MTN Rwanda - Solution Description



Copyright

© Ericsson AB 2022 - All Rights Reserved

**Disclaimer:**

No part of this document may be reproduced in any form without the written permission of the copyright owner. The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

**Trademark List:**

Ericsson is the trademark or registered trademark of Telefonaktiebolaget LM Ericsson. All other products or service names mentioned in this document are trademarks of their respective companies

**About Ericsson**

Ericsson enables communications service providers to capture the full value of connectivity. The company’s portfolio spans Networks, Digital Services, Managed Services, and Emerging Business and is designed to help customers go digital, increase efficiency and find new revenue streams. Ericsson’s investments in innovation have delivered the benefits of telephony and mobile broadband to billions of people around the world. The Ericsson stock is listed on Nasdaq Stockholm and on Nasdaq New York. [www.ericsson.com](http://www.ericsson.com/)

Contributors

|  |  |
| --- | --- |
| Archana Yadav | Business Analyst |
|  |  |

**Revision History**

|  |  |  |
| --- | --- | --- |
| Revision | Date | Comments |
| PA1 | 15-Jan-2025 | First draft version |
| PA2 | 26-Jan-2025 | Design flow update:  1.Ask Pin before showing balance.  2.Getfri call replaced with getbalanceAPI to verify pin  3.Other tenure customer journey is added in the sequence flow. |
| PA3 | 28-Jan-2025 | Test Cases Added.  Bank debit request flow added. |

Table of Contents

[1 Introduction 6](#_Toc189079933)

[1.1 Purpose 6](#_Toc189079934)

[2 Scope 6](#_Toc189079935)

[2.1 Intended Audience 7](#_Toc189079936)

[2.2 Assumptions 7](#_Toc189079937)

[2.3 Out of Scope 7](#_Toc189079938)

[3 Solution Architecture 8](#_Toc189079939)

[3.1 Solution Overview 8](#_Toc189079940)

[4 Use Cases 9](#_Toc189079941)

[4.1 Lock Saving Feature 9](#_Toc189079942)

[4.1.1 Lock savings account – Open account. 9](#_Toc189079943)

[4.1.2 Lock savings account – Deposit. 15](#_Toc189079944)

[4.1.3 Lock savings account - Withdrawal. 21](#_Toc189079945)

[4.1.4 Lock savings account – Activate auto lock savings. 23](#_Toc189079946)

[4.1.5 Bank Debit (Initiated by Mokash) 25](#_Toc189079947)

[4.1.6 Lock Savings account – Cancel auto lock savings. 28](#_Toc189079948)

[4.1.7 Lock savings account – Check Balance 31](#_Toc189079949)

[4.1.8 Lock savings account – Mini Statement 32](#_Toc189079950)

[4.1.9 Lock savings account – Product Information 35](#_Toc189079951)

[4.2 Loan Top Up 36](#_Toc189079952)

[4.2.1 Requesting for a Loan Top-up 36](#_Toc189079953)

[4.2.2 Loan Top-up – Full/Partial Repayment 38](#_Toc189079954)

[4.2.3 Loan Top-up – Check Loan Limit 40](#_Toc189079955)

[4.2.4 Loan Top up – Mini Statement 41](#_Toc189079956)

[4.2.5 Loan Top up – Product Information 43](#_Toc189079957)

[5 Business Rule 44](#_Toc189079958)

[5.1 Lock Saving Account 44](#_Toc189079959)

[6 Customer journey 45](#_Toc189079960)

[6.1 Lock Saving Feature 45](#_Toc189079961)

[6.2 Loan Top Up 45](#_Toc189079962)

[7 API Information 46](#_Toc189079963)

[7.1 GetCustomerDetails/response 46](#_Toc189079964)

[7.2 Locksavingactiverequest/response 46](#_Toc189079965)

[7.3 LockSavingsDepositrequest/response 47](#_Toc189079966)

[7.4 Locksavingsautosavingsrequest/response 48](#_Toc189079967)

[7.5 Locksavingscancelautosavingsrequest/response 49](#_Toc189079968)

[7.6 Lockwithdrawrequest/response 49](#_Toc189079969)

[7.7 Lockbalancerequest/response 50](#_Toc189079970)

[7.8 Ministatement request/response 51](#_Toc189079971)

[7.9 Initiate Loan Application Request/Respose 51](#_Toc189079972)

[7.10 PaymentRequest/Response of Mokash 52](#_Toc189079973)

[7.11 Getbalance v1.2(ECW API) 53](#_Toc189079974)

[7.12 Payment request/response (ECW API) 54](#_Toc189079975)

[7.13 Addpreapproval (ECW API) 56](#_Toc189079976)

[7.14 CancelPreapproval API (ECW API) 58](#_Toc189079977)

[7.15 Bank Debit (SNL 1.6) 60](#_Toc189079978)

[8 Security Description – CLM: 60](#_Toc189079979)

[8.1 Security Impact 60](#_Toc189079980)

[8.2 Security Risk Management 61](#_Toc189079981)

[8.3 Security Requirement 61](#_Toc189079982)

[9 Notifications 64](#_Toc189079983)

[9.1 SMS Notifications 64](#_Toc189079984)

[9.2 USSD Notifications 64](#_Toc189079985)

[9.3 XML Notifications 64](#_Toc189079986)

[10 Test Cases 65](#_Toc189079987)

[10.1 Test Cases 65](#_Toc189079988)

[11 References 66](#_Toc189079989)

[12 Glossary 67](#_Toc189079990)

[Approvals 68](#_Toc189079991)

# Introduction

This design document outlines the enhancements proposed for the MoKash platform, a collaborative financial service between NCBA and MTN Mobile Money, introduced in February 2017. The primary objective of these enhancements is to introduce a lock savings account feature and enable loan top-up capabilities for MoKash customers. These features aim to expand the financial services offered through MoKash, thereby enhancing the mobile money value proposition and promoting financial inclusion.

The lock savings account will allow customers to deposit funds for a defined term, ranging from 1 to 12 months, with an interest rate of up to 10% per annum on the quarterly balance. This feature is designed to encourage savings among users by offering competitive interest rates and flexible terms.

Additionally, the loan top-up feature will enable customers to request additional funds on an existing loan, subject to specific business rules and conditions. This enhancement is intended to provide customers with greater flexibility in managing their financial needs.

The solution document will detail the product features, business and operational rules, user interface requirements, security considerations, and support requirements necessary for the successful implementation of these enhancements. The goal is to ensure a seamless integration of these features into the existing MoKash platform, thereby improving the overall user experience and meeting the evolving needs of our customers.

## Purpose

The purpose of this document is to design solution for business requirement to introduce a **lock savings account** feature and enable **loan top-up capabilities** for MoKash customers.

# Scope

The scope of the design document, based on the provided user requirement specification document, includes the following:

* Integration of new features into the existing MoKash platform, focusing on enhancing customer experience and meeting their financial needs.
* Detail the customer journey for both the **lock savings account** and the **loan top-up feature**, ensuring a seamless user experience.
* Outline the new business rules for **loan top-ups** and the rules for the proposed **lock savings** feature.

**Channels**:

1. USSD
2. MY MTN App (Smart App)
3. SMS (Notifications)

## Intended Audience

The following are included in the target group for this document:

1. Solution Architects
2. Account Managers
3. Project Managers
4. System Integrators

## Assumptions

The below assumptions are considered while designing this solution –

1. ECW and Mokash Integration already exists.
2. The bankdebitrequest already exists and initiated from Mokash towards ECW.
3. CBA-PERSONAL and CBA-SAVINGS OVA account already exist.
4. There is no change in existing loan solution (SNL Solution)

Loan Top UP Configuration are at the 3PP end.

* Loan top up can only be utilized between Day-1 and Day-20 of initial disbursement.
* A user can top up on their loan only once after the initial draw down.
* If a customer partially pays their loan and both drawdowns are still outstanding, the customer cannot draw down again.
* Loan Request cannot be less than Mokash minimum loan request amount RWF 1000.
* For use cases involving only MoKash accounts for debits and credits, EWP will not provide reconciliation logs or reports. MoKash Bank will be responsible for transaction handling.
* For use cases where the MTN Momo wallet is not involved in any transaction, EWP solely authenticates the user by verifying the PIN.

## Out of Scope

Anything which is not explicitly mentioned in the document is out of scope.

# Solution Architecture

## Solution Overview

The solution consists implementation of

1. Mokash Lock Saving feature
2. Loan Top Up feature.

New customer journey for subscriber via USSD /APP channel for new features.

# Use Cases

Below are the use cases -

## Lock Saving Feature

The proposed scope within the implementation of the Lock Savings feature will include the following enhancements and new features:

a. Available to all Mobile Money customers aged 16 and above.

b. Minimum deposit amount of RWF 5,000. (Configurable)

c. Customers can choose a lock-in period ranging from 1 to 12 months as well as tenure months.

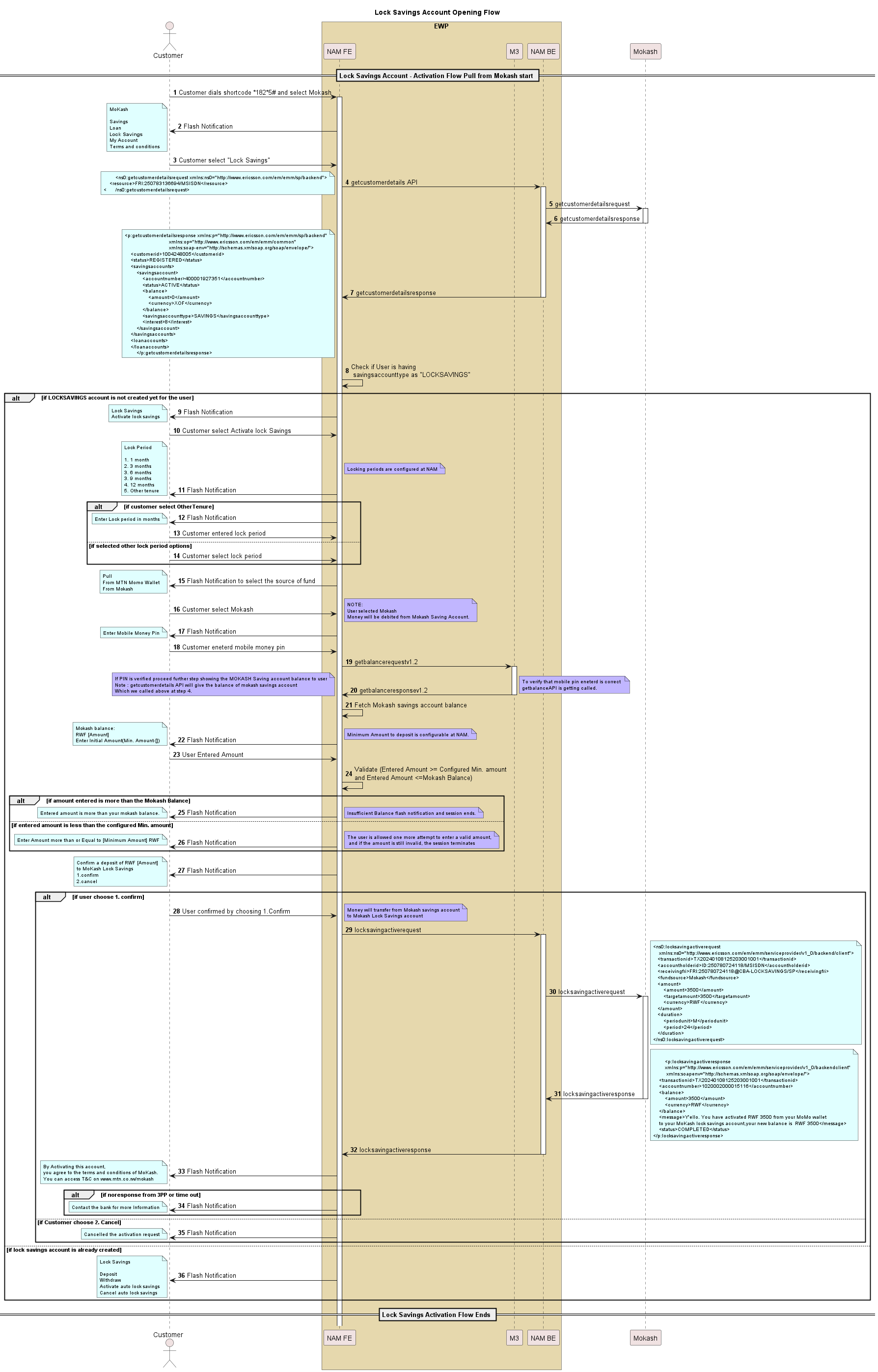
d. Customers can view and manage their Locked Savings Account through the mobile money platform.

e. Customers can view their account balance and mini statement at any time at no cost.

### Lock savings account – Open account.

#### Open Account – From Mokash

Customers activate/open the lock savings account by depositing the minimum configurable amount. Customer is given two option MTN wallet and Mokash savings account from which account they want to debit the money. Below is the sequence for the same:



**Description**:

The UML sequence diagram illustrates the process of opening and activating a Lock Savings Account through Mokash, involving key components such as the Customer, Front End (FE), Middleware (M3), Back End (BE), and the third-party participant Mokash (3PP).

**Process Overview:**

**Initiation**:

* The customer dials a shortcode (1825#) and selects the Mokash service.
* Upon choosing "Lock Savings," the AMFE initiates a getcustomerdetails request to the BE.

**Customer Details Retrieval**:

* The BE forwards a request to Mokash (3PP) to fetch the customer's details, processes the response, and sends the details back to the AMFE.
* Save the Mokash savings account balance in arguments.

**Existing Account Check**:

* If a locked savings account already exists, the customer is presented with options such as Deposit, Withdraw, Activate/Cancel Auto Lock Savings, or Deactivate Lock Savings.
* If no locked account exists, the user is offered the option to activate a new Lock Savings Account.

**Account Activation**:

* The user selects "Activate Lock Savings Account" and specifies the lock period and tenure in months.
* The user is prompted to choose the funding source—either their MTN MoMo account or Mokash savings account.
* After selecting Mokash as the source, the user is asked to enter Mobile Money PIN.PIN is verified after calling get balance API from FE towards M3
* After getting successful response from M3 User is shown Mokash Balance and asked to enter the deposit amount.

**Deposit Validation**:

* If the entered amount is below the configured minimum, the session ends after retrying one more time.
* If the entered amount is less than the balance of Mokash saving account, ussd flash notification with insufficient funds will be shown to user.
* For valid amounts, the customer is prompted to confirm the message that mentioned amount debited for activating lock account for mentioned lock period.

Final Steps:

* Upon confirming the message locksavingsactivation is triggered from FE towards BE
* AMBE forward the locksavingsactivation request towards the 3PP. Refer to the API documentation for detailed API specifications.

**Completion**:

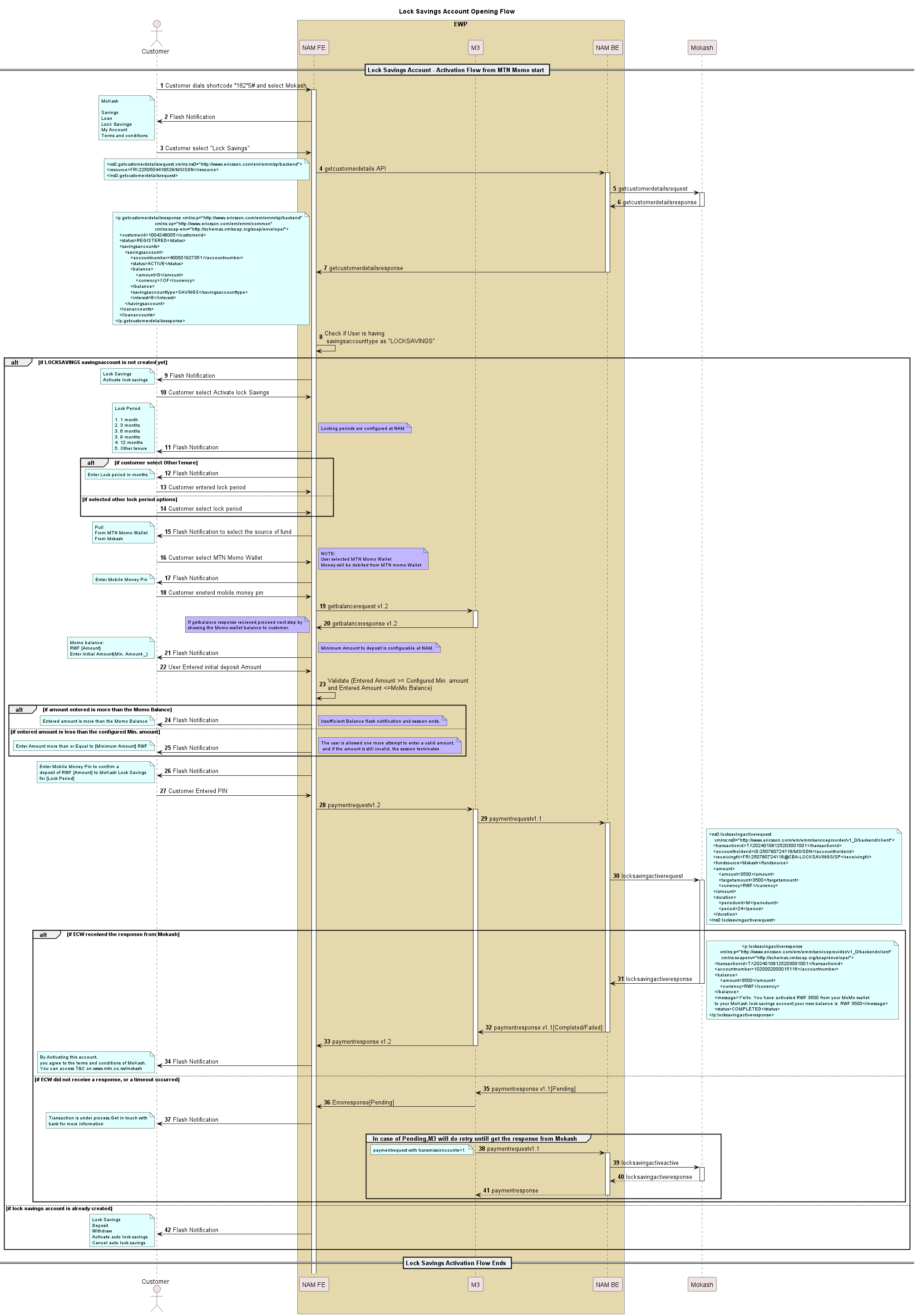
* The AMFE notifies the customer that by activating the Lock Savings Account, they agree to the Mokash terms and conditions by USSD Flash notification after receiving success full response from 3PP.

**Summary**:

This sequence describes the interactive steps and conditional validations required to open a Lock Savings Account via Mokash. It ensures customer authorization, adheres to predefined configurations, and maintains a seamless flow for account activation and deposit handling.

#### Open Account- From Momo Wallet

Customers activate/open the lock savings account by depositing the minimum configurable amount. Customer is given two option MTN wallet and Mokash savings account from which account they want to debit the money. Below is the sequence for the same:



**Description**:

The UML sequence diagram illustrates the process of opening and activating a Lock Savings Account through Mokash, involving key components such as the Customer, Front End (FE), Middleware (M3), Back End (BE), and the third-party participant Mokash (3PP).

**Process Overview:**

**Initiation**:

* The customer dials a shortcode (1825#) and selects the Mokash service.
* Upon choosing "Lock Savings," the AMFE initiates a getcustomerdetails request to the AMBE.

**Customer Details Retrieval**:

* The AMBE forwards a request to Mokash (3PP) to fetch the customer's details, processes the response, and sends the details back to the AMFE.

**Existing Account Check**:

* If a locked savings account already exists, the customer is presented with options such as Deposit, Withdraw, Activate/Cancel Auto Lock Savings, or Deactivate Lock Savings.
* If no locked account exists, the user is offered the option to activate a new Lock Savings Account.

**Account Activation**:

* The user selects "Activate Lock Savings Account" and specifies the lock period.
* The user is prompted to choose the funding source—either their MTN MoMo account or Mokash savings account.
* After selecting MTN Momo wallet as the source, user is asked to enter mobile money pin. Customer enter the PIN and FE initiate get balance API is triggered towards M3 to fetch the balance and then the user asked to enter the deposit amount.

**Deposit Validation**:

* If the entered amount is below the configured minimum, the session ends after user try for second time.
* If the entered amount is less than the balance of momo wallet, ussd flash notification with insufficient funds will be shown to user and session ends.
* For valid amounts, the customer is prompted to enter their Mobile Money PIN to confirm the deposit and lock period.

Final Steps:

* Upon PIN entry, the system triggers a payment request towards M3.
* M3 triggers payment request v1.1 towards AMBE and AMBE initiate the locksavingsactivation request towards the 3PP. Refer to the API documentation for detailed API specifications.
* Locksavingsactivation response contains the parameter status with either “Completed” or “Failed” value, according to which payment response is send back to M3.
* If there is no response from 3PP, M3 will do retry with same transaction id to get the response.

**Completion**:

* The AMFE notifies the customer that by activating the Lock Savings Account, they agree to the Mokash terms and conditions by ussd flash notification if the status is completed.

**Summary**:

This sequence describes the interactive steps and conditional validations required to open a Lock Savings Account. It ensures customer authorization, adheres to predefined configurations, and maintains a seamless flow for account activation and deposit handling.

### Lock savings account – Deposit.

#### Deposit from Mokash Savings Account

Customers can deposit into lock savings account. Customer is given two option MTN wallet and Mokash savings account from which account they want to debit the money. Here Money will debit from Mokash Savings account to Lock Savings Account.

Below is the sequence for the same:



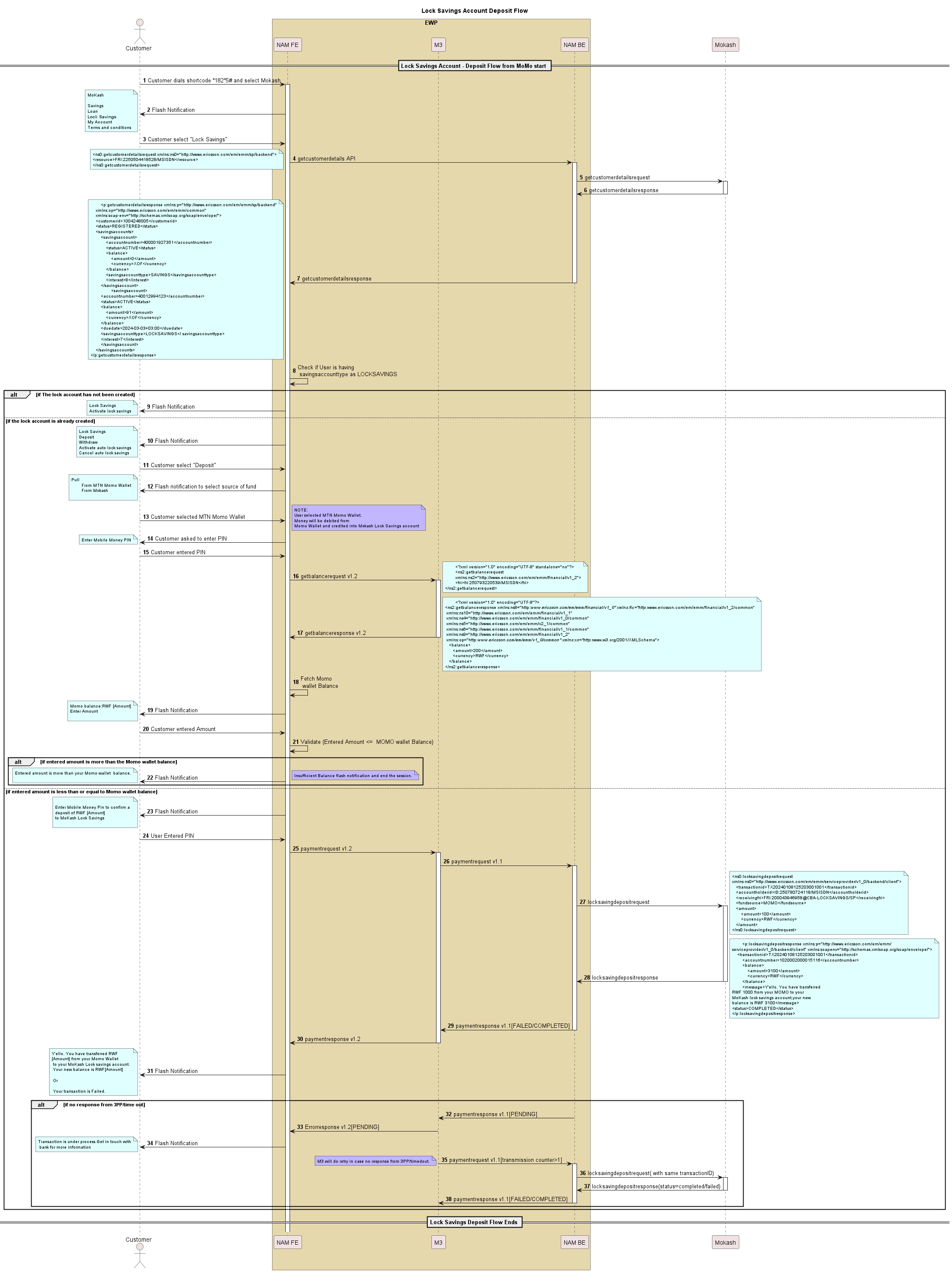
Description:

The sequence flow for making a deposit into a Lock Savings Account via the MoKash service is outlined above. The process begins when a customer dials a shortcode to access MoKash services and selects the "Lock Savings" option.

1. The NAM front-end interacts with the NAM back-end to fetch customer details, including checking if the customer already has a Lock Savings Account, using the getcustomerdetails API call.
2. The NAM front-end checks whether the customer has an existing Lock Savings Account. If no account exists, the customer is prompted to activate one. If an account is found, the customer is given options to manage the account, such as making a deposit or withdrawal.
3. If the customer has a Lock Savings Account and selects the "Deposit" option, they are prompted to choose a source for the deposit (MTN Momo Wallet or MoKash).
4. Customers choose the source fund as Mokash and asked to enter Mobile Money PIN.To verify the PIN, FE initiate the getbalanceAPI towards M3.After getting successful response user is asked to enter the amount.
5. The NAM front-end validates this amount against the customer’s Mokash balance, which is retrieved from the getcustomerdetails response. User should enter amount less than or equal to the MoKash account savings balance.
6. After amount validation, the customer is asked to confirm the message that says entered amount will be debited.
7. On confirming, the NAM FE triggers a locksavingsdeposit request via NAMBE towards 3PP including transaction details like transaction ID (Systemdatetime), account holder ID, the amount to transfer, and fund source (Mokash). Refer to the API information.
8. AMBE communicates with the 3PP to finalize the deposit.
9. A response from the 3PP confirms the transaction's completion, including details such as the transaction ID, updated account balance, and a confirmation message.
10. Finally, the customer receives a notification confirming the transfer amount and the new balance.

#### Deposit from MTN Momo Wallet

Customers can deposit into lock savings account. Customer is given two option MTN wallet and Mokash savings account from which account they want to debit the money. Here Money will debit from MTN MOMO wallet to Lock Savings Account.



**Description**

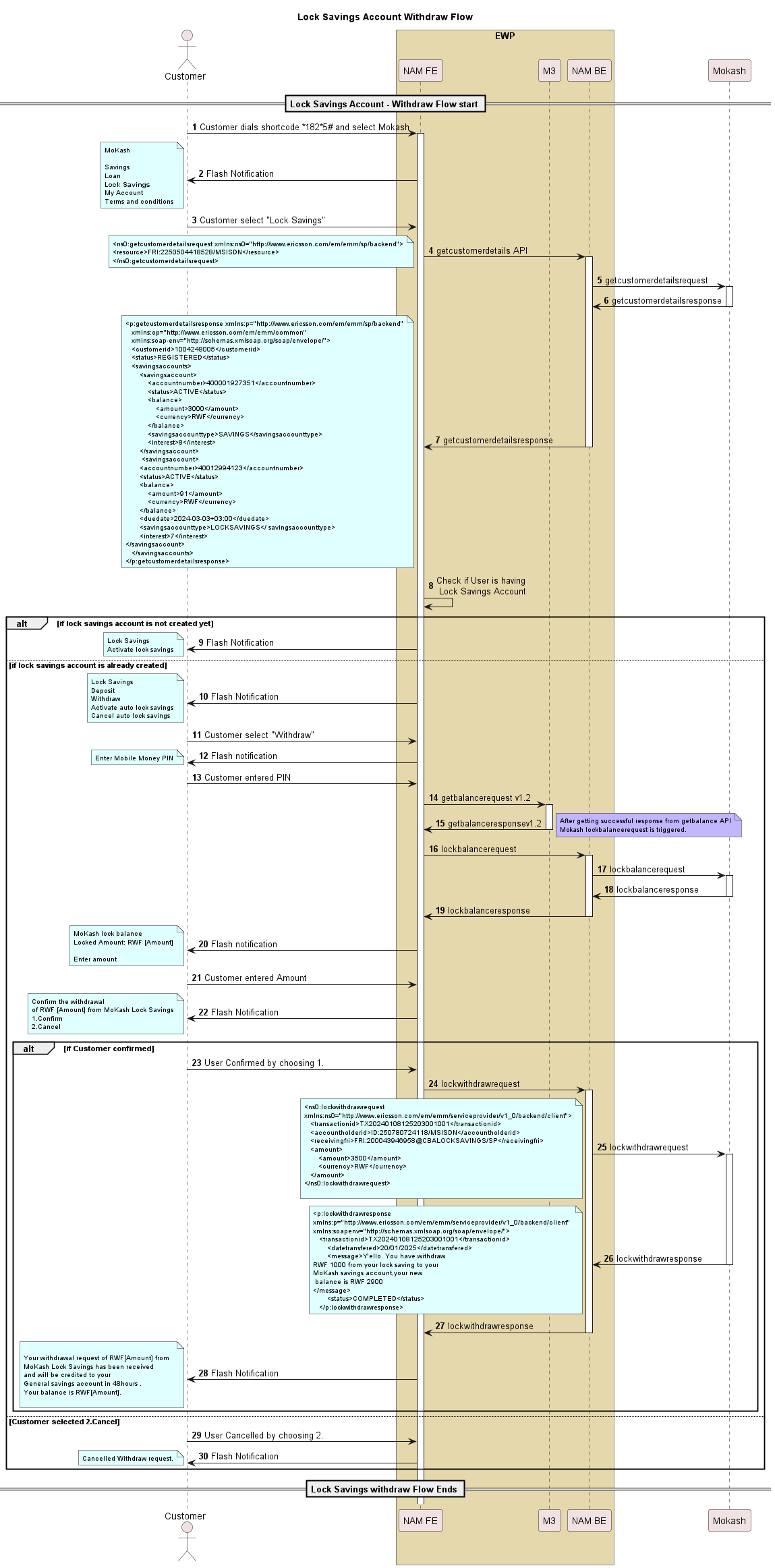
The sequence flow for making a deposit into a Lock Savings Account via the Mokash service (where the source of fund is MTN Momo) is outlined above. The process begins when a customer dials a shortcode to access Mokash services and selects the "Lock Savings" option.

1. The NAM front-end interacts with the NAM back-end to fetch customer details, including checking if the customer already has a Lock Savings Account, using the getcustomerdetails API call towards 3PP.
2. The NAM front-end checks whether the customer has an existing Lock Savings Account. If no account exists, the customer is prompted to activate one. If an account is found, the customer is given options to manage the account, such as making a deposit or withdrawal.
3. If the customer has a Lock Savings Account and selects the "Deposit" option, they are prompted to choose a source for the deposit (MTN Momo Wallet or Mokash)
4. Customers choose the source fund as MTN wallet, then getbalanceAPI is triggered towards M3 to check the balance of Momo wallet after entering the Mobile Money PIN.
5. Customer is asked to enter the amount for deposit.
6. The NAM front-end validates this amount against the customer’s MOMO balance, which is retrieved from the getbalance response. User should enter amount less than or equal to the MTN wallet balance.
7. After amount validation, the customer is asked to enter their Mobile Money PIN to confirm the deposit. AMFE triggers payment v1.2 towards M3 which in turn triggers paymentv1.1 towards AMBE.
8. The NAM back end triggers a locksavingsdeposit request towards 3PP including transaction details like transaction ID (from M3), account holder ID, the amount to transfer, and fund source (MOMO). Refer to the API information.
9. AMBE communicates with the 3PP to finalize the deposit.
10. A response from the 3PP confirms the transaction's completion, including details such as the transaction ID, updated account balance, and a confirmation message with status.
11. Status parameter in locksavingsdeposit can contain either “Completed” or “Failed” according to which payment response is updated.
12. If there is no response from M3 due to network issue or time out M3 will do retry with the same transaction id.
13. The customer receives a USSD Flash notification confirming the transfer amount and the new balance if successful.
14. Be default, product SMS will be sent to Customer.

### Lock savings account - Withdrawal.

#### Withdrawal (Lock Savings Account to Mokash Savings Account)

This use case is for Customers who wants to withdraw money from their lock savings account to Mokash savings account. Below is the sequence flow for the same:



**Description**

The UML diagram depicts the workflow for a customer to withdraw from a Lock Savings Account using Mokash services.

Here is a description of the process shown in the UML:

1. The Customer starts by dialing the shortcode 1825# to access the MoKash menu, then selects "Lock Savings."
2. The Frontend (NAM FE) requests and receives customer details from the Backend (NAM BE), which in turn communicates with the third-party system (Mokash) to obtain detailed information about the customer's accounts.
3. The NAMFE checks if the customer has a Lock Savings account. If the account is found, options such as Withdraw, Deposit, etc are presented to the customer.
4. The Customer selects "Withdraw" and a ussd flash notification is shown to the user to enter PIN. Customer enters PIN.FE initiate get balance API to verify the PIN.

After getting successful response from M3, FE triggers lockbalancerequest API to get lock savings balance. A flash notification is shown to the user mentioning about lock saving balance and asked to enter the withdrawal amount.

1. The system prompts the Customer confirm the withdrawal message.
2. If Customer confirms, NAMFE triggers lockwithdrawrequest towards NAMBE, which communicates with Mokash to initiate a lock withdrawal request by lockwithdrawrequest API.
3. Mokash sends a lockwithdrawresponse back through the system layers that updates the transaction details, including a message confirming the withdrawal amount being processed.
4. The same is shown to the user by USSD Flash Notification.

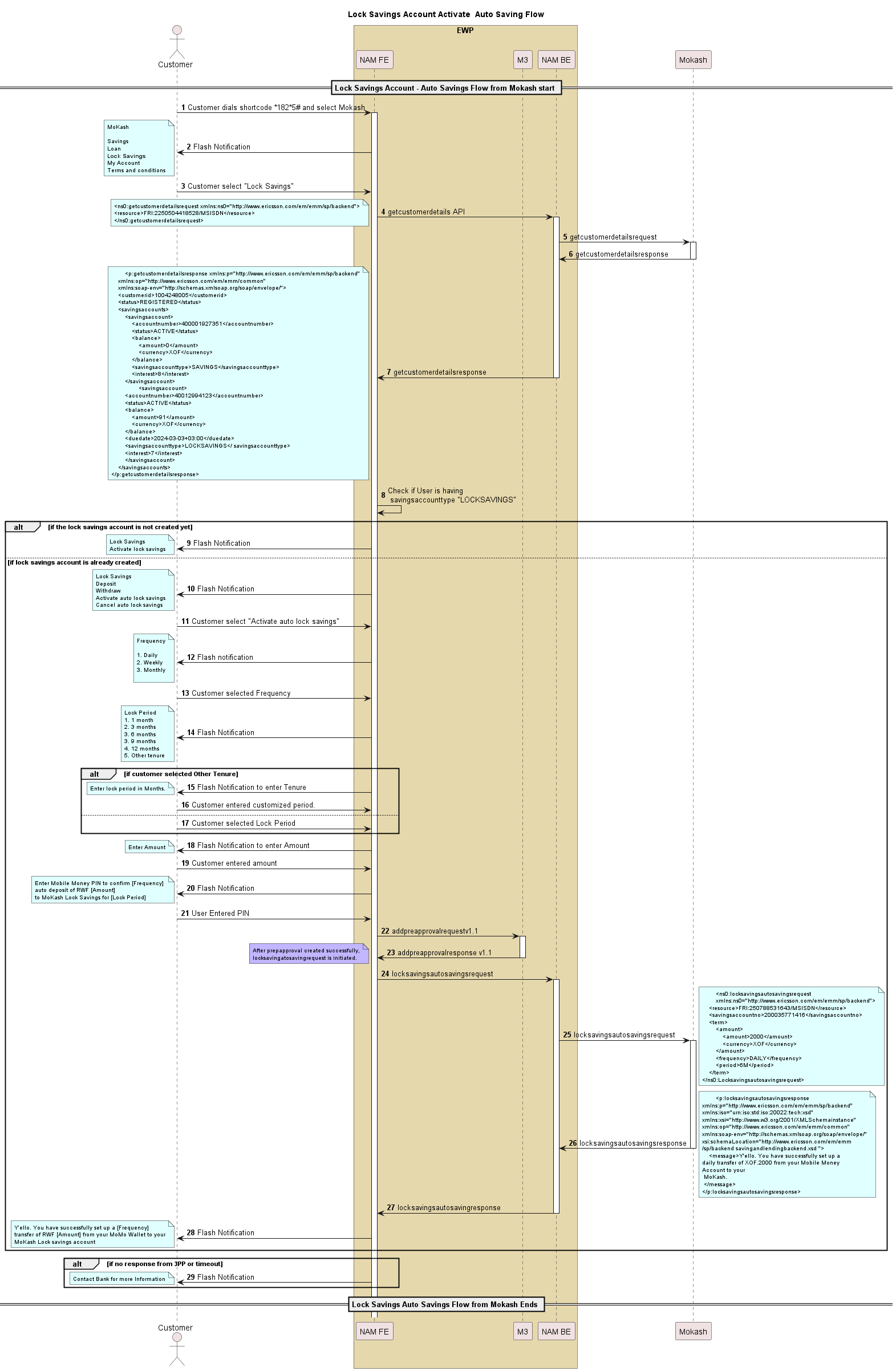
Note: In case there is timeout in case of 3PP API like lockbalancerequest or lockwithdrawrequest a Flash Notification to User “Contact Bank for further Information”.

Throughout this process, the system ensures necessary checks and security validations are performed, such as verifying account activity and user authentication via PIN entry.

### Lock savings account – Activate auto lock savings.

#### Activate auto lock savings.

Customer can give approval for auto deduction for lock savings. The sequence flow is described below.



**Description**

The UML diagram outlines the process for a customer to activate the "Lock Savings Account Auto Savings" feature using the Mokash service. Here's a description based on the provided UML:

1. The process begins with the customer interacting with the Front-End (NAM FE) by dialing a shortcode (1825#) and selecting "Lock Savings." The FE sends a flash notification to the customer, presenting several options including "Mokash Savings Loan," "Lock Savings," "My Account," and "Terms and conditions."
2. Customer selects Lock Savings then the NAMFE sends a getcustomerdetails API request to the NAM Backend (BE), which further sends this request to the external Mokash service (3PP). The 3PP responds with customer details, and this response is relayed by the Backend back to the NAMFE. NAMFE checks if lock saving account exist or not.
3. If an account is already set up, options for activating an auto lock saving are displayed. The customer can then choose the frequency of savings (daily, weekly, monthly) and the lock period (ranging from 1 month to 12 months or other custom tenure).
4. The customer is prompted to enter the amount they wish to save.
5. Upon entry, the FE prompts the customer to enter their Mobile Money PIN to confirm an auto deposit setup to the Mokash Lock Savings for a specified lock period.
6. The FE sends addpreapprovalrequest v1.1 to the M3 system and receives a response, following which it sends a locksavingsautosavingsrequest to the AMBE. The AMBE forwards this to the 3PP, which processes it and sends back a response confirming the setup.
7. The FE sends a final flash notification for the customer to confirm their selected frequency, tenure, and amount for the auto-saving feature.
8. If no response received from 3PP or time out occurs, then a flash notification shown to the user ”Contact bank for more information”.

### Bank Debit (Initiated by Mokash)

The Bank Debit Request, initiated by Mokash, is already operational for Mokash Savings. This functionality will be extended for Mokash Lock Savings as well.

Below is a sample request for a Bank Debit initiated by Mokash for Lock Savings

<?xml version="1.0" encoding="UTF-8"?>

<fron:bankdebitrequest xmlns:fron="http://www.ericsson.com/em/emm/sp/frontend"> <fromfri>FRI:250788312326/MSISDN</fromfri>

**<tofri>FRI: CBA-LOCKSAVINGS/USER</tofri>**

<amount>

<amount>4360</amount>

<currency>RWF</currency>

</amount>

<externaltransactionid>M20240105134344131</externaltransactionid>

<frommessage>Auto Debit Request</frommessage>

<tomessage>Auto Debit Request</tomessage>

<referenceid>M20240105134344131</referenceid>

<simplesweep>false</simplesweep>

</fron:bankdebitrequest>

#### Sequence Flow – Bankdebit



Description

The provided sequence flow diagram illustrates the Bank Debit process.

Below is a detailed description of the process:

The process begins with the service provider initiating a Bank Debit Request.

1. The request contains key details such as account type, MSISDN, and service provider,amount,externaltransaction id,referenceid.
2. EWP validates the account holder details by sending a validateAccountHolderRequest to M3.
3. If the validation is successful, a validateAccountHolderResponse is returned to NAMBE .If validateAccountHolderResponse is false, then throw internal error exception “ACCOUNT\_HOLDER\_NOT\_FOUND”
4. If the account holder is valid and the Savings Account Type is LOCK SAVINGS, a get preapprovals request (Identity and Status = approved) is initiated.
5. Preapproval response will contain list of preapprovals for that user.
6. Filter out based on TOFRI. TOFRI should contain Lock Savings as accountype and CBA as service provider.
7. If a preapproval exists, the process proceeds to the next step otherwise thow internal error exception”PREAPPROVAL\_DOES\_NOT\_EXISTS”.
8. A balance inquiry request (getBalanceRequestV1.2) is sent to M3 to check the available balance.
9. A validation check ensures that the balance is greater than or equal to the debit amount.
10. If the balance is sufficient, a debit request is sent else internal error exception” NOT\_ENOUGH\_FUNDS”.
11. The debit response is received, confirming whether the debit was successful or failed.
12. If the transaction is successful, a bankDebitResponse(status=SUCCESSFUL) is sent to 3PP.
13. If the transaction fails, a bankDebitResponse(status=FAILED) is returned to 3PP.

NOTE:

SavingsAccountType = LOCKSAVINGS should be allowed for processing the bank debit.

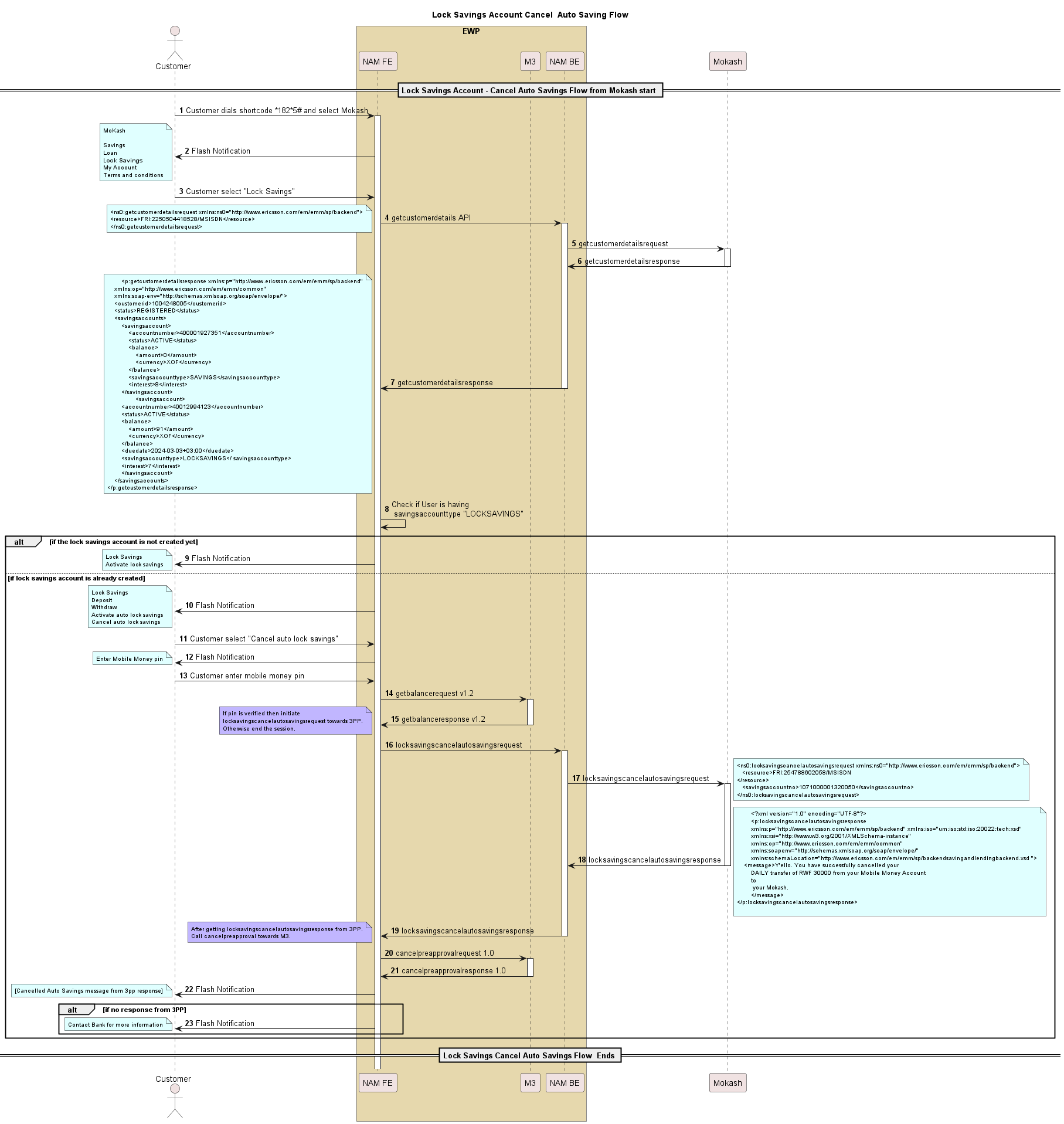
Preapproval must be checked and validated before proceeding.

If the balance is insufficient, the transaction is terminated with a error response.

This sequence flow ensures a secure and structured process for bank debit transactions related to Mokash Lock Savings.

### Lock Savings account – Cancel auto lock savings.

#### Sequence Flow Diagram -Cancel auto Lock Savings



**Description**

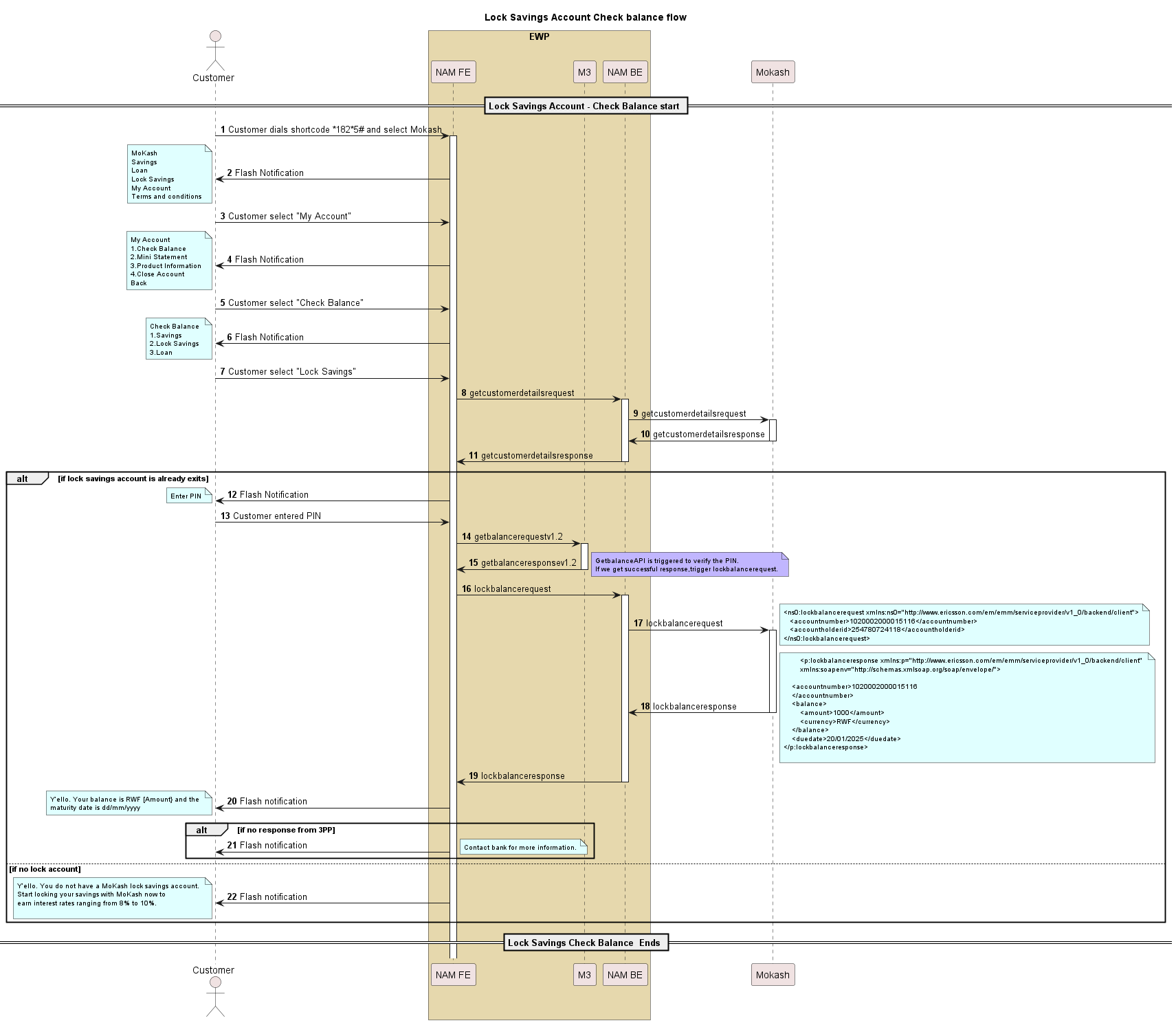
The UML diagram describes the process of cancelling auto savings for a Lock Savings Account in the Mokash service.

1. The flow starts with a customer using a specific shortcode on their phone to access Mokash services, choosing the "Lock Savings" option. The front end (NAM FE) gets customer details through a back-end API call, and checks if the customer already has a Lock Savings account.
2. If a Lock Savings account exists, the customer can choose to cancel auto savings. This triggers a series of API requests from the front end to the back end, and then to an external third-party service (Mokash). The process involves a getbalance to verify pin "locksavingscancelautosavingsrequest" and then "cancelpreapprovalrequest" .
3. The backend finally receives a response from the third-party service, confirming the cancellation of the daily auto transfer, which is then communicated back to the customer via a flash notification.

The diagram also includes XML snippets representing the data involved in the API requests and responses.

### Lock savings account – Check Balance

#### Sequence Flow for check balance



**Description**

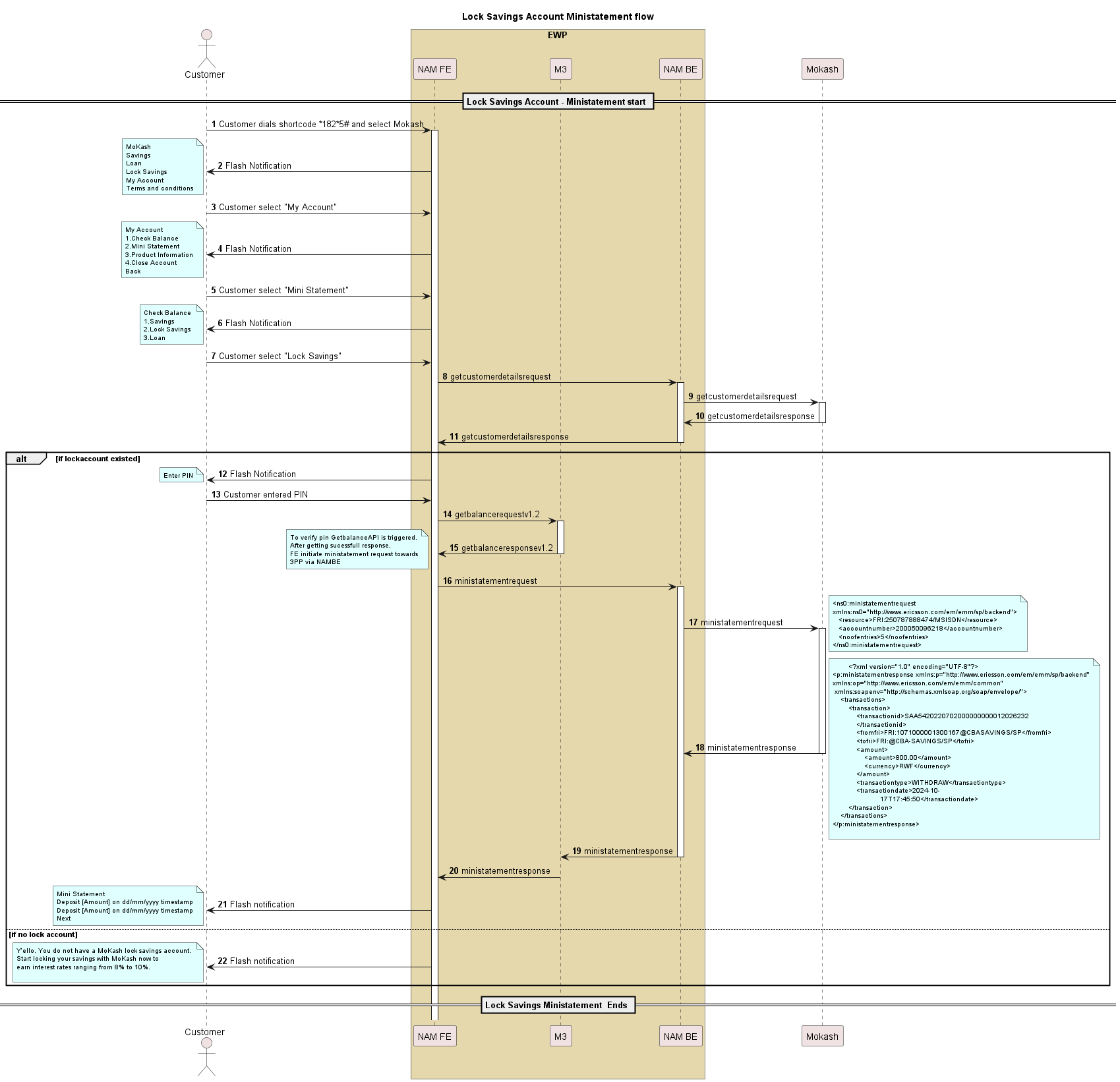
The UML diagram describes a sequence flow for checking the balance of a MoKash Lock Savings Account using a mobile service.

1. The customer initiates the interaction by dialing the shortcode 1825# on their phone and selecting MoKash from the Flash Notification options provided.
2. The customer is presented with another Flash Notification displaying different options under "My Account" (Check Balance, Mini Statement, Product Information, Close Account, Back).
3. The customer selects "Check Balance," which leads to another menu offering options for Savings, Lock Savings, and Loan.
4. The customer selects "Lock Savings" to check the lock savings balance.
5. The Front-End (FE) system sends a request to the Back-End (BE) to get customer details, which is forwarded to a third-party partner (3PP).
6. If a lock account exists, the customer is prompted to enter their PIN.
7. If the lock account exists, the customerdetails response contains locksaving accountnumber which is will be saved for next step.
8. After PIN entry, getbalance request is sent to M3 from NAMFE which verifies the PIN, Then FE trigger backed lockbalancerequest request towards NAMBE.
9. NAMBE triggers lockbalancerequest towards the third-party (3PP) systems, which returns the lock balance details. Refer API information section.
10. The NAMBE receives a lock balance response containing the account number, balance amount, currency, and due date and return the same in response.
11. If no lock account exists, the customer is notified that they do not have a MoKash lock savings account and is encouraged to start one to earn interest rates ranging from 8% to 10%.

### Lock savings account – Mini Statement

#### Sequence Flow for Mini Statement

Customer can view ministatement of lock saving account.



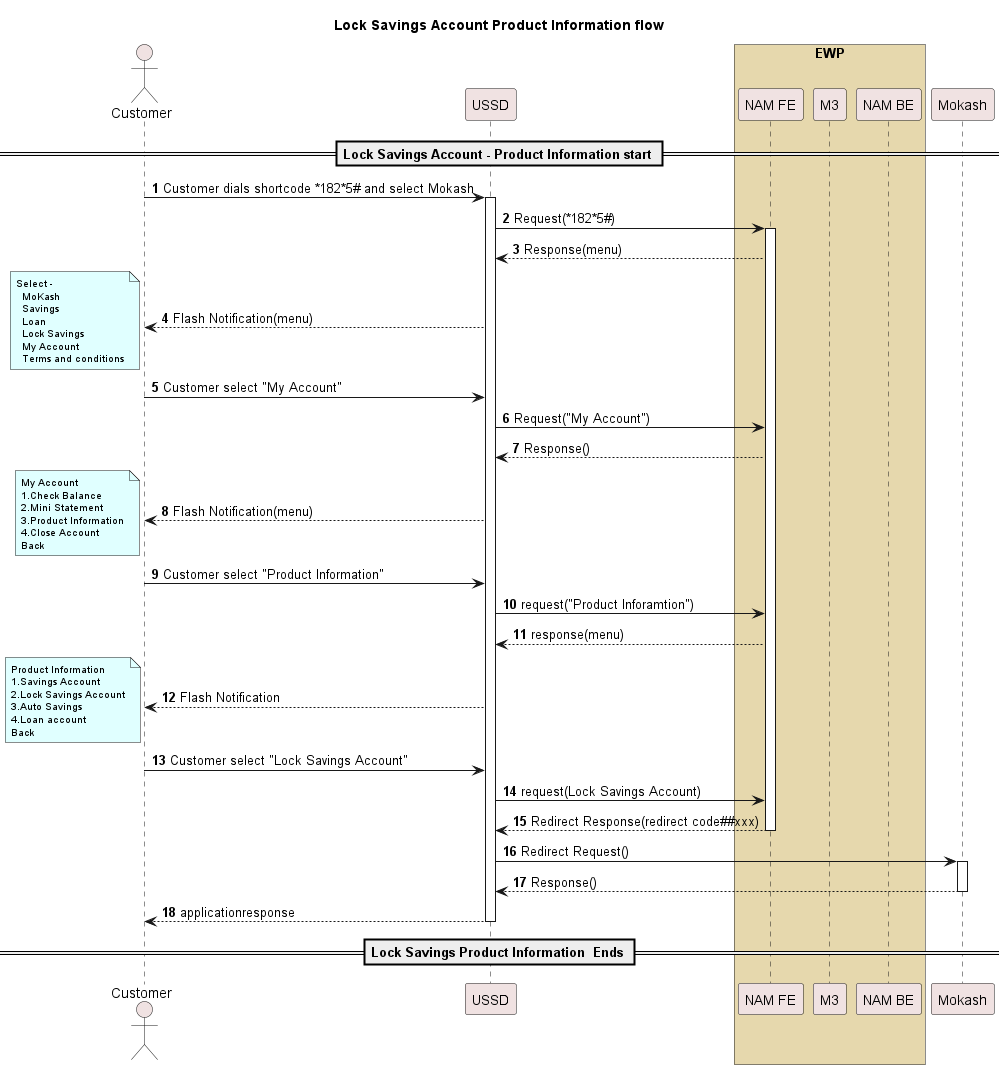
**Description**

The UML diagram outlines the flow for obtaining a ministatement for a Lock Savings Account

1. The customer initiates the process by dialing the shortcode 1825# on their mobile device and selects the MoKash option from the flash notification menu.
2. The customer selects "My Account" and then chooses the "Mini Statement" option in subsequent flash notifications. They further specify "Lock Savings" to get the ministatement for this account type.
3. If the customer has a lock savings account, they receive a flash notification with the ministatement displaying recent deposits.
4. If no lock account is found, the customer is notified with a message encouraging them to start saving with Mokash to earn interest.
5. If the Lock account exist below, are the steps.
6. A request (getcustomerdetailsrequest) is sent from the Front End (FE) to the Backend (BE) and then to a third-party service (3PP) to fetch customer details.
7. If a lock savings account exists, the lock saving account number is fetched, and the customer is prompted to enter their Mobile Money PIN.
8. Once the PIN is entered, a getbalancerequest is made from FE to M3 to verify the PIN.
9. IF Pin is verified by getting successful get balance response from M3. NAMFE triggers Ministatement request towards NAMBE.
10. NAMBE triggers ministatement towards 3PP and 3PP responded to NAMBE with transaction details. The response includes transaction details, such as transaction id, amount, currency, transaction type, and date. Refer API information section.
11. PLEASE NOTE: Ministatement details whatever be the transaction type that should be shown to the end user. (LOCKSAVINGS ,DEPOSITetc)

### Lock savings account – Product Information

#### Sequence flow to get product information.



It’s a USSD redirect to get the product information. The params are configurable in the NAM.

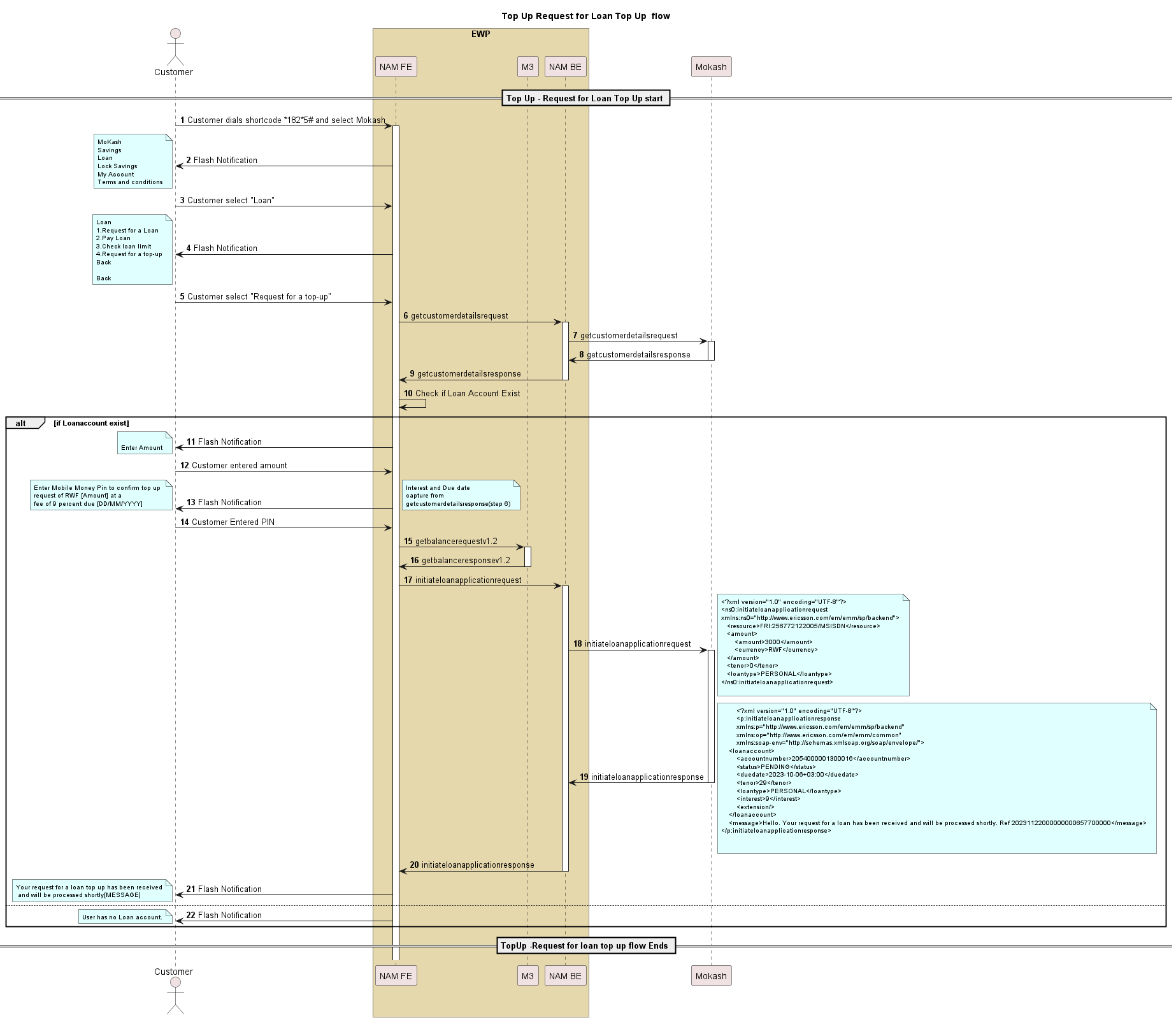
## Loan Top Up

Loan Top Up feature:

1. Top up on loan.
2. Customer can do full and partial payments.
3. Customer can check credit loan limit.
4. Customer can check mini statement.
5. Customer can get Product Info.

### Requesting for a Loan Top-up

#### Sequence Flow for Requesting for a Loan Top-up



Description

The UML diagram illustrates the top-up request process for a loan.

1. The customer dials a shortcode to start the process and navigates through a series of flash notifications to select the "Request for a top up" option.
2. The sequence involves multiple participants: the customer, front-end (FE), back-end (BE), and a third party (Mokash, labeled as 3PP).
3. Once the customer requests a top-up, the FE initiates a request to the BE, which further interacts with the third party to get customer details. If the loan account exist in the customerdetailresponse, the customer is prompted to enter the loan amount and confirm the transaction with their mobile money pin. (Fee and duedate of the loan account fetched from the above customerdetailresponse.)
4. getbalance request is triggered from NAMFE towards M3 to verify the PIN.
5. After Pin is verified, NAMFE triggers initiateloanapplicationrequest towards 3PP via NAMBE with all the details.

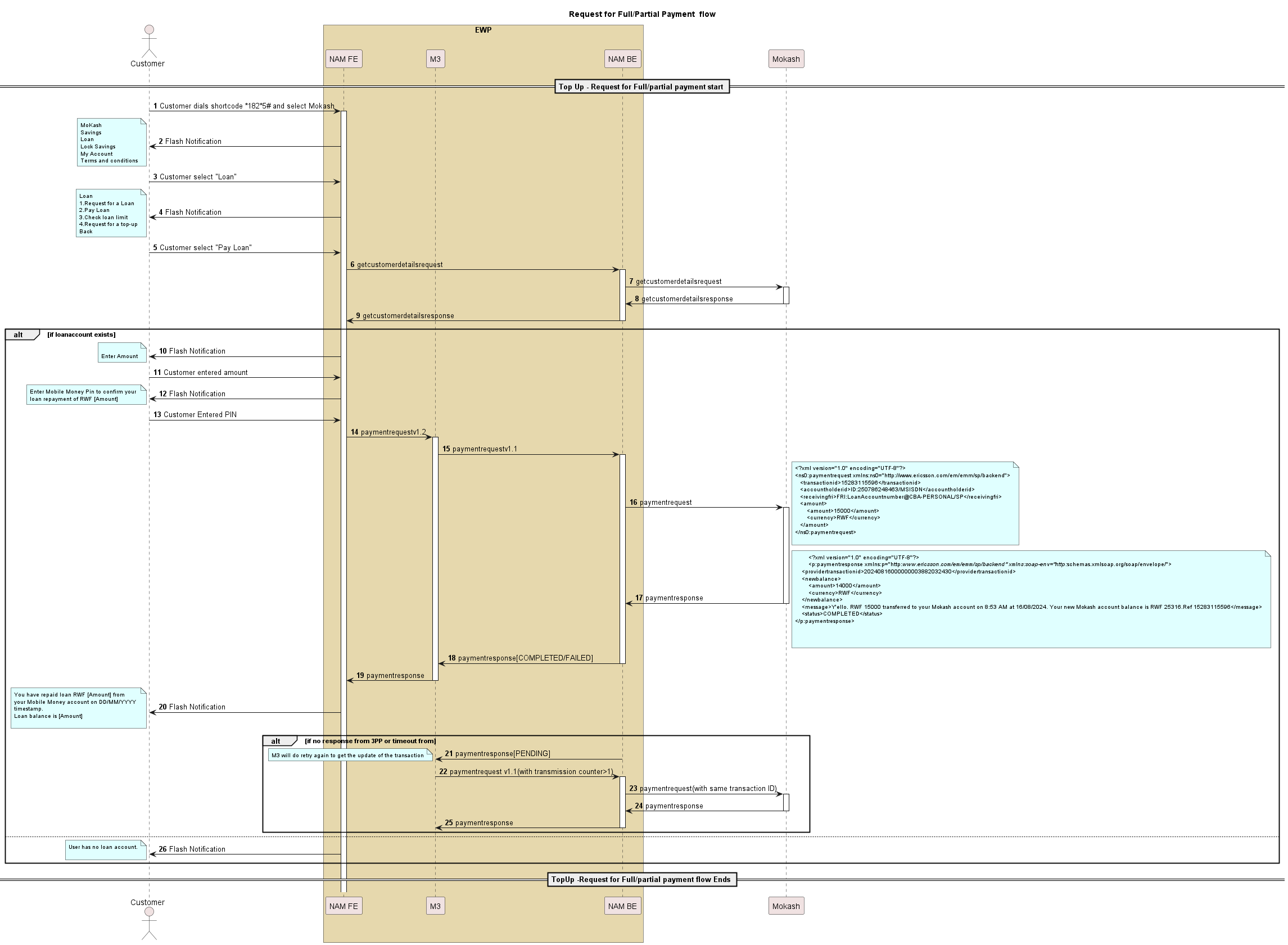
Refer API Information.

1. The response captures loan details, including account number, status, due date, tenor, loan type, and interest rate, which are used to inform the customer via flash notifications.

The process concludes when the top-up request is successfully submitted and processed, and the customer is notified that their request is being handled.

### Loan Top-up – Full/Partial Repayment

#### Sequence Flow for Full/Partial Payment



Description

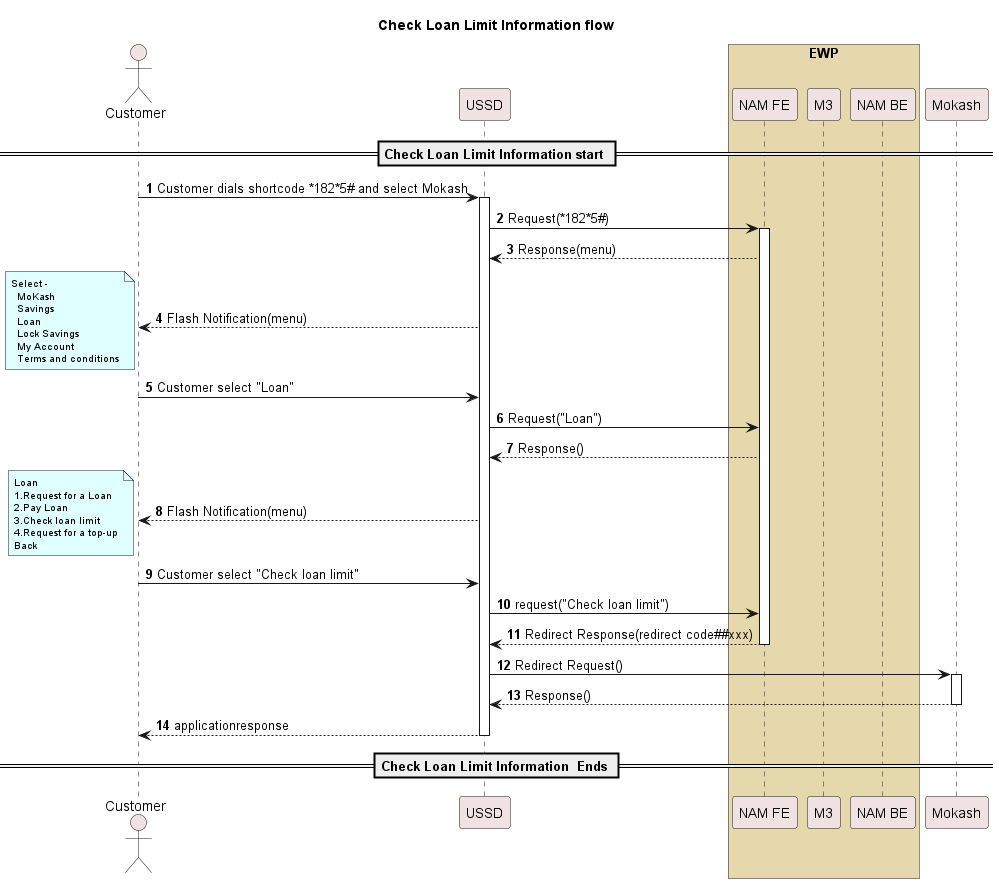
The UML sequence diagram describes the process flow for a customer who requests a full or partial loan repayment using the MoKash service via mobile shortcode 1825#.

1. The customer dials the shortcode and selects the "Loan" option from a list of services presented via a flash notification.
2. The customer chooses to "Pay Loan" from the flash notification options.
3. The FE (Front End) sends a request to the BE (Back End) to retrieve customer details, which in turn, sends a request to a third-party service provider (3PP).
4. If the customer's loan account exists, the FE prompts the customer to enter the repayment amount.
5. After the customer inputs the repayment amount, they are prompted to enter their Mobile Money PIN to confirm the transaction for the specified amount in Rwandan Francs (RWF).
6. The FE sends a payment request to M3, which involves the BE relaying the payment request (Already existing mokash payment request) to the third-party service provider using XML formatted data, specifying transaction details like transaction ID, account holder ID, recipient details, amount, and currency.
7. The payment response from the third-party service provider includes transaction status and updated balance details. If the status is "COMPLETED", the customer receives a notification confirming the repayment and providing the updated loan balance.

If there is no response or a timeout occurs, the transaction status remains "PENDING". The M3 system will retry the transaction to get an updated response.

### Loan Top-up – Check Loan Limit

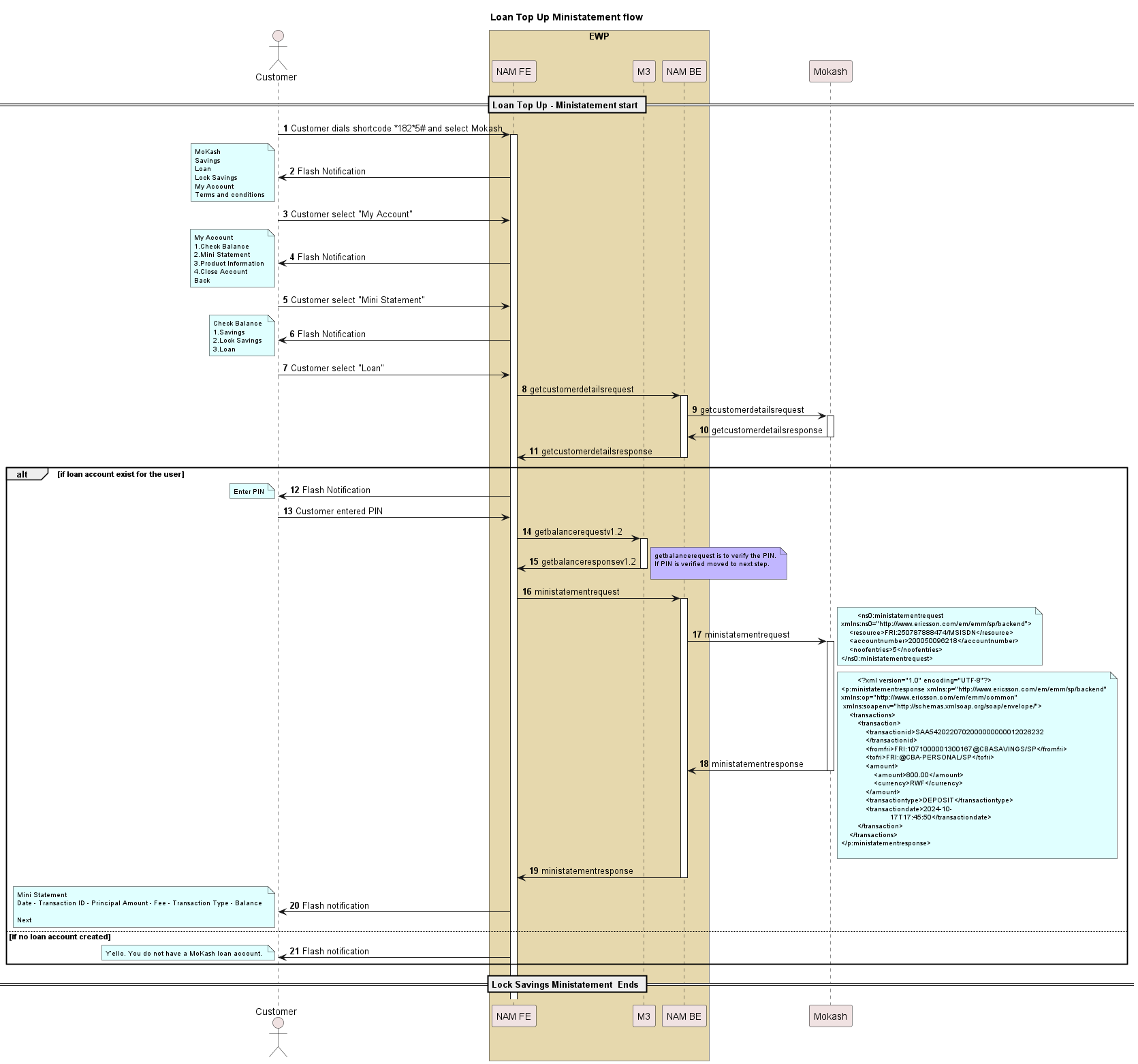
#### Sequence Flow for Check Loan Limit



For creditloanlimit it’s a USSD redirect with the configurable parameter.

### Loan Top up – Mini Statement

#### Sequence Flow for Mini Statement



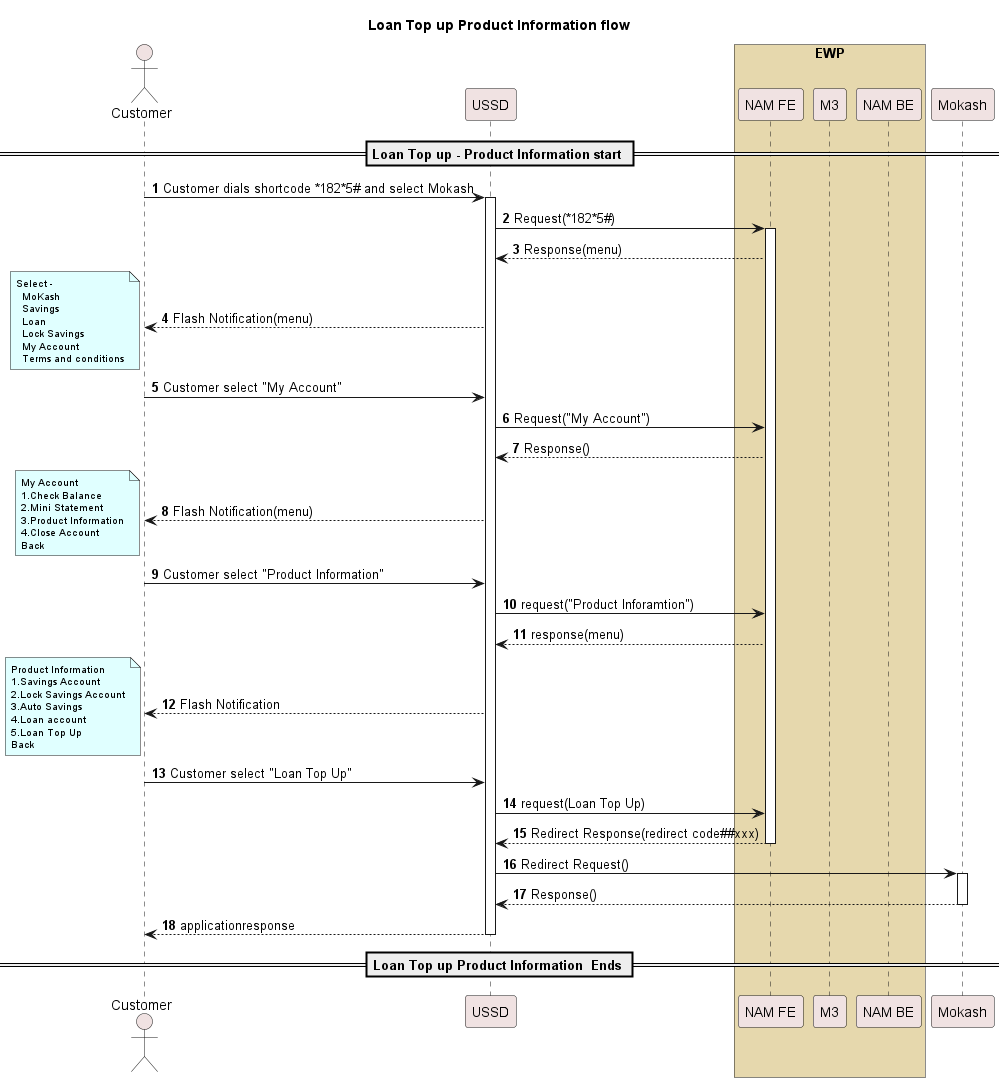
**Description**

The provided diagram describes the process flow for a customer obtaining a mini statement for a Mokash loan top-up.

1. The sequence initiates with the customer dialing a specific shortcode and selecting the Mokash service from the options presented. Once the "My Account" option is chosen, the customer can select "Mini Statement" and then "Loan" to proceed.
2. NAMFE triggers getcustomerdetails API towards 3PP via NAMBE.
3. If the customer has an existing loan account (loan account number fetched from getcustomerdetailsresponse), the system prompts the customer to enter their PIN.
4. NAMFE triggers getbalance request towards M3 to verify the PIN. If successful getbalance response received NAMFE triggers ministatement request towards 3PP via NAMBE.
5. NAMBE triggers 3PP ministatement API towards 3PP.A request in XML format for the mini statement is sent to the 3PP, which responds with transaction details, including transaction ID, amount, currency, type, and date. The response data is then relayed back through the system to the front end, which presents it to the customer as a flash notification with the mini statement details.
6. If the customer does not possess a Mokash loan account, the system immediately informs the customer through a USSD flash notification stating that they do not have such an account.

### Loan Top up – Product Information

#### Sequence flow of Product Information



For product information it’s a USSD redirect with the configurable parameter at NAM.

# Business Rule

Below are business rules for different use cases-

## Lock Saving Account

New OVA account CBA-LOCKSAVINGS will be created.

1. Eligible to all Mobile Money customers
2. Save as low as RWF 5,000 and lock for a defined period.
3. Access Savings at any time through the mobile money platform
4. Option to lock savings for defined term (1 – 12 months as well user input months)
5. View account balance and mini statement at any time at no cost

# Customer journey

Below is updated customer journey for new use cases –

## Lock Saving Feature



## Loan Top Up



# API Information

## GetCustomerDetails/response

**Request**

<ns0:getcustomerdetailsrequest

xmlns:ns0="http://www.ericsson.com/em/emm/sp/backend">

<resource>FRI:250783136694/MSISDN</resource>

</ns0:getcustomerdetailsrequest>

**Response**

<p:getcustomerdetailsresponse xmlns:p="http://www.ericsson.com/em/emm/sp/backend"

xmlns:op="http://www.ericsson.com/em/emm/common"

xmlns:soap-env="http://schemas.xmlsoap.org/soap/envelope/">

<customerid>1004248005</customerid>

<status>REGISTERED</status>

<savingsaccounts>

<savingsaccount>

<accountnumber>400001927351</accountnumber>

<status>ACTIVE</status>

<balance>

<amount>0</amount>

<currency>XOF</currency>

</balance>

<savingsaccounttype>SAVINGS</savingsaccounttype>

<interest>8</interest>

</savingsaccount>

<savingsaccount>

<accountnumber>40012994123</accountnumber>

<status>ACTIVE</status>

<balance>

<amount>91</amount>

<currency>XOF</currency>

</balance>

<duedate>2024-03-03+03:00</duedate>

<savingsaccounttype>LOCKSAVINGS</ savingsaccounttype>

<interest>7</interest>

</savingsaccount>

</savingsaccounts>

</p:getcustomerdetailsresponse>

## Locksavingactiverequest/response

**Request**

<ns0:locksavingactiverequest

xmlns:ns0="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/backend/client">

<transactionid>TX20240108125203001001</transactionid>

<accountholderid>ID:250780724118/MSISDN</accountholderid>

<receivingfri>FRI:250780724118@CBA-LOCKSAVINGS/SP</receivingfri>

<fundsource>MOMO</fundsource>

<amount>

<amount>3500</amount>

<targetamount>3500</targetamount>

<currency>RWF</currency>

</amount>

<duration>

<periodunit>M</periodunit>

<period>24</period>

</duration>

</ns0:locksavingactiverequest>

**Response**

<p:locksavingactiveresponse xmlns:p="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/backendclient" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">

<transactionid>TX20240108125203001001</transactionid>

<accountnumber>1020002000015116</accountnumber>

<balance>

<amount>3500</amount>

<currency>RWF</currency>

</balance>

<message>Y'ello. You have actived RWF 3500 from your MoMo wallet to your MoKash lock savings account,your new balance is RWF 3500</message>

<status>COMPLETED</status>

</p:locksavingactiveresponse>

## LockSavingsDepositrequest/response

**request**

<ns0:locksavingdepositrequest

xmlns:ns0="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/backend/client">

<transactionid>TX20240108125203001001</transactionid>

<accountholderid>ID:250780724118/MSISDN</accountholderid>

<receivingfri>FRI:200043946958@CBA-LOCKSAVINGS/SP</receivingfri>

<fundsource>MOKASH</fundsource>

<amount>

<amount>100</amount>

<currency>RWF</currency>

</amount>

</ns0:locksavingdepositrequest>

**response**

<p:locksavingdepositresponse xmlns:p="http://www.ericsson.com/em/emm/

serviceprovider/v1\_0/backend/client" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">

<transactionid>TX20240108125203001001</transactionid>

<accountnumber>1020002000015116</accountnumber>

<balance>

<amount>3100</amount>

<currency>RWF</currency>

</balance>

<message>Y'ello. You have transferred

RWF 1000 from your Mokash to your

MoKash lock savings account,your new

balance is RWF 3100</message>

<status>COMPLETED</status>

</p:locksavingdepositresponse>

## Locksavingsautosavingsrequest/response

**request**

<ns0:locksavingsautosavingsrequest xmlns:ns0="http://www.ericsson.com/em/emm/sp/backend">

<resource>FRI:250788531643/MSISDN</resource>

<savingsaccountno>200035771416</savingsaccountno>

<term>

<amount>

<amount>2000</amount>

<currency>XOF</currency>

</amount>

<frequency>DAILY</frequency>

<period>6M</period>

</term>

</ns0:Locksavingsautosavingsrequest>

**response**

<p:locksavingsautosavingsresponse

xmlns:p="http://www.ericsson.com/em/emm/sp/backend"

xmlns:iso="urn:iso:std:iso:20022:tech:xsd"

xmlns:xsi="http://www.w3.org/2001/XMLSchemainstance"

xmlns:op="http://www.ericsson.com/em/emm/common"

xmlns:soap-env="http://schemas.xmlsoap.org/soap/envelope/"

xsi:schemaLocation="http://www.ericsson.com/em/emm

/sp/backend savingandlendingbackend.xsd ">

<message>Y'ello. You have successfully set up a

daily transfer of XOF.2000 from your Mobile Money

Account to your

MoKash.

</message>

</p:locksavingsautosavingsresponse>

## Locksavingscancelautosavingsrequest/response

**request**

<ns0:locksavingscancelautosavingsrequest xmlns:ns0="http://www.ericsson.com/em/emm/sp/backend">

<resource>FRI:254788602058/MSISDN

</resource>

<savingsaccountno>1071000001320050</savingsaccountno>

</ns0:locksavingscancelautosavingsrequest>

**response**

<?xml version="1.0" encoding="UTF-8"?>

<p:locksavingscancelautosavingsresponse

xmlns:p="http://www.ericsson.com/em/emm/sp/backend" xmlns:iso="urn:iso:std:iso:20022:tech:xsd"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:op="http://www.ericsson.com/em/emm/common"

xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

xmlns:schemaLocation="http://www.ericsson.com/em/emm/sp/backendsavingandlendingbackend.xsd ">

<message>Y'ello. You have successfully cancelled your

DAILY transfer of RWF 30000 from your Mobile Money Account

to

your Mokash.

</message>

</p:locksavingscancelautosavingsresponse>

## Lockwithdrawrequest/response

**Request**

<ns0:lockwithdrawrequest

xmlns:ns0="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/backend/client">

<transactionid>TX20240108125203001001</transactionid>

<accountholderid>ID:250780724118/MSISDN</accountholderid>

<receivingfri>FRI:200043946958@CBALOCKSAVINGS/SP</receivingfri>

<amount>

<amount>3500</amount>

<currency>RWF</currency>

</amount>

</ns0:lockwithdrawrequest>

**Response**

<p:lockwithdrawresponse xmlns:p="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/backend/client"

xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">

<transactionid>TX20240108125203001001</transactionid>

<datetransfered>20/01/2025</datetransfered>

<message>Y'ello. You have transferred

RWF 1000 from your MoMo wallet to your

MoKash loack savings account,your new

balance is RWF 30612

</message>

<status>COMPLETED</status>

</p:lockwithdrawresponse>

## Lockbalancerequest/response

**Request**

<ns0:lockbalancerequest xmlns:ns0="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/backend/client">

<accountnumber>1020002000015116</accountnumber>

<accountholderid>254780724118</accountholderid>

</ns0:lockbalancerequest>

**Response**

<p:lockbalanceresponse xmlns:p="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/backend/client"

xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">

<accountnumber>1020002000015116

</accountnumber>

<balance>

<amount>1000</amount>

<currency>RWF</currency>

</balance>

<duedate>20/01/2025</duedate>

</p:lockbalanceresponse>

## Ministatement request/response

**Request**

<ns0:ministatementrequest

xmlns:ns0="http://www.ericsson.com/em/emm/sp/backend">

<resource>FRI:250787888474/MSISDN</resource>

<accountnumber>200050096218</accountnumber>

<noofentries>5</noofentries>

</ns0:ministatementrequest>

**Response**

<?xml version="1.0" encoding="UTF-8"?>

<p:ministatementresponse xmlns:p="http://www.ericsson.com/em/emm/sp/backend"

xmlns:op="http://www.ericsson.com/em/emm/common"

xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">

<transactions>

<transaction>

<transactionid>SAA5420220702000000000012026232

</transactionid>

<fromfri>FRI:1071000001300167@CBASAVINGS/SP</fromfri>

<tofri>FRI:@CBA-SAVINGS/SP</tofri>

<amount>

<amount>800.00</amount>

<currency>RWF</currency>

</amount>

<transactiontype>WITHDRAW</transactiontype>

<transactiondate>2024-10-

17T17:45:50</transactiondate>

</transaction>

</transactions>

</p:ministatementresponse>

## Initiate Loan Application Request/Respose

**Request**

<?xml version="1.0" encoding="UTF-8"?>

<ns0:initiateloanapplicationrequest xmlns:ns0="http://www.ericsson.com/em/emm/sp/backend">

    <resource>FRI:256772122005/MSISDN</resource>

    <amount>

        <amount>3000</amount>

        <currency>RWF</currency>

    </amount>

    <tenor>0</tenor>

    <loantype>PERSONAL</loantype>

</ns0:initiateloanapplicationrequest>

**Response**

<?xml version="1.0" encoding="UTF-8"?>

<p:initiateloanapplicationresponse xmlns:p="http://www.ericsson.com/em/emm/sp/backend" xmlns:op="http://www.ericsson.com/em/emm/common" xmlns:soap-env="http://schemas.xmlsoap.org/soap/envelope/">

    <loanaccount>

        <accountnumber>2054000001300016</accountnumber>

        <status>PENDING</status>

        <duedate>2023-10-06+03:00</duedate>

        <tenor>29</tenor>

        <loantype>PERSONAL</loantype>

        <interest>9</interest>

        <extension/>

    </loanaccount>

    <message>Hello. Your request for a loan has been received and will be processed shortly. Ref 20231122000000000657700000</message>

</p:initiateloanapplicationresponse>

## PaymentRequest/Response of Mokash

**Request**

<ns0:paymentrequest xmlns:ns0="http://www.ericsson.com/em/emm/sp/backend">

<transactionid>8897922413</transactionid>

<accountholderid>ID:2250505791906/MSISDN</accountholderid>

<receivingfri>FRI:400055566559@CBA-PERSONAL/SP</receivingfri>

<amount>

<amount>10000</amount>

<currency>XOF</currency>

</amount>

</ns0:paymentrequest>

**Response**

<?xml version="1.0" encoding="UTF-8"?>

<p:paymentresponse xmlns:p="http://www.ericsson.com/em/emm/sp/backend"

xmlns:soap-env="http://schemas.xmlsoap.org/soap/envelope/">

<providertransactionid>20240816000000003882032430</providertransactionid>

<newbalance>

<amount>25316</amount>

<currency>RWF</currency>

</newbalance>

<message>Y'ello. RWF 15000 transferred to your Mokash account on 8:53 AM at 16/08/2024. Your new Mokash account balance is RWF 25316.Ref 15283115596</message>

<status>COMPLETED</status>

</p:paymentresponse>

## Getbalance v1.2(ECW API)

**Sample request/response:**

**Request**

<?xml version="1.0" encoding="UTF-8"?>

<ns0:getbalancerequest xmlns:ns0="http://www.ericsson.com/em/emm/financial/v1\_2">

<fri>FRI:2349062059720/MSISDN</fri>

</ns0:getbalancerequest>

**Response**

<?xml version="1.0" encoding="UTF-8"?>

<ns8:getbalanceresponse xmlns:ns8="http://www.ericsson.com/em/emm/financial/v1\_2" xmlns:fic="http://www.ericsson.com/em/emm/financial/v1\_2/common" xmlns:ns10="http://www.ericsson.com/em/emm/financial/v1\_0" xmlns:ns4="http://www.ericsson.com/em/emm/financial/v1\_0/common" xmlns:ns5="http://www.ericsson.com/em/emm/v2\_1/common" xmlns:ns6="http://www.ericsson.com/em/emm/financial/v1\_1/common" xmlns:ns9="http://www.ericsson.com/em/emm/financial/v1\_2" xmlns:op="http://www.ericsson.com/em/emm/v1\_0/common" xmlns:xs="http://www.w3.org/2001/XMLSchema">

<balance>

<amount>48388.01</amount>

<currency>RWF</currency>

</balance>

</ns8:getbalanceresponse>

## Payment request/response (ECW API)

[payment v1.1](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#TOP)

Initiate a payment request for an account holder towards the Service Provider.

URI: /{baseContext}/payment

Namespace: <http://www.ericsson.com/em/emm/serviceprovider/v1_1/backend/client>

Schema file: payment.xsd

[Parameters](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT" \l "TOP)

**paymentrequest:**

|  |  |  |  |
| --- | --- | --- | --- |
| Table 94    paymentrequest | | | |
| **Name** | **Type** | **Optionality** | **Description** |
| transactionid | long | Mandatory | System internal transaction ID. Parameter can not be NULL. |
| accountholderid | string | Mandatory | The identity of the account holder requesting the payment. Parameter can not be NULL. Validated with [IsIdentity](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT" \l "IsIdentity). |
| receivingfri | string | Mandatory | The FRI of receiving account, to which payment is being made. Parameter can not be NULL. Validated with [IsFRI](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT" \l "IsFRI). |
| amount | [moneydetailstype v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#www.ericsson.com.em.emm.v1_0.common.moneydetailstype) | Mandatory | Amount to be paid. Parameter can not be NULL. |
| message | string | Optional | A descriptive note for service provider. Validated with [Length](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#Length) (max=256). Validated with [IsRestrictedString](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT" \l "IsRestrictedString). |
| transmissioncounter | string | Optional | A counter representing the number of times a particular request has been transmitted. The counter is set to 1 the first time the request is transmitted and is incremented by 1 for each retransmission of the request. Validated with [IsIntegerStringWithinRange](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT" \l "IsIntegerStringWithinRange) (min=0). |
| extension | [extensiontype v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#www.ericsson.com.em.emm.v1_0.common.extensiontype) | Optional | Used for sending any additional information to service provider. |

Values to be signed: transactionid, accountholderid, amount/amount, amount/currency

**paymentresponse:**

|  |  |  |  |
| --- | --- | --- | --- |
| Table 95    paymentresponse | | | |
| **Name** | **Type** | **Optionality** | **Description** |
| providertransactionid | string | Mandatory | Service Provider Transaction ID. Validated with [NotBlank](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT" \l "NotBlank). Validated with [Length](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#Length) (max=256). Validated with [IsExternalTransactionIdString](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT" \l "IsExternalTransactionIdString). |
| scheduledtransactionid | long | Optional | The scheduled transaction ID of the transfer. |
| newbalance | [moneydetailstype v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#www.ericsson.com.em.emm.v1_0.common.moneydetailstype) | Optional | Balance of the account at the service provider. |
| paymenttoken | string | Optional | Service provider payment token. Validated with [Length](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#Length) (max=64). Validated with [IsExternalReferenceString](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT" \l "IsExternalReferenceString). |
| message | string | Optional | A message to the account holder that has made the payment. Validated with [Length](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#Length) (max=256). Validated with [IsRestrictedString](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT" \l "IsRestrictedString). |
| status | [paymentstatus v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#www.ericsson.com.em.emm.serviceprovider.v1_0.common.paymentstatus) | Mandatory | Payment Status. Parameter can not be NULL. |
| extension | [extensiontype v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#www.ericsson.com.em.emm.v1_0.common.extensiontype) | Optional | Used for sending any additional information to front end access module. Only applicable for paymentcompletedrequest and when status is COMPLETED or FAILED. |

Values to be signed: providertransactionid, status

[Expected Error Codes](https://cpistore.internal.ericsson.com/elex?id=32284&SR=BODYTEXT&ORPA=paymentrequest&fn=20_15519-4_FAM901419Uen.X.html&PA=paymentrequest&ST=FULLTEXT#TOP)

|  |  |
| --- | --- |
| Table 96    Error Codes | |
| **Error Code** | **Description** |
| INTERNAL\_ERROR | Sets the withdrawal status to PENDING and retries again according to timers settings. |
| CONNECTION\_ERROR | Sets the withdrawal status to PENDING and retries again according to timers settings. |
| COMMUNICATION\_ERROR | Sets the withdrawal status to PENDING and retries again according to timers settings. |

**Request**

<?xml version="1.0" encoding="UTF-8"?>

<ns4:paymentrequest xmlns:ns4="http://www.ericsson.com/em/emm/serviceprovider/v1\_1/backend/client" xmlns:op="http://www.ericsson.com/em/emm/v1\_0/common" xmlns:xs="http://www.w3.org/2001/XMLSchema">

<transactionid>730007</transactionid>

<accountholderid>ID:26878721737/MSISDN</accountholderid>

<receivingfri>FRI:26878721737@cba-locksavings/SP</receivingfri>

<amount>

<amount>100.00</amount>

<currency>SZL</currency>

</amount>

<transmissioncounter>39868</transmissioncounter>

<extension/>

</ns4:paymentrequest>

**Response**

<?xml version="1.0" encoding="UTF-8"?>

<ns0:paymentresponse xmlns:ns0="http://www.ericsson.com/em/emm/serviceprovider/v1\_1/backend/client">

<providertransactionid>RRN334009924729\_RIIC06000211\_AC0911

</providertransactionid>

<message>Card Issuer Timed Out</message>

<status>PENDING</status>

</ns0:paymentresponse>

NOTE: These are just samples of request/response. Values subject to change.

## Addpreapproval (ECW API)

[addpreapproval v1.1](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html#TOP)

Add a pre-approval for the specified account, to the specified account. In a multi-currency-system, either accounts can be specified either through a particular FRI value (e.g. "FRI:123456/MM"), or through a combination of FRI value and currency (e.g. "FRI:467543219876/MSISDN" and "SEK"). The current release of the system does not support cross-currency transfers, i.e. "fromfricurrency" and "tofricurrency" must be the same, or resolve to the same currency code. If this operation is used in a multi-currency system it is required to use Resolved FRI or AMBIGUOUS\_CURRENCY will be returned. If the requested validity duration exceeds the configured maximum validity duration MAX\_PREAPPROVAL\_VALIDITY\_DURATION\_EXCEEDED will be returned.

Classification: **Undefined**.

URI: /{baseContext}/addpreapproval

Namespace: <http://www.ericsson.com/em/emm/serviceprovider/v1_1/frontend>

Schema file: addpreapproval.xsd

Permissions: ROLE\_UPDATE\_OWN\_ACCOUNTHOLDER

[Parameters](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html#TOP)

**addpreapprovalrequest:**

|  |  |  |  |
| --- | --- | --- | --- |
| Table 1115    addpreapprovalrequest | | | |
| **Name** | **Type** | **Optionality** | **Description** |
| fromfri | string | Mandatory | The Financial Resource Identifier of the sending account. Parameter can not be NULL. Validated with [IsFRI](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html" \l "IsFRI). |
| fromfricurrency | [currencycode v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html#www.ericsson.com.em.emm.v1_0.common.currencycode) | Optional | The currency code of the sending account. |
| tofri | string | Mandatory | The Financial Resource Identifier of the receiving account. Parameter can not be NULL. Validated with [IsFRI](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html" \l "IsFRI). |
| tofricurrency | [currencycode v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html#www.ericsson.com.em.emm.v1_0.common.currencycode) | Optional | The currency code of the receiving account. |
| message | string | Mandatory | Message. Parameter can not be NULL. Validated with [Length](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html#Length) (max=256). Validated with [IsRestrictedString](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html" \l "IsRestrictedString). |
| validityduration | long | Optional | The time duration in seconds that the pre-approval is valid once it is accepted. Validated with [IsPositiveLong](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html" \l "IsPositiveLong). |

**addpreapprovalresponse:**

|  |  |  |  |
| --- | --- | --- | --- |
| Table 1116    addpreapprovalresponse | | | |
| **Name** | **Type** | **Optionality** | **Description** |
| preapprovalid | long | Mandatory | ID of the preapproval. Parameter can not be NULL. |
| status | [preapprovalstatus v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html#www.ericsson.com.em.emm.v1_0.common.preapprovalstatus) | Mandatory | Current status of the preapproval. Parameter can not be NULL. |
| expirytime | [datetime v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html#www.ericsson.com.em.emm.v1_0.common.datetime) | Optional | The expiry time of the pre-approval. Validated with [IsDateTimeValue](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=addpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html" \l "IsDateTimeValue). |
| preexisting | boolean | Mandatory | Indicates if a pre-approval for the involved accounts already exists. Parameter can not be NULL. |

**Request**

<?xml version="1.0" encoding="UTF-8"?>

<ns0:addpreapprovalrequest xmlns:ns0="http://www.ericsson.com/em/emm/serviceprovider/v1\_1/frontend">

<fromfri>FRI:250788312326/MSISDN</fromfri>

<tofri>FRI:CBA-LOCKSAVINGS/USER</tofri>

<message>Preapproval for Saving and Lending Services</message>

</ns0:addpreapprovalrequest>

**Response**

<?xml version="1.0" encoding="UTF-8"?>

<ns4:addpreapprovalresponse xmlns:ns4="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/frontend" xmlns:op="http://www.ericsson.com/em/emm/v1\_0/common" xmlns:xs="http://www.w3.org/2001/XMLSchema">

<preapprovalid>710029</preapprovalid>

<status>APPROVED</status>

< preexisting >false</ preexisting >

</ns4:addpreapprovalresponse>

NOTE: These are just samples of request/response. Values subject to change.

## CancelPreapproval API (ECW API)

**[cancelpreapproval v1.0](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=cancelpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html" \l "TOP)**

Cancel one or all pre-approvals for the logged in account holder. If this operation is used in a multi-currency system it is required to use Resolved FRI or AMBIGUOUS\_CURRENCY will be returned.

URI: /{baseContext}/cancelpreapproval

Namespace: <http://www.ericsson.com/em/emm/serviceprovider/v1_0/frontend>

**cancelpreapprovalrequest:**

|  |  |  |  |
| --- | --- | --- | --- |
| Table 1117    cancelpreapprovalrequest | | | |
| **Name** | **Type** | **Optionality** | **Description** |
| identity | string | Optional | The account holder's internal identity. If set, all pre-approvals for this account holder will be cancelled. Only a logged in user with permission ROLE\_CANCEL\_ANY\_PREAPPROVAL can cancel all pre-approvals for a specific account holder. Validated with [IsIdentity](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=cancelpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html" \l "IsIdentity). |
| fri | string | Optional | The FRI to cancel all pre-approvals from. If no FRI, and no pre-approval identity is supplied, then all pre-approvals for the logged in user are cancelled. Validated with [IsFRI](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=cancelpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html" \l "IsFRI). |
| preapprovalid | string | Optional | Omit to cancel more than one pre-approval. Validated with [IsLongString](https://cpistore.internal.ericsson.com/elex?id=32284&ORPA=cancelpre&SR=TOPIC&FN=13_15519-4_FAM901419Uen.AK.html" \l "IsLongString) (allowNegativeValue=false). |

**cancelpreapprovalresponse:**

An empty response indicates success.

**Request**

<?xml version="1.0" encoding="UTF-8"?>

<ns0:cancelpreapprovalrequest xmlns:ns0="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/frontend">

<identity>ID:2349132059983/MSISDN</identity>

<fri>FRI:accountnumber@CBA-LOCKSAVINGS/SP</fri>

</ns0:cancelpreapprovalrequest>

**Response**

<?xml version="1.0" encoding="UTF-8" standalone="yes"?><ns2:cancelpreapprovalresponse

xmlns:ns2="http://www.ericsson.com/em/emm/serviceprovider/v1\_0/frontend"/>

NOTE: These are just samples of request/response. Values subject to change.

## Bank Debit (SNL 1.6)

**Request**

?xml version="1.0" encoding="UTF-8"?>

<fron:bankdebitrequest xmlns:fron="http://www.ericsson.com/em/emm/sp/frontend">

<fromfri>FRI:250788254148/MSISDN</fromfri>

<tofri>FRI:CBA-LOCKSAVINGS/USER</tofri>

<amount>

<amount>200</amount>

<currency>RWF</currency>

</amount>

<externaltransactionid>M20240112152858151</externaltransactionid>

<frommessage>Auto Savings Request</frommessage>

<tomessage>Auto Savings Request</tomessage>

<referenceid>M20240112152858151</referenceid>

</fron:bankdebitrequest>

**Response**

<?xml version="1.0" encoding="UTF-8"?>

<ns0:bankdebitresponse xmlns:ns0="http://www.ericsson.com/em/emm/sp/frontend">

<transactionid>993267</transactionid>

<status>SUCCESSFUL</status>

<amount>

<amount>200</amount>

<currency>RWF</currency>

</amount>

</ns0:bankdebitresponse>

NOTE: These are just samples of request/response. Values subject to change.

# Security Description – CLM:

## Security Impact

This demand involves new featured of Mokash like lock savings and top Up. Mokash is already integrated with ECW for loan and savings feature. The 3PP is expected to develop their own APIs and adhere to the baseline security guidelines and requirements, details of which have been outlined in the section below.

## Security Risk Management

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S/N | Risk Description | Rating | Vulnerability | Affected Component | Mitigation |
| 1 |  |  |  |  |  |
|  |  |  |  |  |  |

## Security Requirement

The security guidelines for this demand aligns with E// security standard for all services integrating with ECW. All existing security requirements applies.

| **Security Area** | **Domain** | **Description** | **Current Configuration/Evidence** |
| --- | --- | --- | --- |
| Client Authentication | Access Control | Either of following should be supported at minimum for client authentication. - normal authentication (API session based). Authentication of clients is performed by a username/password-based API login service - via the basic authentication scheme as defined in RFC 1945 | Username and password with basic auth |
| Mutual Authentication support | Access Control | TLS v1.2 based Mutual authentication must be supported i.e. Both Client and server must authenticate each other by presenting trusted certificates before allowing any network connection | Supported. |
| Integrity Protection for Data in transit | Data protection | Software shall support message integrity through digital signature verification and creation. The signature algorithm such as RSA encryption over a SHA-256 digest of the signed data must be supported. | Existing EWP SoC application guidelines. |
| Strong encryption methods | Data Protection | If encryption is used for securing the data (rest, or in motion), the key lengths and algorithms shall be “strong” by current definitions. Any weak or broken encryption methods should not be used such as DES, MD5, etc | Existing EWP SoC application guidelines |
| Confidentiality & Integrity of data in motion | Network Data Protection | All communication interfaces internal or external must support secure protocols such as TLS (TLS 1.2 or above) to protect confidentiality and integrity of the messages. | Supported |
| Certificates issued by a CA | Network Data Protection | System must support use of X509 certificates for TLS based connectivity issued by an external certificate authority (CA preference must be agreed with operator). External CA must be trusted by system. | **CA must be either M3 (ECW) or MTN.** |
| Static IP addresses | Network Data Protection | The application should support use of static IP's for communication. If there is a strong need for dynamic IP's for communication, domain name resolution must be done only against an authorized domain names/system with support for DNSSEC. | Supported |
| List of API. |  |  | Refer section 8 |
| List of profiles and permissions |  |  | No new permissions required |

# Notifications

## SMS Notifications

Existing notifications shall apply.

## USSD Notifications

Refer USSD Journey

## XML Notifications

Existing internal notifications shall apply.

# Test Cases

## Test Cases

|  |  |  |  |
| --- | --- | --- | --- |
| S/N | Description | Expectation | Expected Result |
| TC01 | Open Lock Account by selecting “From MTN MoMo Wallet” option. | Saving Lock account should be opened in Mokash having balance entered by User.  Money will be debited from MTN wallet |  |
| TC02 | Open Lock Account by selecting “From Mokash” option. | Saving Lock account should be opened in Mokash having balance entered by User.  Money will be debited from Mokash Savings Account. |  |
| TC03 | User Entered Wrong PIN | Session will end showing default invalid PIN message. |  |
| TC04 | Deposit to Lock Savings Account.  Source “MTN Momo Wallet” | User entered amount will be debited from MTN wallet and credited to Lock Savings account. |  |
| TC05 | Deposit to Lock Savings Account.  Source “Mokash” | User entered amount will be debited from Mokash Savings Account and credited to Lock Savings account. |  |
| TC06 | Withdrawal from Lock Savings Account | User Entered Amount will be debited from User`s lock saving account to Mokash Savings Account. |  |
| TC07 | Activate auto locksavings. | User can create preapproval at ECW and initiate locksavingautosavingrequest |  |
| TC08 | Mokash auto deduction for lock savings if preapproval exists. | If preapproval exist for the FRI :Accountnumber@CBA-LockSavings then initiate debitrequest. |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# References

URS: Mokash Lock savings & Loan Top Up URS.pdf

# Glossary

3PP 3rd Party Product

ECW Ericsson Converged Wallet (Also the mobile money system)

EWP Ericsson Wallet Platform

MMRL Mobile Money Rwanda Limited

PO Partner Orchestrator (used as a gateway into Mobile Money)

RSSB Rwanda Social Security Board

XML Extensible Mark Up Language

USSD Unstructured Supplementary Data

SMS Short Messaging Protocol

SSL Secure Socket Layer

M3 Mobile Money Platform

# Approvals

This document has been mutually reviewed and approved to constitute the full scope of the solution to be delivered by the MTN Rwanda ECW Platform. This document is signed in two (2) copies of which the parties have taken one (1) each.

**MTN Rwanda**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Signature** | **Date** |
| Godwin Mugabe | Manager, Planning & Architecture |  |  |

**Ericsson**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Signature** | **Date** |
| Archana Yadav | Solutions Architect |  |  |
|  |  |  |  |