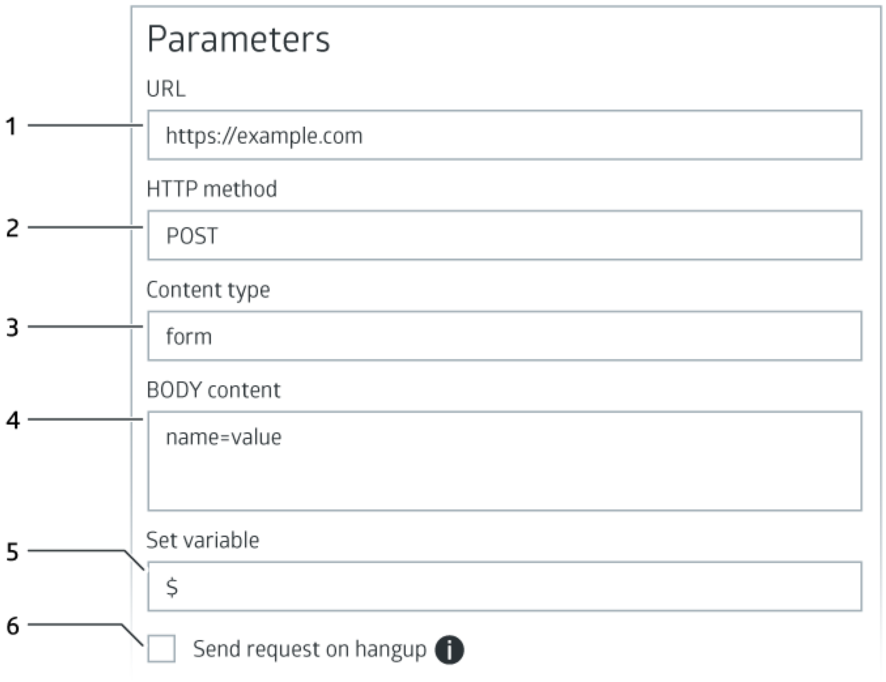
> **Note:** If you need assistance regarding the procedure, please contact our Support Team.

To Add a IVR Webhook Process

1. Select a project from the project list, then select **Channels** > **Voice** > **Inbound**.
2. On the **Actions** tab, select a timesheet or macro, then press **Actions**, and select **IVR Webhook**.
3. Select the item you just added. The properties are located on the right.
4. Set the desired parameters. See **IVR Webhook Process Parameters** section.
5. Press **Apply inbound changes**.

IVR Webhook Process Parameters

name	description
1. URL	Target URL you want to transfer data to.
2. HTTP method	Method of sending: POST, GET, PUT or DELETE.
3. Content type	Content type: form, json.
4. Body content	Body content, available for POST and PUT methods only.
5. Set variable	The variable that will store the received reply from the URL.
6. Send request when call ends	If enabled, data is sent on hang up, and the system disregards the response.


Parameters

URL

1 https://example.com

HTTP method

2 POST

Content type

3 form

BODY content

4 name=value

Set variable

5 $

6 Send request on hangup

IVR Webhook parameters" title="VCC Live **IVR Webhook** parameters illustration">

Note: the received response (stored as a variable, e.g.: vcc_variable) can be used to transfer the call for specific processes or can for a **Read-back** process. (e.g. balance read-back, customerid read-back).

Warning: We strongly recommend using HTTPS instead of HTTP.

Project Login (Sync)

Project Login Webhook makes it easy to add an extra security level by getting real-time information from another system if an agent has the appropriate rights to log in to a specific project. The success is either "true" or "false", encoded in JSON.

In the response JSON object, each project indicated with a projectID needs to be true if the access is granted to an agent, otherwise it is false. If an agent has no access right to at least one project (that is: "false" answer arrived), the system does not allow the agent to log in, and the denied projects will be shown in a pop-up window.

Note: If you set the request method to GET, Webhook parameters are sent in HTTP query parameters. In all other cases (POST or PUT), Webhook parameters are sent in an HTTP body.

Warning: If server side is not well prepared, agents' login is rejected.

Request Object

Name	Type	Comment
customData	object	Agent custom data object.
groupId	integer	Agent group identifier.
projectId	integer	Project identifier.
secondaryProjectIds	array of integers	Secondary project identifier(s).
userId	integer	Agent identifier.
username	string	Agent username.

Sample Request Body

```
{
  "projectId": 1,
  "userId": 2,
  "username": "x",
  "groupId": 2,
  "customData": [],
  "secondaryProjectIds": [2, 3, 4, 5]
}
```

Sample Response Body

```
{
  "1": true,
  "2": true,
  "3": false,
  "4": true,
  "5": true
}
```

Script SDK

Add sophisticated business logic to your scripts.

Introduction

The VCC Live software solution provides a wide range of functionality, with a drag-and-drop visual script editor which allows users to easily create custom layouts as required. It is also very easy to build simple conditions to help handle script flow.

If you need to build more sophisticated scripts, VCC Live's JavaScript-based Script SDK easily allows you to change layouts, retrieve and store data in real time, calculate field values, hide or show controllers depending on previous answers, or create complex conditions for page jumping.

We recommend reading our Script SDK Tutorial before starting to work on sophisticated scripts. Further information on Script SDK-specific functions can be found on our Script SDK Reference page.

You can also find Script SDK examples on our Examples page.

User requirements:

- Basic JavaScript knowledge
- Text editor software (e.g.: Notepad, PSPad, Ultraedit)

Examples

Simple JS codes

Check mandatory field in before disposition set

```
$().beforeSetDisposition = function(disposition) {
 if (disposition.id == 11 && !vcc.getFieldValue("fieldName"))
 {
  alert('This field is mandatory!');
  return false;
 }
};
```

Transfer a call to an inbound process

```
$('oldal_name.button_name').afterSetData = function() {
 vcc.transfer('processid', true);
};
```

Transfer a call to a phone number

```
$('page', 'transfer_phone_button').afterSetData = function() {
 vcc.transfer('3670333444');
};
```

Complex JS codes

Pinning the operators name into a text field

```
$('adatlap').onLoad = function()  {
 var operatorUsername;
 var operatorUsername = vcc.getScriptVariable('agent.name');
 var operators = vcc.getFieldValues('agent_help');
 operators.forEach(function(item) {
  if (item.label === operatorUsername) {
   vcc.setFieldValue('agent_name', item.label);
   vcc.setFieldValue('agent_id', item.export_value);
  }
 });
};
```

Sending an HTTP (AJAX) request

```
// Get required informations on page onload event using AJAX
$('sample_page').onLoad = function() {
 // Create a request object to handle connection
 var request = vcc.httpRequest();

 // We want to send a request to the following url
 request.open('GET', 'http://www.google.com/robots.txt');

 // JavaScript is asynchronous, so we have to define
 // the onload callback before sending the request
 request.onload = function(e) {
  // just dumps the result to the console (your code goes here)
  dump(request.responseText);
 };

 // Send the HTTP request to the URL specified above
 request.send();
};
```

Constants

CONTROLLER_HELPERS

Available values of the assesment property of a disposition.

Type	Comment
vcc.CONTROLLER_ERROR	for showing error
vcc.CONTROLLER_INFO	for showing information
vcc.CONTROLLER_WARNING	for showing warning

DISPOSITION_ASSESMENTS

Available values of the assesment property of a disposition.

Type	Comment
vcc.DISPOSITION_FAILED	failed disposition
vcc.DISPOSITION_ORDERED	ordered disposition
vcc.DISPOSITION_SUCCESS	success disposition

FIELD_QUOTAS

Available values of the quota property of a field.

Type	Comment
vcc.FIELD_QUOTA_CELL	cell quoted field
vcc.FIELD_QUOTA_EDGE	edge quoted field
vcc.FIELD_QUOTA_NONE	not quoted field

FIELD_TYPES

Available values of the type property of a field.

Type	Comment
vcc.FIELD_TYPE_MULTIPLE	multiple field
vcc.FIELD_TYPE_SIMPLE	simple field
vcc.FIELD_TYPE_TEXT	text field

Variables

fields

DEPRECATED, see: getFieldNames, getFieldProp.

isAgent

{Boolean} **true** if the current user is an agent.

Example

There is a special controller which can be seen by a supervisor only.

```
$('page').onLoad = function() {
     vcc.getController('page', 'special').hidden = vcc.isAgent;
};
```

isInbound

{Boolean} **true** if the current call is an inbound one.

Example

Show a special warning message for inbound calls.

```
$('page').onLoad = function() {
     vcc.getController('page', 'message').hidden = !vcc.isInbound;
};
```

isTest

{Boolean} **true** if we are in test mode at the moment.

Example

We want to show a special controller only in test mode. (E.g. a textbox and a button to jump to any page in the script)

```
$('page').onLoad = function() {
     vcc.getController('page', 'special').hidden = !vcc.isTest;
};
```

Functions

addVoicefileTag

Inserts tags during phone call to mark one or more parts of the voice recording that you wish to download separately.

Parameters

Name	Type	Description
tag name	{string}	Name of the voicefile chunk.
time correction (sec)	{int}	Adjusts the timing of tag placement. Example: If the correction is -3, then it means that if the agent executes tag placement at 1:10 during call, the system inserts the tag at 1:07.
Overwrite	{boolean}	-true, -false. If true: when tag placement is executed multiple times, tag is placed at the last attempt. If false: tag is placed at first attempt.

Returns

none

Example

```
$('voicefiletagging','tag1').afterSetData = function() {
  vcc.addVoicefileTag('firstpart',0, true);
}
$('voicefiletagging','tag2').afterSetData = function() {
  vcc.addVoicefileTag('secondpart',-3, true);
}
$('voicefiletagging','tag3').afterSetData = function() {
  vcc.addVoicefileTag('thirdpart',-1, true);
}
```

Downloading Voice File Extracts

For detailed description, see the Voice File Tagging Tutorial tutorial.

alert

Displays a dialog to the operator.

Description

```
vcc.alert(message: string, [title: string]): void
```

A popup window (modal dialog) appears containing a message.

Parameters

message
The message to be shown.

title
The title of the window. If not passed, the window will have no title.

Return values

void

Example

```
vcc.alert('Hello world');
```

clearDtmf

Clears the DTMF digits received during a call.

Description

```
vcc.clearDtmf(): void
```

You can save the DTMF data, then ask the client for another DTMF input.

Parameters

none

Return values

void

Example

After receiving a PIN code, we want to save the code, and then request the client's date of birth. We insert a button into the script that saves the PIN code.

```
vcc.setFieldValue('pin_code', vcc.getScriptVariable('global.dtmf'));
vcc.clearDtmf();
```

confirm

Displays a dialog to the operator with a yes-no question.

Description

```
vcc.confirm(message: string, [title: string]): boolean
```

A popup window (modal dialog) appears, containing a message and **Yes**/**No** buttons.

Parameters

message
The message to be shown.

title
The title of the window. If not passed, the window will have no title.

Return values

Returns **true** if the **Yes** button is pressed, **false** otherwise.

Example

We want to ask again if there are no orders.

```
// it will true or false
const answer = vcc.confirm('Everything was okay with the call?');
```

contains

Checks if a string contains another string.

Description

```
vcc.contains(haystack: string, needle: string): boolean
```

Searches the needle in the haystack.

Parameters

haystack
It looks for the needle within this string.

needle
It looks for this element within the haystack.

Return values

Returns **true** if the *needle* is found in the haystack, **false** otherwise.

Example

We have a comma-separated list of allowed postal codes, and we want to check the given one.

```
const allowedValues = '2034,2579,1069,5173';
if (!vcc.contains(allowedValues, '2034')) {
    vcc.alert('The post code is not allowed');
}
```

dump

Writes information to the console.

Description

```
dump(message: any): void
```

A very useful function during developement. It allows you to quickly check, for example, a condition or the result of a calculation.

Parameters

message
The object that we want to write, it can be anything.

Return values

void

Example

```
dump('Hello world');
```

getChannelVariable

Queries an IVR channel variable.

Description

```
vcc.getChannelVariable(name: string): string
```

Channel variables can be set via inbound settings. See Inbound Calls section.

Parameters

name
Name of the channel variable.

Return values

string - Value of the channel variable or **null** if there is no such variable.

Example

We want to store the DTMF that the client provided via IVR.

```
const ivr = vcc.getChannelVariable('code');
vcc.setFieldValue('ivr_code', ivr);
```

getCommission

Gets the commission of a field.

Description

```
vcc.getCommission(field: string): number
```

Parameters

field
The name of the field. If missing, it returns all fields' commission.

Return values

number
The commission.

Example

We want to display commission to the agent if the value is selected by the agent.

```
const commission = vcc.getCommission('amount');
vcc.setFieldValue('commission', commission);
vcc.getController('page', 'commission').refresh();
```

getController

Gets a specific control item of a page.

Description

```
vcc.getController(name: string, [page]: string): Controller
```

Searches **array** for **element** and returns *true* if successful.

See Types of Controls for a list of available controls.

Parameters

name
Name of control item.

page
The page name which contains the control item. If missing, the current page is used instead.

Return values

Controller - The control item or **null** if there is no such control item.

Example

We want to refresh the control in the *news* page.

```
vcc.getController('news', 'actual').refresh();
```

getCurrentPage

Gets the name of the currently-open script page.

Description

```
vcc.getCurrentPage(): string
```

Gets the name of the currently-open script page that is visible to the agent at the moment.

Parameters

none

Return values

string - The name of the page.

Example

```
const pageName = vcc.getCurrentPage();
dump(pageName);
```

getEdgeData

DEPRECATED, see: getQuotaFieldData

getEdgeValues

DEPRECATED, see: getQuotaFieldValues

getFieldItems

DEPRECATED, see: getFieldNames

getFieldNames

Get the names of the fields.

Parameters

none

Returns

{Array of String} the names of the fields

getFieldProp

Get the properties of a field.

Parameters

Name	Type	Description
name	{String}	the name of the field

Returns

{FIELD_PROPERTY_OBJECT}

getFieldValue

Get the selected value of a field.

Parameters

Name	Type	Description
field	{String}	the name of the field
[key]	{String}	a key of the field to return

Returns

Name	Type	Description
multiple field	{Array of FIELD_VALUE_OBJECT}	if there are no selected values, the array will be empty
simple field	{FIELD_VALUE_OBJECT}	the selected value or {String|Number} if key is specified and available or null if there is no selected object or key is unavailable
text field	{String}	the value of the field

Example

Do not allow to go the next page when the age is between 20 and 30 and there are no selected city.

```
$('page').onNext = function() {
     var age = vcc.getFieldValue('age');
     var city = vcc.getFieldValue('city');
     if (age >= 20 && age <= 30 && !city) {
         return false;
     }
};
```

getFieldValues

Get the available values of a single or multiple field.

Parameters

Name	Type	Description
name	{String}	the name of the field

Returns

{Array of FIELD_VALUE_OBJECT}

getGoogleAccessToken

Opens a new window to get user access from google to specified scope

Parameters

Name	Type	Description
callback	{Function}	will get the access_token in first argument
json	{object}	"Client ID for native application" from https://console.developers.google.com (Download JSON)
scope	{String}	

Example

Get information from google about operator's account

```
$('page').onLoad = function() {
     var json = {"installed":{...}};
     vcc.getGoogleAccessToken(json, 'email profile', function(accessToken) {
         var request = vcc.httpRequest();
         request.open('GET', 'https://www.googleapis.com/plus/v1/people/me');
         request.setRequestHeader('Content-Type', 'application/json');
         request.setRequestHeader('Authorization', 'Bearer '+accessToken);
         request.onload = function(e) {
             var user = JSON.parse(request.responseText);
             dump(user);
             vcc.setFieldValue('op_name', user.displayName);
             vcc.setFieldValue('op_email', user.email[0].value);
         };
         request.send();
     });
};
```

getPages

Get the script pages without the data pages.

Parameters

none

Returns

{Array of String} the page names

getPrice

Get the price of a field or of all fields. Parameters [field] - {String} the name of the field. If it is missing, it will return the summary of price of all fields.

Returns

{Number} the price

Example

Let's show the price to the agent if they select a value.

```
$('page', 'amount').afterSetData = function() {
    var price = vcc.getPrice('amount');
    vcc.setFieldValue('price', price);
    vcc.getController('page', 'price').refresh();
};
```

getQuotaFieldData

Get information of the quota values on a quoted field.

Parameters

Name	Type	Description
field	{String}	the name of the field

Returns

{Object} keys: the valueids of the values; values: { QUOTA_DATA_OBJECT}

Example

We would like sort the selectable values according to the count of remaining items in a quoted field.

```
$('edge_field').onLoadData = function(data) {
     var quotas = vcc.getQuotaFieldData('edge_field');
     return data.sort(function(a, b) {
         return quotas[a.valueid].diff_value - quotas[b.valueid].diff_value;
     });
};
```

getQuotaFieldValues

Get the selectable values for a quoted field.

Parameters

Name	Type	Description
field	{String}	the name of the field

Returns

{Object} keys: the valueid of the selectable values; values: {Boolean}, **true** if selectable.

Example

We want to show only the selectable values in a dropdown controller.

```
$('page1', 'quota_field').onLoadData = function(data) {
     var validValues = vcc.getQuotaFieldValues('quota_field');
     return data.filter(function(value) {
         return !!validValues[value.valueid];
     });
};
```

getScriptVariable

Get every variable what can be used in script text controller. For example {agent.name}.

Parameters

Name	Type	Description
name	{String}	the name of the variable

Returns

{String} the value of the variable or **null** if there is no surch a variable

Available variables

Name	Comment
<field>.commission	commission of the field
<field>.description	description of the field
<field>.exportValue	export value of the field
<field>.label	label of the field
<field>.price	price of the field
<field>.value	value of the field
agent.name	the name of the actual agent
agent.username	the username of the actual agent
destination	the destination phone number
global.ccphone	the phone number of the call center
global.clientphone	the phone number of the client
global.commission	summary of the field commissions
global.dtmf	the DTMF characters pressed during the conversation
global.ivr.<name>	the value of the <name> IVR variable
global.price	summary of the field prices
global.shortid	the short unique ID assigned to the call (unique for the customer)
global.uuid	the UUID
numberid	the unique ID of the client
project.id	the unique ID of the current project
project.name	the name of the current project
source	the source phone number

Example

Put the actual agent name into a field.

```
$().onLoad = function() {
     vcc.setFieldValue('agent', vcc.getScriptVariable('agent.name'));
};
```

getTabs

Get the script data pages without script pages.

Parameters

none

Returns

{Array of String} the page names

goNext

Jump to the next or the given page.

Parameters

Name	Type	Description
[nextPage]	{String}	the name of the next page to jump. If it is missing, it will do like when the user click on the next button of the script

Returns

none

Example

After selecting a value we want to jump to the next page immediately.

```
$('page', 'gender').afterSetData = function() {
     vcc.goNext();
};
```

goTab

Jump to the given data page.

Parameters

Name	Type	Description
[tabId]	{String}	the name of the data page to jump

Returns

none

Example

If the call coming with hidden id, we want load the authentication data page

```
$('data_page').onLoad = function() {
 if (!vcc.getScriptVariable('source')) {
     vcc.goTab('authentication');
 }
};
```

httpRequest

Creates an XMLHttpRequest Object

Returns

XMLHttpRequest Object

Example

see EXAMPLES

inArray

Check that the array contains the given element.

Description

```
vcc.inArray(array: any[], element: any): boolean
```

Searches **array** for **element** and returns true if successful.

Parameters

array
We look for the element within this array.

element
We look for this element within the array.

Return values

Returns **true** if the element is found in the array, **false** otherwise.

Example

In most situations using vcc.isSelected is a good solution, but suppose that we have a list of postal codes and we have to look for the given one.

```
var allowedValues = ['2034', '2579', '1069', '5173'];
vcc.alert(vcc.inArray(allowedValues, '1069') ? 'found' : 'not found');
vcc.alert(vcc.inArray(allowedValues, '2000') ? 'found' : 'not found');
```

inString

DEPRECATED, see contains.

isSelected

Check that the given value is selected in the field or not.

Parameters

Name	Type	Description
field	{String}	the name of the field
[key]	{String}	the name of the property of the value (DEFAULT: valueid)
[value]	{String}	the value we will look for

Returns

{Boolean} see below If the value parameter is missing, it returns true if text field - the field isn't empty simple field - there is a selected value multiple field - there are selected values The value is exists, it returns true if text field - the value of the field is exatly the same as the given value (the 3rd parameter is unused in this case) simple field - there is a selected element and it's property defined by the key is exactly the same as the given value. multiple field - there are selected values which property defined by the key is exactly the same as the given value.

Example

```
$('page1').onNext = function() {
     // text field
     dump(vcc.isSelected('haircolor_text', 'brown'));

     // simple field, the description of the selected value is 'budapest'
     dump(vcc.isSelected('hometown_radio', 'budapest', 'description'));

     // multiple field, there is a selected value which export_value is 'MTV'
     dump(vcc.isSelected('tvchanels_checkbox', 'MTV', 'export_value'));
};
```

isVoiceRecordingActive

Checks whether the voice recording is still ongoing.

Description

```
vcc.isVoiceRecordingActive(): boolean
```

Parameters

none

Return values

Returns **true** if the voice recording is active, **false** otherwise.

Example

```
const isRecording = vcc.isVoiceRecordingActive();
```

runApplication

vcc.runApplication(executable [, arguments, [strict]])

Executes an executable program or may terminate its process.

Parameters

Parameter	Type	Description
executable	String	File path...
arguments	Array of Strings	Arguments... Optional.
strict	Boolean	Possible options: - true: It is strict. - false: It is not strict.

Returns

Returns an Object.

Example

```
// simple
vcc.runApplication('shutdown');

// advanced
vcc.runApplication('shutdown', ['now']);

// more advanced
vcc.runApplication('shutdown', ['now'], true);
```

sendMessage

Request a message sending.

Parameters

Name	Type	Description
data	{Object}	message body

Returns

none

Example

We want to send the values of some fields (e.g. phone_number).

```
$('ba').onLoad = function() {
  var value = vcc.getFieldValue('field');
  var phone = vcc.getFieldValue('phone1');
  var data = {
     field: value,
     phone_number: phone
  };
  vcc.sendMessage(data);
};
```

setEventListener

Set an event listener. You can use this to set script event listeners programmatically. Be aware to use this function within an event! For setting static listeners it is easier to use the dollar-sign notation.

Parameters

Name	Type	Description
callback	{Function}	the function which will handle the event
controller	{String|null}	unique id of the controller in the page
event	{String}	the name of the event
page	{String|null}	page name

Returns

none

Example

We have lots of similar pages, each of them starts with the 'q4_' characters, and there are fields with these names, as well. We wanted to load this pages only if the export value of the connected field is 'allowed'. We can write lots of dollar-sign notation events, but it is more compact to do it in one step. It is important: there can be only one listener for every event, if you set something before, you will replace by setting another one.

```
$().onLoad = function() {
   vcc.getScriptPages().forEach(function(page) {
       if ('q4_' === page.substr(0, 3)) {
           vcc.setEventListener(page, null, 'checkBeforeLoad', function() {
               return vcc.isSelected(page, 'allowed', 'export_value');
           });
       }
   });
};
```

setFieldValue

Set the selected value of the field.

Parameters

Name	Type	Description
field	{String}	the name of the field
[key]		a key of the field to search (optional)
value		the value, see below

Format of the value

Name	Type	Description
multiple field	{Array of Number}	the valueids of the selected elements or {Array of Number|String} the values of the specified key
simple field	{Number}	the valueid of the selected element or {Number|String} the value of the specified key or null
text field	{String}	the value itself

Returns

none

Example

see CONTROLLER_OBJECT.refresh

startVoiceRecording

Starts voice recording.

Description

```
vcc.startVoiceRecording(): void
```

Parameters

none

Return values

void

Example

```
vcc.startVoiceRecording();
```

stopVoiceRecording

Stops voice recording.

Description

```
vcc.stopVoiceRecording(): void
```

Parameters

none

Return values

void

Example

```
vcc.stopVoiceRecording();
```

toggleVoiceRecording

Checks whether the voice recording is still ongoing or not, and stops or starts it accordingly.

Description

```
vcc.toggleVoiceRecording(): void
```

Parameters

none

Return values

void

Example

```
vcc.toggleVoiceRecording();
```

transfer

Transfer current call.

Parameters

Name	Type	Description
[isProcess]	{Boolean}	jump to processid if true, jump to phone otherwise
phone	{String}	telephone number or processid transfer to

Returns

none

Example

We put two button to the script: transfer to external number, and transfer to financial department.

```
$('page', 'transfer_phone_button').afterSetData = function() {
     vcc.transfer('3670333444');
};

$('page', 'transfer_process_button').afterSetData = function() {
   vcc.transfer('345', true);
};
```

Events

You can subscribe to various events to listen any user interactions. In the top level of the code you can use the following notation. The controller and the page are optional, if you miss some of them, you can set controller, page and global event.

Example

We want to set global, page and controller event listeners.

```
// selector usage
// $('<page>', '<controller>').<event> = function() {};

// setting global event listener
$().beforeSetDisposition = function() {};

// setting page event listener
$('page1').onLoad = function() {};

// setting controller event listener
$('page1', 'controller1').afterSetData = function() {};
```

controller.afterSetData

It is fired when a controller set the value of a field. Good for handle user interactions, modify the layout based on them.

Parameters

Name	Type	Descripton
name	{String}	the name of the field
value	{FIELD_VALUE_OBJECT}	the new value of the field

Returns

none

Example

Show our special offer for women if the gender is female (when the gender and the special offer are on the same page).

```
// First: set the correct layout on load
$('page').onLoad = function() {
     vcc.getController('page', 'offer_for_women').hidden =
             !vcc.isSelected('gender', 'female', 'export_value');
};

// Then: change the layout when the gender changes
$('page', 'gender').afterSetData = function() {
     vcc.getController('page', 'offer_for_women').hidden =
             !vcc.isSelected('gender', 'female', 'export_value');
};
```

controller.onLoadData

It is fired when the controller gets its selectable data from the database. Good for filter the allowed values. Important: the controller loads its data on page load and when its CONTROLLER_OBJECT.rebuild function called. (It is not called on CONTROLLER_OBJECT.refresh, and when the user change the selected tab.) Some controller are linked with more than one fields.

Parameters

Name	Type	Description
data	{Array of FIELD_VALUE_OBJECT}	the values of the actual field
field	{String}	the name of the actual field.

Returns

Type	Comment
Array	the modified data. If it is null, this field will be skipped.

Example

The description of the values of the mobile_phone field contains the export_value of a manufacturer. The user has selected the manufacturer before, we want to show only the appropriate phones.

```
$('page', 'mobile_phones').onLoadData = function(data) {
     var manufacturer = vcc.getFieldValue('manufacturer');

     if (!manufacturer) {
         // if there is no selected manufacturer,
         // let be there no selectable phones
         return [];
     }

     // filter the data array
     return data.filter(function(phone) {
         return phone.description = manufacturer.export_value;
     });
};
```

Example

The same as above, but do not hide the the inappropriate values, just make them disabled.

```
$('page', 'mobile_phones').onLoadData = function(data) {
     var manufacturer = vcc.getSelected('manufacturer');

     if (!manufacturer) {
         // if there is no selected manufacturer,
         // let be all phones selectable
         return data;
     }

     // apply a change for every item in the array
     data.forEach(function(phone) {
         phone.disabled = (phone.description !== manufacturer.export_value);
     });

     return data;
};
```

global.beforeSetDisposition

It is fired before setting the selected disposition. Good for prevent terminating the script on special cases or save the comment history into a field.

Parameters

Name	Type	Description
comment	{String}	the comment written by the agent
disposition	{Object}	the selected disposition

Returns

Type	Comment
{Boolean}	If true, disposition is allowed, if false, disposition isn't allowed
{Object}	Disposition object

Disposition object

Key	Type	Description
dispositionId	{Int}	The desired disposition that overwrites the one which was selected by the agent.
next_calldate	{String}	Optional. You can specify the callback time. It has only effect if the selected disposition is a callback. Format: ISO 8601, YYYY-MM-DDTHH:MM:SS

Example

If there are no ordered items, do not allow ordered dispositions.

```
$().beforeSetDisposition = function(disposition, comment) {
     if (!vcc.isSelected('ordered_items') && disposition.assesment === vcc.DISPOSITION_ORDER
ED) {
          vcc.alert('There are no ordered items, you cannot select a successful disposition'
);
          return false;
     }
};
```

Allows updating the disposition (to any dispositionid) selected by an agent in the script.

```
$().beforeSetDisposition = function(disposition){
    return({dispositionId: 1, next_calldate: '2015-03-09T13:45:42'});
}
```

global.onLoad

It is fired during the loading of the script when the database and other resources are ready. Good for setting field values based on external data, like phone numbers or agent properties. Important: there aren't any pages loaded this time.

Parameters

none

Returns

none

Example

We want to save the last phone number used by our client to a field.

```
$().onLoad = function() {
     vcc.setFieldValue('last_phone', vcc.getScriptVariable('global.clientphone'));
};
```

page.checkBeforeLoad

It is fired before the page begins loading. Good for checking if it is possible to go to this page, or we have to skip it. Important: controllers of the page are unavailable within this function.

Parameters

none

Returns

Type	Comment
{Boolean} true	the page is allowed, it will be loaded
{Boolean} false	the page isn't allowed, go to next page instead
{String}	the page isn't allowed, go to

Example

We have special offers to women, we want to show this page only for them.

```
$('page_for_women').checkBeforeLoad = function() {
     return vcc.isSelected('gender', 'female', 'export_value');
};
```

page.onLoad

It is fired immediately after the page has become completely loaded. Good for updating the visible layout according to the values of fields.

Parameters

none

Returns

none

Example

If we know the exactly date of birth, we don't want to ask the age-group.

```
$('page').onLoad = function() {
     vcc.getController('page', 'age_group').hidden = vcc.isSelected('date_of_birth');
};
```

page.onNext

It is fired before going to the next page. Good for both checking the validation of the current page and set the next page based on values of fields.

Parameters

none

Returns

Type	Comment
{Boolean} false	The current page is invalid, stay here and show an error
{Boolean} true	The current page is valid, go to to next page
{String}	The current page is valid, go to

Example

If the user is older than 60 years old and doesn't have internet, he cannot be our new customer, go to the final page.

```
$('page').onNext = function() {
     var age = vcc.getFieldValue('age');
     var hasInternet = vcc.isSelected('internet', 'yes', 'export_value');
     if (age > 60 && hasInternet) {
         return 'final';
     } else {
         return true;
     }
};
```

Objects

CONTROLLER_OBJECT

Structure of the controllers of the script.

Summary

Variables	
disabled	{Boolean} true if the controller is disabled.
hidden	{Boolean} true if the controller is hidden.

Functions	
clearHelper	Hide the small icon.
rebuild	Rebuild the structure of the controller.
refresh	Refresh the controller based on the selected values of the fields belonging to it.
showHelper	Show a small icon beside the controller.

disabled

{Boolean} **true** if the controller is disabled.

VARIABLES

hidden

{Boolean} **true** if the controller is hidden.

FUNCTIONS

clearHelper

Hide the small icon.

Parameters

none

Returns

none

Example

see showHelper

rebuild

Rebuild the structure of the controller. You need to use it if you want to update the available values.

Parameters

none

Returns

none

Example

After updating the manufacturer field we want to filter the mobile phones according the selected manufacturer. You can find the remaining code in the documentation of the EVENTS.onLoadData> event.

```
$('page', 'manufacturer').afterSetData = function() {
     vcc.getController('page', 'mobile_phones').rebuild();
};
```

refresh

Refresh the controller based on the selected values of the fields belonging to it. (Important: it only refreshes the selection/value of the controller, not the selectable values.)

Parameters

none

Returns

none

Example

We have the original data in a text field, but we don't want them to be changed. We have a field for the actual data, and for the simplicity we would like to copy the original data to the actual field and controller by clicking on a button.

```
$('page', 'my_button').afterSetData = function() {
     // copy the value to the new field (it is stored in the database)
     vcc.setFieldValue('actual', vcc.getFieldValue('original'));
     // refresh the controller to load the the value from the database
     vcc.getController('page', 'actual').refresh();
};
```

showHelper

Show a small icon beside the controller. It can be used for errors, warning, or just an information.

Parameters

Name	Type
mode	{vcc.CONTROLLER_HELPERS} the mode of the helper
label	{String} the main text of the helper
[title]	{String} the title of the helper

Example

We want to warn the agent if the amount of the salary is bigger than 2000.

```
$('page', 'salary').afterSetData = function(data) {
     if (data > 2000) {
         vcc.getController('page', 'salary').showHelper(vcc.CONTROLLER_WARNING,
                 'The salary can be mistyped');
     } else {
         vcc.getController('page', 'salary').clearHelper();
     }
};
```

DISPOSITION_OBJECT

The structure of a disposition object.

Properties

Name	Type
assesment	{vcc.DISPOSITION_ASSESMENTS}
commission	{Number}
description	{String}
id	{Number} Unique ID
name	{String}
price	{Number}

FIELD_PROPERTY_OBJECT

The type of the properties object of a field.

Properties

Name	Type
fieldid	{Number} Unique ID
indexed	{Boolean}
label	{String}
name	{String}
quota	{vcc.FIELD_QUOTAS}
type	{vcc.FIELD_TYPES}

FIELD_VALUE_OBJECT

The structure of a value of a simple or multiple field.

Properties

Name	Type
commission	{Number}
description	{String}
export_value	{String}
name	{String}
price	{Number}
valueid	{Number} Unique ID

QUOTA_DATA_OBJECT

The type of the properties of an edge quote data object.

Properties

Name	Type
act_quota	{Number} the actual value of the quota
diff_value	{Number} remaining items
value	{Number} the value of the quota (the maximum value that can be reached)

Click 2 Call

Initiate, hold and hang up calls from a browser or any desktop application.

Introduction

VCC Live's Click to Call (C2C) service makes it easy to manage calls from another desktop software programme, including web browsers.

Once enabled, when an agent successfully logs in a simple, embedded HTTP server is started and bound to the port set by a supervisor. Any software that can send simple HTTP GET requests to VCC Client's Click to Call interface can be used to initiate, hold and hang up calls.

Enable Click to Call Service

To enable the Click to Call service an administrator (e.g. a supervisor) needs to open VCC Live -> Tools -> Call Center Settings -> General tab. Below 'Click 2 Call service', the following steps need to be carried out:

1. Check the 'Enabled' checkbox
2. Enter a port for the HTTP service
3. Press the 'Save' button to save modifications.

The Click to Call service interface is available at http://localhost:[port]/[resource]?[parameters], where:

- *port* is the specified port by an administrator,
- *resource* is one of the three possible commands and
- *parameters* are optional parameters regarding the given resources.

Warning: the Click to Call service is only activated at the end of a successful login process, so already logged in agents need to firstly logout then login again to be able to use this feature.

Resources

Initiating a Call

The VCC Client initiates a call and inserts a new record in the agent's primary project's database using the '/call' resource. Calls can only be initiated when the agent is in 'available' status, otherwise an error message appears instead.

Resource

- Resource: http://localhost:[port]/call/[phone]
- Example: http://localhost:37589/call/36123456789

Optional parameters

Database fields can be set using HTTP parameters.

- Resource: http://localhost:[port]/call/[phone]?field1=value1&field2=value1
- Example: http://localhost:37589/call/36123456789?name=Full Name

Putting a Call on Hold

An ongoing call can be put on hold and retrieved using the '/call/hold' resource.

Resource

- Resource: http://localhost:[port]/call/hold
- Example: http://localhost:37589/call/hold

Hanging Up a Call

An ongoing call can be terminated using the '/call/hangup' resource.

Resource

- Resource: http://localhost:[port]/call/hangup
- Example: http://localhost:37589/call/hangup

Optional parameters

Additionally, it is also possible to set a disposition using the 'dispositionid' parameter. If 'dispositionid' refers to a callback disposition, the callback date is set based on the given disposition's default setting.

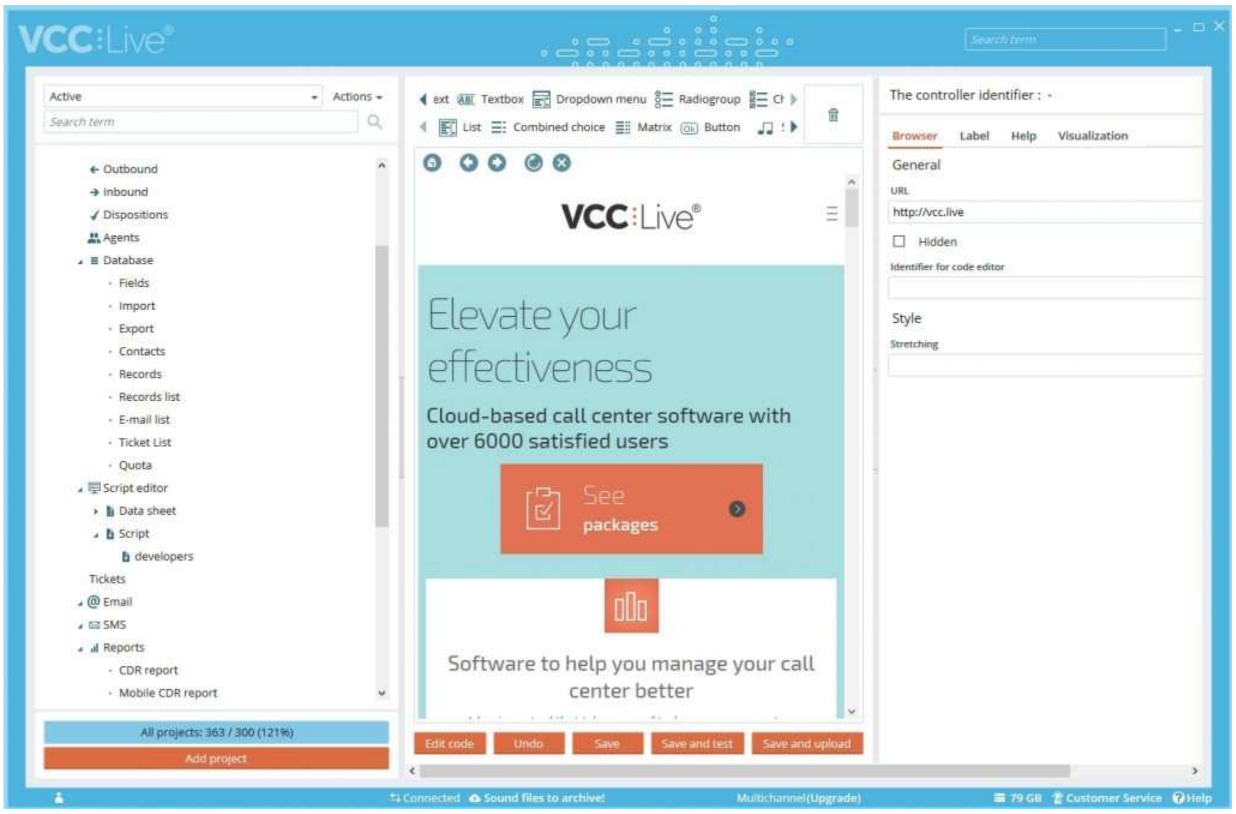
- Resource: http://localhost:[port]/call/hangup?dispostionid=[id]
- Example: http://localhost:37589/call/hangup?dispositionid=1

Website Embedding

Integrate any web-based ERP and CRM system in just a few clicks.

Introduction

Any web-based ERP, CRM and other web-based applications can easily be embedded into VCC's agent interface in just a few seconds allowing, for example, a website with relevant information to appear at the appropriate time on an agent's screen during a conversation.



The controller identifier : -

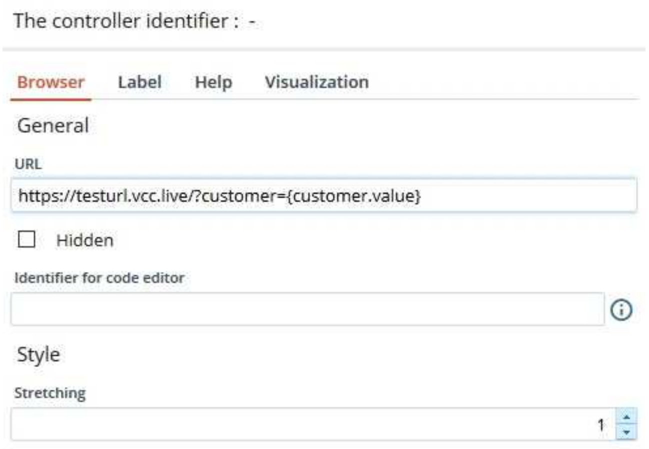
URL: http://vcc.live

If you have any questions regarding how to embed your web-based CRM system into VCC's agent interface, please contact us.

Embed a Website

Any web site can be embedded in a project's Script editor submenu on a datasheet or script page, using the Browser controller:

1. Add a Browser controller to a datasheet or script page
2. Enter your ERP or CRM system's URL
3. If needed, add database fields as variables to the URL


The controller identifier : -

General

URL: https://testurl.vcc.live/?customer={customer.value}

☐ Hidden

Identifier for code editor:

Style

Stretching: 1

The above Browser controller example's script editor settings

Important: We strongly recommend using HTTPS rather than HTTP.

When a call is initiated or answered the relevant datasheet, as well as the appropriate web site based on the previously defined URL, appear.

3rd Party Integrations

Connecting applications to VCC Live through API integration.

Zapier

Introduction

The Zapier development, our goal was to create an intermediate channel through which the VCC in many different applications can be connected, without any development knowledge. The Zapier met these demands, since it can be connected to more than 500 applications.

The Zapier a cloud-based service that connects a central element in a variety of applications. In all cases, the data pass through Zapier individual matching of data structure between two (or more) applications simply by clicking configurations, i.e. Zapier help. All Zap is an application in which an event occurs (e.g. a sign up on the website) , and at least one application, which is generated during the event executed some data (such as the subscriber's name and e-mail address is added to an email list).

Zapier the terminology of the first application it's the trigger, and the second is called ActionScript. The VCC and the Trigger, Action and implements the functionality.

Events (Triggers)

These events can occur in the VCC, which causes the VCC resulting data to be can automatically forwarded to other Zapier connected to the system:

- Create new account
- Change user data

- Fixing operator disposition

Activities (Actions)

A result of events occurring in other systems the following activities can be carried out in the VCC:

- Create new record
- Modify existing record

> The surface of Zapier registration alone is not sufficient to achieve the VCC translation app. To do this, you need a minimum subscription, for instance. Database API password, which can be generated on the surface of the VCC administrator. After that you can request access to the VCC Zapier App. The responsible user will be required to link opening of the VCC client ID, a password -generated database API, or application that you want to get involved in the process of identifying data. In this process, the wizard Zapier is to help our customers, for detailed information about it then visit https://www.zapier.com.

The VCC does not charge any specific cost for providing information through the Zapier app. Using the same time Zapier with different packages to choose from, based on consumption is not free. The cost for each client and therefore the project may vary, for example depending on the number of interconnections. Visit the site https://www.zapier.com/pricing detailed informations about the pricing.

> The VCC and the connecting Zapier knowledge is required for the Database API, but development of an adequate logical connections. To set the proper connections in the VCC does not provide specific support, technical assistance in such a Zapier operators on request.

The Zapier does not store the data transmitted it only carry out the so-called transmission. Always keep track of what applications connected to VCC to know that in addition to the VCC system where data is stored in the access!