Instructions Application Program Interface (API)

The URL generally has the form /api/<resource>/<action>, which could either retrieve a result or perform some action.

When retrieving a result, the API expects a HTTP GET method, and when performing an action it expects a HTTP POST.

The resource /api/instructions refer to all the instructions in the database. To address a specific instruction, use a URL of the form /api/instructions/{unique id}/<action>.

Getting the Unique ID required in this API

The Unique ID required in most of these calls can be obtained by first using the following endpoint, which returns the Unique ID per instruction. See the Accounts API documentation for more information.

GET: /api/accounts/{accountNumber}/instructions

or

GET: /api/accounts/{accountNumber}/instructions/active

Successful results are returned as HTTP 200 or 201 responses, with the body containing JSON data. Defined error responses will be in the HTTP 400 range, and undefined server errors in the HTTP 500 range.

Error responses contain the following JSON format:

```
{
	"errors" : [
		{
			"description" : "<error description>",
			"code" : "<error code>",
			"index" : <integer>
		}
	]
}
```

where the error description is a detailed description of the error, and the error code is a short reference to the error.

Index is an optional error field and is present when the request data is an array of objects. It indicates the position of the offending object in the request array.

Supported API calls

1 Instruction details

GET: /api/instructions/{uniqueId}

Example request:

/api/instructions/2217472

See detail here on where to get this {uniqueId}.

Example JSON response - Investment:


```
{
        "Instruction id":               27647488,
        "Type":                         "Once Off Investment",
        "Status":                       "Submitted",
        "Date created":                 "2005-05-13",
        "Contract id":                  4435261,
        "Bank name":                    "",
        "Bank account holder":          "",
        "Bank account number:          "",
        "Commencement date":            "2005-05-13",
        "Annual income increase":       "",
        "Next escalation date":          "",
        "Next debit order date":        "",
        "Amount":                       {
                "type":         "Money",
                "currency":     "ZAR",
                "value":        "2100.00"
        },
        "Frequency":            "monthly",
        "Currency":             "ZAR",
        "Day of month":         "1",
        "Contributions":        [
                {
                        "Percentage": "100.00%",
                        "Type":                 "Buy",
                        "Instrument code":      "COREQ",
                        "Instrument id":         373901568,
```

```
                "Instrument name":      "Coronation Equity Fund",
                "Amount":               "R 2,100.00",
                "Units":                ""
            }
        ],
        "Next recurrence id": null,
        "Previous recurrence id": null
}
```


Example JSON response - Withdrawal:


```
{
        "Instruction id":               1000762112,
        "Type":                         "Once Off Withdrawal",
        "Status":                       "New",
        "Date created":                 "2005-05-13",
        "Investor account id":          897638656,
        "Bank name":                    "",
        "Bank account holder":          "",
        "Bank account number":          "",
        "Commencement date":            "2005-05-13",
        "Annual income increase":       "",
        "Next escalation date":         "",
        "Next debit order date":        "",
        "Amount":                       {
            "type":         "Money",
            "currency":     "ZAR",
            "value":        "100.00"
        },
        "Frequency":                    "monthly",
        "Currency":                     "ZAR",
        "Day of month":                 "1",
        "Contributions": [
            {
                "Percentage":           "50.00%",
```

```
                "Type":                 "Sell",
                "Instrument code":      "ZAR",
                "Instrument id":        63591680,
                "Instrument name":      "Cash in ZAR",
                "Amount":               "R 100.00",
                "Units": ""
            }
        ],
        "Next recurrence id": null,
        "Previous recurrence id": null
}
```


Example JSON response - Switch Instruction:


```
{
        "Instruction id":               924583168,
        "Type":                         "Switch",
        "Status":                       "Submitted",
        "Date created":                 "2005-05-13",
        "Investor account id":          153434624,
        "Bank name":                    "",
        "Bank account holder":          "",
        "Bank account number":          "",
        "Commencement date":            "2005-05-13",
        "Annual income increase":       "",
        "Next escalation date":         "",
        "Next debit order date":        "",
        "Amount":                       {
            "type":         "Money",
            "currency":     "ZAR",
            "value":        "200.00"
        },
        "Frequency":            "monthly",
        "Currency":             "ZAR",
        "Day of month":         "1",
```

```
        "Contributions":                [
                {
                        "Type":                 "InternalSwitch",
                        "From instrument id":   58530304,
                        "From instrument code": "COREQ",
                        "From instrument name": "Coronation Equity Fund",
                        "From amount":          "",
                        "From units":           "200.0000",
                        "To instrument id":     33106432,
                        "To instrument code":   "COR01",
                        "To instrument name":   "Coronation Balanced Fund",
                        "To amount":            "",
                        "To units":             ""
                }
        ]
}
```


Notes:

1. The Contributions field is a list of the constituent buys, sells, transfer-ins, transfer-outs, and internal switches that make up the instruction.

Error examples:

HTTP response	Code	Description
404	NotFound	Invalid uniqueId

2 Cancel instruction

POST: /api/instructions/{uniqueId}/cancel

Example request:

/api/instructions/2217472/cancel

See detail here on where to get this {uniqueId}.

JSON response

HTTP 200

This call can now cancel a debit order as well

Error examples:

HTTP response	Code	Description
500	NotFound	Invalid uniqueId
500	InternalServerError	Cannot cancel Investment: R 2,100.00 POL3

3 Instruction Purposes

GET: /api/instructions/purposes/

Example request:

/api/instructions/purposes/

Example JSON response:

```
[
    {
        "name":                 "Investment",
        "supportedInstructions":    [ "Investment"],
    },
    {
        "name":                 "Switch or Withdrawal",
        "supportedInstructions":    [ "SwitchInstruction", Withdrawal"],
```

```
        }
    ]
```

4 Create Switch

The below documents the api to create and submit a switch instruction.
Instructions can be sold by amount, unitsAsPercentage or units

POST: /api/instructions/{contractNumber}/switches

Example request to switch using amount for sell:


```
{

        "purpose" : "optional instruction purpose"
        "sellUsing" : "amount",
        “status” : “Authorised”,
        “sells” : [
                {       "instrument" :  "AGNBFB" ,
                        "amount" : 500
                },
                {       "instrument" :  "AGNFSA" ,
                        "amount" : 200.01
                }
        ]
        “buys” : [

                {       "instrument" :  "AGNBFD" ,
                        "percentage" : 40
                },
                {       "instrument" :  "AGNSFD" ,
                        "percentage" : 60
                }
        ]
}
```

Example request to switch using percentage for sell:

```
{
        "purpose" : "optional instruction purpose",
        "sellUsing" : "unitsAsPercentage",
        “sells” : [
                {       "instrument" :  "instrument 1 code" ,
                        "amount" : 20 ,
                },
                {       "instrument" :  "instrument 2 code" ,
                        "amount" : 33,
                }
        ].
        “buys” : [
                {       "instrument" :  "instrument 3 code" ,
                        "percentage" : 50
                }
        ]
}
```

Example request to switch using units for sell:

```
{
        "purpose" : "optional instruction purpose",
        "sellUsing" : "units",
        “sells” : [
                {       "instrument" :  "instrument 1 code" ,
                        "amount" : 104 ,
                },
                {       "instrument" :  "instrument 2 code" ,
                        "amount" : 89.9,
                }
        ].
        “buys” : [
                {       "instrument" :  "instrument 3 code" ,
                        "percentage" : 10.5 ,
                }
        ]
}
```

JSON success response

HTTP 200

```
{
        "instructionId" : 27647488,
        "instructionUrl" : "URL to the fetch the instruction"
}
```

“status” can be provided optionally. If not provided the status will be Submitted. If using Pending Authorisation then instruction must adhere to normal rules pertaining to this status. If using Authorised then the API user must have sufficient access to execute this.

Error examples:

HTTP response	Code	Description
400	ContractNotFound	client account not found / not active

400	InvalidPurpose	Invalid instruction purpose
400	InvalidSellInstrument	Selling instrument not found or available on client account
400	InvalidInstrumentSellAmount	Selling instrument sell amount exceeds available amount
400	InvalidInstrumentSellPercentage	Selling instrument sell percent must be between 1 and 100
400	InvalidInstrumentSellUnits	Selling instrument units exceed available amount
400	ValidateError	sellUsing type must be one of : amount, unitsAsPercentage or units
400	MalformedRequest	sell or buys section missing from json request
400	ValidationFailure	A message due to failure when validating preconditions on switch for submission. Examples : The instrument allocations do not add up to 100.00%

5 Create Unit Transfer In

POST: /api/instructions/{contractNumber}/transferInInstructions

Example JSON request: /api/instructions/CUSTOM1004/transferInInstructions


```
{
        "sourceOfFunds": "Salary",
        "status": "Authorised",
        "purpose": "Transfer In",
        "transferIns": [
                {
                        "instrument": "INSTRMT1",
                        "units": 78.987
                },
                {
                        "instrument": "INSTRMT2",
                        "units": 123.4756
                }
        ]
}
```


Notes:

Mandatory fields are:

- transferIns

  "transferIns" allows specification of what instruments to create transfer ins for. The instrument as specified by the instrument code must also be available on the product of the investor account.

Conditional fields:

- sourceOfFunds is compulsory if ‘Instructions require source of funds’ is ticked on product settings.

Optional fields with their defaults if not present:

- status: Submitted
- purpose:nil

"status" is optional. By default the instruction will be submitted, and will be attempted to be authorised in the case shown in the example. "Authorised", “New” or "Submitted" are the only accepted input; specifying "Submitted" will do nothing as it is submitted by default.

"purpose" is optional and must use a purpose name returned in /api/instructions/purposes where supported Instruction contains UnitTransferOutInstruction

JSON response:

HTTP: 201 {"Unique Id" : 12345678}

The following query can be run to check the instruction created by using the Unique Id in the response:

GET: /api/instructions/12345678/

Error examples:

HTTP response	Code	Description
400	ContractInactiveError	Contract is inactive
400	ContractNotFound	Could not find contract
400	InstrumentNotAvailable	'Instrument with code: "INSTRMT1" could not be found.'
400	InstrumentNotAvailable	'Instrument with code: "INSTRMT1" is not available for Contract: "CUSTOM1004"'

6 Create Unit Transfer Out

POST: /api/instructions/{contractNumber}/transferOutInstructions

Example JSON request: /api/instructions/CUSTOM1004/transferOutInstructions

```
{
        "isCapitalGainEvent": true,
        "status": "Authorised",
        "purpose": "Transfer In",
        "transferOuts": [
                {
                        "instrument": "INSTRMT1",
                        "percentage": 100.0
                },
                {
                        "instrument": "INSTRMT2",
                        "percentage": 100.0
                }
        ]
}
```

Notes:

Mandatory fields are:

- transferOuts

  "transferOuts" allows specification of what instruments to create transfer outs for.

Optional fields with their defaults if not present:

- status: submitted
- isCapitalGainEvent: false
- purpose: nil

"status" is optional. By default the instruction will be submitted, and will be attempted to be authorised in the case shown in the example. "Authorised", “New” or "Submitted" are the only accepted input; specifying "Submitted" will do nothing as it is submitted by default.

"purpose" is optional and must use a purpose name returned in /api/instructions/purposes where supported Instruction contains UnitTransferOutInstruction

JSON response:

HTTP: 201 {"Unique Id" : "12345678"}

The following query can be run to check the instruction created by using the Unique Id in the response:

GET: /api/instructions/12345678/

Error examples:

HTTP response	Code	Description
400	ContractInactiveError	Contract is inactive
400	ContractNotFound	Could not find contract
400	InstrumentNotAvailable	'Instrument with code: "INSTRMT1" could not be found.'
400	InstrumentNotAvailable	'Instrument with code: "INSTRMT1" is not available for Contract: "CUSTOM1004"'

7 Create Internal Unit Transfer

POST: /api/instructions/{contractNumber}/internalUnitTransferInstructions

Example JSON request:
/api/instructions/CUSTOM1004/internalUnitTransferInstructions


```
{
        "toClientAccount": "CUSTOM1005-1",
        "status": "Authorised",
        "workIn": "units",
        "isCapitalGainEvent": true,
        "purpose": "Trust transfer",
        "transationDate": "2024-10-01",
        "transfers": [
                {
                        "instrument": "INSTRMT1",
                        "amount": 5
                }
        ]
}
```


Notes:

Mandatory fields are:

- toClientAccount (Client Account Number of receiving client account)
- transfers

  "transfers" allows specification of what instruments to create internal transfers for.

Optional fields with their defaults if not present:

- status: submitted
- isCapitalGainEvent: false
- purpose: nil
- workIn: units
- transactionDate: Today's date - Future date not allowed

"status" is optional. By default the instruction will be submitted, and will be attempted to be authorised in the case shown in the example. "Authorised", “New” or "Submitted" are the only accepted input; specifying "Submitted" will do nothing as it is submitted by default.

"purpose" is optional and must use a purpose name returned in /api/instructions/purposes where supported Instruction contains InternalUnitTransferInstruction.

"workIn", which can be "percentage" or "units". This is optional. If not present, the default is "units". If "workIn" is "percentage", this will imply "Units as %", which means when specifying "transfers", we expect percentages per instrument.

JSON response:

HTTP: 201 {"Unique Id" : 12345678}

The following query can be run to check the instruction created by using the Unique Id in the response:

GET: /api/instructions/12345678/

Error examples:

HTTP response	Code	Description
400	ContractInactiveError	Contract is inactive
400	ContractNotFound	Could not find contract
400	InstrumentNotAvailable	'Instrument with code: "INSTRMT1" could not be found.'
400	InstrumentNotAvailable	'Instrument with code: "INSTRMT1" is not available for Contract: "CUSTOM1004"'
400	InvalidWorkInType	Invalid "workIn". Allowed values are : "units, percentage"
400	FutureTransactionDateNotAllowed	Future transaction date not allowed

8 Stop recurring sequence

This call will cancel the most recent recurrence of the sent in instruction. This will effectively stop the recurring sequence. The call then returns information about other instructions in the sequence that are still active; whether they can be cancelled and how to cancel them if applicable.

POST: /api/instructions/{uniqueId}/stopRecurringSequence

Example request:

/api/instructions/2217472/stopRecurringSequence

See detail here on where to get this {uniqueId}.

JSON success response

HTTP 200


```
{
        “Active instructions” : [
                {
                "Instruction id" : “27647488”,
                "Can be cancelled" : true,
                “Cancel instruction call” : “/api/instructions/27647488/cancel”
                },
                {
                "Instruction id" : “27647489”,
                "Can be cancelled" : false,
                “Cancel instruction call” : “”
                }
        ]
}
```


Error examples:

HTTP response	Code	Description
500	NotFound	Invalid uniqueId

500	InternalServerError	Cannot cancel Investment: R 2,100.00 POL3