**FUNCTIONAL SPECIFICATION DOCUMENT**

**Back-End API & Web Platform**

**Version 1.0**

**COPYRIGHT NOTICE**

**Copyright © (2016) by PT. Code Development Indonesia**

All rights reserved. This material is confidential and proprietary to PT. Code Development Indonesia and no part of this material should be reproduced, published in any form by any means, electronic or mechanical including photocopy or any information storage or retrieval system nor should the material be disclosed to third parties without the express written authorization of PT. Code Development Indonesia.

This Functional Specification Document v 1.0 is released for Back-End API & Web Platform, PT. Code Development Indonesia (CODE.ID).

|  |  |  |
| --- | --- | --- |
| Prepared by |  |  |

**REVISION HISTORY**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Revision No.** | **Revision Date** | **Author** | **Sections Changed** | **Page #** | **Descriptions** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BR Reff** | **Description** | **FR ID** | **Functional Requirement Description** | **UC ID** | **Use Case** |
| BR-1 | Has feature to handle investor registration from Online Distributors | FR-1 | Back-End API shall be able to save multiple accounts for a single investor if registered from different Online Distributor and pull Investor’s data from Dukcapil | 1.01.01 | First time registration |
| 1.01.02 | Pull Investor data from Dukcapil |
| FR-2 | Back-End API shall be able to create task and send email registration request notification | 1.01.03 | Send email registration notification to Marketing |
| 1.01.04 | Send email registration notification to Compliance |
| FR-3 | Back-End API shall be able to update approval status of registration data and send email notification registration rejected | 1.01.05 | Registration approval by Marketing |
| 1.01.06 | Registration approval by Compliance |
| 1.01.07 | Send email registration rejected to Investor |
| FR-4 | Back-End API shall be able to provide approved registration data for Back-Office | 1.01.08 | Provide approved registration data |
| FR-5 | Back-End API shall be able to save SID and IFUA number provided from Back-Office and send email notification registration approved | 1.01.09 | Save investor data from Back-Office |
| 1.01.10 | Send email registration approved to Investor |
| 1.01.11 | Provide Investor data for Online Distributor |
| BR-2 | Has feature to handle update data investor from Online Distributors | FR-6 | Back-End API shall be able to save investor’s updated data without replacing the old data | 1.01.12 | Save investor updated data from Online Distributor |
| FR-7 | Back-End API shall be able to create task and send email update data request notification | 1.01.13 | Send email update data notification to Marketing |
| 1.01.14 | Send email update data notification to Compliance |
| FR-8 | Back-End API shall be able to update approval status of investor’s update data and send email notification update data rejected | 1.01.15 | Updated data approval by Marketing |
| 1.01.16 | Updated data approval by Compliance |
| 1.01.17 | Send email update data rejected to Investor |
| FR-9 | Back-End API shall be able to provide approved investor’s updated data for Back-Office | 1.01.18 | Provide approved updated data |
| FR-10 | Back-End API shall be able to save IFUA number provided from Back-Office and send email notification update data approved | 1.01.19 | Send email update data approved to Investor |
| BR-3 | Has feature to handle Subsciption Transaction where Clients can buy Bahana’s product from Online Distributor | FR-11 | Back-End API shall be able to save subscription data from Online Distributors | 1.01.20 | Save subscription data |
| FR-12 | Back-End API shall be able to provide subscription data for Back-Office | 1.01.21 | Provide subscription data |
| FR-13 | Back-End API shall be able to save update unit allocation and NAV data for subscription transaction provided from Back-Office and save subscription data from send by Back-Office | 1.01.22 | Save subscription unit allocation and NAV data |
| FR-14 | Back-End API shall be able to provide update unit allocation for subscription transaction for Online Distributors | 1.01.23 | Provide subscription data for Online Distributor |
| FR-15 | Back-End API shall be able to send email subscription status | 1.01.24 | Send email subscription successful to Investor |
| 1.01.25 | Send email subscription rejected to Investor and Marketing |
| BR-4 | Has feature to handle Redemption Transaction where Clients can sell their products from Online Distributor | FR-16 | Back-End API shall be able to save redemption data from Online Distributors | 1.01.26 | Save redemption data |
| FR-17 | Back-End API shall be able to provide redemption data for Back-Office | 1.01.27 | Provide redemption data |
| FR-18 | Back-End API shall be able to save update unit allocation and NAV data for redemption transaction provided from Back-Office | 1.01.28 | Save redemption unit allocation and NAV data |
| FR-19 | Back-End API shall be able to provide update unit allocation for redemption transaction for Online Distributors | 1.01.29 | Provide redemption data for Online Distributor |
| FR-20 | Back-End API shall be able to send email redemption status | 1.01.30 | Send email redemption successful to Investor |
| 1.01.31 | Send email redemption rejected to Investor and Marketing |
| BR-5 | Has feature to handle Switching Transaction where Clients can change product that they have to another product | FR-21 | Back-End API shall be able to save switching data from Online Distributors | 1.01.32 | Save switching data |
| FR-22 | Back-End API shall be able to provide switching data for Back-Office | 1.01.33 | Provide switching data |
| FR-23 | Back-End API shall be able to save update unit allocation and NAV data for switching transaction provided from Back-Office | 1.01.34 | Save switching’s unit allocation and NAV data |
| FR-24 | Back-End API shall be able to provide update unit allocation for switching transaction for Online Distributors | 1.01.35 | Provide switching data for Online Distributor |
| FR-25 | Back-End API shall be able to send email switching status | 1.01.36 | Send email switching successful to Investor |
| 1.01.37 | Send email switching rejected to Investor and Marketing |
| BR-6 | Has feature to reconcile Investor Unit Holding where Back-Office compare Investor portfolio (Products and unit count they have) in Back-Office with Investor portfolio in Back-End | FR-26 | Back-End API shall be able to calculate margin of difference between unit holding in Back-End and unit holding provided from Back-Office and log the margin | 1.01.38 | Reconcile unit holding |
| 1.01.39 | Log margin of difference |
| BR-7 | Has feature to Revise Transaction where Back-Office can edit transaction data in Back-End | FR-27 | Back-End API shall be able to revise transaction that provided by Back-Office | 1.01.40 | Revise transaction |
| BR-8 | Has feature to Void Transaction where Back-Office can delete transaction in Back-End | FR-28 | Back-End API shall be able to void transaction provided from Back-Office | 1.01.41 | Void Transaction |
| BR-9 | Has feature to provide company and products information from Back-Office to Online Distributor | FR-29 | Back-End API shall be able to receive master data from Back-Office | 1.01.42 | Receive master data |
| FR-30 | Back-End API shall be able to provide master data for Online Distributors | 1.01.43 | Provide master data |
| BR-10 | Has feature to periodically send email (Birthday Greeting, Reminder of KTP Expiry) and data summary (Product Summary, Transaction Summary and Transaction Log Report) | FR-31 | Back-End API shall be able to send email about data summary periodically | 1.01.44 | Cronjob to send Portfolio Summary to Investor |
| 1.01.45 | Cronjob to send Transaction Summary to Marketing |
| 1.01.46 | Cronjob to send email Birthday Greeting |
| 1.01.47 | Cronjob to send email Reminder of KTP Expiry |
| 1.01.48 | Cronjob to send Unit Holding Reconciliation Log Report to Marketing and Operation |
| 1.01.49 | Cronjob to send Revise Transaction Log Report to Marketing and Operation |
| 1.01.50 | Cronjob to send Void Transaction Log Report to Marketing and Operation |
| BR-11 | Web Platform to maintain master data, admin users, data approval, task notification and report generation. | FR-32 | Web Platform shall be able to maintain master data | 1.01.51 | View master data |
| 1.01.52 | Create master data |
| 1.01.53 | Edit master data |
| 1.01.54 | Delete master data |
| FR-33 | Web Platform shall be able to maintain admin users | 1.01.55 | View user |
| 1.01.56 | Create user |
| 1.01.57 | User creation approval |
| 1.01.58 | Change password |
| 1.01.59 | Login by email |
| 1.01.60 | Logout |
| FR-34 | Web Platform shall be able to maintain investor data approval | 1.01.61 | View registration data |
| 1.01.62 | Registration data approval |
| 1.01.63 | View updated data and old data |
| 1.01.64 | Updated data approval |
| FR-35 | Web Platform shall be able to view transaction data | 1.01.65 | View transaction data |
| FR-36 | Web Platform shall be able to maintain report request approval and generate report | 1.01.66 | View report |
| 1.01.67 | Create task request report |
| 1.01.68 | Generate report |
| 1.01.69 | View report request |
| 1.01.70 | Report request approval |
| 1.01.71 | Cronjob to delete expire report |

**TABLE OF CONTENTS**

[1. Introduction 12](#_Toc501097880)

[1.1. Background of Request 12](#_Toc501097881)

[1.1.1. Existing Process Diagram 12](#_Toc501097882)

[1.1.2. Existing Process Description 12](#_Toc501097883)

[1.1.3. Future Process Diagram 13](#_Toc501097884)

[1.1.4. Future Process Description 13](#_Toc501097885)

[1.2. Scope of Work 14](#_Toc501097886)

[1.2.1. Scope of Work 14](#_Toc501097887)

[1.2.2. Design Constraint 16](#_Toc501097888)

[1.2.3. Consideration for Performance 16](#_Toc501097889)

[1.3. Dependent/Related Projects 16](#_Toc501097890)

[2. Business Process Overview 17](#_Toc501097891)

[2.1. Business Requirement (BR) 17](#_Toc501097892)

[2.2. Functional Requirements (FR) 17](#_Toc501097893)

[2.3. Non Functional Requirement (NFR) 19](#_Toc501097894)

[2.3.1. Hardware Interfaces 19](#_Toc501097895)

[2.3.2. Software Interfaces 19](#_Toc501097896)

[2.3.3. System Interfaces 19](#_Toc501097897)

[2.4. Business Flow (BF) 20](#_Toc501097898)

[2.4.1. General Flow 20](#_Toc501097899)

[2.4.2. Detail Flow 21](#_Toc501097900)

[2.5. Assumptions 26](#_Toc501097901)

[3. Systems Overview 27](#_Toc501097902)

[3.1. System Context 27](#_Toc501097903)

[3.2. Impacted Systems 27](#_Toc501097904)

[3.3. Use Case Back-End 28](#_Toc501097905)

[3.3.1. Investor Registration 28](#_Toc501097906)

[3.3.2. Pull Investor data from Dukcapil 30](#_Toc501097907)

[3.3.3. Send email registration notification to Marketing 31](#_Toc501097908)

[3.3.4. Send email registration notification to Compliance 32](#_Toc501097909)

[3.3.5. Registration approval by Marketing 32](#_Toc501097910)

[3.3.6. Registration approval by Compliance 33](#_Toc501097911)

[3.3.7. Send email registration rejected to Investor 33](#_Toc501097912)

[3.3.8. Provide approved registration data 34](#_Toc501097913)

[3.3.9. Save investor data from Back-Office 36](#_Toc501097914)

[3.3.10. Send email registration approved to Investor 39](#_Toc501097915)

[3.3.11. Provide Investor data for Online Distributor 40](#_Toc501097916)

[3.3.12. Save investor updated data from Online Distributor 42](#_Toc501097917)

[3.3.13. Send email update data notification to Marketing 45](#_Toc501097918)

[3.3.14. Send email update data notification to Compliance 45](#_Toc501097919)

[3.3.15. Updated data approval by Marketing 46](#_Toc501097920)

[3.3.16. Updated data approval by Compliance 47](#_Toc501097921)

[3.3.17. Send email update data rejected to Investor 47](#_Toc501097922)

[3.3.18. Provide approved updated data 48](#_Toc501097923)

[3.3.19. Send email update data approved to Investor 50](#_Toc501097924)

[3.3.20. Save subscription data 50](#_Toc501097925)

[3.3.21. Provide subscription data 52](#_Toc501097926)

[3.3.22. Save subscription unit allocation and NAV data 53](#_Toc501097927)

[3.3.23. Provide subscription data for Online Distributor 53](#_Toc501097928)

[3.3.24. Send email subscription successful to Investor 54](#_Toc501097929)

[3.3.25. Send email subscription rejected to Investor and Marketing 55](#_Toc501097930)

[3.3.26. Save redemption data 55](#_Toc501097931)

[3.3.27. Provide redemption data 57](#_Toc501097932)

[3.3.28. Save redemption unit allocation and NAV data 58](#_Toc501097933)

[3.3.29. Provide redemption data for Online Distributor 59](#_Toc501097934)

[3.3.30. Send email redemption successful to Investor 60](#_Toc501097935)

[3.3.31. Send email redemption rejected to Investor and Marketing 61](#_Toc501097936)

[3.3.32. Save switching data 61](#_Toc501097937)

[3.3.33. Provide switching data 63](#_Toc501097938)

[3.3.34. Save switching unit allocation and NAV data 64](#_Toc501097939)

[3.3.35. Provide switching data for Online Distributor 65](#_Toc501097940)

[3.3.36. Send email switching successful to Investor 66](#_Toc501097941)

[3.3.37. Send email switching rejected to Investor and Marketing 67](#_Toc501097942)

[3.3.38. Reconcile unit holding 67](#_Toc501097943)

[3.3.39. Log margin of difference 68](#_Toc501097944)

[3.3.40. Revise transaction 68](#_Toc501097945)

[3.3.41. Void Transaction 69](#_Toc501097946)

[3.3.42. Receive master data 70](#_Toc501097947)

[3.3.43. Provide master data 71](#_Toc501097948)

[3.3.44. Cronjob to send Portfolio Summary to Investor 72](#_Toc501097949)

[3.3.45. Cronjob to send Transaction Summary to Marketing 73](#_Toc501097950)

[3.3.46. Cronjob to send email Birthday Greeting 73](#_Toc501097951)

[3.3.47. Cronjob to send email Reminder of KTP Expiry 74](#_Toc501097952)

[3.3.48. Cronjob to send Unit Holding Reconciliation Log Report to Marketing and Operation 74](#_Toc501097953)

[3.3.49. Cronjob to send Revise Transaction Log Report to Investor, Marketing and Operation 74](#_Toc501097954)

[3.3.50. Cronjob to send Void Transaction Log Report to Investor, Marketing and Operation 75](#_Toc501097955)

[3.4. Use Case User Action - CMS (Admin Panel) 76](#_Toc501097956)

[3.4.1. View Master Data 76](#_Toc501097957)

[3.4.2. Create Master Data 76](#_Toc501097958)

[3.4.3. Edit Master Data 77](#_Toc501097959)

[3.4.4. Delete Master Data 78](#_Toc501097960)

[3.4.5. View User 78](#_Toc501097961)

[3.4.6. Create user 79](#_Toc501097962)

[3.4.7. User creation approval 79](#_Toc501097963)

[3.4.8. Change password 80](#_Toc501097964)

[3.4.9. Login by email 81](#_Toc501097965)

[3.4.10. Logout 82](#_Toc501097966)

[3.4.11. View registration data 83](#_Toc501097967)

[3.4.12. Registration data approval 84](#_Toc501097968)

[3.4.13. View updated data and old data 85](#_Toc501097969)

[3.4.14. Update Data approval 86](#_Toc501097970)

[3.4.15. View transaction data 87](#_Toc501097971)

[3.4.16. View report 88](#_Toc501097972)

[3.4.17. Create task request report 89](#_Toc501097973)

[3.4.18. Generate report 92](#_Toc501097974)

[3.4.19. View Report Request 92](#_Toc501097975)

[3.4.20. Report Request approval 93](#_Toc501097976)

[3.4.21. Cronjob to delete expired report 93](#_Toc501097977)

**LIST OF TABLES**

[Table 1 Dependent/Related Projects 16](#_Toc499827086)

[Table 2 Requirement Summary 17](#_Toc499827087)

[Table 3 Functional Requirements 17](#_Toc499827088)

[Table 4 Hardware Interfaces 19](#_Toc499827089)

[Table 5 Software Interfaces 19](#_Toc499827090)

[Table 6 System Interfaces 19](#_Toc499827091)

[Table 7 Impacted Systems 27](#_Toc499827092)

**LIST OF FIGURES**

[Figure 1 General Flow 20](#_Toc499827185)

[Figure 2 Business Flow 1 – Account/KYC Registration 21](#_Toc499827186)

[Figure 3 Business Flow 2 – Account/KYC Update 22](#_Toc499827187)

[Figure 4 Business Flow 3 – Subscription 23](#_Toc499827188)

[Figure 5 Business Flow 4 - Redemption 24](#_Toc499827189)

[Figure 6 Business Flow 5 – Switching 25](#_Toc499827190)

[Figure 7 System Context 27](#_Toc499827191)

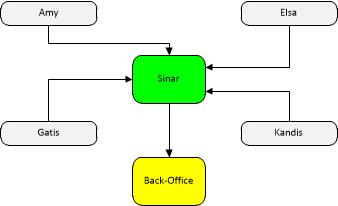
# Introduction

## Background of Request

Bahana TCW is one of the corporation that moved in the field of investment management and provide best innovative products for its clients. To make transactions with these products, clients have to go through a web called SiNar, register account there and then can proceed with transactions. Clients also have to access SiNar web to know what products provide by Bahana TCW.

To enable clients see product’s info and make transactions not only through SiNar will be in need for another system. Back-End API & Web Platform is an interface that connect Online Distributors with Back-Office Bahana. With Back-End API to provide product’s info for Online Distributor, Online Distributor will then can show product’s info to more clients. Back-End API also enable clients form Online Distributors to make transaction, so clients doesn’t need to have SiNar’s Account to make transaction.

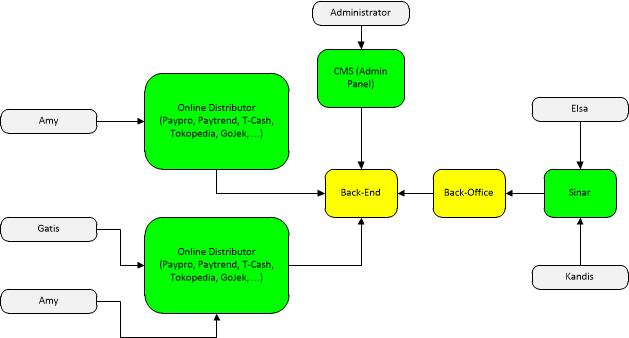
### Existing Process Diagram



### Existing Process Description

As of now, if clients want to make transaction with Bahana TCW’s product, then the only way is to register account in SiNar web and then clients can make transaction.

### Future Process Diagram



### Future Process Description

With Back-End API and Web Platform, clients can make transaction from Online Distributor that they familiar with. Bahana’s staff also can monitor investor data and transactions record from Web Platform. Tha main feature of Back-End API and Web Platform will be:

* Has feature to Register Investor from Online Distributor to Back-Office
* Has feature to Update Data Investor from Online Distributor to Back-Office
* Has feature to handle Subsciption Transaction where Clients can buy Bahana’s product from Online Distributor
* Has feature to handle Redemption Transaction where Clients can sell their products from Online Distributor
* Has feature to handle Switching Transaction where Clients can change product that they have to another product
* Has feature to reconcile Investor Unit Holding where Back-Office compare Investor portfolio (Products and unit count they have) in Back-Office with Investor portfolio in Back-End
* Has feature to Revise Transaction where Back-Office can edit transaction data in Back-End
* Has feature to Void Transaction where Back-Office can delete transaction in Back-End
* Has feature to provide company and products information from Back-Office to Online Distributor
* Has feature to periodically send email (Birthday Greeting, Reminder of KTP Expiry) and data summary (Product Summary, Transaction Summary and Transaction Log Report)
* Has feature to generate Reports in Back-End

## Scope of Work

### Scope of Work

Scope of Work of this project is defined as follows:

* Back-End
  + - Registration
      * Receive Registration data from Online Distributor
      * More than one accounts for one investor
      * Calculate Risk Profile based on Questionnaire
      * Send email registration notification to Marketing and Complience
      * Registration approval by Marketing and Compliance
      * Send email registration approval status notification to Investor
      * Provide Registration data for Back-Office
      * Receive Response from Back-Office and provide Response for Online Distributor
    - Update Data
      * Receive Updated data from Online Distributor
      * Send email update data notification to Marketing and Complience
      * Updated data approval by Marketing and Compliance
      * Send email update data approval status notification to Investor
      * Provide Updated data for Back-Office
      * Receive Response from Back-Office and provide Response for Online Distributor
    - Subscription
      * Receive Subscription data from Online Distributor
      * Provide Subscription data for Back-Office
      * Receive Subscription update data from Back-Office
      * Provide Subscription update data for Online Distributor
    - Redemption
      * Receive Redemption data from Online Distributor
      * Provide Redemption data for Back-Office
      * Receive Redemption update data from Back-Office
      * Provide Redemption update data for Online Distributor
    - Switching
      * Receive Switching data from Online Distributor
      * Provide Switching data for Back-Office
      * Receive Switching update data from Back-Office
      * Provide Switching update data for Online Distributor
    - Unit Reconciliation
      * Receive Investor Portfolio from Back-Office
      * Log the difference of unit value
    - Revise Transaction
    - Void Transaction
    - Master data
      * Receive master data from Back-Office
      * Provide master data for Online Distributor
* Web Platform
  + Admin User
    - * Create User
      * User Creation approval
      * View User
      * Change password sent by email
      * Login by email
      * Logout
  + Manage Master data
    - * Occupation
      * Product
      * Risk Category
      * Questionnaire
      * Promo
      * Company Profile
  + Investor data approval
    - * View Registration data
      * Registration data approval
      * View Updated data
      * Updated data approval
  + Report request approval and generate report
    - * View Report
      * Create report request
        + Dashboard (All sub reports exported in Excel)

1st Report (Tabular Dashboard)

2nd Report (Graphic AUM Growth and Tabular Distributor Statictic)

3rd Report (Tabular Latest Approval Request)

4th Report (Tabular Latest Transaction)

* + - * + Distributor (All sub reports exported in Excel)

AUM Growth YTD per Distributor (Graphic)

AUM Growth YTD Summary (Graphic)

Distributor Transaction Statistic per Distributor (Tabular)

Distributor Transaction Statistic Summary (Tabular)

* + - * + Investor (All sub reports exported in Excel)

Investor Demography By Age Range (Graphic and Tabular)

Investor Demography By Gender (Graphic and Tabular)

Investor Demography By Occupation (Graphic and Tabular)

Investor Demography By Earning Range (Graphic and Tabular)

Avarage Transaction per Month of All Investors (Graphic Transaction Value, Graphic Transaction Count and Tabular)

Avarage AUM per Investor (Graphic and Tabular)

Total Investor by Avarage AUM Range (Graphic and Tabular)

Investor Statistic (Tabular)

* + - * + Transaction

Transaction Statistic per Distributor (Tabular) in Excel

Transaction Statistic Summary (Tabular) in Excel

Top 10 Investor based on Subscription Transaction (Tabular) in Excel

Top 10 Investor based on AUM (Tabular) in Excel

Unit Holding Reconciliation in PDF

Revise Transaction in PDF

Void Transaction in PDF

Trx Summary in PDF

* + - * + OTP Statisctic (Tabular) in Excel
        + Product Statictic (Tabular) in Excel
        + Promo (All sub reports exported in Excel)

List of Promos (Tabular)

Total Subscription Growth during Promotion Date Range (Graphic)

PromoCode usage per day (Graphic)

Promo usage by Age Range (Graphic and Tabular)

Promo usage by Gender (Graphic and Tabular)

Promo usage by Occupation (Graphic and Tabular)

Promo usage by Earning Range (Graphic and Tabular)

* + - * Generate Report
      * View task request report
      * Report request approval
      * Cronjob to delete expired report

### Design Constraint

Design Constraint for Watervall is defined as follow :

* Back-End API will be built on Java
* Web Platform will be built on Angular2
* UI Language will use English
* Web Platform will be prioritized running well in one Web Browser that is Google Chrome
* Investor can only register once from one Online Distributor but can register from different Online Distributor
* Part of master data will not be provided with UI:
  + Educational
  + Religion
  + Income Level
  + Maritial Status
  + Risk Category
  + Investment Objective
  + Source Fund
  + Flow Type
  + SIDStatus
  + Investor Type
  + ID Type
  + Gender
  + Country
  + District
  + City
  + Statement Type
  + FATCA
  + BI Member
  + Currency
  + Transaction Type
  + Transfer Type
  + Transaction Status
  + Switching Fee Charge Fund

### Consideration for Performance

## Dependent/Related Projects

Table 1 Dependent/Related Projects

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Project Name** | **Dependency Description** | **Reference Documents Name** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Business Process Overview

## Business Requirement (BR)

Table 2 Requirement Summary

|  |  |  |
| --- | --- | --- |
| **BR ID** | **Description** | **Document Reff** |
| BR-1 | Has feature to handle investor registration from Online Distributors | Proposal\* |
| BR-2 | Has feature to handle update data investor from Online Distributors | Proposal\* |
| BR-3 | Has feature to handle Subsciption Transaction where Clients can buy Bahana’s product from Online Distributor | Proposal\* |
| BR-4 | Has feature to handle Redemption Transaction where Clients can sell their products from Online Distributor | Proposal\* |
| BR-5 | Has feature to handle Switching Transaction where Clients can change product that they have to another product | Proposal\* |
| BR-6 | Has feature to reconcile Investor Unit Holding where Back-Office compare Investor portfolio (Products and unit count they have) in Back-Office with Investor portfolio in Back-End | Proposal\* |
| BR-7 | Has feature to Revise Transaction where Back-Office can edit transaction data in Back-End | Proposal\* |
| BR-8 | Has feature to Void Transaction where Back-Office can delete transaction in Back-End | Proposal\* |
| BR-9 | Has feature to provide company and products information from Back-Office to Online Distributor | Proposal\* |
| BR-10 | Has feature to periodically send email (Birthday Greeting, Reminder of KTP Expiry) and data summary (Product Summary, Transaction Summary and Transaction Log Report) | Proposal\* |
| BR-11 | Web Platform to maintain master data, admin users, data approval, task notification and report generation | Proposal\* |

\*: TECHNICAL PROPOSAL – CODE.ID – BTIM – BACK-END APIs & WEB PLATFORM (File proposal)

## Functional Requirements (FR)

Table 3 Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **FR ID** | **BR Reff** | **Functional Requirement Description** | **Document Reff** |
| FR-1 | BR-1 | Back-End API shall be able to save multiple accounts for a single investor if registered from different Online Distributor | Proposal\* |
| FR-2 | BR-1 | Back-End API shall be able to create task and send email registration request notification | Proposal\* |
| FR-3 | BR-1 | Back-End API shall be able to update approval status of registration data and send email notification registration rejected | Proposal\* |
| FR-4 | BR-1 | Back-End API shall be able to provide approved registration data for Back-Office | Proposal\* |
| FR-5 | BR-1 | Back-End API shall be able to save SID (Single Investor Identification) and IFUA (Investor Fund Unit Account) number provided from Back-Office and send email notification registration approved | Proposal\* |
| FR-6 | BR-2 | Back-End API shall be able to save investor’s updated data without replacing the old data | Proposal\* |
| FR-7 | BR-2 | Back-End API shall be able to create task and send email update data request notification | Proposal\* |
| FR-8 | BR-2 | Back-End API shall be able to update approval status of investor’s update data and send email notification update data rejected | Proposal\* |
| FR-9 | BR-2 | Back-End API shall be able to provide approved investor’s updated data for Back-Office | Proposal\* |
| FR-10 | BR-2 | Back-End API shall be able to save IFUA number provided from Back-Office and send email notification update data approved | Proposal\* |
| FR-11 | BR-3 | Back-End API shall be able to save subscription data from Online Distributors | Proposal\* |
| FR-12 | BR-3 | Back-End API shall be able to provide subscription data for Back-Office | Proposal\* |
| FR-13 | BR-3 | Back-End API shall be able to save update unit allocation and NAV data for subscription transaction provided from Back-Office | Proposal\* |
| FR-14 | BR-3 | Back-End API shall be able to provide update unit allocation for subscription transaction for Online Distributors | Proposal\* |
| FR-15 | BR-3 | Back-End API shall be able to send email subscription status | Proposal\* |
| FR-16 | BR-4 | Back-End API shall be able to save redemption data from Online Distributors | Proposal\* |
| FR-17 | BR-4 | Back-End API shall be able to provide redemption data for Back-Office | Proposal\* |
| FR-18 | BR-4 | Back-End API shall be able to save update unit allocation and NAV data for redemption transaction provided from Back-Office | Proposal\* |
| FR-19 | BR-4 | Back-End API shall be able to provide update unit allocation for redemption transaction for Online Distributors | Proposal\* |
| FR-20 | BR-4 | Back-End API shall be able to send email redemption status | Proposal\* |
| FR-21 | BR-5 | Back-End API shall be able to save switching data from Online Distributors | Proposal\* |
| FR-22 | BR-5 | Back-End API shall be able to provide switching data for Back-Office | Proposal\* |
| FR-23 | BR-5 | Back-End API shall be able to save update unit allocation and NAV data for switching transaction provided from Back-Office | Proposal\* |
| FR-24 | BR-5 | Back-End API shall be able to provide update unit allocation for switching transaction for Online Distributors | Proposal\* |
| FR-25 | BR-5 | Back-End API shall be able to send email switching status | Proposal\* |
| FR-26 | BR-6 | Back-End API shall be able to calculate margin of difference between unit holding in Back-End and unit holding provided from Back-Office and log the margin | Proposal\* |
| FR-27 | BR-7 | Back-End API shall be able to revise transaction that provided by Back-Office | Proposal\* |
| FR-28 | BR-8 | Back-End API shall be able to void transaction provided from Back-Office | Proposal\* |
| FR-29 | BR-9 | Back-End API shall be able to receive master data from Back-Office | Proposal\* |
| FR-30 | BR-9 | Back-End API shall be able to provide master data for Online Distributors | Proposal\* |
| FR-31 | BR-10 | Back-End API shall be able to send email about data summary periodically | Proposal\* |
| FR-32 | BR-11 | Web Platform shall be able to maintain admin users | Proposal\* |
| FR-33 | BR-11 | Web Platform shall be able to maintain master data | Proposal\* |
| FR-34 | BR-11 | Web Platform shall be able to maintain investor data approval | Proposal\* |
| FR-35 | BR-11 | Web Platform shall be able to view transaction data | Proposal\* |
| FR-36 | BR-11 | Web Platform shall be able to maintain report request approval and generate report | Proposal\* |

\*: TECHNICAL PROPOSAL - CODE.ID - BTIM - BACK-END APIs & WEB PLATFORM (File proposal)

## Non Functional Requirement (NFR)

### Hardware Interfaces

Table 4 Hardware Interfaces

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NFR ID** | **BR Reff** | **Interface Name** | **Interface Description** | **Functionality Supported by this Interface** | **Document Reff** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

### Software Interfaces

Table 5 Software Interfaces

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NFR ID** | **BR Reff** | **Interface Name** | **Interface Description** | **Functionality Supported by this Interface** | **Document Reff** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

### System Interfaces

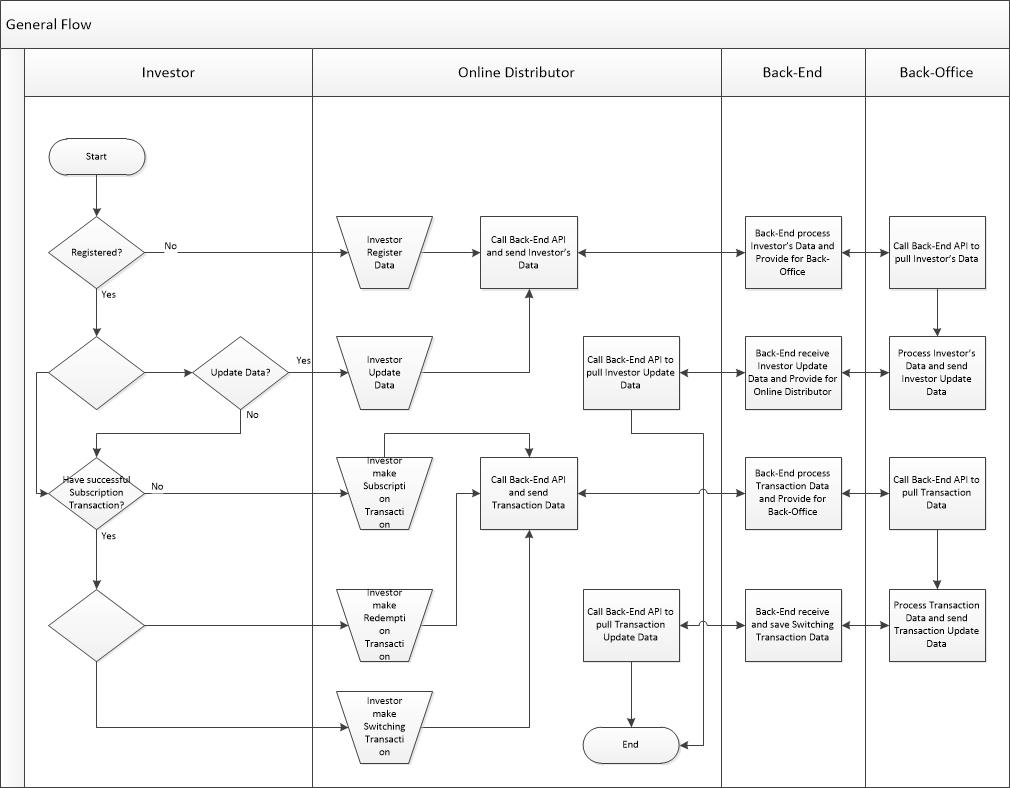
Table 6 System Interfaces

| **System Name** | **Protocol** | **Port(s) /**  **Type(s)** | **Direction**  **(Application**  **Perspective)** | **Details** |
| --- | --- | --- | --- | --- |
| Web Platform | HTTP | 80 |  | Web panel for admin to monitor transactions data, manage master data and manage report |
| Back-End API | HTTP | 1080 |  | API for Back-Office and Online Distributor to provide and retrieve data |

## Business Flow (BF)

### General Flow

Figure 1 General Flow



### Detail Flow

Figure 2 Business Flow 1 – Account/KYC Registration

| **BF ID** | BF-1 |
| --- | --- |
| **BR Reff** | BR-1 |
| **FR Reff** | FR-1, FR-2, FR-3, FR-4, FR-5 |
| **Flow Process** |  |

Figure 3 Business Flow 2 – Account/KYC Update

| **BF ID** | BF-2 |
| --- | --- |
| **BR Reff** | BR-2 |
| **FR Reff** | FR-6, FR-7, FR-8, FR-9, FR-10 |
| **Flow Process** |  |

Figure 4 Business Flow 3 – Subscription

| **BF ID** | BF-3 |
| --- | --- |
| **BR Reff** | BR-3 |
| **FR Reff** | FR-11, FR-12, FR-13, FR-14 |
| **Flow Process** |  |

Figure 5 Business Flow 4 - Redemption

| **BF ID** | BF-4 |
| --- | --- |
| **BR Reff** | BR-4 |
| **FR Reff** | FR-15, FR-16, FR-17, FR-18 |
| **Flow Process** |  |

Figure 6 Business Flow 5 – Switching

| **BF ID** | BF-5 |
| --- | --- |
| **BR Reff** | BR-5 |
| **FR Reff** | FR-19, FR-20, FR-21, FR-22 |
| **Flow Process** |  |

## Assumptions

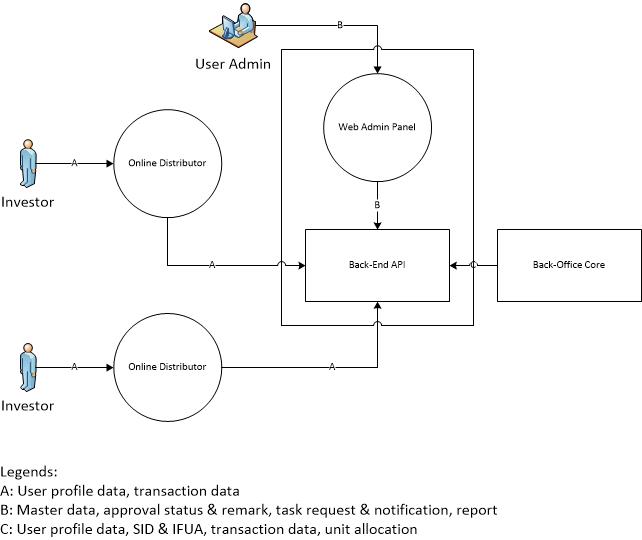
To make sure this application runs smoothly, we assume that this system meet the following requirements, such as:

* Internet connection is available
* Online Distributor must follow Back-End API data structure
* Online Distributor will provide UI Form for Investors activity
* Online Distributor will provide OTP Generation and Verification
* Online Distributor will handle Transaction Invoice
* Online Distributor will handle Payment process

# Systems Overview

## System Context

Figure 7 System Context



## Impacted Systems

Table 7 Impacted Systems

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **System Name** | **Impacted Area/Functional** | **Description** | **Document Reff** |
|  |  |  |  |  |

## Use Case Back-End

### Investor Registration

|  |  |
| --- | --- |
| **UC ID** | UC-1 |
| **Description** | Investor Registration from Online Distributor |
| **FR Reff** | FR-1 |
| **Actor** | Online Distributor |
| **Pre-Condition** | Registration data saved in Online Distributor |
| **Trigger** | Online Distributor call Back-End API to register investor |
| **Normal Scenarios** | * Back-End will validate the registration data, saved data and then return status as API response * If Questionnaire exist then Back-End calculate Risk Profile based on total score of Investor’s answer from questionnaire. * UC-3 will be triggered |
| **UI Validation** | - |
| **Business Rules** | * Fields First name, Place of birth, Date of birth, Distributor ID, Nationality, ID type, ID no, Gender, Email, Educational, Religion, Occupation, Income level, Investment objective, Source fund, Correspondence address, Correspondence city code, Bank code 1, Bank Country 1, Account currency 1, Account number 1, Account Name 1, KTP image, KTP selfie image and questionnaires – question & answer must not be empty * ID no must be unique * KTP address and KTP city code will be filled with value of Correspondence address and Correspondence city code if Nationality is “ID”. * Risk Profile define by comparing total score answer of Investor from Questionnaire with range score in Master Data Risk Category * Flow Type value will be set to 1 (First Input) if it’s the first time registration and will be set to 2 if it’s for data amendment * If SID Status value is empty by default will be set to 2 (N/A) means it is not processed yet * Investor Type value by default will be set to 1 (Individual Investor) * Statement Type value will be set to 2 (e-Statement) if it’s empty from Online Distributor * Investor can only register once per Online Distributor but can register in more than one Online Distributor. If Investor have already registered once (Online Distributor 1) and register again from different Online Distributor (Online Distributor 2) then Investor will have to choose either to keep existing data (Data registered from Online Distributor 1) or update with new data (Data registered from Online Distributor 2). |
| **Success Result** | Online Distributor will receive successful status as API response |
| **Minimal Result** | Online Distributor will receive failed status as API response with message “Invalid data provided” |
| **Extention** | Invalid data ([type]) provided |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | API used for this use case is:  “[POST]investors” with JSON body that contain:   * [M]DistributorID (String) * [M]FirstName (String) * [O]MiddleName (String) * [O]LastName (String) * [O]CountryOfBirth (String) * [M]PlaceOfBirth (String) * [M]DateOfBirth (long) * [O]SID (String) * [O]SIDStatus (int) * [M]Nationality (String) * [M]IDType (int) * [M]IDNo (long) * [O]IDExpDate (String, YYYYMMDD) * [M]Gender (String) * [O]NPWP (int) * [O]NPWPRegDate (String, YYYMMDD) * [O]MotherName (String) * [O]SpouseName (String) * [M]Email (String) * [O]HomePhone (String) * [M]MobilePhone (String) * [O]Facsimile (String) * [M]Educational (int) * [M]Religion (int) * [M]Occupation (int) * [O]CompanyPositionType (int) * [O]BadanHukumType (int) * [O]InstitutionType (int) * [M]IncomeLevel (int) * [M]MaritialStatus (int) * [M]InvestmentObjective (int) * [M]SourceFund (int) * [C]KTPAddress (String), this field mandatory if field Nationality is “ID” * [C]KTPCityCode (int), this field mandatory if field Nationality is “ID” * [O]KTPPostalCode (int) * [M]CorrespondenceAddress (String) * [M]CorrespondenceCityCode (int) * [O]CorrespondenceCityName (String) * [O]CorrespondencePostalCode (int) * [O]CountryOfCorrespondence (String) * [O]DomicileAddress (String) * [O]DomicileCityCode (int) * [O]DomicileCityName (String) * [O]DomicilePostalCode (int) * [O]CountryOfDomicile (String) * [O]StatementType (int) * [O]FATCA (int) * [O]TIN (String) * [O]TINCountry (String) * [M]REDMBankCode1 (String) * [M]REDMBankCountry1 (String) * [O]REDMBankBranch1 (String) * [M]REDMAccountCcy1 (String) * [M]REDMAccountNo1 (String) * [M]REDMAccountName1 (String) * [O]REDMBankCode2 (String) * [O]REDMBankCountry2 (String) * [O]REDMBankBranch2 (String) * [O]REDMAccountCcy2 (String) * [O]REDMAccountNo2 (String) * [O]REDMAccountName2 (String) * [O]REDMBankCode3 (String) * [O]REDMBankCountry3 (String) * [O]REDMBankBranch3 (String) * [O]REDMAccountCcy3 (String) * [O]REDMAccountNo3 (String) * [O]REDMAccountName3 (String) * [M]KTPImage (Image) * [M]KTPSelfieImage (Image) * [O]FamilyEmergencyPhone (String) * [O]FamilyEmergencyAddress (String) * [O]Questionnaires (int[])   M: Mandatory  C: Conditional  O: Optional  and the response will be JSON (meta) with:   * meta   + http\_code (int)   + message (String) |

### Pull Investor data from Dukcapil

|  |  |
| --- | --- |
| **UC ID** | UC-2 |
| **Description** | Pull Investor data from Dukcapil |
| **FR Reff** | FR-1 |
| **Actor** | Back-End |
| **Pre-Condition** | Investor data from Online Distributor already saved in Back-End |
| **Trigger** | Back-End call Dukcapil API |
| **Normal Scenarios** | Back-End call Dukcapil API then receive and save data from response body |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor data from Dukcapil will be saved in separate table from Investor data from Online Distributor |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | Dukcapil API fro this case is:  [POST]<http://172.16.160.128:8080/dukcapil/get_json/bahana_tcw/cal_nik>  with body contain:   * NIK (String) * user\_id (String) * password (String) * ip\_user (String)   and the response contain:   * NO\_KK (String) * NIK (String) * NAMA\_LGKP (String) * NAMA\_LGKP\_AYAH (String) * NAMA\_LGKP\_IBU (String) * TMPT\_LHR (String) * TGL\_LHR (String) * JENIS\_KLMIN (String) * STATUS\_KAWIN (String) * JENIS\_PEKERJAAN (String) * NO\_PROP (int) * PROP\_NAME (String) * NO\_KAB (int) * KAB\_NAME (String) * NO\_KEC (int) * KEC\_NAME (String) * NO\_KEL (int) * KEL\_NAME (String) * NO\_RT (int) * NO\_RW (int) |

### Send email registration notification to Marketing

|  |  |
| --- | --- |
| **UC ID** | UC-3 |
| **Description** | Send email registration notification to Marketing |
| **FR Reff** | FR-2 |
| **Actor** | Back-End |
| **Pre-Condition** | * UC-1 have been passed * Account created with status inactive |
| **Trigger** | UC-1 |
| **Normal Scenarios** | Back-End will send email notification to all Admin user with role “Marketing Staff” |
| **UI Validation** | - |
| **Business Rules** | Registration data must be valid |
| **Success Result** | All Admin user with role “Marketing Staff” will receive email notification for Registartion approval |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Send email registration notification to Compliance

|  |  |
| --- | --- |
| **UC ID** | UC-4 |
| **Description** | Send email registration notification to Compliance |
| **FR Reff** | FR-2 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-5 has been passed with status Approved |
| **Trigger** | UC-5 |
| **Normal Scenarios** | Back-End send email notification to all Admin user with role “Compliance Staff” after Registration data approved by Marketing Admin |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | All Admin user with role “Compliance Staff” will receive email notification for Registration approval |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Registration approval by Marketing

|  |  |
| --- | --- |
| **UC ID** | UC-5 |
| **Description** | Registration approval by Marketing |
| **FR Reff** | FR-3 |
| **Actor** | Admin Panel (User Marketing Admin) |
| **Pre-Condition** | * User Marketing Admin has logged in to Admin Panel * User Marketing Admin is on Registration approval page |
| **Trigger** | User Marketing Admin taps “Approve” button |
| **Normal Scenarios** | * Back-End receive Account ID and Approval Status from Admin Panel * Back-End update Approval Status by Marketing to Investor’s account * If data Approved, then UC-4 will be triggered * If data Rejected, then UC-7 will be triggered |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Approval status by Marketing for Registration data will be changed |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End API administrator |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | API used for this use case is :  “[PUT]investors” with body contain:   * AccountID (int) * ApprovalStatus (int)   and the response will be:   * meta   + http\_code (int)   + message (String) |

### Registration approval by Compliance

|  |  |
| --- | --- |
| **UC ID** | UC-6 |
| **Description** | Registration approval by Compliance |
| **FR Reff** | FR-3 |
| **Actor** | Admin Panel (User Compliance Admin) |
| **Pre-Condition** | * User Compliance Admin has logged in to Admin Panel * User Compliance Admin is on Registration approval page |
| **Trigger** | User Compliance Admin taps “Approve” button |
| **Normal Scenarios** | * Back-End API receive Account ID and Approval Status from Admin Panel * Back-End API update Approval Status by Compliance to Investor’s account * If data Rejected, then UC-7 will be triggered |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Approval status by Compliance for Registration data will be changed |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End API administrator |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | API used for this use case is :  “[PUT]investors” with body contain:   * AccountID (int) * ApprovalStatus (int)   and the response will be :   * meta   + http\_code (int)   + message (String) |

### Send email registration rejected to Investor

|  |  |
| --- | --- |
| **UC ID** | UC-7 |
| **Description** | Send email registration rejected to Investor |
| **FR Reff** | FR-4 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-5 or UC-6 passed with status Rejected |
| **Trigger** | UC-5 or UC-6 |
| **Normal Scenarios** | Back-End will send email notification Registration Rejected to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification Registration Rejected |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Provide approved registration data

|  |  |
| --- | --- |
| **UC ID** | UC-8 |
| **Description** | Provide approved registration data for Back-Office |
| **FR Reff** | FR-5 |
| **Actor** | Back-End |
| **Pre-Condition** | Back-Office connected with Back-End |
| **Trigger** | Back-Office call Back-End API to pull Registration data |
| **Normal Scenarios** | * Back-Office call Back-End API to pull Registration data * Back-End query Registration data with Approval status Approved by Compliance * Back-End return Registration data as API response |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Back-Office will receive list of Registration data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[GET]investors”  and the response will be JSON (meta & data) with:   * meta   + http\_code (int)   + message (String) * data   + string(JSON): list of Registration data with fields:     - [M]ClientID (String)     - [M]DistributorID (String)     - [M]FirstName (String)     - [O]MiddleName (String)     - [O]LastName (String)     - [O]CountryOfBirth (String)     - [M]PlaceOfBirth (String)     - [M]DateOfBirth (long)     - [M]FlowType (int)     - [O]SID (String)     - [M]SIDStatus (int)     - [M]InvestorType (int)     - [M]Nationality (String)     - [M]IDType (int)     - [M]IDNo (long)     - [O]IDExpDate (String, YYYYMMDD)     - [M]Gender (String)     - [O]NPWP (int)     - [O]NPWPRegDate (String, YYYMMDD)     - [O]MotherName (String)     - [O]SpouseName (String)     - [M]Email (String)     - [O]HomePhone (String)     - [M]MobilePhone (String)     - [O]Facsimile (String)     - [M]Educational (int)     - [M]Religion (int)     - [M]Occupation (int)     - [O]CompanyPositionType (int)     - [O]BadanHukumType (int)     - [O]InstitutionType (int)     - [M]IncomeLevel (int)     - [M]MaritialStatus (int)     - [O]RiskProfile (int)     - [M]InvestmentObjective (int)     - [M]SourceFund (int)     - [C]KTPAddress (String), this field mandatory if field Nationality is “ID”     - [C]KTPCityCode (int), this field mandatory if field Nationality is “ID”     - [O]KTPPostalCode (int)     - [M]CorrespondenceAddress (String)     - [M]CorrespondenceCityCode (int)     - [O]CorrespondenceCityName (String)     - [O]CorrespondencePostalCode (int)     - [O]CountryOfCorrespondence (String)     - [O]DomicileAddress (String)     - [O]DomicileCityCode (int)     - [O]DomicileCityName (String)     - [O]DomicilePostalCode (int)     - [O]CountryOfDomicile (String)     - [M]StatementType (int)     - [O]FATCA (int)     - [O]TIN (String)     - [O]TINCountry (String)     - [M]REDMBankCode1 (String)     - [M]REDMBankCountry1 (String)     - [O]REDMBankBranch1 (String)     - [M]REDMAccountCcy1 (String)     - [M]REDMAccountNo1 (String)     - [M]REDMAccountName1 (String)     - [O]REDMBankCode2 (String)     - [O]REDMBankCountry2 (String)     - [O]REDMBankBranch2 (String)     - [O]REDMAccountCcy2 (String)     - [O]REDMAccountNo2 (String)     - [O]REDMAccountName2 (String)     - [O]REDMBankCode3 (String)     - [O]REDMBankCountry3 (String)     - [O]REDMBankBranch3 (String)     - [O]REDMAccountCcy3 (String)     - [O]REDMAccountNo3 (String)     - [O]REDMAccountName3 (String)     - [M]KTPImage (Image)     - [M]KTPSelfieImage (Image)     - [O]FamilyEmergencyPhone (String)     - [O]FamilyEmergencyAddress (String)   M: Mandatory  C: Conditional  O: Optional |

### Save investor data from Back-Office

|  |  |
| --- | --- |
| **UC ID** | UC-9 |
| **Description** | Back-Office want to update Investor data in Back-End (either SID and IFUA number or other data) |
| **FR Reff** | FR-6 |
| **Actor** | Back-End |
| **Pre-Condition** | Back-Office connected with Back-End |
| **Trigger** | Back-Office call Back-End API to update Investor’s data |
| **Normal Scenarios** | * Back-Office call Back-End API to update Investor’s data * Back-End receive and update SID and IFUA to Investor’s Account * If data sent from Back-Office included SID, SIDStatus with value 1 (Completed), IFUA and FlowType with value 1 (first registration input) then Investor’s Account will be set to active and UC-10 will be triggered * If data sent from Back-Office included IFUA and FlowType with value 2 (amendment) then UC-19 will be triggered |
| **UI Validation** | - |
| **Business Rules** | * SID must be unique per Investor * IFUA must be unique per Investor and per Online Distributor |
| **Success Result** | Investor’s account on Back-End will be updated with SID and IFUA number |
| **Minimal Result** | - |
| **Extention** | * SID already exist * IFUA already exist on Online Distributor * Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | API used for this use case is :  “[PUT]investors” with body contain JSON list of field data   * [M]ClientID (String) * [O]SID (String) * [O]IFUA (String) * [O]FirstName (String) * [O]MiddleName (String) * [O]LastName (String) * [O]CountryOfBirth (String) * [O]PlaceOfBirth (String) * [O]DateOfBirth (long) * [O]SIDStatus (int) * [O]InvestorType (int) * [O]Nationality (String) * [O]IDExpDate (String, YYYYMMDD) * [O]Gender (String) * [O]NPWP (int) * [O]NPWPRegDate (String, YYYMMDD) * [O]MotherName (String) * [O]SpouseName (String) * [O]Email (String) * [O]HomePhone (String) * [O]MobilePhone (String) * [O]Facsimile (String) * [O]Educational (int) * [O]Religion (int) * [O]Occupation (int) * [O]CompanyPositionType (int) * [O]BadanHukumType (int) * [O]InstitutionType (int) * [O]IncomeLevel (int) * [O]MaritialStatus (int) * [O]RiskProfile (int) * [O]InvestmentObjective (int) * [O]SourceFund (int) * [O]KTPAddress (String) * [O]KTPCityCode (int) * [O]KTPPostalCode (int) * [O]CorrespondenceAddress (String) * [O]CorrespondenceCityCode (int) * [O]CorrespondenceCityName (String) * [O]CorrespondencePostalCode (int) * [O]CountryOfCorrespondence (String) * [O]DomicileAddress (String) * [O]DomicileCityCode (int) * [O]DomicileCityName (String) * [O]DomicilePostalCode (int) * [O]CountryOfDomicile (String) * [O]StatementType (int) * [O]FATCA (int) * [O]TIN (String) * [O]TINCountry (String) * [O]REDMBankCode1 (String) * [O]REDMBankCountry1 (String) * [O]REDMBankBranch1 (String) * [O]REDMAccountCcy1 (String) * [O]REDMAccountNo1 (String) * [O]REDMAccountName1 (String) * [O]REDMBankCode2 (String) * [O]REDMBankCountry2 (String) * [O]REDMBankBranch2 (String) * [O]REDMAccountCcy2 (String) * [O]REDMAccountNo2 (String) * [O]REDMAccountName2 (String) * [O]REDMBankCode3 (String) * [O]REDMBankCountry3 (String) * [O]REDMBankBranch3 (String) * [O]REDMAccountCcy3 (String) * [O]REDMAccountNo3 (String) * [O]REDMAccountName3 (String) * [O]FamilyEmergencyPhone (String) * [O]FamilyEmergencyAddress (String) * [O]Disclaimer (Boolean)   M: Mandatory  O: Optional  and the response will be JSON (meta) with:   * meta   + http\_code (int)   + message (String) |

### Send email registration approved to Investor

|  |  |
| --- | --- |
| **UC ID** | UC-10 |
| **Description** | Send email registration approved to Investor |
| **FR Reff** | FR-4 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-9 has been passed with SID, SIDStatus completed, IFUA and FlowType with value 1 (first registration input) saved in Back-End |
| **Trigger** | UC-9 |
| **Normal Scenarios** | Back-End will send email notification Registration Approved included Investor’s data to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification Registration Approved |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Provide Investor data for Online Distributor

|  |  |
| --- | --- |
| **UC ID** | UC-11 |
| **Description** | Provide Investor data for Online Distributor |
| **FR Reff** | FR-5 |
| **Actor** | Back-End |
| **Pre-Condition** | Online Distributor connected with Back-End |
| **Trigger** | Online Distributor call Back-End API to pull Registration update data |
| **Normal Scenarios** | * Online Distributor call Back-End API to pull Registration update data * Back-End return Registration update data as API response |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Online Distributor will receive list of Registration update data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[GET]investors/update-data”  and the response will be JSON (meta & data) with:   * meta   + http\_code (int)   + message (String) * data   + string(JSON): list of Registration data with fields:     - ClientID (String)     - ClientCode (String)     - FirstName (String)     - MiddleName (String)     - LastName (String)     - CountryOfBirth (String)     - PlaceOfBirth (String)     - DateOfBirth (long)     - FlowType (int)     - SID (String)     - SIDStatus (int)     - IFUA (String)     - InvestorType (int)     - Nationality (String)     - IDType (int)     - IDNo (String)     - IDExpDate (String, YYYYMMDD)     - Gender (String)     - NPWP (int)     - NPWPRegDate (String, YYYMMDD)     - MotherName (String)     - SpouseName (String)     - Email (String)     - HomePhone (String)     - MobilePhone (String)     - Facsimile (String)     - Educational (int)     - Religion (int)     - Occupation (int)     - CompanyPositionType (int)     - BadanHukumType (int)     - InstitutionType (int)     - IncomeLevel (int)     - MaritialStatus (int)     - RiskProfile (int)     - InvestmentObjective (int)     - SourceFund (int)     - KTPAddress (String)     - KTPCityCode (int)     - KTPPostalCode (int)     - CorrespondenceAddress (String)     - CorrespondenceCityCode (int)     - CorrespondenceCityName (String)     - CorrespondencePostalCode (int)     - CountryOfCorrespondence (String)     - DomicileAddress (String)     - DomicileCityCode (int)     - DomicileCityName (String)     - DomicilePostalCode (int)     - CountryOfDomicile (String)     - StatementType (int)     - FATCA (int)     - TIN (String)     - TINCountry (String)     - REDMBankCode1 (String)     - REDMBankCountry1 (String)     - REDMBankBranch1 (String)     - REDMAccountCcy1 (String)     - REDMAccountNo1 (String)     - REDMAccountName1 (String)     - REDMBankCode2 (String)     - REDMBankCountry2 (String)     - REDMBankBranch2 (String)     - REDMAccountCcy2 (String)     - REDMAccountNo2 (String)     - REDMAccountName2 (String)     - REDMBankCode3 (String)     - REDMBankCountry3 (String)     - REDMBankBranch3 (String)     - REDMAccountCcy3 (String)     - REDMAccountNo3 (String)     - REDMAccountName3 (String)     - FamilyEmergencyPhone (String)     - FamilyEmergencyAddress (String)     - Disclaimer (Boolean) |

### Save investor updated data from Online Distributor

|  |  |
| --- | --- |
| **UC ID** | UC-12 |
| **Description** | Use Case to save investor updated data |
| **FR Reff** | FR-7 |
| **Actor** | Online Distributor |
| **Pre-Condition** | Updated data saved in Online Distributor |
| **Trigger** | Online Distributor call Back-End API to update data investor |
| **Normal Scenarios** | * Back-End receive updated data * Back-End validate updated data, saved data and then return status as API response * UC-13 will be triggered |
| **UI Validation** | - |
| **Business Rules** | * ID No must be unique * FlowType value will be set to 2 (amendment) for update data |
| **Success Result** | Online Distributor will receive successful status as API response |
| **Minimal Result** | Online Distributor will receive failed status as API response with message “Invalid data provided” |
| **Extention** | Invalid data ([type]) provided |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | API used for this use case is:  “[PUT]investors” with JSON body that contain:   * [M]IDNo (String) * [O]FirstName (String) * [O]MiddleName (String) * [O]LastName (String) * [O]CountryOfBirth (String) * [O]PlaceOfBirth (String) * [O]DateOfBirth (long) * [O]Nationality (String) * [O]IDExpDate (String, YYYYMMDD) * [O]Gender (String) * [O]NPWP (int) * [O]NPWPRegDate (String, YYYMMDD) * [O]MotherName (String) * [O]SpouseName (String) * [O]Email (String) * [O]HomePhone (String) * [O]MobilePhone (String) * [O]Facsimile (String) * [O]Educational (int) * [O]Religion (int) * [O]Occupation (int) * [O]CompanyPositionType (int) * [O]BadanHukumType (int) * [O]InstitutionType (int) * [O]IncomeLevel (int) * [O]MaritialStatus (int) * [O]InvestmentObjective (int) * [O]SourceFund (int) * [O]KTPAddress (String) * [O]KTPCityCode (int) * [O]KTPPostalCode (int) * [O]CorrespondenceAddress (String) * [O]CorrespondenceCityCode (int) * [O]CorrespondenceCityName (String) * [O]CorrespondencePostalCode (int) * [O]CountryOfCorrespondence (String) * [O]DomicileAddress (String) * [O]DomicileCityCode (int) * [O]DomicileCityName (String) * [O]DomicilePostalCode (int) * [O]CountryOfDomicile (String) * [O]StatementType (int) * [O]FATCA (int) * [O]TIN (String) * [O]TINCountry (String) * [O]REDMBankCode1 (String) * [O]REDMBankCountry1 (String) * [O]REDMBankBranch1 (String) * [O]REDMAccountCcy1 (String) * [O]REDMAccountNo1 (String) * [O]REDMAccountName1 (String) * [O]REDMBankCode2 (String) * [O]REDMBankCountry2 (String) * [O]REDMBankBranch2 (String) * [O]REDMAccountCcy2 (String) * [O]REDMAccountNo2 (String) * [O]REDMAccountName2 (String) * [O]REDMBankCode3 (String) * [O]REDMBankCountry3 (String) * [O]REDMBankBranch3 (String) * [O]REDMAccountCcy3 (String) * [O]REDMAccountNo3 (String) * [O]REDMAccountName3 (String) * [M]KTPSelfieImage (image) * [M]KTPImage (image) * [O]FamilyEmergencyPhone (String) * [O]FamilyEmergencyAddress (String)   M: Mandatory  O: Optional  and the response will be JSON (meta) with:   * meta   + http\_code (int)   + message (String) |

### Send email update data notification to Marketing

|  |  |
| --- | --- |
| **UC ID** | UC-13 |
| **Description** | Use Case to send email update data notification to Marketing |
| **FR Reff** | FR-8 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-12 has been passed |
| **Trigger** | UC-12 |
| **Normal Scenarios** | Back-End will send email notification to all Admin user with role “Marketing Staff” |
| **UI Validation** | - |
| **Business Rules** | Updated Data must be valid |
| **Success Result** | All Admin user with role “Marketing Staff” will receive email notification for Update Data approval |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Send email update data notification to Compliance

|  |  |
| --- | --- |
| **UC ID** | UC-14 |
| **Description** | Use Case to send email update data notification to Compliance |
| **FR Reff** | FR-8 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-15 has been passed with status Approved |
| **Trigger** | UC-15 |
| **Normal Scenarios** | Back-End send email notification to all Admin user with role “Compliance Staff” after request Update Data approved by Marketing Admin |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | All Admin user with role “Compliance Staff” will receive email notification for Update Data approval |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Updated data approval by Marketing

|  |  |
| --- | --- |
| **UC ID** | UC-15 |
| **Description** | Use Case for updated data approval by Marketing |
| **FR Reff** | FR-9 |
| **Actor** | Admin Panel (User Marketing Admin) |
| **Pre-Condition** | * User Marketing Admin has logged in to Admin Panel * User Marketing Admin is on Update Data approval page |
| **Trigger** | User Marketing Admin taps “Approve” button |
| **Normal Scenarios** | * Back-End receive Account ID and Approval Status from Admin Panel * Back-End update Approval Status by Marketing to Investor’s account * If data Approved, then UC-14 will be triggered * If data Rejected, then UC-17 will be triggered |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Approval status by Marketing for Update Data will be changed to approved or rejected |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | API used for this use case is :  “[PUT]investorsUpdate” with body contain:   * AccountID (int) * ApprovalStatus (int)   and the response will be :   * meta   + http\_code (int)   + message (String) |

### Updated data approval by Compliance

|  |  |
| --- | --- |
| **UC ID** | UC-16 |
| **Description** | Use Case for Updated Data approval by Compliance |
| **FR Reff** | FR-9 |
| **Actor** | Admin Panel (User Compliance Admin) |
| **Pre-Condition** | * User Compliance Admin has logged in to Admin Panel * User Compliance Admin is on Update Data approval page |
| **Trigger** | User Compliance Admin taps “Approve” button |
| **Normal Scenarios** | * Back-End receive Account ID and Approval Status from Admin Panel * Back-End update Approval Status by Compliance to Investor’s account * If data Rejected, then UC-17 will be triggered |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Approval status by Compliance for Update Data will be changed |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | API used for this use case is :  “[PUT]investorsUpdate” with body contain:   * AccountID (int) * ApprovalStatus (int)   and the response will be :   * meta   + http\_code (int)   + message (String) |

### Send email update data rejected to Investor

|  |  |
| --- | --- |
| **UC ID** | UC-17 |
| **Description** | Use Case to send email Update Data rejected to Investor |
| **FR Reff** | FR-10 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-15 or UC-16 passed with status Rejected |
| **Trigger** | UC-15 or UC-16 |
| **Normal Scenarios** | Back-End will send email notification Update Data Rejected to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification Update Data Rejected |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Provide approved updated data

|  |  |
| --- | --- |
| **UC ID** | UC-18 |
| **Description** | Use Case to provide approved Update Data for Back-Office |
| **FR Reff** | FR-11 |
| **Actor** | Back-End |
| **Pre-Condition** | Back-Office connected with Back-End |
| **Trigger** | Back-Office call Back-End API to pull Updated Data |
| **Normal Scenarios** | * Back-Office call Back-End API to pull Updated Data * Back-End query Updated Data with Approval status Approved by Compliance * Back-End return Updated Data as API response |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Back-Office will receive list of Updated Data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[GET]investorsUpdate”  and the response will be JSON (meta & data) with:   * meta   + http\_code (int)   + message (String) * data   + string(JSON): list of Updated Data with fields:     - [M]ClientID (String)     - [O]FirstName (String)     - [O]MiddleName (String)     - [O]LastName (String)     - [O]CountryOfBirth (String)     - [O]PlaceOfBirth (String)     - [O]DateOfBirth (long)     - [M]FlowType (int)     - [O]Nationality (String)     - [O]IDExpDate (String, YYYYMMDD)     - [O]Gender (String)     - [O]NPWP (int)     - [O]NPWPRegDate (String, YYYMMDD)     - [O]MotherName (String)     - [O]SpouseName (String)     - [O]Email (String)     - [O]HomePhone (String)     - [O]MobilePhone (String)     - [O]Facsimile (String)     - [O]Educational (int)     - [O]Religion (int)     - [O]Occupation (int)     - [O]CompanyPositionType (int)     - [O]BadanHukumType (int)     - [O]InstitutionType (int)     - [O]IncomeLevel (int)     - [O]MaritialStatus (int)     - [O]InvestmentObjective (int)     - [O]SourceFund (int)     - [O]KTPAddress (String)     - [O]KTPCityCode (int)     - [O]KTPPostalCode (int)     - [O]CorrespondenceAddress (String)     - [O]CorrespondenceCityCode (int)     - [O]CorrespondenceCityName (String)     - [O]CorrespondencePostalCode (int)     - [O]CountryOfCorrespondence (String)     - [O]DomicileAddress (String)     - [O]DomicileCityCode (int)     - [O]DomicileCityName (String)     - [O]DomicilePostalCode (int)     - [O]CountryOfDomicile (String)     - [O]StatementType (int)     - [O]FATCA (int)     - [O]TIN (String)     - [O]TINCountry (String)     - [O]REDMBankCode1 (String)     - [O]REDMBankCountry1 (String)     - [O]REDMBankBranch1 (String)     - [O]REDMAccountCcy1 (String)     - [O]REDMAccountNo1 (String)     - [O]REDMAccountName1 (String)     - [O]REDMBankCode2 (String)     - [O]REDMBankCountry2 (String)     - [O]REDMBankBranch2 (String)     - [O]REDMAccountCcy2 (String)     - [O]REDMAccountNo2 (String)     - [O]REDMAccountName2 (String)     - [O]REDMBankCode3 (String)     - [O]REDMBankCountry3 (String)     - [O]REDMBankBranch3 (String)     - [O]REDMAccountCcy3 (String)     - [O]REDMAccountNo3 (String)     - [O]REDMAccountName3 (String)     - [M]KTPSelfieImage (image)     - [M]KTPImage (image)     - [O]FamilyEmergencyPhone (String)     - [O]FamilyEmergencyAddress (String)   M: Mandatory  O: Optional |

### Send email update data approved to Investor

|  |  |
| --- | --- |
| **UC ID** | UC-19 |
| **Description** | Use Case to send email Update Data approved to Investor |
| **FR Reff** | FR-10 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-9 has been passed with IFUA and FlowType with value 2 (amendment) saved in Back-End |
| **Trigger** | UC-9 |
| **Normal Scenarios** | Back-End API will send email notification Update Data Approved included Investor current data to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification Update Data Approved |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Save subscription data

| **UC ID** | UC-20 |
| --- | --- |
| **Description** | Use Case to receive and save subscription data from Online Distributor |
| **FR Reff** | FR-13 |
| **Actor** | Online Distributor |
| **Pre-Condition** | * Online Distributor connected with Back-End API * Subscription Data saved in Online Distributor |
| **Trigger** | Online Distributor call Back-End API to send Subscription Data |
| **Normal Scenarios** | * Online Distributor call Back-End API to send Subscription Data * Back-End receive Subscription Data * Back-End validate and then save Subscription Data |
| **UI Validation** | - |
| **Business Rules** | * Fields Transaction Date, Transaction Type, IFUA, Fund Code, Amount Nominal, Transaction ID, Distributor ID and OTP Time must not be empty * Promo Code can be included * If Promo Code in not empty and valid (active), Back-End will generate another (2nd) Transaction Subscription record with Amount Nominal is Promo Amount of the Promo Code without changing the original (1st) Transaction Subscription data. * SACode value will be set to “DX002” if it’s empty |
| **Success Result** | Subscription Data will be save in Back-End and Online Distributor will receive successful status &message as API response |
| **Minimal Result** | Online Distributor will receive failed status as API response with message “Invalid data provided” |
| **Extention** | Invalid data ([type]) provided |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is :  “[POST]transaction/subscription” with body contain JSON:   * [M]TransactionDate (String, YYYYMMDD) * [M]TransactionType (int) * [O]SACode (String) * [M]IFUA (String) * [M]FundCode (int) * [M]AmountNominal (long) * [O]FeeNominal (long) * [O]FeeUnit (long) * [O]FeePercentage (long) * [O]SAReferenceNo (String) * [M]TrxID (String) * [M]DistributorID (int) * [O]PromoCode (String) * [M]OTPTime (String)   M: Mandatory  O: Optional  and the response will be JSON (meta) with:   * meta   + http\_code (int)   + message (String) |

### Provide subscription data

|  |  |
| --- | --- |
| **UC ID** | UC-21 |
| **Description** | Use Case to provide Subscription Data for Back-Office |
| **FR Reff** | FR-14 |
| **Actor** | Back-End |
| **Pre-Condition** | Back-Office connected with Back-End API |
| **Trigger** | Back-Office call Back-End API to pull Subscription Data |
| **Normal Scenarios** | * Back-Office call Back-End API to pull Subscription Data * Back-End query Subscription Data and returned as API response |
| **UI Validation** | - |
| **Business Rules** | Subscription Data must be valid |
| **Success Result** | Back-Office will receive list of Subscription Data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[GET]transactions/subscription”  and the response will be JSON (meta & data) with:   * meta   + http\_code (int)   + message (String) * data   + string(JSON): list of Transaction Data with fields:     - [M]TransactionDate (String, YYYYMMDD)     - [M]TransactionType (int)     - [M]SACode (String)     - [M]IFUA (String)     - [M]FundCode (int)     - [M]AmountNominal (long)     - [C]AmountUnit (long), field value will be 0 if subscription     - [C]AmountAllUnit (String), field value will be empty String if subscription     - [O]FeeNominal (long)     - [O]FeeUnit (long)     - [O]FeePercentage (long)     - [C]REDMSequentialCode (int), field value will be 0     - [C]REDMBankBICCode (String), field value will be empty String     - [C]REDMBankBICode (String), field value will be emptry String if subscription     - [C]REDMAccountNo (String), field value will be empty String if subscription     - [C]PaymentDate (String), field value will be empty String if subscription     - [C]TransferType (int), field value will be 0 if subscription     - [O]SAReferenceNo (String)     - [M]TrxID (String)     - [M]DistributorID (int)     - [O]PromoCode (String)     - [M]OTPTime (String)   M: Mandatory  C: Conditional  O: Optional |

### Save subscription unit allocation and NAV data

| **UC ID** | UC-22 |
| --- | --- |
| **Description** | Use Case to save subscription unit allocation and NAV data |
| **FR Reff** | FR-15 |
| **Actor** | Back-End |
| **Pre-Condition** | * Back-Office connected with Back-End API * Transaction exist in both Back-End and Back-Office |
| **Trigger** | Back-Office call Back-End API to update Transaction Data |
| **Normal Scenarios** | * Back-Office call Back-End API to update Transaction Data * Back-End receive request data from Back-Office and update Transaction Data on Back-End * If Transaction Status “Success” then UC-25 will be triggered * If Transaction Status “Failed” then UC-26 will be triggered |
| **UI Validation** | - |
| **Business Rules** | Transaction with Promo Code will send the transacation data as it is without sum Promo Amount to the original Subscription Amount |
| **Success Result** | Transaction Data on Back-End will be updated with data from Back-Office |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[PUT]transactions/subscription” with body contain JSON:   * [M]TrxID (String) * [M]TrxStatus (int) * [M]AmountUnit (long) * [M]NAVDate (String, YYYYMMDD) * [M]NAVPrice (long)   M: Mandatory  and the response will be JSON (meta) with:   * meta   + http\_code (int)   + message (String) |

### Provide subscription data for Online Distributor

|  |  |
| --- | --- |
| **UC ID** | UC-23 |
| **Description** | Use Case to provide Subscription data for Online Distributor |
| **FR Reff** | FR-16 |
| **Actor** | Back-End |
| **Pre-Condition** | Online Distributor connected with Back-End API |
| **Trigger** | Online Distributor call Back-End API to pull Subscription Data |
| **Normal Scenarios** | * Online Distributor call Back-End API to pull Subscription Data * Back-End query Subscription Data and returned as API response |
| **UI Validation** | - |
| **Business Rules** | Subscription Data must be valid |
| **Success Result** | Online Distributor will receive list of Subscription Data with Unit Allocation and NAV Data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[GET]transactions/subscription”  and the response will be JSON (meta & data) with:   * meta   + http\_code (int)   + message (String) * data   + string(JSON): list of Subscription Transaction Update Data with fields:     - TransactionDate (String, YYYYMMDD)     - TransactionType (int)     - SACode (String)     - IFUA (String)     - FundCode (int)     - AmountNominal (long)     - AmountUnit (long)     - FeeNominal (long)     - FeeUnit (long)     - FeePercentage (long)     - SAReferenceNo (String)     - TrxID (String)     - DistributorID (int)     - PromoCode (String)     - OTPTime (String)     - NAVDate (String , YYYYMMDD)     - NAVPrice (long)     - TrxStatus (int)   M: Mandatory |

### Send email subscription successful to Investor

|  |  |
| --- | --- |
| **UC ID** | UC-24 |
| **Description** | Use Case to send email subscription successful to Investor |
| **FR Reff** | FR-17 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-22 has been passed with Transaction Status “Success” |
| **Trigger** | UC-22 |
| **Normal Scenarios** | Back-End API will send email notification Subscription Success to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification Subscription Success |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** |  |

### Send email subscription rejected to Investor and Marketing

|  |  |
| --- | --- |
| **UC ID** | UC-25 |
| **Description** | Use Case to Send email subscription rejected to Investor and Marketing |
| **FR Reff** | FR-17 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-22 has been passed with Transaction Status “Failed” |
| **Trigger** | UC-22 |
| **Normal Scenarios** | Back-End API will send email notification Subscription Failed to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification Subscription Failed |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Save redemption data

| **UC ID** | UC-26 |
| --- | --- |
| **Description** | Use Case to receive and save redemption data from Online Distributor |
| **FR Reff** | FR-18 |
| **Actor** | Online Distributor |
| **Pre-Condition** | * Online Distributor connected with Back-End API * Redemption Data saved in Online Distributor |
| **Trigger** | Online Distributor call Back-End API to send Redemption Data |
| **Normal Scenarios** | * Online Distributor call Back-End API to send Redemption Data * Back-End receive Redemption Data * Back-End validate and then save Redemption Data |
| **UI Validation** | - |
| **Business Rules** | * Fields Transaction Date, Transaction Type, IFUA, Fund Code, Amount Nominal or Amount Unit, Bank BI Code, Bank Account No, Transaction ID, Distributor ID and OTP Time must not be empty * SACode value will be set to “DX002” if it’s empty * TransferType value will be set to 1 (SKNBI) if it’s empty |
| **Success Result** | Redemption Data will be save in Back-End and Online Distributor will receive successful status &message as API response |
| **Minimal Result** | Online Distributor will receive failed status as API response with message “Invalid data provided” |
| **Extention** | Invalid data ([type]) provided |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is :  “[POST]transaction/redemption” with body contain JSON:   * [M]TransactionDate (String, YYYYMMDD) * [M]TransactionType (int) * [O]SACode (String) * [M]IFUA (String) * [M]FundCode (int) * [C]AmountNominal (long), this field mandatory if field AmountUnit and AmountAllUnit is empty * [C]AmountUnit (long), this field mandatory if field AmountNominal is empty * [O]AmountAllUnit (char) * [O]FeeNominal (long) * [O]FeeUnit (long) * [O]FeePercentage (long) * [M]REDMBankBICode (String) * [M]REDMBankAccountNo (String) * [O]PaymentDate (String, YYYYMMDD) * [O]TransferType (int) * [O]SAReferenceNo (String) * [M]TrxID (String) * [M]DistributorID (int) * [M]OTPTime (String)   M: Mandatory  O: Optional  C: Conditional  and the response will be JSON (meta) with:   * meta   + http\_code (int)   + message (String) |

### Provide redemption data

|  |  |
| --- | --- |
| **UC ID** | UC-27 |
| **Description** | Use Case to provide Redemption Data for Back-Office |
| **FR Reff** | FR-19 |
| **Actor** | Back-End |
| **Pre-Condition** | Back-Office connected with Back-End API |
| **Trigger** | Back-Office call Back-End API to pull Redemption Data |
| **Normal Scenarios** | * Back-Office call Back-End API to pull Redemption Data * Back-End query Redemption Data and returned as API response |
| **UI Validation** | - |
| **Business Rules** | Redemption Data must be valid |
| **Success Result** | Back-Office will receive list of Redemption Data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[GET]transactions/redemption”  and the response will be JSON (meta & data) with:   * meta   + http\_code (int)   + message (String) * data   + string(JSON): list of Transaction Data with fields:     - [M]TransactionDate (String, YYYYMMDD)     - [M]TransactionType (int)     - [M]SACode (String)     - [M]IFUA (String)     - [M]FundCode (int)     - [C]AmountNominal (long), this field mandatory if field AmountUnit and AmountAllUnit is empty     - [C]AmountUnit (long), this field mandatory if field AmountNominal is empty     - [O]AmountAllUnit (String)     - [O]FeeNominal (long)     - [O]FeeUnit (long)     - [O]FeePercentage (long)     - [C]REDMSequentialCode (int), field value will be 0     - [C]REDMBankBICCode (String), field value will be empty String     - [M]REDMBankBICode (String)     - [M]REDMAccountNo (String)     - [O]PaymentDate (String, YYYYMMDD)     - [O]TransferType (int)     - [O]SAReferenceNo (String)     - [M]TrxID (String)     - [M]DistributorID (int)     - [M]OTPTime (String)   M: Mandatory  O: Optional  C: Conditional |

### Save redemption unit allocation and NAV data

| **UC ID** | UC-28 |
| --- | --- |
| **Description** | Use Case to save redemption unit and NAV data |
| **FR Reff** | FR-20 |
| **Actor** | Back-End |
| **Pre-Condition** | * Back-Office connected with Back-End API * Transaction exist in both Back-End and Back-Office |
| **Trigger** | Back-Office call Back-End API to update Transaction Data |
| **Normal Scenarios** | * Back-Office call Back-End API to update Transaction Data * Back-End receive request data from Back-Office and update Transaction Data on Back-End * If Transaction Status “Success” then UC-30 will be triggered * If Transaction Status “Failed” then UC-31 will be triggered |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Transaction Data on Back-End will be updated with data from Back-Office |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[PUT]transactions/redemption” with body contain JSON:   * [M]TrxID (String) * [M]TrxStatus (int) * [C]AmountNominal (long), This field Mandatory if field AmountUnit or AmountAllUnit in Redemption Transaction is not empty, Reff: UC-26 * [C]AmountUnit (long), This field Mandatory if field AmountNominal in Redemption Transaction is not empty, Reff: UC-26 * [M]NAVDate (String, YYYYMMDD) * [M]NAVPrice (long)   M: Mandatory  C: Conditional  and the response will be JSON (meta) with:   * meta   + http\_code (int)   + message (String) |

### Provide redemption data for Online Distributor

|  |  |
| --- | --- |
| **UC ID** | UC-29 |
| **Description** | Use Case to provide Redemption data for Online Distributor |
| **FR Reff** | FR-21 |
| **Actor** | Back-End |
| **Pre-Condition** | Online Distributor connected with Back-End API |
| **Trigger** | Online Distributor call Back-End API to pull Redemption Data |
| **Normal Scenarios** | * Online Distributor call Back-End API to pull Redemption Data * Back-End query Redemption Data and returned as API response |
| **UI Validation** | - |
| **Business Rules** | Redemption Data must be valid |
| **Success Result** | Online Distributor will receive list of Redemption Data with Unit and NAV Data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[GET]transactions/redemption”  and the response will be JSON (meta & data) with:   * meta   + http\_code (int)   + message (String) * data   + string(JSON): list of Redemption Transaction Update Data with fields:     - TransactionDate (String, YYYYMMDD)     - TransactionType (int)     - SACode (String)     - IFUA (String)     - FundCode (String)     - AmountNominal (long)     - AmountUnit (long)     - AmountAllUnit (char)     - FeeNominal (long)     - FeeUnit (long)     - FeePercentage (long)     - REDMSequentialCode (int)     - REDMBankBICCode (String)     - REDMBankBICode (String)     - REDMAccountNo (String)     - PaymentDate (String, YYYYMMDD)     - TransferType (int)     - SAReferenceNo (String)     - TrxID (String)     - DistributorID (int)     - PromoCode (String)     - OTPTime (String)     - NAVDate (String , YYYYMMDD)     - NAVPrice (long)     - TrxStatus (int) |

### Send email redemption successful to Investor

|  |  |
| --- | --- |
| **UC ID** | UC-30 |
| **Description** | Use Case to send email redemption successful to Investor |
| **FR Reff** | FR-22 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-28 has been passed with Transaction Status “Success” |
| **Trigger** | UC-28 |
| **Normal Scenarios** | Back-End API will send email notification Redemption Success to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification RedemptionSuccess |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | - |

### Send email redemption rejected to Investor and Marketing

|  |  |
| --- | --- |
| **UC ID** | UC-31 |
| **Description** | Use Case to Send email redemption rejected to Investor and Marketing |
| **FR Reff** | FR-22 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-28 has been passed with Transaction Status “Failed” |
| **Trigger** | UC-28 |
| **Normal Scenarios** | Back-End API will send email notification Redemption Failed to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification Redemption Failed |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Save switching data

| **UC ID** | UC-32 |
| --- | --- |
| **Description** | Use Case to receive and save switching data from Online Distributor |
| **FR Reff** | FR-23 |
| **Actor** | Online Distributor |
| **Pre-Condition** | * Online Distributor connected with Back-End API * Switching Data saved in Online Distributor |
| **Trigger** | Online Distributor call Back-End API to send Switching Data |
| **Normal Scenarios** | * Online Distributor call Back-End API to send Switching Data * Back-End receive Switching Data * Back-End validate and then save Switching Data |
| **UI Validation** | - |
| **Business Rules** | * Fields Transaction Date, Transaction Type, IFUA, Switch out Fund Code, Switch out Amount Nominal or Switch out Amount Unit, Switch in Fund Code, Transfer Type, Transaction ID, Distributor ID and OTP Time must not be empty * SACode value will be set to “DX002” if it’s empty * SwitchingOutFeeChargeFund value will be set to 1 (Out) if it’s empty, means that switch out fee will be charge to fund that switched out |
| **Success Result** | Switching Data will be save in Back-End and Online Distributor will receive successful status &message as API response |
| **Minimal Result** | Online Distributor will receive failed status as API response with message “Invalid data provided” |
| **Extention** | Invalid data ([type]) provided |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is :  “[POST]transactions/switching” with body contain JSON:   * [M]TransactionDate (String, YYYYMMDD) * [M]TransactionType (int) * [O]SACode (String) * [M]IFUA (String) * [M]SwitchOutFundCode (String) * [C]SwitchOutAmountNominal (long), this field mandatory if field SwitchOutAmountUnit and SwitchOutAmountAllUnit is empty * [C]SwitchOutAmountUnit (long), this field mandatory if field SwitchOutAmountNominal is empty * [O]SwitchOutAllUnits (char) * [O]SwitchingOutChargeFund (int) * [O]FeeNominal (long) * [O]FeeUnit (long) * [O]FeePercentage (long) * [M]SwitchInFundCode (String) * [O]PaymentDate (String, YYYYMMDD) * [M]TransferType (int) * [O]SAReferenceNo (String) * [M]TrxID (String) * [M]DistributorID (int) * [M]OTPTime (long)   M: Mandatory  O: Optional  C: Conditional  and the response will be JSON (meta) with:   * meta   + http\_code (int)   + message (String) |

### Provide switching data

|  |  |
| --- | --- |
| **UC ID** | UC-33 |
| **Description** | Use Case to provide Switching Data for Back-Office |
| **FR Reff** | FR-24 |
| **Actor** | Back-End |
| **Pre-Condition** | Back-Office connected with Back-End API |
| **Trigger** | Back-Office call Back-End API to pull Switching Data |
| **Normal Scenarios** | * Back-Office call Back-End API to pull Switching Data * Back-End query Switching Data and returned as API response |
| **UI Validation** | - |
| **Business Rules** | Switching Data must be valid |
| **Success Result** | Back-Office will receive list of Switching Data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[GET]transactions/switching”  and the response will be JSON (meta & data) with:   * meta   + http\_code (int)   + message (String) * data   + string(JSON): list of Redemption Transaction Data with fields:     - [M]TransactionDate (String, YYYYMMDD)     - [M]TransactionType (int)     - [M]SACode (String)     - [M]IFUA (String)     - [M]SwitchOutFundCode (String)     - [C]SwitchOutAmountNominal (long), this field mandatory if field SwitchOutAmountUnit and SwitchOutAmountAllUnit is empty     - [C]SwitchOutAmountUnit (long), this field mandatory if field SwitchOutAmountNominal is empty     - [O]SwitchOutAllUnit (char)     - [O]SwitchingOutChargeFund (int)     - [O]FeeNominal (long)     - [O]FeeUnit (long)     - [O]FeePercentage (long)     - [M]SwitchInFundCode (String)     - [O]PaymentDate (String, YYYYMMDD)     - [M]TransferType (int)     - [O]SAReferenceNo (String)     - [M]TrxID (String)     - [M]DistributorID (int)     - [M]OTPTime (long)   M: Mandatory  O: Optional  C: Conditional |

### Save switching unit allocation and NAV data

| **UC ID** | UC-34 |
| --- | --- |
| **Description** | Use Case to save switching unit and NAV data |
| **FR Reff** | FR-25 |
| **Actor** | Back-End |
| **Pre-Condition** | * Back-Office connected with Back-End API * Transaction exist in both Back-End and Back-Office |
| **Trigger** | Back-Office call Back-End API to update Transaction Data |
| **Normal Scenarios** | * Back-Office call Back-End API to update Transaction Data * Back-End receive request data from Back-Office and update Transaction Data on Back-End * If Transaction Status “Success” then UC-36 will be triggered * If Transaction Status “Failed” then UC-37 will be triggered |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Transaction Data on Back-End will be updated with data from Back-Office |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[PUT]transactions/switching” with body contain JSON:   * [M[TrxID (String) * [M]TrxStatus (int) * [C]SwitchOutAmountNominal (long), this field mandatory if field SwitchOutAmountUnit or SwitchOutAmountAllUnit is not empty, Reff: UC-32 * [C]SwitchOutAmountUnit (long), this field mandatory if field SwitchOutAmountNominal is not empty, Reff: UC-32 * [M]SwitchInAmountUnit (long) * [M]SwitchOutNAVDate (String, YYYYMMDD) * [M]SwitchOutNAVPrice (long) * [M]SwitchInNAVDate (String, YYYYMMDD) * [M]SwitchInNAVPrice (long)   M: Mandatory  C: Conditional  and the response will be JSON (meta) with:   * meta   + http\_code (int)   + message (String) |

### Provide switching data for Online Distributor

|  |  |
| --- | --- |
| **UC ID** | UC-35 |
| **Description** | Use Case to provide Switching data for Online Distributor |
| **FR Reff** | FR-26 |
| **Actor** | Back-End |
| **Pre-Condition** | Online Distributor connected with Back-End API |
| **Trigger** | Online Distributor call Back-End API to pull Switching Data |
| **Normal Scenarios** | * Online Distributor call Back-End API to pull Switching Data * Back-End query Switching Data and returned as API response |
| **UI Validation** | - |
| **Business Rules** | Switching Data must be valid |
| **Success Result** | Online Distributor will receive list of Switching Data with Unit and NAV Data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is:  “[GET]transactions/switching”  and the response will be JSON (meta & data) with:   * meta   + http\_code (int)   + message (String) * data   + string(JSON): list of Switching Transaction Data with fields:     - TransactionDate (String, YYYYMMDD)     - TransactionType (int)     - SACode (String)     - IFUA (String)     - SwitchOutFundCode (String)     - SwitchOutAmountNominal (long)     - SwitchOutAmountUnit (long)     - SwitchOutAmountAllUnit (char)     - SwitchingFeeChargeFund (int)     - FeeNominal (long)     - FeeUnit (long)     - FeePercentage (long)     - SwitchInFundCode (String)     - SwitchInAmountUnit (long)     - PaymentDate (String, YYYYMMDD)     - TransferType (int)     - SAReferenceNo (String)     - TrxID (String)     - DistributorID (int)     - SwitchOutNAVDate (String , YYYYMMDD)     - SwitchOutNAVPrice (long)     - SwitchInNAVDate (String , YYYYMMDD)     - SwitchInNAVPrice (long)     - OTPTime (String)     - TrxStatus (int) |

### Send email switching successful to Investor

|  |  |
| --- | --- |
| **UC ID** | UC-36 |
| **Description** | Use Case to send email switching successful to Investor |
| **FR Reff** | FR-27 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-34 has been passed with Transaction Status “Success” |
| **Trigger** | UC-34 |
| **Normal Scenarios** | Back-End API will send email notification Switching Success to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification Switching Success |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | - |

### Send email switching rejected to Investor and Marketing

|  |  |
| --- | --- |
| **UC ID** | UC-37 |
| **Description** | Use Case to Send email switching rejected to Investor and Marketing |
| **FR Reff** | FR-27 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-34 has been passed with Transaction Status “Failed” |
| **Trigger** | UC-34 |
| **Normal Scenarios** | Back-End API will send email notification Switching Failed to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor receive email notification Switching Failed |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | - |

### Reconcile unit holding

|  |  |
| --- | --- |
| **UC ID** | UC-38 |
| **Description** | Use Case to reconcile Unit Holding |
| **FR Reff** | FR-28 |
| **Actor** | Back-End |
| **Pre-Condition** | * Back-Office connected with Back-End * Investor’s account exist in both Back-Office and Back-End |
| **Trigger** | Back-Office call Back-End API to do Unit Holding reconciliation |
| **Normal Scenarios** | * Back-Office call Back-End API to do Unit Holding reconciliation * Back-End compare Investor portfolio (unit) in Back-End with portfolion send from Back-Office * If there are margin of difference then UC-39 will be triggered |
| **UI Validation** | - |
| **Business Rules** | Fields IFUA, Date, Product ID and Unit Holding must not be empty |
| **Success Result** | Back-Office will receive successful status as API response |
| **Minimal Result** | Back-Office will receive failed status & message as API response with message “Invalid data provided” |
| **Extention** | Invalid data ([type]) provided |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is :  “[POST]UnitHolding” with body contain JSON:   * [M]IFUA (String) * [M]Date (String, YYYYMMDD) * [M]FundCode (String) * [M]Unit (long)   and the response will be:   * meta   + http\_code (int)   + message (String) |

### Log margin of difference

|  |  |
| --- | --- |
| **UC ID** | UC-39 |
| **Description** | Use Case to log margin of Unit Holding |
| **FR Reff** | FR-28 |
| **Actor** | Back-End |
| **Pre-Condition** | UC-38 passed with unit margin of difference >= 1 |
| **Trigger** | UC-38 |
| **Normal Scenarios** | Back-End will record Investor portfolio with margin of difference and save in log table |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Log record of margin of difference will be added in log table |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | Function used for this use case is :  “logUnitHolding(SID, Date, DetailUnitMargin)”:   * [M]IFUA (String) * [M]Date (String, YYYYMMDD) * [M]FundCode (String) * [M]UnitDifference (long)   M: Mandatory |

### Revise transaction

|  |  |
| --- | --- |
| **UC ID** | UC-40 |
| **Description** | Use Case to revise transaction |
| **FR Reff** | FR-29 |
| **Actor** | Back-End |
| **Pre-Condition** | * Back-Office connected with Back-End * Transaction exist in both Back-Office and Back-End |
| **Trigger** | Back-Office call Back-End API to revise transaction |
| **Normal Scenarios** | * Back-Office call Back-End API to revise transaction * Back-End receive data transaction to revise * Back-End update data transaction with data received |
| **UI Validation** | - |
| **Business Rules** | Fields Transaction ID, IFUA, Transaction Type must not be empty |
| **Success Result** | Transaction status in Back-End will be changed |
| **Minimal Result** | Back-Office will receive failed status & message as API response with message “Invalid data provided” |
| **Extention** | Invalid data ([type]) provided |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is :  “[POST]transaction/revise” with body contain JSON:   * [M]TrxID (String) * [M]IFUA (String) * [M]TransactionType (int) * [O]AmountNominal (long) * [O]AmountUnit (long) * [O]SwitchInAmountUnit (long) * [O]NAVDate (String, YYYYMMDD) * [O]NAVPrice (long) * [O]SwitchInNAVDate (String, YYYYMMDD) * [O]SwitchInNAVPrice (long) * [O]TrxStatus (int)   M: Mandatory  O: Optional  and the response will be:   * meta   + http\_code (int)   + message (String) |

### Void Transaction

|  |  |
| --- | --- |
| **UC ID** | UC-41 |
| **Description** | Use Case to void transaction |
| **FR Reff** | FR-30 |
| **Actor** | Back-End |
| **Pre-Condition** | * Back-Office connected with Back-End * Transaction exist in both Back-Office and Back-End |
| **Trigger** | Back-Office call Back-End API to void transaction |
| **Normal Scenarios** | * Back-Office call Back-End API to void transaction * Back-End receive data transaction to void * Back-End update transaction status |
| **UI Validation** | - |
| **Business Rules** | Fields SID, Date, Transaction ID must not be empty |
| **Success Result** | Transaction status in Back-End will be changed |
| **Minimal Result** | Back-Office will receive failed status & message as API response with message “Invalid data provided” |
| **Extention** | Invalid data ([type]) provided |
| **UI Image** | - |
| **Frequency of Usage** | Often |
| **Notes** | API used for this use case is :  “[POST]transaction/void” with body contain JSON:   * [M]IFUA (String) * [M]TrxID (String)   M: Mandatory  and the response will be:   * meta   + http\_code (int)   + message (String |

### Receive master data

|  |  |
| --- | --- |
| **UC ID** | UC-42 |
| **Description** | Use Case to received Master Data from Back-Office |
| **FR Reff** | FR-31 |
| **Actor** | Back-End |
| **Pre-Condition** | Back-Office connedted with Back-End |
| **Trigger** | Back-Office call Back-End API to send Master Data |
| **Normal Scenarios** | * Back-Office call Back-End API to send Master Data * Back-End API receive and save Master Data |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Master Data will be saved in Back-End |
| **Minimal Result** | Back-Office will receive failed status & message as API response with message “Invalid data provided” |
| **Extention** | Invalid data ([type]) provided |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | API used for this use case is :  “[POST]masterData” with body contain JSON list data of all master data with “Key -> Value” format where master data name as “Key” and list of record as “Value”.  List of Master Data:   * Occupation * Product * Risk Category * Questionnaire (Question) * Questionnaire (Answer) * Company Profile * Educational * Religion * Income Level * Maritial Status * Investment Objective * Source Fund * Flow Type * SIDStatus * Investor Type * ID Type * Gender * Country * District * City * Statement Type * FATCA * BI Member * Currency * Transaction Type * Transfer Type * Transaction Status * Switching Fee Charge Fund   and the response will be:   * meta   + http\_code (int)   + message (String |

### Provide master data

|  |  |
| --- | --- |
| **UC ID** | UC-43 |
| **Description** | Use Case to provide Master Data for Online Distributor |
| **FR Reff** | FR-32 |
| **Actor** | Back-End |
| **Pre-Condition** | Online Distributor connedted with Back-End |
| **Trigger** | Online Distributor call Back-End API to pull Master Data |
| **Normal Scenarios** | * Online Distributor call Back-End API to pull Master Data * Back-End API query Master Data and returned as API response |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Online Distributor will receive Master Data |
| **Minimal Result** | - |
| **Extention** | Service unavailable, please contact Back-End administrator |
| **UI Image** | - |
| **Frequency of Usage** | Rarely |
| **Notes** | API used for this use case is :  “[GET]masterData”  and the response will be JSON (meta & data):   * meta   + http\_code (int)   + message (String) * data (String JSON) contain list data of all master data with “Key -> Value” format where master data name as “Key” and list of record as “Value” :   + Occupation   + Product   + Risk Category   + Questionnaire (Question)   + Questionnaire (Answer)   + Company Profile   + Educational   + Religion   + Income Level   + Maritial Status   + Investment Objective   + Source Fund   + Flow Type   + SIDStatus   + Investor Type   + ID Type   + Gender   + Country   + District   + City   + Statement Type   + FATCA   + BI Member   + Currency   + Transaction Type   + Transfer Type   + Transaction Status   + Switching Fee Charge Fund |

### Cronjob to send Portfolio Summary to Investor

|  |  |
| --- | --- |
| **UC ID** | UC-44 |
| **Description** | Use Case for scheduled send portfolio summary to Investor |
| **FR Reff** | FR-33 |
| **Actor** | Back-End |
| **Pre-Condition** | Cronjob setting to send portfolio summary already set in Back-End |
| **Trigger** | Automatic |
| **Normal Scenarios** | * Cronjob get data portfolio summary and create in PDF format * Cronjob send email notification with attached file summary to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor will receive email notification with file summary |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Monthly |
| **Notes** | - |

### Cronjob to send Transaction Summary to Marketing

|  |  |
| --- | --- |
| **UC ID** | UC-45 |
| **Description** | Use Case for scheduled send Transaction Summary to Marketing |
| **FR Reff** | FR-33 |
| **Actor** | Back-End |
| **Pre-Condition** | Cronjob setting to send Transaction Summary already set in Back-End |
| **Trigger** | Automatic |
| **Normal Scenarios** | * Cronjob get data transaction summary that happen in the day and create in PDF format * Cronjob send email notification with attached file summary to Marketing |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Email with file summary attached will sent to all admin user with role “Marketing Staff” |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Daily |
| **Notes** | - |

### Cronjob to send email Birthday Greeting

|  |  |
| --- | --- |
| **UC ID** | UC-46 |
| **Description** | Use Case for scheduled send email Birthday Greeting to Investor |
| **FR Reff** | FR-33 |
| **Actor** | Back-End |
| **Pre-Condition** | Cronjob setting to send Birthday Greeting already set in Back-End |
| **Trigger** | Automatic |
| **Normal Scenarios** | * Cronjob will check Investor profile and list Investor(s) that have birthday in the day * Cronjob send email Birthday Greeting to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor will receive email Birthday Greeting |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Daily |
| **Notes** | - |

### Cronjob to send email Reminder of KTP Expiry

|  |  |
| --- | --- |
| **UC ID** | UC-47 |
| **Description** | Use Case for scheduled send email Reminder KTP Expiry to Investor |
| **FR Reff** | FR-33 |
| **Actor** | Back-End |
| **Pre-Condition** | Cronjob setting to send Reminder KTP Expiry already set in Back-End |
| **Trigger** | Automatic |
| **Normal Scenarios** | * Cronjob will check Investor profile and list Investor(s) that have KTP Expiry in the next 7 day * Cronjob send email Reminder KTP Expiry to Investor |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Investor will receive email Reminder of KTP Expiry |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Daily |
| **Notes** | - |

### Cronjob to send Unit Holding Reconciliation Log Report to Marketing and Operation

|  |  |
| --- | --- |
| **UC ID** | UC-48 |
| **Description** | Use Case for scheduled send Unit Holding Reconciliation Log Report to Marketing and Operation |
| **FR Reff** | FR-33 |
| **Actor** | Back-End |
| **Pre-Condition** | Cronjob setting to send Unit Holding Log Report already set in Back-End |
| **Trigger** | Automatic |
| **Normal Scenarios** | * Cronjob get Log Report of Unit Holding Reconciliation in the day and create in PDF format * Cronjob send email with attached file report to Marketing Admin and Operation Admin |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Email will be sent to all user with role “Marketing Staff” or “Operation Staff” |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Daily |
| **Notes** | - |

### Cronjob to send Revise Transaction Log Report to Investor, Marketing and Operation

|  |  |
| --- | --- |
| **UC ID** | UC-49 |
| **Description** | Use Case for scheduled send Revise Transaction Log Report to Investor, Marketing and Operation |
| **FR Reff** | FR-33 |
| **Actor** | Back-End |
| **Pre-Condition** | Cronjob setting to send Revise Transaction Log Report already set in Back-End |
| **Trigger** | Automatic |
| **Normal Scenarios** | * Cronjob get Log Report of Revise Transaction in the day and create in PDF format * Cronjob send email with attached file report to Investor Marketing Admin and Operation Admin |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Email will be sent to Investor and all admin user with role “Marketing Staff” or “Operation Staff” |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Daily |
| **Notes** | - |

### Cronjob to send Void Transaction Log Report to Investor, Marketing and Operation

|  |  |
| --- | --- |
| **UC ID** | UC-50 |
| **Description** | Use Case for scheduled send Void Transaction Log Report to Investor, Marketing and Operation |
| **FR Reff** | FR-33 |
| **Actor** | Back-End |
| **Pre-Condition** | Cronjob setting to send Void Transaction Log Report already set in Back-End |
| **Trigger** | Automatic |
| **Normal Scenarios** | * Cronjob get Log Report of Void Transaction in the day and create in PDF format * Cronjob send email with attached file report to Investor Marketing Admin and Operation Admin |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Email will be sent to Investor and all admin user with role “Marketing Staff” or “Operation Staff” |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Daily |
| **Notes** | - |

## Use Case User Action - CMS (Admin Panel)

### View Master Data

|  |  |
| --- | --- |
| **UC ID** | UC-51 |
| **Description** | Use Case to view master data |
| **FR Reff** | FR-32 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is already in master data menu :   * Occupation * Product * Risk Category * Questionnaire * Promo * Company Profile |
| **Trigger** | Administrator loads master data |
| **Normal Scenarios** | CMS shows master data page |
| **UI Validation** | - |
| **Business Rules** | * There are paging and filter features * By default, 10 items per page are shown * Options of items per page are : 10, 25, 50, 100 * Filter will be triggered if administrator select field and types in box of minimum 2 chars * Fields to be filtered vary on each master data, they are:   + Occupation: Name and Description   + Product: Product ID, Product Name, Product Type, Latest NAV, Unit, AUM and Annual Return Assumption   + Risk Category: No Filter   + Questionnaire: No Filter   + Promo: Promo Code, Promo Name, Promo Amount, Start Date and End Date   + Company Profile: No Filter |
| **Success Result** | CMS shows master data page |
| **Minimal Result** | No data shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Low |
| **Notes** | - |

### Create Master Data

|  |  |
| --- | --- |
| **UC ID** | UC-52 |
| **Description** | Use Case to create master data |
| **FR Reff** | FR-32 |
| **Actor** | Administrator |
| **Pre-Condition** | * Administrator is already in add page of the master data menu :   + Questionnaire (Question & Answer)   + Promo * Administrator entry mandatory fields:   + Question: Question sentence   + Answer: Answer sentence, Question ID, Score   + Promo: Promo Code, Promo Name, Promo Amount, Distributor ID, Min Subscription Amount, Product Type, Product Name, Start Date and End Date |
| **Trigger** | Admin click save button |
| **Normal Scenarios** | * Administrator fill mandatory fields * When save button clicked, Back-End will save new master data |
| **UI Validation** | * All fields must not be empty * Number-based field must be filled with numbers * Date-based field must be filled with date * Master data is unique for each ID |
| **Business Rules** | * All mandatory fields must not be empty * End Date must be greater than Start Date * For master data Answer, master data Question must be created first |
| **Success Result** | New master data is saved on Back-End |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | Invalid data provided |
| **UI Image** |  |
| **Frequency of Usage** | Low |
| **Notes** | - |

### Edit Master Data

|  |  |
| --- | --- |
| **UC ID** | UC-53 |
| **Description** | Use Case to edit master data |
| **FR Reff** | FR-32 |
| **Actor** | Administrator |
| **Pre-Condition** | * Administrator is already in edit page of master data menu :   + Product   + Questionaire (Question & Answer)   + Promo * Administrator edit some fields as needed:   + Product: Fund Factsheet (PDF) and Prospectus (PDF)   + Question: Question sentence   + Answer: Answer sentence, Question ID, Score   + Promo: Promo Code, Promo Name, Promo Amount, Distributor ID, Min Subscription Amount, Product Type, Product Name, Start Date and End Date |
| **Trigger** | Administrator click save button |
| **Normal Scenarios** | * Administrator edit master data fields * When save button clicked, Back-End will save updated master data |
| **UI Validation** | * All mandatory fields must not be empty * Number-based field must be filled with numbers * Date-based filled must be filled with date |
| **Business Rules** | Promo with “Active” status can not be edited |
| **Success Result** | Updated data is saved on server |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | Invalid data provided |
| **UI Image** |  |
| **Frequency of Usage** | Low |
| **Notes** | - |

### Delete Master Data

|  |  |
| --- | --- |
| **UC ID** | UC-54 |
| **Description** | Use Case to delete master data |
| **FR Reff** | FR-32 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is already in one of the master data menu :   * + Questionnaire (Question & Answer)   + Promo |
| **Trigger** | Admin click delete link |
| **Normal Scenarios** | * Show delete prompt * If the administrator click ok, data will be deleted * If the administrator click no, deletion process will be canceled |
| **UI Validation** | **-** |
| **Business Rules** | Promo that has already been used (referenced as a foreign key can not be deleted |
| **Success Result** | Data will be deleted |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | Invalid data provided |
| **UI Image** |  |
| **Frequency of Usage** | Low |
| **Notes** | - |

### View User

|  |  |
| --- | --- |
| **UC ID** | UC-55 |
| **Description** | Use Case to view all admin user |
| **FR Reff** | FR-33 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is already in users page |
| **Trigger** | Administrator load data users |
| **Normal Scenarios** | * Administrator click view link on a row in verify ID list view * Image of user’s uploaded image ID will be shown |
| **UI Validation** | - |
| **Business Rules** | * There is paging features * By default, 10 items per page are shown * Options of items per page are : 10, 25, 50, 100 |
| **Success Result** | Data list administratoe will be shown |
| **Minimal Result** | No data shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### Create user

|  |  |
| --- | --- |
| **UC ID** | UC-56 |
| **Description** | Use Case to create new administrator user |
| **FR Reff** | FR-33 |
| **Actor** | Administrator (Manager) |
| **Pre-Condition** | * Administrator is already in page add user * Administrator entry all manatory fields: Name, Email, Position, Role |
| **Trigger** | Administrator click save button |
| **Normal Scenarios** | * Administrator fill all mandatory fields * When save button clicked, new administrator data will be saved |
|  |  |
| **UI Validation** | All mandatory fields must not be empty |
| **Business Rules** | * Image is compressed to minimize its size * Compression is done by both front-end and server * Only PNG image is supported * There are paging and search features * By default, 10 items per page are shown * Options of items per page are : 10, 25, 50, 100 * Search will be triggered if administrator types in search box of minimum 2 chars, using ‘OR’ logic of all shown columns |
| **Success Result** | New master data is saved on Back-End |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | Invalid data provided |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### User creation approval

|  |  |
| --- | --- |
| **UC ID** | UC-57 |
| **Description** | Use Case to approve created new administrator user |
| **FR Reff** | FR-33 |
| **Actor** | Administrator (Director) |
| **Pre-Condition** | Administrator is already in page list user |
| **Trigger** | Administrator click approve button |
| **Normal Scenarios** | * Administrator click approve button on a row in list view users * User status will be approved |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | User status will be approved |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | Invalid data provided |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### Change password

|  |  |
| --- | --- |
| **UC ID** | UC-58 |
| **Description** | Use Case to change administrator user’s password |
| **FR Reff** | FR-33 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator logged in Admin Panel |
| **Trigger** | Administrator click Change Password menu |
| **Normal Scenarios** | * Administrator click Change Password in Setting * Show prompt change password * If the administrator click ok, email will be sent to administraor * If the administrator click no, process will be canceled |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Administrator will receive email to Change Password |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### Login by email

|  |  |
| --- | --- |
| **UC ID** | UC-59 |
| **Description** | Use Case to approve user address image |
| **FR Reff** | FR-33 |
| **Actor** | Administrator |
| **Pre-Condition** | * Administrator is on login page * Administrator has filled email and password field |
| **Trigger** | Administrator click login button |
| **Normal Scenarios** | * CMS will send email and password to Back-End * Back-End validate email and password * If valid then send privilege as API response |
| **UI Validation** | * Email field must be filled * Password field must be filled |
| **Business Rules** | * Email and password must be registered on Back-End * Email and password must be match with administrator’s input |
| **Success Result** | CMS navigate to Dashboard page |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | Email or Password invalid |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### Logout

|  |  |
| --- | --- |
| **UC ID** | UC-60 |
| **Description** | Use Case to logout from CMS |
| **FR Reff** | FR-33 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is logged in CMS |
| **Trigger** | Administrator click logout button |
| **Normal Scenarios** | * Administrator clik logout button in Setting Menu * CMS will navigate to login page |
| **UI Validation** | **-** |
| **Business Rules** | - |
| **Success Result** | CMS will navigate to login page |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### View registration data

|  |  |
| --- | --- |
| **UC ID** | UC-61 |
| **Description** | Use Case to view registration data |
| **FR Reff** | FR-34 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is on Data & Approval page |
| **Trigger** | Administrator load Registration Data |
| **Normal Scenarios** | CMS shows Registration Data |
| **UI Validation** | **-** |
| **Business Rules** | * There are paging and filter features * By default, 10 items per page are shown * Options of items per page are : 10, 25, 50, 100 * Filter will be triggered if administrator select field and types in box of minimum 2 chars * Fields shown: Investor Name, Online Distributor, Sign Up Date, Phone, Email, Account Activation Date, Status and Remark * Fields to be filtered are: Investor Name, Online Distributor (Dropdown), Sign Up Date (Date Range), Phone, Email, Account Activation Date (Date Range) and Status (Dropdown) |
| **Success Result** | CMS shows Registration Data |
| **Minimal Result** | No data shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### Registration data approval

|  |  |
| --- | --- |
| **UC ID** | UC-62 |
| **Description** | Use Case for registration data approval |
| **FR Reff** | FR-34 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is on detail registration data page |
| **Trigger** | Administrator click approve button |
| **Normal Scenarios** | * Administrator click approval button in view list registration data * CMS navigate to detail registration data page * Administrator click approve button |
| **UI Validation** | **-** |
| **Business Rules** | - |
| **Success Result** | Registration Data status will be updated in Back-End |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | * This use case will be used for approval by Marketing and Complience * Complience do approval after Marketing approved Registration Data |

### View updated data and old data

|  |  |
| --- | --- |
| **UC ID** | UC-63 |
| **Description** | Use Case to view updated data and old data |
| **FR Reff** | FR-34 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is on Data & Approval page |
| **Trigger** | Administrator load Update Data |
| **Normal Scenarios** | CMS shows Update Data |
| **UI Validation** | **-** |
| **Business Rules** | * There are paging and filter features * By default, 10 items per page are shown * Options of items per page are : 10, 25, 50, 100 * Filter will be triggered if administrator select field and types in box of minimum 2 chars * Fields shown: Investor Name, Online Distributor, Sign Up Date, Phone, Email, Account Activation Date, Status and Remark * Fields to be filtered are: Investor Name, Online Distributor (Dropdown), Sign Up Date (Date Range), Phone, Email, Account Activation Date (Date Range) and Status (Dropdown) |
| **Success Result** | CMS shows Update Data |
| **Minimal Result** | No data shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### Update Data approval

|  |  |
| --- | --- |
| **UC ID** | UC-64 |
| **Description** | Use Case for Update Data approval |
| **FR Reff** | FR-34 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is on detail Update Data page |
| **Trigger** | Administrator click approve button |
| **Normal Scenarios** | * Administrator click approval button in view list Update Data * CMS navigate to detail Update Data page * Administrator click approve button |
| **UI Validation** | **-** |
| **Business Rules** | - |
| **Success Result** | Update Data status will be updated and Investor data will be changed if approved by Complience |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | * This use case will be used for approval by Marketing and Complience * Complience do approval after Marketing approved Update Data |

### View transaction data

|  |  |
| --- | --- |
| **UC ID** | UC-65 |
| **Description** | Use Case to view transaction data |
| **FR Reff** | FR-35 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is on Transaction page |
| **Trigger** | Administrator load Transaction Data |
| **Normal Scenarios** | CMS shows Transaction Data |
| **UI Validation** | **-** |
| **Business Rules** | * There are tabs for each transaction (Subscription, Redemption and Switching) * There are paging and filter features * By default, 10 items per page are shown * Options of items per page are : 10, 25, 50, 100 * Filter will be triggered if administrator select field and types in box of minimum 2 chars * Fields shown for   + Subscription: Investor Name, Online Distributor, Transaction Date, Transaction ID, Product Name, Subscription Amount and Transaction Status   + Redemption: Investor Name, Online Distributor, Transaction Date, Transaction ID, Product Name, Redemption Amount, Unit and Transaction Status   + Swithing: Investor Name, Online Distributor, Transaction Date, Transaction ID, Product Switch Out, Switch Out Amount, Switch Out Unit, Product Switch In, Switch In Unit and Transaction Status * Fields to be filtered are: Investor Name, Online Distributor (Dropdown), Transaction Date (Date Range), Transaction ID, Product (Dropdown), Product Switch Out (Dropdown), Product Switch In (Dropdown), Transaction Amount (Amount Range) and Transaction Status (Dropdown) |
| **Success Result** | CMS shows Transaction Data |
| **Minimal Result** | No data shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### View report

|  |  |
| --- | --- |
| **UC ID** | UC-66 |
| **Description** | Use Case to view updated data and old data |
| **FR Reff** | FR-36 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is on Data & Approval page |
| **Trigger** | Administrator load Update Data |
| **Normal Scenarios** | CMS shows Update Data |
| **UI Validation** | **-** |
| **Business Rules** | * There are paging and filter features * By default, 10 items per page are shown * Options of items per page are : 10, 25, 50, 100 * Filter will be triggered if administrator select field and types in box of minimum 2 chars * Fields shown will be vary based on report:   + Distributor: Net Subscription YTD, Total Investor YTD, New Investor YTD, Subscription Value YTD, Subscription Count YTD, Redemption Value YTD, Redemption Count YTD and Distributor Name (If report is Detail per Distributor)   + Investor: Name, SID, Age, Occupation, AUM, Invest Days, Subscription Value, Subscription Count, Redemption Value, Redemption Count, Last Transaction Data, Inactive Days and Detail Portfolio (Product Name, Avarage Cost, Investment Value and Return)   + Transaction: Subscription Count, Avarage Subcription Value, Avarage Subcription Count, Redemption Count, Avarage Redemption Value, Avarage Redemption Count and Distributor Name (If report is Detail per Distributor)   + OTP: Investor, Phone, OTP Date, Activity and Status   + Product: Product Name, Total AUM, AUM Growth, Net Subscription, Subscription Value, Redemption Value, Total Investor and New Investor YTD   + Promo: Code, Distributor, Start Date, End Date, Subscription Value, Subscription Count and Promotion Cost. * Fields to be filtered will be vary based on report:   + Dashboard: Distributor and Date Range   + Distributor: Distributor and Date Range   + Transaction: Distributor and Date Range   + OTP: Distributor, Investor, Date Range and Status   + Product: Distributor and Date Range   + Promo: Distributor and Date Range |
| **Success Result** | CMS shows Report Data |
| **Minimal Result** | No data shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### Create task request report

|  |  |
| --- | --- |
| **UC ID** | UC-67 |
| **Description** | Use Case to create task request for report |
| **FR Reff** | FR-36 |
| **Actor** | Administrator |
| **Pre-Condition** | * Administrator is already in view page of one of the report menu :   + Dashboard   + Distributor   + Investor   + Transaction   + OTP   + Product   + Promo * Administrator tick checklist in the left-side of each row in table that want to be exported |
| **Trigger** | Admin click download button |
| **Normal Scenarios** | * Administrator click download button * Task request will be created and assign to correspond Director/Head Administrator * UC-69 will be triggered |
| **UI Validation** | - |
| **Business Rules** | * All mandatory fields must not be empty * End Date must be greater than Start Date * For master data Answer, master data Question must be created first |
| **Success Result** | New master data is saved on Back-End |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Low |
| **Notes** | Reports will be categorize as:   * Dashboard   + 1st Report (Tabular Dashboard): Total AUM, Net Subs, Net Subs Growth, Investor, New Investor, Investor Growth, Subscription, Top Up, Initial Subs and Redemption   + 2nd Report (Graphic AUM Growth and Tabular Distributor Statictic): Distributor, AUM, Subs, Reds and Investor   + 3rd Report (Tabular Latest Approval Request): Investor Name, Distributor, Sign Up, Status   + 4th Report (Tabular Latest Transaction): Trx ID, Distributor, Trx Date, Trx Status, Investor, Product and Trx Amount * Distributor   + AUM Growth YTD per Distributor (Graphic)   + AUM Growth YTD Summary (Graphic)   + Distributor Transaction Statistic per Distributor (Tabular): Distributor, Net Subs YTD, Total Investor, New Investor YTD, Subs Value YTD, Subs Count YTD, Reds Value YTD and Reds Count YTD   + Distributor Transaction Statistic Summary (Tabular): Net Subs YTD, Total Investor, New Investor YTD, Subs Value YTD, Subs Count YTD, Reds Value YTD and Reds Count YTD * Investor   + Investor Demography By Age Range (Graphic and Tabular): No, Age Range and Investor Count   + Investor Demography By Gender (Graphic and Tabular): No, Gender and Investor Count   + Investor Demography By Occupation (Graphic and Tabular): No, Occupation and Investor Count   + Investor Demography By Earning Range (Graphic and Tabular): No, Earning Range and Investor Count   + Avarage Transaction per Month of All Investors (Graphic Transaction Value, Graphic Transaction Count and Tabular): Periode, Subs Value, Subs Count, Reds Value, Reds Count   + Avarage AUM per Investor (Graphic and Tabular): No, Age Range, Total AUM, Total Investor, Avarage AUM   + Total Investor by Avarage AUM Range (Graphic and Tabular): No, Avarage AUM, Investor Count   + Investor Statistic (Tabular): Name, SID, Age, Occupation, AUM, Invest Days, Subscription Value, Subscription Count, Redemption Value, Redemption Count, Last Transaction Data, Inactive Days and Detail Portfolio (Product Name, Avarage Cost, Investment Value and Return) * Transaction   + Transaction Statistic per Distributor (Tabular): Distributor, Subs Count YTD, Avarage Subs Value, Avarage Subs Count per Day, Reds Count YTD, Avarage Reds Value and Avarage Reds Count per Day   + Transaction Statistic Summary (Tabular): Subs Count YTD, Avarage Subs Value, Avarage Subs Count per Day, Reds Count YTD, Avarage Reds Value and Avarage Reds Count per Day   + Top 10 Investor based on Subscription Transaction (Tabular): No, Investor Name and Subs Count   + Top 10 Investor based on AUM (Tabular): No, Investor Name and AUM   + Unit Holding Reconciliation (PDF): Name, SID, Distributor, Date and Detail (Product Name, Portfolio Unit, Back-Office Unit)   + Revise Transaction (PDF): Trx ID, Distributor, Trx Type, Trx Date, Product, Trx Amount, NAV Date, NAV Price, Trx Unit, Revise Date, NAV Date Revise, NAV Price Revise and Revise Unit   + Void Transaction (PDF): Trx ID, Distributor, Trx Date, Product, Trx Amount, Trx Unit and Void Date   + Trx Summary (PDF): Agent, Product, Investor Name, Trx ID, Trx Date, Trx Amount, Trx Unit and Trx Type * OTP (Tabular): Distributor, Investor Name, Phone, OTP Date, Activity and Status * Product (Tabular): Product Name, Total AUM, AUM Growth, Net Subs, Subs Value, Reds Value, Total Investor an New Investor * Promo   + List of Promos (Tabular): Code, Distributor, Periode (Start & End), Subs Count, Subs Value and Promotion Cost   + Total Subscription Growth during Promotion Date Range (Graphic)   + Promo Code usage per day (Graphic)   + Promo usage by Age Range (Graphic and Tabular): No, Age Range and Usage count   + Promo usage by Gender (Graphic and Tabular): No, Gender and Usage count   + Promo usage by Occupation (Graphic and Tabular): No, Occupation and Usage count   + Promo usage by Earning Range (Graphic and Tabular): No, Earning Range and Usage count |

### Generate report

|  |  |
| --- | --- |
| **UC ID** | UC-68 |
| **Description** | Use Case to generate report |
| **FR Reff** | FR-36 |
| **Actor** | Administrator |
| **Pre-Condition** | UC-69 has been passed |
| **Trigger** | UC-69 |
| **Normal Scenarios** | Report will be generated and saved in Back-End server |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | New file report is saved on Back-End |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Low |
| **Notes** | - |

### View Report Request

|  |  |
| --- | --- |
| **UC ID** | UC-69 |
| **Description** | Use Case to view Report Request |
| **FR Reff** | FR-36 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is on Data & Approval page |
| **Trigger** | Administrator load Report Request |
| **Normal Scenarios** | CMS shows list Report Request |
| **UI Validation** | **-** |
| **Business Rules** | * There are paging and filter features * By default, 10 items per page are shown * Options of items per page are : 10, 25, 50, 100 * Filter will be triggered if administrator select field and types in box of minimum 2 chars * Fields shown: Report Name, Request by, Assign to, Request Date, Expiry Date, Link Download, Status and Remark * Fields to be filtered are: Report Name, Request by, Assign to, Request Date (Date Range), Expiry Date (Date Range) and Status (Dropdown) |
| **Success Result** | CMS shows list Report Request |
| **Minimal Result** | No data shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### Report Request approval

|  |  |
| --- | --- |
| **UC ID** | UC-70 |
| **Description** | Use Case for Report Request approval |
| **FR Reff** | FR-36 |
| **Actor** | Administrator |
| **Pre-Condition** | Administrator is on detail Report Request page |
| **Trigger** | Administrator click approve button |
| **Normal Scenarios** | * Administrator click approval button in view list Report Request * CMS navigate to detail Report Request page * Administrator click approve button |
| **UI Validation** | **-** |
| **Business Rules** | - |
| **Success Result** | Report Request status will be updated in Back-End |
| **Minimal Result** | Error dialog will be shown |
| **Extention** | - |
| **UI Image** |  |
| **Frequency of Usage** | Medium |
| **Notes** | - |

### Cronjob to delete expired report

|  |  |
| --- | --- |
| **UC ID** | UC-71 |
| **Description** | Use Case for scheduled delete expired report |
| **FR Reff** | FR-36 |
| **Actor** | Back-End |
| **Pre-Condition** | Cronjob setting to delete expired report already set in Back-End |
| **Trigger** | Automatic |
| **Normal Scenarios** | * Cronjob check table report request for expired report request based on Expiry Date * If Expiry Date was a day before then Cronjob will delete report file from Back-End |
| **UI Validation** | - |
| **Business Rules** | - |
| **Success Result** | Report file will be deleted |
| **Minimal Result** | - |
| **Extention** | - |
| **UI Image** | - |
| **Frequency of Usage** | Daily |
| **Notes** | - |