

Accounts 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/accounts` refer to all the accounts in the database. To address a specific account, use a URL of the form `/api/accounts/{account number}/<action>`.

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 Total market value**
- 2 Market value**
- 3 Create account and person**
- 4 Create account and company**
- 5 Create account**
- 6 Create investment**
- 7 Create withdrawal**
- 8 Individual statement**
- 9 Combined statement**
- 10 Holdings**
- 11 Available Holdings**
- 12 Advisor accrued fees**
- 13 Advisor fees (taken)**
- 14 Policy document**
- 15 Account details**
- 16 Update Account details**
- 17 Available instruments**
- 18 Documents**
- 19 Instructions**
- 20 Investment Summary Since Inception**
- 21 Investment Summary for Period**
- 22 Products**
- 23 Product available instruments**
- 24 Transactions for Period**

1 Total market value

GET: /api/accounts/totalMarketValue (as of today)

GET: /api/accounts/totalMarketValue?date={date} (as of date specified)

Note: Date must be ISO format string YYYY-MM-DD

Example JSON response:

```
{
  "amount" : "1199.9800",
  "currency" : "ZAR",
  "accrued fees" : "0.60",
  "date" : "2021-12-31"
}
```

2 Market value

GET: /api/accounts/{accountNumber}/marketValue (as of today)

GET: /api/accounts/{accountNumber}/marketValue?date={date} (as of date specified)

Note: Date must be ISO format string YYYY-MM-DD

Example JSON response:

```
{
  "amount" : "599.9900",
  "currency" : "ZAR",
  "accrued fees" : "0.02",
  "amount in default currency" : "599.9900",
  "accrued fees in default currency" : "0.02",
  "amount in preferred currency" : "599.9900",
  "accrued fees in preferred currency" : "0.02",
  "date" : "2021-12-31"
}
```

Error examples:

HTTP response	Code	Description
400	AccountInactiveError	Account is inactive
404	AccountNotFound	Could not find account

3 Create account and person

POST: /api/accounts/createPersonAndAccount

Example JSON request:

```
{
  "idNumber": "1234567890",
  "firstNames": "Jane",
  "surname": "Doe",
  "initials": "JD",
  "middleNames": "Angela Prime",
  "title": "Mrs",
  "countryOfBirth": "MU", //New
  "accountNumber": "CUSTOM1004",
  "productName": "The Investment",
  "clientGroup": "Direct Client",
  "dateOfBirth": "1991-11-19",
  "gender": "Female",
  "email": "jane@doe.com",
  "taxNumber": "P1234056",
  "cellPhoneNumber": "+27128092343",
  "workPhoneNumber": "+27128092343",
  "homePhoneNumber": "+27128092343",
  "distributionOption": "reinvestDistribution",
  "contractStatus": "New business", //New
  "address": {
    "line1": "Unit 2",
    "line2": "17 Awkward Road",
    "line3": "",
    "city": "Gotham",
    "code": "1234",
    "country": "ZA",
    "type": "Residential"
  },
  "nationality": "ZA",
  "countryForTaxPurposes": "ZA",
  "isTaxResidentInCountryForTaxPurposes": true,
  "isUSACitizen": false,
  "isPoliticallyExposed": false, //New
  "receivesIncomeFromUSA": false,
  "employmentStatus": "Self-employed",
  "occupation": "Accountant",
  "occupationIndustry": "Accountancy",
  "sourceOfFunds": "Transfer from RFI",
  "totalAnnualEarnings": "<25000",
  "bankDetails": {
    "holderName": "John Doe",
    "accountNumber": "1234567890",
    "branchCode": "222626",
    "type": "Current",
    "bankName": "First National Bank",
    "branchName": "Centurion",
  }
}
```

```

    "currencyIdentifier": "zar",
    "swiftBankIdentifierCode": "777",
    "intermediaryName": "NEDWXYZYYYY",
    "internationalBankAccountNumber": "GB82WEST12345698765432"
  },

  "communicationOptions": {
    "sendCorrespondenceTo": "Client",
    "automaticallyEmailStatement": true,
    "communicationPreference": "Email",
    "transactionNotificationPreferences": {
      "CompleteWithdrawal": "Email",
      "SuccessfulLogin": "Both"
    }, //New
  },

  "entityRelationships": [ //New
    {
      "relationshipType": "Spouse",
      "idNumber": "8888",
      "firstNames": "John",
      "initials": "J",
      "surname": "Doe"
    }
  ]
}

```

Notes:

Mandatory fields are:

- idNumber
- firstNames
- surname
- initials (defaults to first letter of firstNames)
- productName

The following restrictions apply:

- productName: must be the name of an existing product in the system
- dateOfBirth: a valid date in various formats, preferably ISO format
- clientGroup: the name of a valid client group as set up in the system configuration
- gender: Male or Female
- email: a valid email address
- address - type: Residential, Registered, Business or Postal
- address - country: a valid ISO country code (as set up in the system configuration)
- countryOfBirth: a valid ISO country code (as set up in the system configuration)
- title: a valid title (as set up in the system configuration)
- contractStatus: One of the statuses in System settings/ Account statuses. The transition from the existing status to the new status should also be valid
- distributionOption : can be one of: reinvestDistribution,payoutDistribution or leaveDistributionInCash

- under communicationOptions, the following apply:
 - sendCorrespondenceTo: can be one of: 'None' 'Both' 'Client' or 'Advisor'
 - communicationPreference can be one of: 'Post' 'Email' 'No consent given' or 'Opted-out'
 - automaticallyEmailStatement is a boolean
 - transactionNotificationPreferences is a list of types: SubmitInvestment, CompleteInvestment, SubmitWithdrawal, CompleteWithdrawal, SubmitSwitchInstruction, CompleteSwitchInstruction, SubmitUnitTransferInInstruction, CompleteUnitTransferInInstruction, SubmitUnitTransferOutInstruction, CompleteUnitTransferOutInstruction, SubmitReinvestmentSwitchInstruction, CompleteReinvestmentSwitchInstruction, ApprovedChangeRequest, SuccessfulLogin, the preference can be one of 'Both' 'Email' 'SMS' or 'None' . By default, none are set

Optional fields with their defaults if not present:

- accountNumber: a system assigned number starting with the prefix set up in the system configuration
- firstNames: blank
- surname: blank
- clientGroup: set to the system default
- dateOfBirth: nil
- gender: blank
- email: blank
- cellPhoneNumber: blank
- workPhoneNumber: blank
- homePhoneNumber: blank
- address:
 - city, line1, line2, line3, code: blank
 - type: Residential
 - country: default country set up in system configuration
- taxNumber: blank
- employmentStatus: Options - System - Risk Assessment Employment Status Type
- occupation: Options - System - Risk Assessment Occupation Type
- occupationIndustry: Options - System - Industries
- sourceOfFunds: Options - System - Risk Assessment Source Of Funds
- totalAnnualEarnings: Options - System - Risk Assessment Total Annual Earning
- bankDetails
- advisor

Example JSON response:

```
{  
  "accountNumber": "CUSTOM1004",
```

"entityUniqueId": 123456

}

Error examples:

HTTP response	Code	Description
400	ExistingPersonWithID NumberFound	A person matching the idNumber already exists
400	MalformedRequest	Malformed request. Field "idNumber" is compulsory
400	MalformedRequest	Malformed request. Invalid JSON
400	MalformedRequest	Invalid title \"Mrs\"

4 Create account and company

New call: Added Jan 2025

POST: /api/accounts/createCompanyAndAccount

Example JSON request:

```
{
  "companyName": "ABC PRINTERS (PTY)LTD",
  "companyType": "Listed company",
  "registrationNumber": "71/12707/07",
  "registeredCountry": "ZA",
  "registrationDate": "1999-01-02",
  "accountNumber": "CUSTOM1005",
  "clientGroup": "Direct Client",
  "distributionOption": "reinvestDistribution",
  "productName": "The Investment",
  "email": "abcprinters@mail.com",
  "taxNumber": "1234/123/12/1",
  "isTaxResidentInCountryForTaxPurposes": false,
  "cellPhoneNumber": "+27123457890",
  "workPhoneNumber": "+27123457891",
  "homePhoneNumber": "+27123457892",
  "address": {
    "line1": "Unit 2",
    "line2": "17 Awkward Road",
    "line3": "Ghost Town",
    "city": "Gotham",
    "code": "1234",
    "country": "ZA",
    "type": "Residential"
  },
  "advisor": {
    "advisorCode": "AGFSPB",
    "brokerageCode": "000017"
  },
  "bankDetails": {
    "holderName": "John Doe",
    "accountNumber": "1234567890",
    "branchCode": "222626",
    "type": "Current",
    "bankName": "First National Bank",
    "branchName": "Centurion",
    "currencyIdentifier": "zar",
    "swiftBankIdentifierCode": "777",
    "intermediaryName": "NEDWXYZYYYY",
    "internationalBankAccountNumber": "GB82WEST12345698765432"
  },
  "entityRelationships": [
    {
```



```
        "relationshipType": "Director",  
        "idNumber": "8888",  
        "firstNames": "John",  
        "initials": "J",  
        "entityType": "Person",  
        "surname": "Doe"  
    }  
]  
}
```

Notes:

Mandatory fields are:

- productName
- companyName
- companyType
- registrationNumber
- entityRelationships-entityType

The following restrictions apply:

- productName: must be the name of an existing product in the system
- companyType: the name of a valid Company type as set up in the system configuration
- clientGroup: the name of a valid client group as set up in the system configuration

Optional fields with their defaults if not present:

- accountNumber: a system assigned number starting with the prefix set up in the system configuration
- registeredCountry: default country set up in system configuration
- registrationDate: blank
- email: blank
- cellPhoneNumber: blank
- workPhoneNumber: blank
- homePhoneNumber: blank
- address:
 - city, line1, line2, line3, code: blank
 - type: Residential
 - country: default country set up in system configuration
- taxNumber: blank
- bankDetails
- advisor

Example JSON response:

```
{
```

```
"accountNumber": "CUSTOM1005",  
"entityUniqueId": 123456  
}
```

Error examples:

HTTP response	Code	Description
400	ExistingCompanyWithRegistrationNumberFound	A company matching the registrationNumber already exists
400	CompanyNotFound	Could not find company

5 Create account

POST: /api/accounts/createAccount

Example JSON request:

```
{  
  "idNumber" : "1234567890",  
  "accountNumber" : "CUSTOM1004",  
  "productName" : "The Investment",  
  "clientGroup" : "Direct Client",  
  "advisor": {  
    "advisorCode": "AD1234",//New  
    "brokerageCode": "BR777",//New  
  }  
}
```

Notes:

Mandatory fields are:

- idNumber (for person) or registrationNumber (for company) or entityUniqueId (for either company or person)
- productName

The following restrictions apply:

- productName: must be the name of an existing product in the system
- clientGroup: the name of a valid client group as set up in the system configuration

Optional fields with their defaults if not present:

- accountNumber: a system assigned number starting with the prefix set up in the system configuration
- clientGroup: set to the system default
- advisor: An existing advisor with the code = advisorCode, and brokerage code = brokerageCode will be set on the contract.

Example JSON response:

```
{"accountNumber" : "CUSTOM1004"}
```

Error examples:

HTTP response	Code	Description
400	PersonNotFound	Could not find person
400	CompanyNotFound	Could not find company
400	MissingEntityIdentifierField	One of the following fields is required: "idNumber" or "registrationNumber" or "entityUniqueid"
400	MalformedRequest	Malformed request. Field "idNumber" is compulsory
400	MalformedRequest	Malformed request. Invalid JSON
400	MalformedRequest	Advisor not linked to brokerage with given brokerage code
404	AdvisorNotFound	Could not find an advisor with the given code

6 Create investment

POST: [/api/accounts/{accountNumber}/investments](#)

Example JSON request: [/api/accounts/CUSTOM1004/investments](#)

```
{
  "amount" : 1234.56,
```

```
"currency" : "ZAR",  
"sourceOfFunds" : "Salary",  
"authoriseAutomatically" : true,  
"purpose" : "Investment",  
"buys" : {  
    "INSTRMT1" : "75.00%",  
    "INSTRMT2" : "25.00%"  
}  
}
```

Notes:

Currencies with identifiers are configured in the system configuration.

"sourceOfFunds" is optional and is also configured in the system configuration.

"authoriseAutomatically" is also optional. By default the instruction will be submitted, however, an investment cannot be authorised unless it is matched. In order to streamline the process of authorising investments this field can be set to true, in which case the investment will be authorised automatically when it has been matched. Leaving this field out, or setting it to false, will leave the investment in the "submitted" state, where it will remain until the investment is manually authorised.

"buys" is also optional. This allows specification of what instruments to create buys for. The instrument as specified by the instrument code must also be available on the product of the investor account. The percentage for each buy must add up to 100%.

"purpose" is optional and must use a purpose name returned in `/api/instructions/purposes` where `supportInstruction` contains `Investment`

JSON response:

HTTP: 201

```
{"Unique Id" : 12345678}
```

Error examples:

HTTP response	Code	Description
400	AccountInactiveError	Account is inactive
404	AccountNotFound	Could not find account
400	MalformedRequest	Malformed request. Amount value "-100.00" must be greater than 0
400	MalformedRequest	Malformed request. Invalid currency "LIBRE"
400	MalformedRequest	Malformed request. Field "amount" is compulsory
400	InstrumentNotAvailable	Instrument with code: "INSTRMT3" could not be found.
400	InstrumentNotAvailable	Instrument with code: "INSTRMT4" is not available for Investor Account: "accountNumber"
400	CantSubmitInstruction	The instrument allocations do not add up to 100.00% for the R 1,234.56 investment on "accountNumber"
400	InvalidInstruction Purpose	The instruction purpose is invalid for the instruction type or the instruction purpose could not be found

7 Create withdrawal

POST: /api/accounts/{accountNumber}/withdrawals

Here are some important fields that determine how the withdrawal instruction will sell.

1. "workIn", which can be "amount" or "units". This is optional. If not present, the default is "amount". If "workIn" is "units", this will imply "Units as %", which means when specifying "sells", we expect percentages per instrument.
2. "captureBy", which can be "entire instruction" or "per instrument". This is also optional.
3. "amount", which is a positive currency amount. This is used when working in amounts, capturing by entire instruction and no sells are specified.
4. "instructionPercentage", which is a number > 0 and ≤ 100 , and only used when working in units, capturing by entire instruction and no sells are specified.
5. "status", which can be "Authorised", "Pending Authorisation", or "Submitted". If this field is not set then the withdrawal will be submitted only.

Summary of default behaviour

- * If there is an "amount" and "sells", the default is "per instrument".
- * If there is an "amount", and no "sells" (this field is not present), it is "entire instruction".
- * If there is no "amount", "sells" are compulsory, the default is "per instrument".

Selling behaviour options

This section shows how to specify what the withdrawal should sell.

Proportional amount sell over all instruments

You post an "amount" field, and no "sells". The system will create a withdrawal that will proportion the amount over available instruments. If there is only one instrument available, the system will sell the entire amount from that instrument. It will pay the specified amount to the investor.

Example JSON request:

```
{
  "workIn" : "amount", # Optional, but clarifies that this is the default
  "captureBy" : "entire instruction", # Optional, but clarifies that this is the default
  "amount" : 1234.56,
  "currency" : "ZAR",
  "status" : "Authorised",
  "bankDetails" : {"accountNumber" : "1234567890"}
}
```

Proportional percentage sell over all instruments

You post the "instructionPercentage" field, and no "sells". The system will create a withdrawal that the provided percentage over all available instruments. It will pay the proceeds received from the instrument dealers to the investor.

Example JSON request:

```
{
  "workIn" : "units", # Compulsory for this type of withdrawal
  "captureBy" : "entire instruction", # Compulsory for this type of withdrawal.
  "instructionPercentage" : 40, # Compulsory for this type of withdrawal.
  "bankDetails" : {"accountNumber" : "1234567890"}
}
```

Sell per instrument by amount

You post "amount" fields with your "sells". The system will create a withdrawal that sells each instrument by the specified amount for each sell. It will pay the total of all the amounts of the sells to the investor.

Example JSON request:

```
{
  "workIn" : "amount", # Optional, but clarifies the default.
  "captureBy" : "per instrument", # Optional, but clarifies the default.
  "amount" : 1234.56, # Optional. But validated to be = to the sum of amounts on the sells
  "bankDetails" : {"accountNumber" : "1234567890"},
  "sells" : {
    "INSTRMT1" : 234.56,
    "INSTRMT2" : 1000.00
  }, # Compulsory per instrument. Format: {"Instrument code", amount value}
}
```

Sell per instrument by percentage

You post "percentage" fields with your "sells". The system will create a withdrawal that sells the specified percentage of available units in each instrument. It will pay the sum of the proceeds for each sell received from the instrument dealers to the investor.

Example JSON request:

```
{
  "workIn" : "units", # Compulsory
  "captureBy" : "per instrument", # Compulsory
  "bankDetails" : {"accountNumber" : "1234567890"},
  "sells" : {
    "INSTRMT1" : 100.0,
    "INSTRMT2" : 50.0
  }, # Compulsory per instrument. Format: {"Instrument code", amount value}
}
```

The bank account

For creating a withdrawal, we have 2 options: use an existing bank account or create a new bank account.

The system will check for the existence of a bank account matching the “accountNumber” on the contract owner. If none is found, the system will check for a third party bank account matching the “accountNumber”. If found and the API user does not have the permission “Third party withdrawal”, an error is raised. If a matching bank account is found, this bank account is used and no further “bankAccount” fields are used. This means that if the user of the API intends to use an existing bank account, only the “accountNumber” has to be provided and all the other fields become optional. Some examples of this are in the examples above.

Please use the Entities API to update bank accounts.

To create a new bank account with a withdrawal, the example below shows what fields are available.

Example JSON showing optional fields that can be specified when creating a new bank account.

Example Partial JSON request:

```
"bankDetails" :
{
  "holderName" : "John Doe",
  "accountNumber" : "1234567890",
  "branchCode" : "222626",
  "type" : "Current",
  "bankName" : "First National Bank",
  "branchName" : "Centurion", # Optional
  "currencyIdentifier" : "zar", # Optional, defaults to preferred currency
  "swiftBankIdentifierCode" : "777", # Optional
  "intermediaryName" : "NEDWXYZYYYY", # Optional
  "internationalBankAccountNumber" : "GB82WEST12345698765432" # Optional
}
```

Currency

The currency of a withdrawal is the same as the currency of the bank account (that will be paid to) on a withdrawal. All amount fields are interpreted in this currency. The system currently does not support a currency on a withdrawal that is different from the currency of a bank account. The “currencyIdentifier” in the “bankDetails” field has to be provided if the denominated currency of the bank account being paid to determines the currency of the withdrawal.

If the denominated currency of the bank account being paid to is not correct on the system, the payment instruction sent to the payment agent (or the bank directly) will be received in

the contract preferred currency. This simply implies that the bank will do foreign exchange outside of the system.

Notes:

Possible account types: "**Current**", "**Transmission**" or "**Savings**".

Possible bank names: as configured under FinWorks system configuration -> Banks.

Currency identifier as configured under FinWorks system configuration.

"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" and "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 supportInstruction contains Withdrawal

The value for each sell depends on presence of other fields as follows:

Field	Specified	Sells value meaning
work in amount or units	no	An amount for each sell in the same currency as specified in the JSON request.
work in amount or units	yes with value "instruction"	sells not allowed to be specified as. Applies "instruction %" percentage to all instruments
work in amount or units	yes with value "instrument"	sell required. The meaning of the value is determined by "capture unit type" as follows: "percent" - value = percent of instrument to sell. "unit" - value = no. of units to instrument to sell.

JSON response:

HTTP: 201

{"Unique Id" : 12345678}

Error examples:

HTTP response	Code	Description
400	AccountInactiveError	Account is inactive
404	AccountNotFound	Could not find account
400	MalformedRequest	Malformed request. Amount value "-100.00" must be greater than 0
400	MalformedRequest	Malformed request. Invalid currency "LIBRE"
400	MalformedRequest	Malformed request. Field "amount" is compulsory
400	MalformedRequest	Malformed request. Field "bankDetails" is compulsory
400	MalformedRequest	Malformed request. Bank does not exist "Invalid bank"
400	MalformedRequest	Malformed request. Field "accountNumber" is compulsory
400	AvailableAmount Exceeded	There is not enough funds available to sell the requested amount
400	InvalidStatus	"AnInvalidStatus" is not a valid instruction status
400	InstrumentNotAvailable	Instrument with code: "INSTRMT3" could not be found.
400	InstrumentNotAvailable	Instrument with code: "INSTRMT4" is not available for Investor Account: "accountNumber"
400	NoAvailableHoldings	There are no available holdings in instrument: "INSTRMT1"
400	AvailableAmount Exceeded	There are not enough funds available in instrument: "INSTRMT1" to sell the requested amount

400	InvalidSellPercentage	The value on an instrument sell is not a valid percentage. Must be between 0 and 100 inclusive.
400	InvalidAmountComparisonWithSells	The total sell amount of all instruments does not match the optional amount specified

8 Individual statement

GET:

/api/accounts/{accountNumber}/statement?
start_date={startDate}&end_date={endDate}§ions={list-of-section-
names}&add_password={boolean}'

Content-Type: application/pdf

Example request:

/api/accounts/ACC123/statement?start_date=2005-01-01&end_date=2005-01-31

JSON response:

HTTP: 200

Content-Type: application/pdf

Notes:

Response contains pdf binary data.

All fields are mandatory except for 'sections' and 'add_password'

sections = comma separated list of section names

URLs must be *percent encoded*, eg. space = %20, comma = %2C

add_password = 'true' or 'false'

if add_password is not specified, a password will be added to the document based on the value of the "Password protect PDF documents?" configuration setting.

Error examples:

HTTP response	Code	Description
400	Malformed request	Invalid date
500	MultipleAccountsFound	Found more than one matching account
400	AccountInactiveError	Account is inactive
404	AccountNotFound	Could not find account

Available sections:

- Account details
- Advisor details
- Advisor fee details
- Beneficiary details
- Investment Summary For Statement Period
- Investment Allocation
- Asset Allocation
- Default investment portfolio
- Investment summary since inception
- Investment Summary Graph
- Investment return
- Investment return graph
- Monthly market values
- Instrument performance
- Instrument cash flows
- Instrument cash flow summary
- Combined cash instrument transactions per currency
- Account holdings summary
- Account holdings summary (opening balance)
- Insurance instrument benefits
- Benefit description
- Currency exposure
- Instructions
- Cash flows
- Transaction history
- Total expense ratio
- Administration fee rates per year
- Fees
- Statement notes
- Income Review
- Bond Maturity
- Statement footer
- Statement cover letter
- Net Replacement Ratio

9 Combined statement

Multiple accounts statement for a particular account owner.

GET: /api/accounts/statement?
search_key={searchKey}&start_date={startDate}&end_date={endDate}§ions={list-of-section-names}&add_password={boolean}

Content-Type: application/pdf

Notes:

Response contains pdf binary data.

All fields are mandatory except for 'sections' and 'add_password'

search_key = company registration number or person identity number.

sections = comma separated list of section names

URLs must be percent encoded, eg. space = %20, comma = %2C

add_password = 'true' or 'false'

if add_password is not specified, a password will be added to the document based on the value of the "Password protect PDF documents?" configuration setting.

Example request:

/api/accounts/statement?search_key=123456789&start_date=2005-01-01&end_date=2005-01-31§ions=Account%20details%2CBeneficiary%20details

JSON response:

HTTP: 200

Content-Type: application/pdf

Error examples:

HTTP response	Code	Description
400	Malformed request	Invalid date
403	Security	
404	AccountOwnerNotFound	Could not find account owner
404	AccountNotFound	Could not find account (** if there are no active accounts for the account owner)
500	MultipleAccountOwnersFound	Found more than one matching account owner

Available sections:

- Account details
- Advisor details
- Advisor fee details
- Beneficiary details
- Investment Summary For Statement Period
- Investment Allocation
- Asset Allocation
- Default investment portfolio
- Investment summary since inception
- Investment Summary Graph
- Investment return
- Investment return graph
- Monthly market values
- Instrument performance
- Instrument cash flows
- Instrument cash flow summary
- Combined cash instrument transactions per currency
- Account holdings summary
- Account holdings summary (opening balance)
- Insurance instrument benefits
- Benefit description
- Currency exposure
- Instructions
- Cash flows
- Transaction history
- Total expense ratio
- Administration fee rates per year
- Fees
- Statement notes
- Income Review
- Bond Maturity
- Statement footer
- Statement cover letter
- Net Replacement Ratio
- Combined holdings summary
- Combined holdings summary graph
- Combined Asset Allocation

10 Holdings

GET: /api/accounts/{accountNumber}/holdings (as of today)

GET: /api/accounts/{accountNumber}/holdings?date={date} (as of date specified)

Note: Date must be ISO format string YYYY-MM-DD

Example JSON response:

```
[ {
  "Contract id": 123123123,
  "Contract number": "POL3",
  "Client account id": 48044772352,
  "Instrument id": 1122334455,
  "Instrument code": "COR01",
  "Instrument name": "Coronation Balanced Fund",
  "Instrument account number": "0987654321",
  "Price date": "2022-02-01",
  "Date": "2022-02-01",
  "Units": {
    "type": "Unit",
    "Instrument id": 1122334455,
    "value": "440"
  },
  "Market Value in System Currency": {
    "type": "Money",
    "value": "440",
    "currency": "ZAR"
  },
  "Market Value in Fund Currency": {
    "type": "Money",
    "value": "440",
    "currency": "ZAR"
  },
  "Accrued fees": {
    "type": "Money",
    "value": "0.60",
    "currency": "ZAR"
  },
}
```

```

    "Accrued interest": {
      "type": "Money",
      "value": "0.00",
      "currency": "ZAR"
    },
    "Latest available price": {
      "type": "Price",
      "Instrument id": 1122334455,
      "value": "0.60",
      "currency": "ZAR"
    }
  }
}

```

Error examples:

HTTP response	Code	Description
400	AccountInactiveError	Account is inactive
404	AccountNotFound	Could not find account

11 Available Holdings

GET: /api/accounts/{accountNumber}/availableHoldings (as of today)

GET: /api/accounts/{accountNumber}/availableHoldings?date={date} (as of date specified)

Note: Date must be ISO format string YYYY-MM-DD

Example JSON response:

```

[ {
  "Contract id": 123123123,
  "Contract number": "POL3",
  "Client account id": 48044772352,
  "Instrument id": 1122334455,
  "Instrument code": "COR01",
  "Instrument name": "Coronation Balanced Fund",
  "Instrument account number": "0987654321",

```

```

    "Price date": "2022-02-01",
    "Date": "2022-02-01",
    "Units": {
      "type": "Unit",
      "Instrument id": 1122334455,
      "value": "440"
    },
    "Market Value in System Currency": {
      "type": "Money",
      "value": "440",
      "currency": "ZAR"
    },
    "Market Value in Fund Currency": {
      "type": "Money",
      "value": "440",
      "currency": "ZAR"
    },
    "Accrued fees": {
      "type": "Money",
      "value": "0.60",
      "currency": "ZAR"
    },
    "Latest available price": {
      "type": "Price",
      "Instrument id": 1122334455,
      "value": "0.60",
      "currency": "ZAR"
    }
  }
}

```

Error examples:

HTTP response	Code	Description
400	AccountInactiveError	Account is inactive
404	AccountNotFound	Could not find account



12 Advisor accrued fees

GET: /api/accounts/advisorAccruedFees?

code={code}&start_date={startDate}&end_date={endDate}

Example JSON response:

```
{
  "Instrument code" : "ZAR",
  "Investor name" : "ivan I",
  "Fee type" : "Annual adviser fee",
  "Amount in system currency" : {
    "currency" : "ZAR",
    "value" : "10.49",
    "type" : "Money"
  },
  "Amount in fee currency" : {
    "currency" : "ZAR",
    "value" : "10.49",
    "type" : "Money"
  },
  "Adviser code" : "AGFSPB",
  "Instrument name" : "Cash in ZAR",
  "Adviser name" : "Pompies P",
  "Transaction date" : "2005/06/12",
  "Account number" : "POL3",
  "Product" : "Discretionary",
  "Preferred currency" : "ZAR",
  "Fee currency" : "ZAR"
}
```

Error examples:

HTTP response	Code	Description
404	AdvisorNotFound	Invalid advisor code

13 Advisor fees (taken)

GET: /api/accounts/advisorFees?

code={code}&start_date={startDate}&end_date={endDate}

Example JSON response:

```
{
  "Instrument Code" : "ZAR",
  "Investor Name" : "ivan, I",
  "Investor ID" : "6611235090089",
  "Units" : {
    "currency" : "ZAR",
    "value" : "10.49",
  }
}
```

```

    "type" : "Money"
  },
  "Adviser code" : "AGFSPB",
  "Adviser name" : "Pompies P",
  "Transaction Date" : "2005-06-12",
  "Adviser Fee Period" : "2005/05/13 to 2005/06/12",
  "Amount" : {
    "currency" : "ZAR",
    "value" : "10.49",
    "type" : "Money"
  },
  "Product" : "Discretionary",
  "Instrument" : "Cash in ZAR",
  "Fee Type" : "Ongoing Adviser Fee",
  "Account Number" : "POL3",
  "Price" : {
    "value" : "1.0000",
    "currency" : "ZAR",
    "Instrument id" : <cash instrument id>,
    "type" : "Price"
  }
}

```

Error examples:

HTTP response	Code	Description
404	AdvisorNotFound	Invalid advisor code

14 Policy document

GET: /api/accounts/{accountNumber}/policyDocument?add_password={boolean}'

Content-Type: application/pdf

Example request:

/api/accounts/ACC123/policyDocument

/api/accounts/ACC123/policyDocument?add_password=true

Note: if add_password is not specified, a password will be added to the document based on the value of the "Password protect PDF documents?" configuration setting.

Notes:

Response contains pdf binary data.

All fields are mandatory except for 'add_password'

add_password = 'true' or 'false'

if add_password is not specified, a password will be added to the document based on the value of the "Password protect PDF documents?" configuration setting.

JSON response:

HTTP: 200

Content-Type: application/pdf

Response contains pdf binary data.

Error examples:

HTTP response	Code	Description
500	MultipleAccountsFound	Found more than one matching account
404	AccountNotFound	Could not find account

15 Account details

GET: /api/accounts/{accountNumber}

Example request:

/api/accounts/ACC123

Example JSON response:

```
{
  "Unique Id": 123456789014,
  "Product": "Discretionary",
  "Account Number": "ACC123",
  "Contract Number": "ACC123",
  "Model Portfolio Name": "Clever Conservative",
  "Owners" : [
    {
      "Entity": "/api/entities/123456789015",
      "Entity Id": 123456789015,
      "Name": "Mr Gene Fraser"
    }
  ],
  "Advisor" : {
    "Entity": "/api/entities/378231296", //To be removed
    "Advisor Entity Unique Id": 378231296, //New
    "Name": "John Clever", //To be removed
    "Advisor Name": "John Clever", //New
    "Code": "ADV001", //To be removed
    "Advisor Code": "ADV001", //New
    "Brokerage": "/api/entities/275446528", //To be removed
    "Brokerage Entity Unique Id": 275446528, //New
    "Brokerage Code": "CLE001",
    "Brokerage Name": "Clever Advice"
  },
  "Beneficiaries" : [
    {
      "Entity": "/api/entities/123456789017",
      "Entity Id": 123456789017,
      "Name": "Mr John Seymor",
      "Benefit": {
        "type": "Percentage",
        "value": "100.00"
      }
    }
  ],
  "Product": "Itransact Securities Investment Plan",
  "Product id": 160275969,
  "Description": "Itransact Securities Investment Plan",
  "Status": {
```

```

    "active": "true",
    "identifier": "Existing business"
  },

  "Communication Options": {
    "communicationPreference": "None",
    "transactionNotificationPreferences": {
      "CompleteSwitchInstruction": "None",
      "SubmitUnitTransferOutInstruction": "None",
      "SubmitReinvestmentSwitchInstruction": "None",
      "SubmitSwitchInstruction": "None",
      "SubmitUnitTransferInInstruction": "None",
      "SubmitWithdrawal": "None",
      "CompleteReinvestmentSwitchInstruction": "None",
      "CompleteWithdrawal": "None",
      "CompleteUnitTransferOutInstruction": "Both",
      "ApprovedChangeRequest": "Email",
      "SuccessfulLogin": "Sms",
      "SubmitInvestment": "None",
      "CompleteInvestment": "None",
      "CompleteUnitTransferInInstruction": "None"
    },
    "sendCorrespondenceTo": "None",
    "automaticallyEmailStatement": false
  }, /New (2025-01)

  "Market value": {
    "currency": "ZAR",
    "amount": "77444.0600",
    "accrued fees": "27.9200",
    "date": "2023-06-29"
  },
  "Representative" : {
    "Entity" : "/api/entities/123456789018",
    "Name" : "Ms Sarah Ann Fraser"
  },
  "Client Accounts" : [ {
    "Unique Id" : 12345,
    "Name" : "Default",
    "Model Portfolio Id" : ,
    "Model Portfolio Name" : ""
  }, {
    "Unique Id" : 34567,
    "Name" : "Savings - ABC Conservative",
    "Model Portfolio Id" : 45678,
    "Model Portfolio Name" : "ABC Conservative"
  } ]
}
New fields: Added Sept 2024

```

Deprecated fields: To be removed Nov 2024

Error examples:

HTTP response	Code	Description
404	AccountNotFound	Could not find account

16 Update Account details //New

POST: /api/accounts/{accountNumber}

Example request:

```
{
  "contractStatus": "New business",
  "distributionOption": "reinvestDistribution",
  "advisor": {
    "advisorCode": "AGFSPB",
    "brokerageCode": "000017"
  },
  "communicationOptions": {
    "communicationPreference": "No consent given",
    "sendCorrespondenceTo": "None",
    "automaticallyEmailStatement": false,
    "transactionNotificationPreferences": {
      "SuccessfulLogin": "SMS",
      "SubmitInvestment": "Email"
    }
  }
}
```

With this call one can update the following:

- distributionOption : can be one of: reinvestDistribution,payoutDistribution or leaveDistributionInCash
- contract status: One of the statuses in System settings/ Account statuses. The transition from the existing status to the new status should also be valid
- advisor
- communication options:

communicationPreference: valid options: Post, Email, No consent given, Opted-out

sendCorrespondenceTo: valid options:'None';'Both';'Client';'Adviser';'Advisor';

17 Available instruments

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

Example request:

/api/accounts/ACC123/instruments

Example JSON response:

```
{
  "data": [
    {
      "Instrument id": 390850560,
      "Code": "ZAR",
      "Name": "Cash in ZAR",
      "Instrument provider": ""
    },
    {
      "Instrument id": 517341184,
      "Code": "COREQ",
      "Name": "Coronation Equity Fund",
      "Instrument provider": "Coronation"
    }
  ],
  "size": 2
}
```

Error examples:

HTTP response	Code	Description
404	AccountNotFound	Could not find account

18 Documents

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

Example request:

/api/accounts/ACC123/documents

Example JSON response:

```
{
  "data": [
    {
      "Document id":      "/api/documents/825280512",
      "Document Unique Id": 825280512,
      "Type":             "Contribution Certificate",
      "Label":             "a document",
      "Date":              "2021/08/11"
    },
    {
      "Document id":      "/api/documents/214897664",
      "Document Unique Id": 214897664,
      "Type":             "Application Form",
      "Label":             "another document",
      "Date":              "2021/08/11"
    }
  ],
  "size": 2
}
```

Error examples:

HTTP response	Code	Description
404	AccountNotFound	Could not find account

19 Instructions

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

For active instructions only:

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

Example request:

/api/accounts/ACC123/instructions

Example JSON response:

```
{
  "data": [
    {
      "Instruction id": "/api/instructions/516098816",
      "Instruction Unique Id": 516098816,
      "Type": "Once Off Investment",
      "Status": "Submitted",
      "Amount": "R 2,100.00",
      "Contributions": [
        {
          "Type": "Buy",
          "Sell Instrument": "",
          "Buy Instrument": "aFundCode",
          "Amount": "R 2,100.00",
          "Price": "2.1000",
          "Units": "1000.00",
          "Status": "Active"
        }
      ]
    },
    {
      "Instruction id": "/api/instructions/418818304",
      "Instruction Unique Id": 418818304,
      "Type": "Once Off Withdrawal",
      "Status": "New",
      "Amount": "R 0.00",
      "Contributions": [
        {
          "Type": "Sell",
          "Sell Instrument": "aFundCode",
          "Buy Instrument": "",
          "Amount": "0.00",
          "Price": "2.1000",
          "Units": "0.00",
          "Status": "Active"
        }
      ]
    }
  ]
}
```

```
    ],  
    "size": 2  
}
```

Error examples:

HTTP response	Code	Description
404	AccountNotFound	Could not find account

20 Investment Summary Since Inception

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

- as of today

GET: /api/accounts/{accountNumber}/investmentSummarySinceInception?end_date={date}

- as of date specified

Note: Date must be ISO format string YYYY-MM-DD

Example JSON response:

```
{
  "Withdrawals / Transfers Out": {
    "type": "Money",
    "currency": "USD",
    "value": "0.00"
  },
  "End date": "2022-06-01",
  "IRR Cumulative": {
    "type": "Percentage",
    "value": "-25.07"
  },
  "Start date": "2022-01-01",
  "Closing balance": {
    "type": "Money",
    "currency": "USD",
    "value": "231844.20"
  },
  "Opening balance": {
    "type": "Money",
    "currency": "USD",
    "value": "308826.60"
  },
  "Gain/Loss": {
    "type": "Money",
    "currency": "USD",
    "value": "-76982.40"
  },
  "Investments / Transfers In": {
    "type": "Money",
    "currency": "USD",
    "value": "0.00"
  },
  "IRR Annualised": {
    "type": "Percentage",
    "value": "-50.00"
  }
}
```

Error examples:

HTTP response	Code	Description
400	AccountInactiveError	Account is inactive
404	AccountNotFound	Could not find account

21 Investment Summary for Period

GET: `api/accounts/{accountNumber}/investmentSummaryForPeriod?
start_date={startDate}&end_date={endDate}`

Note: Date must be ISO format string YYYY-MM-DD

Example JSON response:

```
{
  "Withdrawals / Transfers Out": {
    "type": "Money",
    "currency": "USD",
    "value": "0.00"
  },
  "End date": "2022-06-01",
  "IRR Cumulative": {
    "type": "Percentage",
    "value": "-25.07"
  },
  "Start date": "2022-01-01",
  "Closing balance": {
    "type": "Money",
    "currency": "USD",
    "value": "231844.20"
  },
  "Opening balance": {
    "type": "Money",
    "currency": "USD",
    "value": "308826.60"
  },
  "Gain/Loss": {
    "type": "Money",
    "currency": "USD",
    "value": "-76982.40"
  },
  "Investments / Transfers In": {
    "type": "Money",
    "currency": "USD",
    "value": "0.00"
  },
  "IRR Annualised": {
    "type": "Percentage",
    "value": "-50.00"
  }
}
```

Error examples:

HTTP response	Code	Description
400	AccountInactiveError	Account is inactive
404	AccountNotFound	Could not find account

22 Products

GET: </api/accounts/products>

Example request:

</api/accounts/products>

Example JSON response:

```
{
  "products":
    [
      {
        "Product id": 609157888,
        "Product name": "Discretionary"},
      {"Product id": 896356096,
        "Product name": "Retirement annuity"}
    ]
}
```

23 Product available instruments

GET: /api/accounts/products/{uniqueId}/availableInstruments

Example request:

/api/accounts/products/16420686081/availableInstruments

Example JSON response:

```
{
  "data": [
    {
      "Instrument id": 52565504,
      "Code": "COREQ",
      "Name": "Coronation Equity Fund",
      "Instrument provider": "Coronation"
    },
    {
      "Instrument id": 389797632,
      "Code": "ZAR",
      "Name": "Cash in ZAR",
      "Instrument provider": ""
    }
  ],
  "size": 2
}
```

Error examples:

HTTP response	Code	Description
500	NotFound	Invalid uniqueId

24 Transactions for Period

GET: `api/accounts/{accountNumber}/transactionsForPeriod?`
`start_date={startDate}&end_date={endDate}`

Note: Date must be ISO format string YYYY-MM-DD

Example JSON response:

```
{
  "data": [
    {
      "Description": "\"Buy (Once Off Investment)\"",
      "Date": "2005-05-13",
      "Instrument code": "AGMF",
      "Transaction id": 375453696,
      "Instrument name": "Allan Gray Money Market Fund",
      "Amount": {
        "type": "Money",
        "currency": "ZAR",
        "value": "600.00"},
      "Balance": {
        "type": "Money",
        "currency": "ZAR",
        "value": "600.00"}
    },
    {
      "Description": "\"Sell (Once Off Withdrawal)\"",
      "Date": "2005-05-16",
      "Instrument code": "AGMF",
      "Transaction id": 948439808,
      "Instrument name": "Allan Gray Money Market Fund",
      "Amount": {"type": "Money", "currency": "ZAR", "value": "-360.00"},
      "Balance": {"type": "Money", "currency": "ZAR", "value": "240.00"}
    },
    {
      "Description": "\"Deposit matched\"",
      "Date": "2005-05-13",
      "Instrument code": "ZAR",
      "Transaction id": 110048000,
      "Instrument name": "Cash in ZAR",
      "Amount": {"type": "Money", "currency": "ZAR", "value": "600.00"},
      "Balance": {"type": "Money", "currency": "ZAR", "value": "600.00"}
    },
    {
      "Description": "\"Buy in Allan Gray Money Market Fund (AGMF)\"",
      "Date": "2005-05-13",
      "Instrument code": "ZAR",
      "Transaction id": 271667968,
      "Instrument name": "Cash in ZAR",
      "Amount": {"type": "Money", "currency": "ZAR", "value": "-600.00"},
      "Balance": {"type": "Money", "currency": "ZAR", "value": "0.00"}
    }
  ]
}
```

```
{
  "Description": "\"Sell of Allan Gray Money Market Fund\"",
  "Date": "2005-05-16",
  "Instrument code": "ZAR",
  "Transaction id": 732626944,
  "Instrument name": "Cash in ZAR",
  "Amount": {"type": "Money", "currency": "ZAR", "value": "360.00"},
  "Balance": {"type": "Money", "currency": "ZAR", "value": "360.00"}
},
{
  "Description": "\"Payment 2-POL3      to ***566 (Capitec-ZAR)\"",
  "Date": "2005-05-21",
  "Instrument code": "ZAR",
  "Transaction id": 601679872,
  "Instrument name": "Cash in ZAR",
  "Amount": {"type": "Money", "currency": "ZAR", "value": "-360.00"},
  "Balance": {"type": "Money", "currency": "ZAR", "value": "0.00"}
}
],
"size": 6
}
```

Error examples:

HTTP response	Code	Description
400	AccountInactiveError	Account is inactive
404	AccountNotFound	Could not find account