Business Requirements Document

Project Name: Notification Service

Document Version: v1.4

Date: 05.12.2024

Document History

Version	Date	Author	Description
1.1	27.11.2024	Vishnu Prasanth	Initial draft for stakeholder feedback
1.2	29.11.2024	Vishnu Prasanth	Explained Routing Logic in detail. Added new features.
1.3	02.12.2024	Vishnu Prasanth	Incorporated feedback from Santosh Jha
1.4	05.12.2024	Vishnu Prasanth	Added new section- 'Additional Features'

Contents

1. Purpose/Objective:

The purpose of this notification service is to provide a centralized, robust, scalable, secure and domain-agnostic microservice that facilitates seamless communication (across email, SMS, WhatsApp, Push, In-App, Webhook channels with an option to add MS Teams, Slack and Discord in the future) with all stakeholders (both Internal and External) including but not limited to Investors, distributors, AMCs, Channel Partners, Banks, regulators, etc., It will ensure efficient, secure and cost-effective delivery of notifications across multiple communication channels. This will serve as a critical component of the platform, ensuring timely and accurate dissemination of information to various stakeholders identified above.

This service is designed to meet the unique needs of CAMS where regulatory compliance, security and cost-efficiency are of paramount importance. By automating notification workflows, enabling dynamic recipient management, and integrating advanced routing and analytics, this microservice will improve operational efficiency, reduce manual effort, and enhance stakeholder engagement.

The service will support diverse use cases, ranging from transactional alerts and compliance updates (Risk, Fraud, etc.,) to marketing communications and operational notifications, all of these while maintaining a high degree of personalization and reliability. Additionally, it will provide administrators with a user-friendly interface for managing templates, channels, and configurations, ensuring ease of use and adaptability to evolving business needs.

1. Automation of Notifications:
    a. Automate the generation and delivery of notification based on predefined triggers and workflows (emanating from application) thereby reducing manual intervention and operational delays
    b. Enable use of APIs for real-time notification requests and dynamic content injection.
2. Domain-Agnostic Communication:
    a. Provide a versatile service that can be leveraged across various business domains and processes, from investor communications to regulatory reporting across MF/AIF & KRA for internal as well as external stakeholders.
    b. Ensure flexibility to cater to diverse use cases, such as transaction alerts, compliance notices, bulk updates, and event-driven notifications.
3. Multi-Channel Delivery with provider failover mechanism:

    a. Support multiple communication channels, including WhatsApp, SMS, email, and push notifications, with dynamic channel prioritization based on recipient preferences, delivery reliability, and cost-effectiveness.
    b. Implement channel redundancy and failover mechanisms not only between channels but also between providers to ensure message delivery even in case of channel/provider failures.
4. Personalization and Template Management:
    a. Facilitate the creation and management of notification templates with placeholders for dynamic content, ensuring tailored communication for each recipient, variations as per AMC/Channel requirements, etc.,
    b. Introduce an approval workflow for templates to ensure compliance with financial and regulatory standards.
    c. The approval process must be robust so that notifications templates comply with SEBI/AMFI guidelines. This process should involve compliance/Risk officers to review and approve the content.
5. Real-Time Monitoring and Analytics:
    a. Offer detailed reporting and analytics on notification delivery, engagement metrics, and costs.
    b. Enable administrators to monitor system health, delivery success rates, and potential issues in real time.
    c. Maintain detailed logs of all communications, including content, recipients, and delivery status to facilitate audits and demonstrate compliance with regulatory requirements.
6. Cost Optimization:
    a. Enable intelligent routing of notifications based on channel cost implications and business priorities.
    b. Provide detailed analytics to monitor and optimize notification costs across channels.
7. Security and Compliance:
    a. Ensure end-to-end encryption of notification content and recipient data, adhering to industry standards like GDPR, SEBI, and RBI guidelines.
    b. Implement robust access control and auditing mechanisms to maintain the confidentiality and integrity of sensitive communications.
    c. Ensure that the service complies with data protection regulations as stated in SEBI, RBI and DPDPA, safeguarding personal information and providing mechanisms for recipients to manage their communication
    d. Ensure that Keys are managed in a way such that content delivery provider is not able to store the communication date post decrypting the message before delivery.
8. Scalability and Availability:

a. Design the service to handle high notification volumes with low latency, ensuring scalability to meet the demands of a growing platform.
b. Guarantee high availability and uptime to always support critical communication needs.
c. Enable a mechanism which guarantees 100% delivery of all notifications as mandated by SEBI, if system is not available, the notifications should land in a queue, that picks it up from the point of failure.

9. Ease of Integration and Testing:
    a. Provide APIs, SDKs, and tools for seamless integration with existing systems and workflows.
    b. Include a sandbox environment for testing notifications, templates, and channel configurations before deployment.
10. Enhancing Stakeholder Engagement:
    a. Deliver timely, relevant, and reliable notifications to improve transparency and trust with stakeholders, including investors, regulators, and partners.
    b. Support feedback mechanisms to gather insights and improve communication strategies.

Key Features:

Feature	Description
Recipient management	Define recipient types (e.g., internal teams, investors, distributors, AMCs, regulators). - Allow dynamic grouping and segmentation of recipients based on predefined rules (e.g., geographic location, transaction history). - Consent Management: Ensure recipients' explicit opt-in for communications, in line with DPDP Act. - Implement mechanisms to manage recipient preferences for communication channels.
Channel Configuration	Support multiple channels: WhatsApp, SMS, Email, Push Notifications, and more. - Prioritize channels based on delivery cost, recipient preferences, and regulatory constraints. - Enable channel failover mechanisms to ensure critical notifications are delivered even if a preferred channel fails.
Template Management	Build a template library with pre-approved templates for various use cases, including compliance-driven communications. - Implement a multi-level approval workflow for template creation and modification, involving compliance and legal teams to ensure adherence to SEBI regulations and AMFI guidelines. - Support dynamic placeholders for personalization (e.g., name, transaction details) while ensuring compliance with DPDP Act (no over-collection of personal data).

Routing Logic	Enable intelligent routing to select the most cost-effective and reliable channel based on the recipient's preferences and notification type (e.g., critical vs. non-critical). - Implement fallback routing to switch channels in case of delivery failure on the primary channel.
Security Features	Ensure end-to-end encryption of all notifications and sensitive recipient data, in compliance with SEBI regulations and DPDP Act. - Implement role-based access control (RBAC) and audit trails to log all user actions related to notification creation, approval, and delivery. - Support anonymization and pseudonymization of recipient data for analytics and debugging.
Compliance Mechanisms	Embed compliance rules within the notification system to flag or block unauthorized communication (e.g., ensuring content is SEBI-compliant). - Maintain an audit trail of all notifications, including content, recipient details, timestamps, and approvals, for easy reference during audits. - Provide regulatory templates tailored to SEBI and AMFI standards to simplify compliance.
File Management	Support bulk uploads of recipient data in formats such as CSV, with validation for errors (e.g., invalid email IDs or phone numbers). - Include file versioning and access controls to prevent unauthorized modifications.
Testing and Debugging Tools	Offer a sandbox environment to test notifications, templates, and channel configurations without sending live messages. - Provide detailed logs for debugging failed notifications, including reasons for failure (e.g., invalid recipient details, channel unavailability).
Analytics and Reporting	Provide real-time delivery reports (e.g., success rates, failure reasons) segmented by channel, recipient type, and priority. - Enable engagement analytics, such as open rates, click-through rates, and read receipts, while anonymizing data in compliance with the DPDP Act. - Generate cost analysis reports to optimize channel usage and identify high-cost delivery patterns.
Scalability and Reporting	Design for horizontal scalability to handle up to 100 million notifications daily with sub-second latency for critical messages. - Ensure high availability (99.999% uptime) to support business-critical communication needs.
Integration Features	Provide APIs and SDKs for seamless integration with existing platforms (e.g., CRMS, AMC platforms, etc.,). - Ensure integration mechanisms comply with data minimization principles under the DPDP Act.
Personalization	Enable targeted notifications by dynamically personalizing messages based on recipient attributes, ensuring compliance with AMFI and SEBI content guidelines. - Implement audience segmentation for custom campaigns or regulatory updates.
Feedback and Audit Mechanisms	Provide a feedback mechanism to capture recipient responses (e.g., acknowledgment, opt-out requests) in compliance with the DPDP Act's provisions for user rights. - Maintain a comprehensive audit trail for all notifications, approvals, and recipient interactions, ensuring readiness for regulatory audits.

2. Scope

The notification service will offer a comprehensive range of functionalities, including:

- **Creation and Management of Notification Messages**:
    - Supports multiple communication channels such as email, SMS, Webhook, WhatsApp, voice, Push, In-App, and future use cases.
- **Event Definition and Mapping**:
    - Allows the definition of events and the mapping of these events to notification messages.
- **Sending Notifications**:
    - Sends notifications based on user-defined events.
- **Routing Hierarchy and Service Provider Setup**:
    - Configures a routing hierarchy and setup for multiple service providers for each communication channel at the client/AMC level.
- **External Client Service Layer**:
    - Provides this notification layer as a service to external clients within the Mutual Fund (MF), Alternative Investment Fund (AIF), and Fintech ecosystem. It includes an admin interface and configuration at the AMC/Organisation level.
- **Dashboard for Performance Analysis**:
    - Offers a dashboard to analyse the performance of notifications.
- **Logging and Monitoring**:
    - Logs and monitors sent notifications.
- **Billing Module**:
    - Includes a billing module to create invoices and reports based on the pricing setup at the client/AMC level.

3. Stakeholders

- Product Manager: Oversee the project and align it with business goals.
- Development Team: Build and maintain the notification service.

- Design Team: Create user interfaces and APIs for Message management and notification requests. Product Manager will share design instructions before design phase initiation.
- QA Team: Test the notification service for reliability and performance.
- Operations: Manage AMC (or org/sub-org) specific templates
- Compliance: Approve/Reject templates

4. Pre-requisites

Integration with Multiple Vendors:

- The application must be fully integrated with various vendors across different communication channels to ensure seamless functionality. The SLA with vendors for delivery success rates, latency, throughput and reliability must be defined. System must also support vendor failover mechanisms (Airtel to Jio or Vodafone, etc.,) also these vendors should provide real time APIs for message delivery status which should integrate with the notification service.

Recipient Information Handling:

- The details of recipients and other necessary information for transactional notifications will be provided by an external or third-party system, following an agreed-upon format. The application should validate data (phone numbers, email addresses, etc.,) upon ingestion to prevent errors during delivery. Also, recipient information from 3rd party systems should be standardized to a common format.
- All the recipient data that is processed by Notification System should adhere to DPDP, SEBI and AMFI requirements
- Users should be able to perform consent management within the app. This is covered under DND.

Pre-approved Domain-Specific Messages:

- The application must have a repository of pre-approved messages, particularly for SMS and WhatsApp channels. This should have version control to track changes and roll-back if there is a requirement. Pre-Approved templates should allow placeholders for dynamic content (name, transaction details, applicable date, etc.,)

Template Whitelisting:

- The notification system should support the whitelisting of templates for SMS and WhatsApp.

- Vendors will expose an API for template whitelisting, which will be integrated within the application. System should fetch and display real-time statis of templates (pending, approved, etc.,) from the Vendor's API>
- This integration will allow clients (AMCs) to navigate the template approval process smoothly and efficiently. If a template fails there should be notification to administrators an users and logging of the event should happen.
- Templates should be categorized as per the type of notification process as described in section 5.2 (e.g., transactional, promotional, regulatory, etc.,)

5. Notification

5.1. Overview

The Notification Process involves the Notification Service application sending messages to the designated recipients. This can be done through either an automated routing logic or manually defined channels and vendors.

5.2. Types of Notification processes

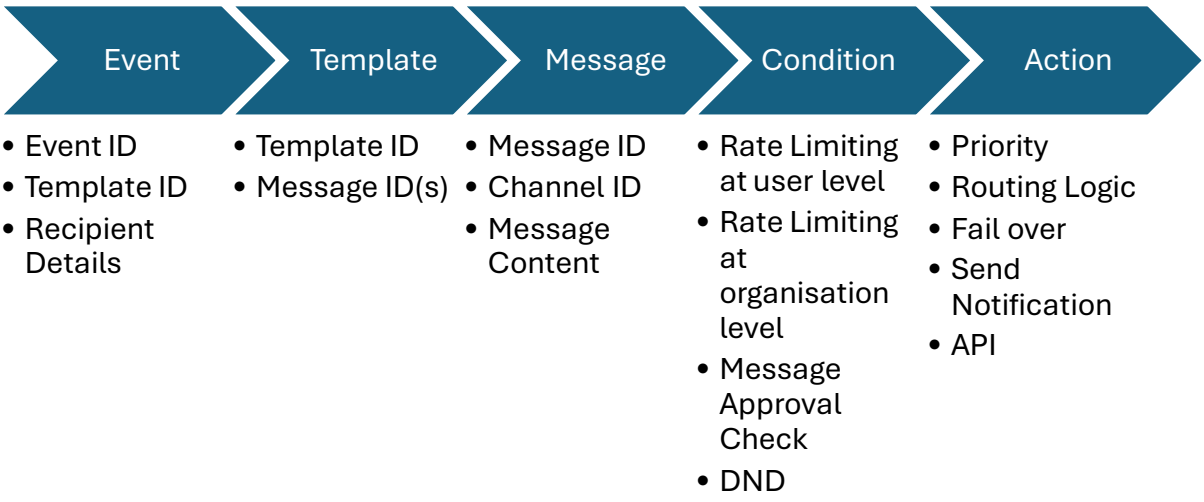
5.2.1. Transactional/Event based

A transactional or event-based notification is initiated by an external system based on a predefined event. The external system provides recipient-related information and other necessary details required for the notification. Various use cases where this type of notification is applicable are:

- Acceptance of transaction (Fresh/Additional Purchase, SIP, Redemption, etc.,)
- Updates on Processing stages (e.g., acceptance, reconciliation, unit allocation (both +ve and -ve)
- Account or portfolio changes (creation of a new folio, credit/debit of units, etc.,)
- Generation of account statements, CAS, etc.,

Example: A customer receives a notification about unit allocation and Net Asset Value (NAV) once a transaction is posted.

5.2.1.1. Diagram



5.2.1.2. Explanation of Diagram

The notification services application is designed to listen for events or signals from external applications. Upon receiving an event, the following steps are executed:

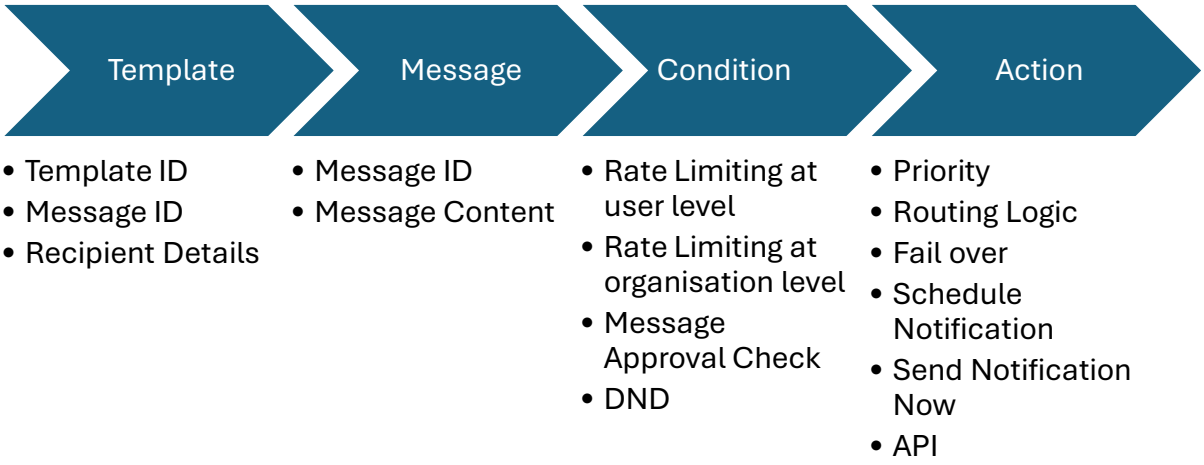
1. Event Reception: The application captures the event from the external system.
2. Template Processing: The event is processed using the relevant template, which includes Message ID(s) and recipient details such as contact information and other parameters.
3. Message Population: The messages associated with the template are populated with the parameters received from the event. This includes dynamic information specific to each recipient.
4. Condition Verification: The populated messages are then checked against both recipient-level and organization-level conditions to ensure all criteria are met.
5. Notification Dispatch: Once all conditions are verified, the notification is sent to the recipient.

5.2.2. Promotional

Promotional Notifications are bulk messages sent to a large group of recipients for marketing purposes. These notifications aim to inform recipients about promotions, events, or new offers related to a product or service. The recipients are typically defined at the message level, meaning the list of recipients is determined when the message is created.

Example: Emails and SMS notifications regarding the launch of a New Fund Offer (NFO).

5.2.2.1. Diagram



5.2.2.2. Explanation of Diagram

Promotional notifications do not rely on external events to initiate the notification flow. Instead, the process is triggered by an internal template. This template contains recipient details and the associated messages that need to be sent.

Steps Involved**:**

1. Template Initiation**:** The notification process begins with the triggering of an internal template. This template includes recipient details such as contact information and other necessary parameters.
2. Message Population**:** Once the template is triggered, the associated messages are called. These messages are populated with dynamic information pertaining to the recipients. This information can either be stored within the notification services application or uploaded by the user and stored at the template level for specific use within that template.
3. Condition Verification**:** After the messages are populated with recipient-specific dynamic parameters, both recipient-level and organization-level conditions are verified to ensure that all criteria are met.
4. Notification Dispatch**:** Upon successful verification, the notification is sent to the recipient.

5.2.3 Regulatory Notifications:

Regulatory notifications can include the below communication.

1. Communication of TER changes
2. Updates on Riskometer classifications

3. Other Regulatory changes (Investment amount, consolidation of folios, etc.,)

Feature/Requirement	Transactional	Regulatory	Promotional
Delivery Speed	Real-Time	Scheduled	Bulk with less urgency
Personalization	High	Low	High
Approval Workflow	Low	High	Medium
Recipient Preferences	Mandatory	Mandatory	Optional
Failure Escalation	Critical	High	Moderate
Compliance Priority	High	Very High	High

6. Functional Requirements

6.1. Transactional/Event Based Notifications

6.1.1. Message creation

- A message is the content of the notification in a pre-defined format for a specific channel.
- Users should have the capability to create, edit, view, and remove messages as needed. This flexibility ensures that the notification system can be easily maintained and updated.
- The messages should be easily searchable. The system should offer robust filter and sort options, allowing users to quickly find and manage messages.
- Users should be able to assign label and description to each message. These make it easier to locate and identify messages through the search functionality.

6.1.1.1. Channel 1 -Email

Message configuration

Static Content

Messages can contain static content that does not change based on recipient details.

Example:
Dear Customer,
Thanks for visiting our website. Looking forward to you seeing you again.
Best regards,
Application Team

Dynamic content (Variables)

Messages can also include dynamic parameters that are populated based on the recipient's details.

Example:

Dear [user],
Here is your OTP [1234]
Best regards,
Application Team

Variable Label Creation and Configuration

Variable labels can be created and configured at two levels:

- Organization Level Variable Creation:
  - **Definition**: Variables are defined centrally and can be used across multiple messages.
  - **Details Included**: Variable Label, Data Type, Length, and Constraints.
- Message Level Variable Creation:
  - **Definition**: New variable labels can be added specifically for use within a particular message.
- Search & tagging of pre-created variables to Messages
  - Pre-created variables at the organization level can be searched and added directly to messages. This facilitates the reuse of commonly used dynamic parameters.

Example: API contract for message from external system to notification service.

```
{
  "recipient ID": "101",
   "Name": "Ashok",
   "Email": "ashok@camsonline.com",
  }
```

Here, Name and Email could be populated as dynamic parameters within the message.

Formatting the Message

What you see is what you get Editor

The content editor should be a WYSIWYG (What You See Is What You Get) Editor. This means that the content seen in the editor for a particular message will be exactly what is displayed in the actual notification sent to the recipient in the corresponding channel.

A WYSIWYG editor is a user interface that allows users to create and edit content in a form that closely resembles its appearance when displayed to the end user. The key feature of a WYSIWYG editor is that it provides a visual representation of the content, which makes it easier for users to see and understand what the final output will look like.

Key Features

- **Real-Time Preview**:
    - Users can see how the content will look as they create and edit it. This real-time feedback helps ensure that the formatting, style, and layout are accurate before the content is sent out.
- **User-Friendly Interface**:
    - The WYSIWYG editor typically includes a toolbar with common text formatting options such as bold, italic, underline, font size, color, and alignment. This makes it easy for users to style their content without needing to know HTML or other coding languages.
- **Drag-and-Drop Functionality**:
    - WYSIWYG editor supports drag-and-drop functionality, allowing users to easily add and arrange images, videos, tables, and other multimedia elements within the content.

URL Shortening and QR Generation

- **URL Shortening**: The content editor should include a URL shortening feature that allows long URLs to be dynamically converted into shorter, more manageable links. This helps in creating cleaner and more visually appealing messages, especially for SMS or email notifications where space is limited.
- **QR Code Generation**: The editor should also have a QR code generation feature for both URLs and files. This feature enables the creation of QR codes that recipients can scan to easily access web pages, download files, or gain additional information without manually entering long URLs.

Multilingual Capability

The application must have the ability to create, edit, and send notifications in multiple languages. This ensures that the notification system can cater to a diverse audience, enhancing user experience and accessibility.

Virtual keyboards for regional languages should be available.

Voice to text option should be available for regional (multiple) languages.

HTML capability

The application must have the ability to create, edit, and send messages in HTML format. This feature ensures that notifications are visually appealing and can include rich formatting options such as images, links, and styles, enhancing the overall user experience.

Embedding iframe

The application must have the capability to embed iframes into messages. This feature allows the inclusion of content from external sources, such as dashboards, videos, or other web content, directly within the notification. The system should support embedding iframes from various sources, including but not limited to:

- Data visualization tools (e.g., Looker)
- Video platforms (e.g., YouTube)
- Other web-based applications and content providers

Attachments

The application must have the ability to attach different types of files within the Message.

File Formats

The application must have the ability to attach different types of files within the message. This functionality allows users to enhance notifications with additional documents, images, or other file types, providing recipients with comprehensive information.

- Image files .jpg, .jpeg, .gif, .bmp
- Document files .txt, .doc, .docx, .html, .htm, .pdf, .rtf, .xls, .xlsx, .csv
- Multimedia files .3gp, .amr, .mov, .mp3, .mp4, .mpeg, .mpg, .png, .tif, .tiff, .wav
- Presentation files .ppt, .pptx
- Video files .mp4, .mov, .wmv
- Audio files .m4a, .mp3, .wav

PDF

- **Encrypted and Normal Form**: The application must have the capability to attach PDF files in both encrypted and normal forms. This ensures that sensitive information can be securely transmitted while also allowing for standard PDF attachments where encryption is not required.
- **Size Limit**: The size limit for PDF attachments will be defined at the organizational level. An upper limit will also be set at the application level to ensure consistent handling of large files.

Version Control

The application should support message versioning to manage changes and updates effectively. This ensures that users can track the history of changes, revert to previous versions if necessary, and maintain an organized record of message revisions.

Key Features of Version Control:

1. **Message Versioning**:

    - Each message should have a version control system in place to manage updates and changes. Users should be able to create new versions of messages while preserving the history of previous versions.

2. **Change Tracking**:
    - The system should track all changes made to messages, including who made the changes and when. This audit trail helps maintain transparency and accountability.

3. **Revert to Previous Versions**:
    - Users should have the ability to revert to previous versions of a message if needed. This feature ensures that any inadvertent changes can be undone, restoring the message to an earlier state.

4. **Version History**:
    - The system should display a version history for each message, showing all the changes made over time. Users can review this history to understand the evolution of the message content.

5. **Comparison of Versions**:
    - The application should allow users to compare different versions of a message side-by-side to easily identify changes and differences.

Priority- Can be avoided- Check

Users must have the ability to set the priority level of a message. The priority levels ensure that notifications are transmitted in the order of importance, with critical messages being sent first.Critical

- Critical
- High
- Medium
- Low

The notifications in the pipeline will be transmitted in the order of priority. The notification with priority ‘Critical’ will be transmitted first and the notification with priority ‘Low’ will be transmitted at the last.

Notifications in the pipeline will be transmitted according to their assigned priority. This means:

- Notifications with **Critical** priority will be sent first.
- Notifications with **High** priority will be sent next.
- Notifications with **Medium** priority will follow.

- Notifications with **Low** priority will be sent last.

Content Generation

The application must enable the generation of content for messages using Generative AI (Gen AI). This feature allows users to leverage AI technology to create, edit, and enhance message content, ensuring higher efficiency and creativity.

Message Lifecycle

The message life cycle encompasses various stages that a message undergoes from creation to deletion. These stages ensure proper management, approval, and deployment of messages within the notification system.

Stages:

1. **Draft**:
    - **Description**: The message is saved as a draft during its creation. At this stage, the message can be edited and modified by the user.
    - **Purpose**: Allows users to work on the message content without sending it out immediately.
2. **Waiting for Approval**:
    - **Description**: Once the message has been finalized, it is sent for approval. Users can share the message for review to ensure its accuracy and appropriateness.
    - **Purpose**: Ensures that messages are reviewed and approved by authorized personnel before being sent out.
3. **Approved**: (Define who all should approve)
    - **Description**: Messages that have been reviewed and approved by users with the necessary authority. Approved messages are ready to be sent through the designated channels.
    - **Purpose**: Confirms that the message meets all necessary standards and can be dispatched to recipients.
4. **Inactive/Activate**:
    - **Inactive**:
        - **Description**: Messages can be deactivated by users with the appropriate permissions. Deactivated messages will not be used to send notifications, and any ongoing communication through the message will be discontinued.
        - **Tag**: Messages are tagged as 'inactive'.
    - **Activate**:
        - **Description**: Deactivated messages can be reactivated by authorized users. Reactivated messages are tagged as 'active'. Note that ongoing communications that were happening through the message before deactivation will not resume upon reactivation.
        - **Tag**: Messages are tagged as 'active'.
    - **Purpose**: Provides flexibility to manage message availability and usage within the system.
5. **Remove**:

- **Description**: Messages can be permanently deleted from the system by users with the necessary permissions.
- **Purpose**: Ensures that outdated or unnecessary messages are removed, maintaining the system's organization and relevance.

Approval process

Users must have the ability to send draft messages to designated approvers for approval. Messages must be approved before they can be sent as notifications.

Key Features:

- **Sending Draft Messages for Approval**:
  - **Functionality**: Users should be able to send messages saved in the draft state to approvers for review. This initiates the approval process.
  - **State Change**: Once a draft message is sent for approval, its status changes to 'Waiting for approval'.
- **Waiting for Approval State**:
  - **Description**: Messages in this state are pending review and approval by authorized personnel. During this period, the message cannot be edited by the original creator but can be reviewed and either approved or rejected by the approver.
  - **Approval Workflow**: The system should notify the approvers about messages awaiting their review. Approvers can then access these messages, review the content, and decide whether to approve or reject them.
- **Approval State**:
  - **Description**: Once a message is reviewed and approved by the approver, its status changes to 'Approved'. Only messages in this state can be pushed as notifications to recipients.
  - **Restriction**: Messages that are not in the 'Approved' state cannot be dispatched as notifications. This ensures that only vetted and verified messages are sent out.
- **Rejection and Feedback**:
  - **Functionality**: If a message is rejected, the approver should provide feedback or reasons for the rejection. The message will then revert to the draft state, allowing the creator to make necessary edits based on the feedback and resubmit for approval.

- **User Permissions**:
    - **Access Control**: Only users with appropriate permissions can send messages for approval, review pending messages, and approve or reject them. This ensures a secure and controlled approval process.

6.1.1.2. Channel 2-SMS

The notification service system should be able to send SMS notifications to recipients, adhering to specific formatting rules and compliance with regulations in India.

Formatting Rules:

- **Variable Limitations**:
    - Each SMS message can include up to five dynamic variables.
    - Each variable can have a maximum length of 30 characters.
    - Due to these limitations, the URL shortening feature is critical to ensure that long URLs do not exceed the character limit.
- **Character Limit**:
    - The total length of the SMS message cannot exceed 160 characters. This includes both static content and dynamic variables.
    - Careful consideration is required to balance the static message content and dynamic parameters within this character limit.

General Rules for SMS Notifications in India:

- **Sender ID**:
    - SMS messages must include a sender ID, which is a unique identifier associated with the sender. This helps recipients recognize the sender and adds authenticity to the message.
- **DND Compliance**:
    - Adherence to Do Not Disturb (DND) regulations is mandatory. SMS notifications should not be sent to recipients who have opted out of receiving promotional messages.
- **Timing Restrictions**:
    - Transactional SMS messages can be sent at any time but must comply with local regulations to avoid recipient inconvenience.

Whitelisting Message Formats

The notification system must facilitate the whitelisting of message formats for SMS. This process allows message formats to be approved before use, ensuring compliance and standardization.

Key Features:

- **Vendor API Integration**:
    - The vendor will expose an API for whitelisting message formats. This API will be integrated into the notification system to automate the message format approval process.
- **Seamless Approval Process**:
    - Clients, such as Asset Management Companies (AMCs), can submit message formats for approval directly through the application. The system will interface with the vendor's API to whitelist the message formats, streamlining the approval process.
- **Message Management**:
    - Users should be able to create, edit, and submit message formats for whitelisting. Once submitted, the status of the message formats (e.g., pending, approved, rejected) should be tracked within the application.
- **Real-Time Feedback**:
    - The system should provide real-time feedback on the status of submitted message formats. This includes notifications for approvals, rejections, and any required modifications.
- **Compliance Check**:
    - Before submission, message formats should be checked for compliance with regulatory and vendor-specific guidelines to increase the likelihood of approval.

6.1.1.3. Channel 3- Push

The notification service system must have the capability to send push notifications to recipients on various platforms, including:

- Android Devices
- iOS Devices
- Web Push Notifications on:
    - Windows systems

- Mac systems

6.1.1.4. Channel 4- WhatsApp

The notification system must facilitate the whitelisting of message formats for WhatsApp. This process allows message formats to be approved before use, ensuring compliance and standardization.

Key Features:

- **Vendor API Integration**:
  - The vendor will expose an API for whitelisting message formats. This API will be integrated into the notification system to automate the message format approval process.
- **Seamless Approval Process**:
  - Clients, such as Asset Management Companies (AMCs), can submit message formats for approval directly through the application. The system will interface with the vendor's API to whitelist the message formats, streamlining the approval process.
- **Message Management**:
  - Users should be able to create, edit, and submit message formats for whitelisting. Once submitted, the status of the message formats (e.g., pending, approved, rejected) should be tracked within the application.
- **Real-Time Feedback**:
  - The system should provide real-time feedback on the status of submitted message formats. This includes notifications for approvals, rejections, and any required modifications.
- **Compliance Check**:
  - Before submission, message formats should be checked for compliance with regulatory and vendor-specific guidelines to increase the likelihood of approval.

6.1.1.5. Channel 5- In App

An in-app notification system allows app developers to send messages directly to users within the app environment. This can include updates, reminders, promotional offers, or any important information that enhances user engagement and interaction. Here are some key aspects:

Example Use Cases:

- **Welcome Messages**: Greeting new users and providing a quick tour of the app's features.
- **Promotional Offers**: Notifying users about discounts or special offers.
- **Reminders**: Reminding users about incomplete actions, such as items left in the cart.
- **Feature Updates**: Informing users about new features or updates within the app.

6.1.1.6. Channel 6- Voice

The notification service system must support both inbound and outbound voice notifications, with the capability to provide interactive responses based on recipients' selections.

Key Features:

- **Inbound/Interactive Voice Response (IVR)**:
    - **Functionality**: Provides a menu system that allows recipients to interact with the system using their phone keypad or voice commands. The system responds based on the selections made by the recipient.
    - **Examples**:
        - Press 1 for account balance
        - Press 2 for recent transactions
        - Press 3 to speak with a representative
- **Outbound Voice Notifications**:
    - **Functionality**: Enables the system to place automated calls to recipients to deliver messages or alerts.
    - **Use Cases**: Reminders for appointments, notifications of overdue payments, or alerts for suspicious account activity.

6.1.2. Template creation

- **Template Creation and Integration**:
    - **Message Association**: When creating a template for transactional or event-based notifications, the associated messages need to be added to the template.
    - **API Contract**: The template's API contract must be shared with the external application that will be the source of the event. This contract defines how the external system will interact with the notification system.

- **Event-Triggered Notifications**:
    - **Event Activation**: Transactional or event-based notifications are initiated by an event. This event activates the template via an API call and shares recipient-related information along with any other necessary data required by the message.
    - **Dynamic Parameter Mapping**: The recipient details provided by the template are mapped to dynamic parameters within the message. These parameters are then populated with personalized content.
- **Mapping Recipient Information**:
    - The template serves as the source for mapping recipient information from the external system to the dynamic parameters used in the messages. This ensures that the notifications are personalized based on the recipient's details.
- **Template Accessibility and Governance**:
    - **Organization-Wide Availability**: Templates created under an organization should be accessible to all entities within the organization, including admins, users, and sub-organizations. This facilitates consistent use of standardized templates across the entire organization.
    - **Governance Structure**: The governance structure will ensure that template management, including creation, editing, and usage, follows established guidelines and protocols within the organization.

6.1.3. Preview

Users should have the permission to view messages as a preview across multiple devices and browsers to ensure consistency and proper rendering before sending notifications.

Key Features:

- **Device Compatibility**:
    - **Functionality**: The system should support previewing messages on various devices, including smartphones, tablets, and desktops.
    - **Platforms**:
        - **Mobile/Tablet**: iOS and Android
        - **Desktops/Laptops**: Windows and MacOS
- **Browser Support**:

    - **Functionality**: Users should be able to preview messages across different web browsers to ensure compatibility and consistent appearance.
    - **Browsers**:
        - Google Chrome
        - Mozilla Firefox
        - Microsoft Edge
        - Safari

- **Responsive Design Testing**:
    - **Functionality**: The preview feature should allow users to test how messages adapt to different screen sizes and orientations, ensuring a responsive design.

6.1.4. Mapping Messages to Template

The notification system must facilitate the integration of messages with templates to streamline the process of sending notifications triggered by events.

- Each message will contain a unique Message ID. This ID is shared with the template component.
- The template component will map which message it should refer to using the Message ID.
- The template component is called by an external application or event. When a template is initiated by an event, the corresponding message is activated.
- A single template can be associated with more than one message. This allows for flexibility in sending different messages based on the same event.
- Only approved messages can be added to a template. This ensures that all messages sent through the system have been reviewed and meet quality standards.
- Users can enable or disable templates as needed. When a template is disabled, it will not trigger any messages, even if the associated event occurs.

6.1.5. Contract Generation

- Once a template is created, the user/administrator can generate API contract for the template through which external applications can share information with the notification service system (template).

- The API contract will include the dynamic parameters required to populate messages mapped with the template.
- The external system needs to make sure that whenever an event occurs, the associated template needs to be evoked using the corresponding API contract. This integration is done on the external system.

6.1.6. Event

- An event is something that happens outside the notification service system. For example, the user getting units of a fund allotted to their DEMAT accounts is an event.

Example Workflow:

1. **Template Creation**:
    - A template is created in the notification service system, and the required messages are mapped to it.
2. **API Contract Generation**:
    - The user or administrator generates an API contract for the template. This contract includes dynamic parameters such as [Name], [EventDetails], [Date], etc.
3. **Event Triggering**:
    - An event occurs in the external system (e.g., the user getting units of a fund allotted to their DEMAT accounts is an event.)
4. **API Call**:
    - The external system makes an API call to the notification service system, invoking the template using the provided API contract. It includes all the necessary recipient information and event details.
5. **Message Population and Dispatch**:
    - The notification service system uses the dynamic parameters to populate the messages and dispatches the notifications to the recipients.

6.1.6.1. Event List (For CAMS RTA use- To be provided)

6.1.7. Send Notification

Routing logic overview

The routing logic will determine the vendor through which a particular message and its associated channel will reach the recipient. This ensures efficient and reliable delivery of notifications. The routing logic is defined at a template level.

Routing configuration

Omni-Channel Routing

- All messages within the template are sent as notifications based on the vendor selected manually or using auto-routing logic.
- **Re-Try Configuration**:
    - Users can set up how many retries can be attempted until the send operation is successful for each route.
    - Users can define the time period between each retry for each route.
- **Fail-Over Logic**:
    - If a notification is not delivered successfully for a particular message after the retries are exhausted, the fail-over logic is implemented.
    - Users can select the number of fail-overs.
    - In each fail-over, the user can select one of the logics below.
        - Manual- Users can manually select alternate vendors
        - Cost Based- The cheapest alternative (vendor) is chosen
        - Performance Based- The alternative (vendor) with best delivery rate is chosen
    - If an attempt is successful, subsequent fail-overs will not be called.

Hierarchical Routing

- Users pick the order in which messages are sent from the template. Users can select the order manually or based on low cost or high-performance logic.
- **Re-Try Configuration**:
    - Users can set up how many retries can be attempted until the send operation is successful for each route.
    - Users can define the time period between each retry for each route.
- User can define exactly how many messages from the template needs to reach the recipient.
- If the defined number of messages are successful in reaching the recipient, subsequent messages in the hierarchy will not be attempted.
    - Manual- Users can manually select order of the messages within the template in which they need to be shared

    - Cost Based- The cheapest channel and vendor combination from the message is chosen. Channel is already given in the message; vendor will be chosen to populate based on cost (low).
    - Performance Based- The channel and vendor combination with the highest delivery rate is chosen. Channel is already given in the message; vendor will be chosen to populate based on delivery rate (high).

- **Fail-Over Logic**:
    - If a notification is not delivered successfully for a particular message after the retries are exhausted, the fail-over logic is implemented.
    - Users can select the number of fail-overs.
    - In each fail-over, the user can select one of the logics below.
        - Manual- Users can manually select alternate vendors
        - Cost Based- The cheapest alternative (vendor) is chosen
        - Performance Based- The alternative (vendor) with best delivery rate is chosen
    - If an attempt is successful, subsequent fail-overs will not be called.

Testing Notification

Testing a template can be done the users at any point of time. Organisation can set the limit on the number of times a message or template can be tested by users at each level (super admin, admin and users).

Pre-Defined Recipient Details:

- **Setup by Admin**: Admins have the option to set up pre-defined email IDs, phone numbers, and other recipient details for testing purposes.
- **Usage**: Users can use these pre-defined details to test the templates or messages, ensuring they work correctly before sending out notifications.

Ad Hoc Recipient Testing:

- **User-Entered Details**: Users can enter ad hoc recipient details to test the messages. This allows for flexible testing scenarios and ensures that messages are personalized and formatted correctly.
- **Testing Process**: Before finalizing and sending out notifications, users can simulate the notification process using these ad hoc details to check for accuracy and functionality.

API Call

- Notifications are typically sent through API calls to the vendor.
- The API contract and endpoints are defined by the vendor.
- These details are configured at Super-Admin level.

6.1.8. Response Handler

Event Trigger Acknowledgement

- **Receipt Confirmation**:
    - When an event is triggered from the external system and reaches the notification system, the notification system shares an acknowledgment confirming it has received the communication. This acknowledgment can be shared as an API response.
    - The external system should provide a callback URL along with the event itself.
- **Error Handling**:
    - If the API contract does not adhere to the requested template format or satisfy the prerequisites (such as missing mandatory fields), the notification system sends an error message stating the reason along with the acknowledgment message.
- **Notification ID**:
    - The acknowledgment includes the Notification ID, which can be used for polling purposes to track the status of the notification.

Notification Status

- The notification system should inform the external system about the status of the notification by sharing the following details:
    - Notification ID
    - Template ID
    - Message ID
    - Status of the Message:
        - Notification Successful
        - Notification not received by the customer yet -> Fall back is in progress
        - Notification Failed

- Recipient ID
- Channel
- Vendor
- Timestamp

The notification status can be shared when there is an update in the status of the message or external systems can find the status through polling in real time.

6.1.9. Conditional setups

Rate Limiting

Rate Limiting at Organisation Level

- Limits the number of notifications sent out (channel-wise and vendor-wise) and the number of recipients added in a single notification.
- Sets limits based on the requests from events or the organization itself. It checks the client/event calling the notification services application to ensure the allowed number of calls.
- Limit the number of times a message can be tested by an individual at role level and at an organisation and sub-organisation level.

Rate Limiting at Recipient Level

- Users can set limits, as requested by the recipient, on the number of notifications sent via a particular channel or across channels on a daily, weekly, or monthly basis.

Rate Limiting at Channel and Vendor Level

- Rate limiting can also be set at a channel and vendor level. Both recipient and organization limits must comply with this limit, which is particularly important for SMS notifications.

Do Not Disturb (DND)

DND at Recipient Level

- DND can be set for a particular channel and vendor at the recipient level.
- Users can set DND based on time periods during which notifications should not be shared with recipients.
- Notifications cannot be sent if the DND conditions for channel, vendor, or time are met.
- Both recipients and administrators can set up DND.

DND at Organisation Level

- Only administrators can set up DND criteria at the organization level for channel, vendor, and time. Notifications cannot be sent out through the specified channel and vendor or at particular time intervals based on these conditions.

DND at Channel and Vendor Level

- DND can also be set at a channel and vendor level, ensuring that both recipient and organization limits comply with this setting, which is crucial for SMS notifications.

6.2. Promotional Notifications

6.2.1. Message creation

- A message is the content of the notification in a pre-defined format for a specific channel and vendor.
- Users should have the capability to create, edit, view, and remove messages as needed. This flexibility ensures that the notification system can be easily maintained and updated.
- The messages should be easily searchable. The system should offer robust filter and sort options, allowing users to quickly find and manage messages.
- Users should be able to assign label and description to each message. These make it easier to locate and identify messages through the search functionality.

6.2.1.1. Channel 1 -Email

Message configuration

Static Content

Messages can contain static content that does not change based on recipient details.

Example:
Dear Customer,
Thanks for visiting our website. Looking forward to you seeing you again.
Best regards,
Application Team

Dynamic content (Variables)

Messages can also include dynamic parameters that are populated based on the recipient's details.

Example:
Dear [user],
Here is your OTP [1234].
Best regards,
Application Team

Variable Label Creation and Configuration

Variable labels can be created and configured at two levels:

- Organization Level Variable Creation:
  - **Definition**: Variables are defined centrally and can be used across multiple messages.
  - **Details Included**: Variable Label, Data Type, Length, and Constraints.
- Message Level Variable Creation:
  - **Definition**: New variable labels can be added specifically for use within a particular message.
- Search & tagging of pre-created variables to Messages
  - Pre-created variables at the organization level can be searched and added directly to messages. This facilitates the reuse of commonly used dynamic parameters.

Example: API contract for message from external system to notification service.

```
{
  "recipient ID": "101",
   "Name": "Ashok",
   "Email": "ashok@camsonline.com",
  }
```

Here, Name and Email could be populated as dynamic parameters within the message.

Formatting the Message

What you see is what you get Editor

The content editor should be a WYSIWYG (What You See Is What You Get) Editor. This means that the content seen in the editor for a particular message will be exactly what is displayed in the actual notification sent to the recipient in the corresponding channel.

A WYSIWYG editor is a user interface that allows users to create and edit content in a form that closely resembles its appearance when displayed to the end user. The key feature of a WYSIWYG editor is that it provides a visual representation of the content, which makes it easier for users to see and understand what the final output will look like.

Key Features

- **Real-Time Preview**:
  - Users can see how the content will look as they create and edit it. This real-time feedback helps ensure that the formatting, style, and layout are accurate before the content is sent out.

- **User-Friendly Interface**:
    - The WYSIWYG editor typically includes a toolbar with common text formatting options such as bold, italic, underline, font size, color, and alignment. This makes it easy for users to style their content without needing to know HTML or other coding languages.
- **Drag-and-Drop Functionality**:
    - WYSIWYG editor supports drag-and-drop functionality, allowing users to easily add and arrange images, videos, tables, and other multimedia elements within the content.

URL Shortening and QR Generation

- **URL Shortening**: The content editor should include a URL shortening feature that allows long URLs to be dynamically converted into shorter, more manageable links. This helps in creating cleaner and more visually appealing messages, especially for SMS or email notifications where space is limited.
- **QR Code Generation**: The editor should also have a QR code generation feature for both URLs and files. This feature enables the creation of QR codes that recipients can scan to easily access web pages, download files, or gain additional information without manually entering long URLs.

Multilingual Capability

The application must have the ability to create, edit, and send notifications in multiple languages. This ensures that the notification system can cater to a diverse audience, enhancing user experience and accessibility.

HTML capability

The application must have the ability to create, edit, and send messages in HTML format. This feature ensures that notifications are visually appealing and can include rich formatting options such as images, links, and styles, enhancing the overall user experience.

Embedding iframe

The application must have the capability to embed iframes into messages. This feature allows the inclusion of content from external sources, such as dashboards, videos, or other web content, directly within the notification. The system should support embedding iframes from various sources, including but not limited to:

- Data visualization tools (e.g., Looker)
- Video platforms (e.g., YouTube)
- Other web-based applications and content providers

Attachments

The application must have the ability to attach different types of files within the Message.

File Formats

The application must have the ability to attach different types of files within the message. This functionality allows users to enhance notifications with additional documents, images, or other file types, providing recipients with comprehensive information.

- Image files .jpg, .jpeg, .gif, .bmp
- Document files .txt, .doc, .docx, .html, .htm, .pdf, .rtf, .xls, .xlsx, .csv
- Multimedia files .3gp, .amr, .mov, .mp3, .mp4, .mpeg, .mpg, .png, .tif, .tiff, .wav
- Presentation files .ppt, .pptx
- Video files .mp4, .mov, .wmv
- Audio files .m4a, .mp3, .wav

PDF

- **Encrypted and Normal Form**: The application must have the capability to attach PDF files in both encrypted and normal forms. This ensures that sensitive information can be securely transmitted while also allowing for standard PDF attachments where encryption is not required.
- **Size Limit**: The size limit for PDF attachments will be defined at the organizational level. An upper limit will also be set at the application level to ensure consistent handling of large files.

Version Control

The application should support message versioning to manage changes and updates effectively. This ensures that users can track the history of changes, revert to previous versions if necessary, and maintain an organized record of message revisions.

Key Features of Version Control:

6. **Message Versioning**:
    - Each message should have a version control system in place to manage updates and changes. Users should be able to create new versions of messages while preserving the history of previous versions.
7. **Change Tracking**:
    - The system should track all changes made to messages, including who made the changes and when. This audit trail helps maintain transparency and accountability.
8. **Revert to Previous Versions**:

- Users should have the ability to revert to previous versions of a message if needed. This feature ensures that any inadvertent changes can be undone, restoring the message to an earlier state.

9. **Version History**:
    - The system should display a version history for each message, showing all the changes made over time. Users can review this history to understand the evolution of the message content.
10. **Comparison of Versions**:
    - The application should allow users to compare different versions of a message side-by-side to easily identify changes and differences.

Priority

Users must have the ability to set the priority level of a message. The priority levels ensure that notifications are transmitted in the order of importance, with critical messages being sent first.Critical

- Critical
- High
- Medium
- Low

The notifications in the pipeline will be transmitted in the order of priority. The notification with priority 'Critical' will be transmitted first and the notification with priority 'Low' will be transmitted at the last.

Notifications in the pipeline will be transmitted according to their assigned priority. This means:

- Notifications with **Critical** priority will be sent first.
- Notifications with **High** priority will be sent next.
- Notifications with **Medium** priority will follow.
- Notifications with **Low** priority will be sent last.

Content Generation

The application must enable the generation of content for messages using Generative AI (Gen AI). This feature allows users to leverage AI technology to create, edit, and enhance message content, ensuring higher efficiency and creativity.

Message Lifecycle

The message life cycle encompasses various stages that a message undergoes from creation to deletion. These stages ensure proper management, approval, and deployment of messages within the notification system.

Stages:

6. **Draft**:
    - **Description**: The message is saved as a draft during its creation. At this stage, the message can be edited and modified by the user.
    - **Purpose**: Allows users to work on the message content without sending it out immediately.
7. **Waiting for Approval**:
    - **Description**: Once the message has been finalized, it is sent for approval. Users can share the message for review to ensure its accuracy and appropriateness.
    - **Purpose**: Ensures that messages are reviewed and approved by authorized personnel before being sent out.
8. **Approved**:
    - **Description**: Messages that have been reviewed and approved by users with the necessary authority. Approved messages are ready to be sent through the designated channels.
    - **Purpose**: Confirms that the message meets all necessary standards and can be dispatched to recipients.
9. **Inactive/Activate**:
    - **Inactive**:
        - **Description**: Messages can be deactivated by users with the appropriate permissions. Deactivated messages will not be used to send notifications, and any ongoing communication through the message will be discontinued.
        - **Tag**: Messages are tagged as 'inactive'.
    - **Activate**:
        - **Description**: Deactivated messages can be reactivated by authorized users. Reactivated messages are tagged as 'active'. Note that ongoing communications that were happening through the message before deactivation will not resume upon reactivation.
        - **Tag**: Messages are tagged as 'active'.
    - **Purpose**: Provides flexibility to manage message availability and usage within the system.
10. **Remove**:

- **Description**: Messages can be permanently deleted from the system by users with the necessary permissions.
- **Purpose**: Ensures that outdated or unnecessary messages are removed, maintaining the system's organization and relevance.

Approval process

Users must have the ability to send draft messages to designated approvers for approval. Messages must be approved before they can be sent as notifications.

Key Features:

- **Sending Draft Messages for Approval**:
  - **Functionality**: Users should be able to send messages saved in the draft state to approvers for review. This initiates the approval process.
  - **State Change**: Once a draft message is sent for approval, its status changes to 'Waiting for approval'.
- **Waiting for Approval State**:
  - **Description**: Messages in this state are pending review and approval by authorized personnel. During this period, the message cannot be edited by the original creator but can be reviewed and either approved or rejected by the approver.
  - **Approval Workflow**: The system should notify the approvers about messages awaiting their review. Approvers can then access these messages, review the content, and decide whether to approve or reject them.
- **Approval State**:
  - **Description**: Once a message is reviewed and approved by the approver, its status changes to 'Approved'. Only messages in this state can be pushed as notifications to recipients.
  - **Restriction**: Messages that are not in the 'Approved' state cannot be dispatched as notifications. This ensures that only vetted and verified messages are sent out.
- **Rejection and Feedback**:
  - **Functionality**: If a message is rejected, the approver should provide feedback or reasons for the rejection. The message will then revert to the draft state, allowing the creator to make necessary edits based on the feedback and resubmit for approval.

- **User Permissions**:
    - **Access Control**: Only users with appropriate permissions can send messages for approval, review pending messages, and approve or reject them. This ensures a secure and controlled approval process.

6.2.1.2. Channel 2-SMS

The notification service system should be able to send SMS notifications to recipients, adhering to specific formatting rules and compliance with regulations in India.

Formatting Rules:

- **Variable Limitations**:
    - Each SMS message can include up to five dynamic variables.
    - Each variable can have a maximum length of 30 characters.
    - Due to these limitations, the URL shortening feature is critical to ensure that long URLs do not exceed the character limit.
- **Character Limit**:
    - The total length of the SMS message cannot exceed 160 characters. This includes both static content and dynamic variables.
    - Careful consideration is required to balance the static message content and dynamic parameters within this character limit.

General Rules for SMS Notifications in India:

- **Sender ID**:
    - SMS messages must include a sender ID, which is a unique identifier associated with the sender. This helps recipients recognize the sender and adds authenticity to the message.
- **DND Compliance**:
    - Adherence to Do Not Disturb (DND) regulations is mandatory. SMS notifications should not be sent to recipients who have opted out of receiving promotional messages.
- **Timing Restrictions**:
    - Transactional SMS messages can be sent at any time but must comply with local regulations to avoid recipient inconvenience.

Whitelisting Message Formats

The notification system must facilitate the whitelisting of message formats for SMS. This process allows message formats to be approved before use, ensuring compliance and standardization.

Key Features:

- **Vendor API Integration**:
    - The vendor will expose an API for whitelisting message formats. This API will be integrated into the notification system to automate the message format approval process.
- **Seamless Approval Process**:
    - Clients, such as Asset Management Companies (AMCs), can submit message formats for approval directly through the application. The system will interface with the vendor's API to whitelist the message formats, streamlining the approval process.
- **Message Management**:
    - Users should be able to create, edit, and submit message formats for whitelisting. Once submitted, the status of the message formats (e.g., pending, approved, rejected) should be tracked within the application.
- **Real-Time Feedback**:
    - The system should provide real-time feedback on the status of submitted message formats. This includes notifications for approvals, rejections, and any required modifications.
- **Compliance Check**:
    - Before submission, message formats should be checked for compliance with regulatory and vendor-specific guidelines to increase the likelihood of approval.

6.2.1.3. Channel 3- Push

The notification service system must have the capability to send push notifications to recipients on various platforms, including:

- Android Devices
- iOS Devices
- Web Push Notifications on:
    - Windows systems

- Mac systems

6.2.1.4. Channel 4- WhatsApp

The notification system must facilitate the whitelisting of message formats for WhatsApp. This process allows message formats to be approved before use, ensuring compliance and standardization.

Key Features:

- **Vendor API Integration**:
  - The vendor will expose an API for whitelisting message formats. This API will be integrated into the notification system to automate the message format approval process.
- **Seamless Approval Process**:
  - Clients, such as Asset Management Companies (AMCs), can submit message formats for approval directly through the application. The system will interface with the vendor's API to whitelist the message formats, streamlining the approval process.
- **Message Management**:
  - Users should be able to create, edit, and submit message formats for whitelisting. Once submitted, the status of the message formats (e.g., pending, approved, rejected) should be tracked within the application.
- **Real-Time Feedback**:
  - The system should provide real-time feedback on the status of submitted message formats. This includes notifications for approvals, rejections, and any required modifications.
- **Compliance Check**:
  - Before submission, message formats should be checked for compliance with regulatory and vendor-specific guidelines to increase the likelihood of approval.

6.2.1.5. Channel 5- In App

An in-app notification system allows app developers to send messages directly to users within the app environment. This can include updates, reminders, promotional offers, or any important information that enhances user engagement and interaction. Here are some key aspects:

Example Use Cases:

- **Welcome Messages**: Greeting new users and providing a quick tour of the app's features.
- **Promotional Offers**: Notifying users about discounts or special offers.
- **Reminders**: Reminding users about incomplete actions, such as items left in the cart.
- **Feature Updates**: Informing users about new features or updates within the app.

6.2.1.6. Channel 6- Voice

The notification service system must support both inbound and outbound voice notifications, with the capability to provide interactive responses based on recipients' selections.

Key Features:

- **Inbound/Interactive Voice Response (IVR)**:
    - **Functionality**: Provides a menu system that allows recipients to interact with the system using their phone keypad or voice commands. The system responds based on the selections made by the recipient.
    - **Examples**:
        - Press 1 for account balance
        - Press 2 for recent transactions
        - Press 3 to speak with a representative
- **Outbound Voice Notifications**:
    - **Functionality**: Enables the system to place automated calls to recipients to deliver messages or alerts.
    - **Use Cases**: Reminders for appointments, notifications of overdue payments, or alerts for suspicious account activity.

6.2.2. Template creation

- When creating a template for promotional notifications, the associated messages need to be added to the template.
- Recipient details and other relevant information required to populate the message are added to the template. These details can be fetched from the notification system database or added on an ad-hoc basis at the template level.
- Users can add recipient information and other details to the template through an Excel file for ad-hoc user additions.

- The promotional notification is initiated within the notification system itself, without reliance on an external event trigger.
- Recipient details shared by the template should be mapped to dynamic parameters (if available), which are then used to populate the message with personalized content.
- Templates and messages created under an organization should be available to all entities within the organization, including admins, users, and sub-organizations.

6.2.3. Preview

- Users should have the permission to view the Messages as a preview across multiple devices and browsers.

6.2.4. Mapping Messages to Template

Message ID and Template Mapping:

- **Message ID**: Each message will contain a unique Message ID. This ID is shared with the template component.
- **Template Mapping**: The template component maps which message it should refer to using the Message ID. Whenever a template is initiated by the 'Send Action' operation, the corresponding message will be called.

Multiple Message Association:

- **Flexibility**: A single template can be associated with more than one message. This allows for flexible and diverse messaging within a single notification flow.

Template Management:

- **Enable/Disable Templates**: Users can enable or disable templates as needed. When a template is disabled, it will not trigger any messages, even if the 'Send Action' operation is performed.
- **Approved Messages**: Only approved messages can be added to a template. This ensures that all messages sent through the system have been reviewed and meet quality standards.

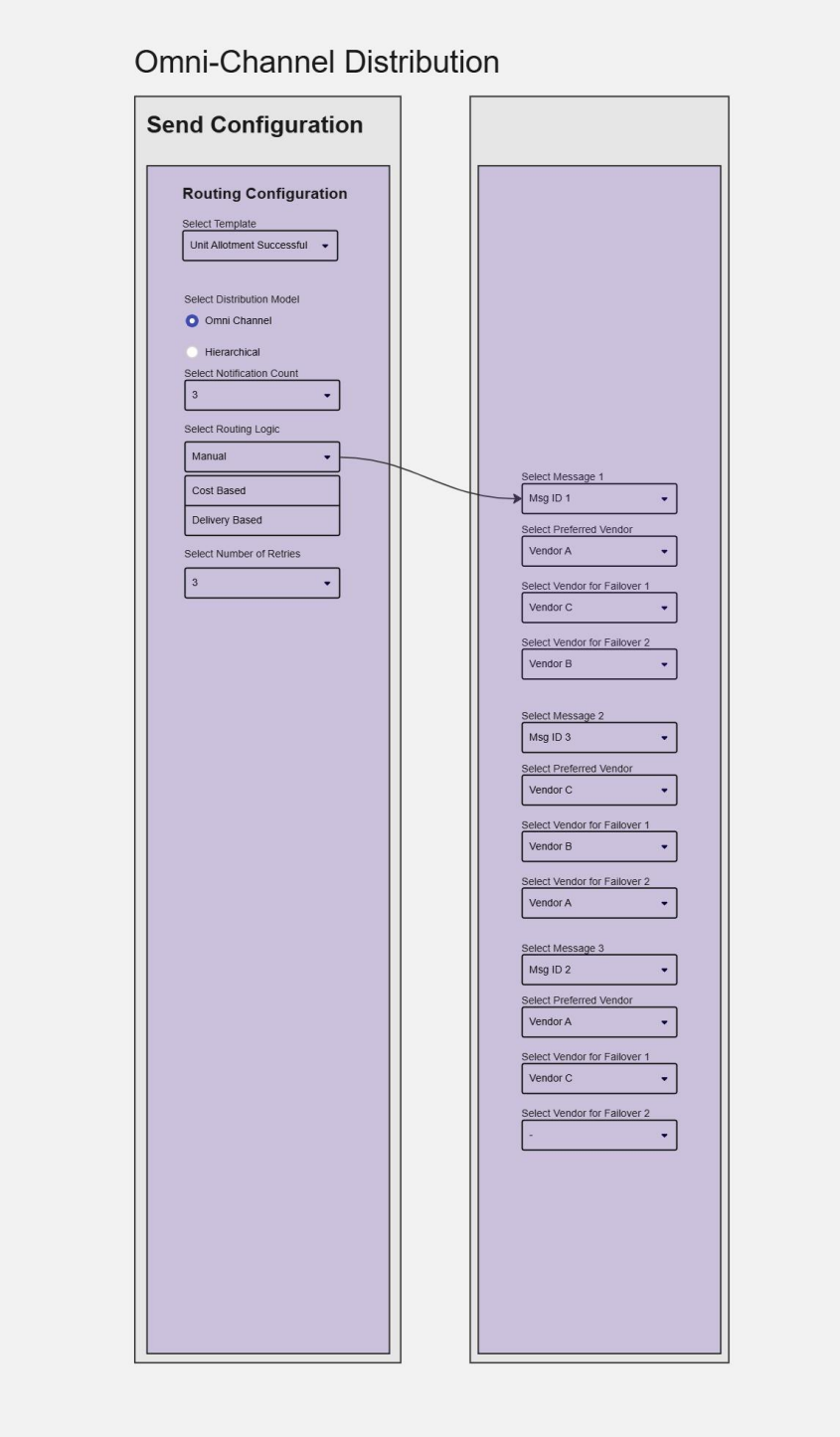
6.2.5. Send Notification

Routing logic overview

The routing logic will determine the vendor through which a particular message and its associated channel will reach the recipient. This ensures efficient and reliable delivery of notifications. The routing logic is defined at a template level.

Routing Logic Configuration

Omni-Channel Distribution Model

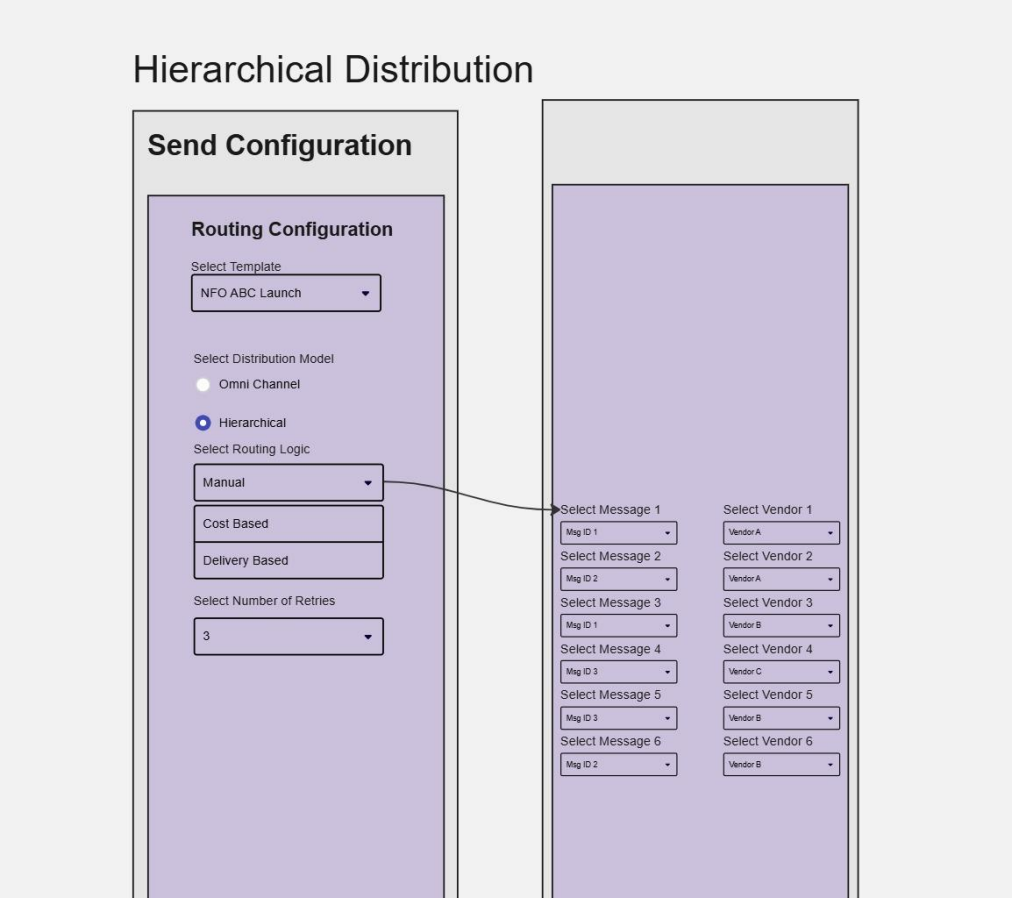


- Users can configure the number of messages within the template that can be sent as notifications based on the vendor selected manually or using auto-routing logic. The number cannot be more than the number of messages within the template.
- There are three routing logics

- Manual: User will have to select manually, the message ID and associated vendor through which it has to reach the recipient. User can select up to two failovers.
- Cost Based: System will automatically sort the cheapest message and vendor combination along with the fail-overs. Messages and failovers will be triggered based on this logic. No message will be sent more than once across any vendor. This means that the recipient will not receive more than one message on a single channel.
- Delivery Based: System will automatically sort the message and vendor combination with the best delivery rates along with the fail-overs. Messages and failovers will be triggered based on this logic.

- **Re-Try Configuration**:
  - Users can set up how many retries can be attempted until the send operation is successful for each route.
  - Retry will be initiated only if notification status from the vendor is marked as ‘failed’
  - If a notification is not delivered successfully for a particular message after the retries are exhausted, the fail-over logic is implemented.

Hierarchical Routing



- The difference between omnichannel and hierarchical routing is that, in hierarchical distribution model, the recipient will only receive one message from a template.
- Users pick the order in which messages are sent from the template. Users can select the order manually or based on low cost or high-performance logic.
- There are three routing logics
    - Manual: User will have to select manually, the message ID and associated vendor through which it has to reach the recipient. User can select up to two failovers. Here, the user can select same message ID, multiple times but with different vendors.
    - Cost Based: System will automatically sort the cheapest message and vendor combination. Messages will be triggered based on this logic.
    - Delivery Based: System will automatically sort the message and vendor combination with the best delivery rate. Messages and failovers will be triggered based on this logic.
- **Re-Try Configuration**:

- Users can set up how many retries can be attempted until the send operation is successful for each route.
- Retry will be initiated only if notification status from the vendor is marked as 'failed'
- If a notification is not delivered successfully for a particular message after the retries are exhausted, the fail-over logic is implemented.

Testing Notification

Testing a template can be done the users at any point of time. Organisation can set the limit on the number of times a message or template can be tested by users at each level (super admin, admin and users).

Pre-Defined Recipient Details:

- **Setup by Admin**: Admins have the option to set up pre-defined email IDs, phone numbers, and other recipient details for testing purposes.
- **Usage**: Users can use these pre-defined details to test the templates or messages, ensuring they work correctly before sending out notifications.

Ad Hoc Recipient Testing:

- **User-Entered Details**: Users can enter ad hoc recipient details to test the messages. This allows for flexible testing scenarios and ensures that messages are personalized and formatted correctly.
- **Testing Process**: Before finalizing and sending out notifications, users can simulate the notification process using these ad hoc details to check for accuracy and functionality.

Schedule Notification

This is done at a Message level. User can define when a notification should be shared with the designated recipients.

Send now

This is done at a Message level. User can send the notification manually as an when required.

API Call

- Notifications are typically sent through API calls to the vendor.
- The API contract and endpoints are defined by the vendor.
- These details are configured at Super-Admin level.

6.2.6. Response Handler

Template Trigger Confirmation

- When a template is triggered in the notification system, the system shares an acknowledgment confirming the send operation for a particular template has started. This acknowledgment can be shared as an API response.
- The acknowledgment includes the Notification ID, which can be used for polling purposes to track the status of the notification.

Notification Status

- The notification system should inform the external system about the status of the notification by sharing the following details:
    - Notification ID
    - Template ID
    - Message ID
    - Status of the Message:
        - Notification Successful
        - Notification not received by the customer yet -> Fall back is in progress
        - Notification Failed
    - Recipient ID
    - Channel
    - Vendor
    - Timestamp

The notification status can be shared when there is an update in the status of the message or external systems can find the status through polling in real time.

6.2.7. Conditional setups

Rate Limiting

Rate Limiting at Organisation Level

- Limits the number of notifications sent out (channel-wise and vendor-wise) and the number of recipients added in a single notification.
- Sets limits based on the requests from events or the external system. It checks the client/event calling the notification services application to ensure the allowed number of calls.
- Limit the number of times a message can be tested by an individual at role level and at an organisation and sub-organisation level.

Rate Limiting at Recipient Level

- Users can set limits, as requested by the recipient, on the number of notifications sent via a particular channel or across channels on a daily, weekly, or monthly basis.

Rate Limiting at Channel and Vendor Level

- Rate limiting can also be set at a channel and vendor level. Both recipient and organization limits must comply with this limit, which is particularly important for SMS notifications.

Do Not Disturb (DND)

DND at Recipient Level

- DND can be set for a particular channel and vendor at the recipient level.
- Users can set DND based on time periods during which notifications should not be shared with recipients.
- Notifications cannot be sent if the DND conditions for channel, vendor, or time are met.
- Both recipients and administrators can set up DND.

DND at Organisation Level

- Only administrators can set up DND criteria at the organization level for channel, vendor, and time. Notifications cannot be sent out through the specified channel and vendor or at particular time intervals based on these conditions.

DND at Channel and Vendor Level

- DND can also be set at a channel and vendor level, ensuring that both recipient and organization limits comply with this setting, which is crucial for SMS notifications.

6.3. Recipient Management

6.3.1. Recipient Overview

Recipients are people who receive the messages from organisations through notification system. Each recipient that is stored within the notification system has to be mapped under an organisation.

6.3.2. Recipient Data Dictionary

Data Dictionary consists of various fields, conditions/validations related to recipients pertaining to an organization. Any recipient who is added to the notification system under an organization has to adhere to the data dictionary of the organisation.

The data dictionary of an organisation is defined and configured on the notification system by the organisation administrator.

Recipients of transactional notifications and those who are directly added at the template level do not have to adhere to the data dictionary of the organization. They only have to comply to the data dictionary at the template level.

6.3.3. Recipient Types

- Individual Recipients- Individual recipients can be created directly on the notification system or imported from external systems and maintained at an organisation or sub-organisation level within the application.
- Recipient Groups- Collection of multiple recipients who are created on the notifications system.

The recipient information added to the notification system should follow the data dictionary set up at an organisation level. This field level validation happens before a recipient is added to the notification system. If recipient information fails to adhere to the data dictionary of the organisation, then the recipient cannot be added to the organisation.

6.3.4. Add Recipient

- Recipients can be added directly via Notification System (mapped against each org [AMC] and sub-org).
    a. Ad-hoc
        i. Created directly on notification system
    b. Bulk Upload
        i. Excel addition
        ii. System integration where data is maintained on both notification system and external system in real time basis
        iii. Added through Webhook

6.3.5. Delete Recipient

- Recipients added through Notification System alone can be removed using the application
- Delete recipient operation on notification system can also be performed through webhook

6.3.6. Edit Recipient

- Recipients added through Notification System alone can be edited using the application
- Edit operation on notification system can also be performed through webhook

6.3.7. Recipient Group Management

Recipient Groups are nothing, but multiple recipients put together in a cluster so that the same template logic can be applied on them.

Create Recipient group

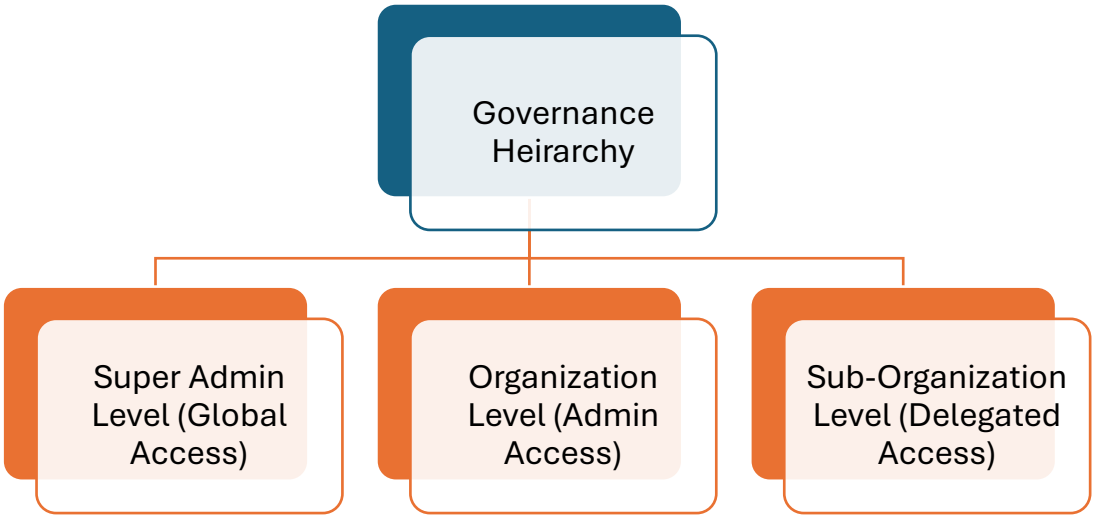
In order to create a recipient group, the user needs to add name and description

Add/Edit/Remove Recipients

Recipients can be added or edited in the recipient group by the following means

- Manual selection through sort and filter or query operation
- Webhook- External systems should be allowed to add/edit recipients from a group so that they can manage the recipient groups also based on event.

6.4. Governance Structure



Provisioning super admins for the notification service will be through a well-defined SOP approved by Project Odin Program Director. Super-Admins will have global oversight and unrestricted access to all features, and therefore their provisioning needs to be highly sensitive. Other than approval mentioned above, there should be user identity verification through MFA, via Zero Trust policy. Super admins should be provisioned via centralized identity management system of the platform. These super-admin roles should be linked with specific policies to restrict unintended actions (e.g., configuring templates that violate SEBI regulations). Security Measurement for these super-admins should be through

1. Least privilege principle
2. MFA
3. Access time restrictions
4. Activity Monitoring

6.4.1. Super Admin Level (Global Access)

Responsibilities

- Manage and configure all organisational entities (e.g., CAMS, AMCs, Channel Partners, CAMSPay, AIF, KRA, Sterling, etc.,)
- Add/Remove organizational Admins and Users
- Define and manage global policies, such as notification types, templates, and compliance rules
- Oversee all access permissions across channels and organizations
- Monitor service provider integrations (Email, SMS, WhatsApp, IVR, Push, etc.,)

Capabilities

- Perform cross-organization template creation and approvals
- Access analytics and monitoring dashboards for all organizations
- Approve and audit templates in line with SEBI, and other regulators Acts.

6.4.2. Organization Level (Admin Access)

Organization Admins:

- Manage their specific organization's users, sub-organizations, and permissions
- Add/remove sub-organization admins and users
- Define differential access for specific notification channels (e.g., Email, SMS, WhatsApp)
- Approve messages and templates for their organizations
- Oversee compliance with organizational and regulatory requirements

Organization Users:

- Have restricted access based on assigned roles
- Can initiate and manage notifications based on pre-defined workflows
- Perform template, message creation and channel-specific approvals tasks as per permissions

6.4.3. Sub-Organization Level (Delegated Access)

Sub-Organization Admins

- Manage users and permissions within the sub-organization
- Define templates and messages specific to their sub-organization
- Approve or reject notifications in alignment with organizational policies

Sub-Organization Users:

- Limited permissions to execute specific notification tasks
- Access to approved templates and specific channels (e.g., IVR, Push Notifications, etc.,)

6.5. Analytics

Organisation admins should be able to see the performance of the Messages, templates on a dashboard. Some of the metrics are given below for reference.

- Confirmed Click Through Rate (total clicks/confirmed delivered) * 100%
- # of notifications Opened/ Not Opened
- # of notifications Reported.
- # of notifications Unsubscribed

6.6. Logging and Monitoring

Log Notifications: All sent notifications will be logged with details including:

- Notification ID
- Event ID
- Message ID
- Recipient
- Status
- Timestamp
- Retry Count
- Client ID
- Monitoring Dashboard: Admins can view notification statistics, including success and failure rates.
- Audit Trail for Message- Maintain an audit trail of who created, edited or approved along with timestamps.

6.7. System Alerts

System should have the ability to send notifications to admin and other designated users.

- Failed Delivery (Bounce/Error) Alert- Email needs to be shared with administrators
- API Integration Alert
- System Load Alert- High CPU, memory etc., usage
- Service Unavailable alert
- Alert for Queue Backlog not cleared or clearing slower than expected

6.8. Billing Management

- Pricing will be based on different tiers. Each tier will be associated with certain number of users, Messages, channel wise notifications, recipients, recipient groups.
- Payment Gateway integration is required to enable swift payments.
- Billing cycle should be configured at org level.

6.9. Security Features

- Data should be encrypted both at rest and transit
- File Encryption for attachments in emails, WhatsApp messages etc.
- Two Factor Authentication
- RBAC to be implemented
- Rate limiting and Throttling
- Capture Consent
- Capture Opt-Out
- Ability to delete target user's personal data based on request

7. Non-Functional Requirements

7.1. Availability

Type of Notification	Business Hours		Non-Business Hours	
	Time Period	Availability	Time Period	Availability
Transactional Notifications	24*7	99.99% ("four nines")	Not Applicable	Not Applicable
Promotional Notifications	9 AM to 9 PM, Monday to Saturday	99.8% ("two nines eight")	9 PM to 9 AM Monday to Saturday and Sunday	99% ("two nines")

- Ensure delivery within 2 seconds for all notifications
- Encrypt all sensitive transactions details in transit, in use and at rest
- Adhere to SEBI & DPDPA standards (This needs to be explicitly defined)
- Notifications should be immutable, should also be audit-ready, and have a success rate of 100% (transactional and Regulatory) and 99.9% for promotional.
- We should be able to handle as many notifications as there are # of investors folios on the platform (SEBI may ask to send a notification to all folio holders) – this should be highly scalable.
- The business hours of promotional notification have been defined based on TRAI's rules for SMS communication around promotional campaigns.
- There should be no planned deployment/downtime during business hours. If planned, then we need to send clear communication to all clients which may be impacted mentioning the downtime a week before.
- Reference calculation is added below for availability %

Availability %	Downtime per year	Downtime per month	Downtime per week	Downtime per day (24 hours)
90% ("one nine")	36.53 days	73.05 hours	16.80 hours	2.40 hours
95% ("one nine five")	18.26 days	36.53 hours	8.40 hours	1.20 hours
97% ("one nine seven")	10.96 days	21.92 hours	5.04 hours	43.20 minutes

98% ("one nine eight")	7.31 days	14.61 hours	3.36 hours	28.80 minutes
99% ("two nines")	3.65 days	7.31 hours	1.68 hours	14.40 minutes
99.5% ("two nines five")	1.83 days	3.65 hours	50.40 minutes	7.20 minutes
99.8% ("two nines eight")	17.53 hours	87.66 minutes	20.16 minutes	2.88 minutes
99.9% ("three nines")	8.77 hours	43.83 minutes	10.08 minutes	1.44 minutes
99.95% ("three nines five")	4.38 hours	21.92 minutes	5.04 minutes	43.20 seconds
99.99% ("four nines")	52.60 minutes	4.38 minutes	1.01 minutes	8.64 seconds

7.2. Compatibility

- Ensure that any new release maintains backward compatibility with existing API interactions and data formats, targeting a zero-compatibility break for existing users.

7.3. Maintainability

- Code Changes should be able to be deployed within 15 minutes, with no more than 1% of users affected during deployment.
- Maximum planned downtime for scheduled maintenance should not exceed 1 hour per week. The planned downtime should be on Sunday 4 AM-5 AM.
- All changes should undergo an impact analysis process that identifies potential risks and effects on existing functionalities and receive sign off from product before implementation.
- Mean time to recovery (MTTR) should be less than 15 minutes after a system failure.
- Logging and monitoring tools should be in place to alert on failures within 5 minutes.
- The backend support (via email) should be available (in real time) from 9 AM till 7 PM with an option for 24/7 support during emergencies and time periods required by customers. All queries or issues (shared through support email of CAMS) should be resolved within below time frame. Refer SLA document for further information.
    - Critical: 1-2 business hours
    - Major: 4-8 business hours
    - Medium: 16-24 business hours
    - Low: 24-40 business hours

7.4. Performance Efficiency

- The service should handle up to 50,000 Transactional notifications per second (concurrent requests) under peak load conditions, 30,000 transactional notifications per second during regular business hours and 20,000 transactional notifications per second during non-business hours without degradation in performance.
- The system must be able to scale dynamically to support 100% increase in processing concurrent requests (during critical periods or regulatory deadlines) while maintaining same consistent response time for 95% of requests for all notifications.

- The system should have less than 0.0001% errors for transactional notifications and 0.05% for promotional notifications. Errors could include time-outs, failed requests, delivery failures etc.,
- Failure should be logged with detailed diagnostic information for real-time troubleshooting.
- Notification delivery latency must be under 500 ms for transactional notification and 1 second for bulk regulatory and bulk promotional notifications.
- CPU and memory usage must not exceed 60% during normal operational conditions and should remain under 85% during business hours.
- Network latency should be maintained at under 50 milliseconds for all interactions between the client and server.

7.5. Web Performance Metrics

Metric	Definition	Range	
		Min	Max
Page Load Time	The time it takes for a web page to fully load in a browser after a user clicks a link or types in a URL.		Desktop: 200 ms Mobile: 200 ms
Time To First Byte	Amount of time it takes for a browser to receive the first byte of a file from a server after a request is made	100 ms	100 ms
Time To Last Byte	Amount of time it takes to receive the last byte of a response from a server after a browser sends a request	150 ms	150 ms
Start Render Time	Time it takes for a website or web app to load enough that a user can interact with the page. It's the time it takes for the DOM to complete and the window to load the event.	175 ms	175 ms
First Input Delay	The time from when a site visitor first interacts with a page to the time when the browser is able to begin processing event handlers.		Desktop: 50 ms Mobile: 50 ms

7.6. Portability

- The application must function effectively across the below specified web browsers and devices.
    - **Web Browsers:**
        - Support the latest stable versions and the previous two versions of:
            - Google Chrome (latest version- 129)
            - Mozilla Firefox (latest version- 131)
            - Safari (latest version- 17)
            - Microsoft Edge (latest version- 127)
    - **Mobile Browsers:**
        - Support the latest stable versions of:
            - Safari (latest version- 17)
            - Google Chrome (latest version- 129)

        - Mozilla Firefox
        - Microsoft Edge
    - **Devices:**
        - Ensure compatibility with devices that meet the following specifications:
            - **Desktop:** Windows 10 and Windows 11, macOS
            - **Mobile and Tablet:** Latest iOS, Android, Windows and Bhar OS versions, 4 versions before that as well.
    - Testing Requirement: Achieve 95% of success for test cases across all specified browsers and devices during quality assurance testing. There should not be any critical or major issues.

7.7. Reliability

- System must recover from failures without data loss. Back up should be taken automatically on a daily basis and manually before maintenance and deployment activities.
- The system should have an MTTF (Mean Time To Failure- Average time between failures) of at least 5000 hours (~6 months) of operation without failure during the first 12 months of launch and 1300 hours (~18 months) of MTTF post 12 months of launch.

7.8. Scalability

- The system should be able to automatically scale resources up or down within 5 minutes based on predefined thresholds for CPU or memory usage.

7.9. Security

- All sensitive data (e.g., user information, conversation logs) must be encrypted both in transit (using TLS 1.3 or higher) and at rest (using AES-256 encryption).
- System should produce audit trail logs as mentioned in BRD whenever required
- Multi-factor authentication needs to be enabled by requiring email address and OTP for access
- Ensure all communications are conducted over secure protocols (HTTPS) to protect data in transit
- Rate Limiting is defined within the system and is dynamic. The rate limiting constraints configured within the application should be adhered.

7.10. Usability

- At least 90% of users should complete tasks mentioned in BRD (such as logging in, creating a message, and mapping a message to a template) within 5 minutes without external assistance, measured through user testing sessions.
- Maintain a task success rate of at least 95% for common user interactions (e.g., creating a message, and mapping a message to a template
- Ensure compliance with WCAG 2.1 Level AA accessibility standards, achieving 100% compliance.
    - **Perceivable**

        - Text must have a contrast ratio of at least 4.5:1 against its background. Large text (18pt and larger) should have a contrast ratio of at least 3:1.
        - Text must be resizable up to 200% without loss of content or functionality. Users should be able to zoom in and still have the layout work.
        - If the same visual presentation can be achieved with text, images of text should not be used unless a specific exception applies (e.g., logos).
        - Content must be able to reflow to fit within a viewport of 320 pixels wide without loss of information or functionality.
        - User interface components and graphical objects must have a contrast ratio of at least 3:1 against adjacent colours.
    - **Operable**
        - Headings and labels must be clear and descriptive to help users understand the content structure.
        - Any interactive element must have a visible focus indicator, such as a border or change in color, to show which element is currently selected.
        - When an input field has a label, the label must be programmatically associated with the input, ensuring that assistive technologies can interpret it correctly.
    - **Understandable**
        - If parts of the content are in a different language, the language must be programmatically indicated.
        - Changes in context that result from user actions (like submitting a form) must not occur unexpectedly. Users should be informed about changes.
        - Navigation mechanisms that are repeated on multiple pages must be consistent across those pages.
        - Components that have the same functionality must be identified consistently across the site.
        - If an input error is detected and suggestions for correction are available, they must be provided to the user.
        - For web pages that require user input, users must be able to review and confirm before finalizing the submission (like voice-based search).
    - **Robust**
        - For all user interface components (like buttons, checkboxes, etc.), the name and role must be programmatically determined, and the current value must be exposed to assistive technologies.
        - Status changes that occur after user input must be programmatically determined and communicated to assistive technologies, such as notifying users of successful form submissions.

7.11. Compliance

- Maintain comprehensive audit trails (as mentioned in BRD) for all processes related to compliance certifications, retaining records for at least 6 years (5 years in active and 1 year in archive).

7.12. Localization

- The application must support multiple global languages (text) both within the message and on the application UI. Multiple global dialects (voice) should be supported within the voice channel.

- Users should be able to switch between available languages seamlessly, with at least 90% accuracy in translation and localization tasks verified through user feedback and testing.

7.13. Service Level Agreements (SLAs)

- All the performance metrics specified above must be met each month and quarter
- The backend support (via email) should be available (in real time) from 9 AM till 7 PM with an option for 24/7 support during emergencies and time periods required by customers. All queries or issues should be resolved within below time frame.
    - Critical: 1-2 business hours
    - Major: 4-8 business hours
    - Medium: 16-24 business hours
    - Low: 24-40 business hours

7.14. Business Continuity/ Disaster Recovery

- Recovery Time Objective: 5 minutes
- Recovery Point Objective: 0 seconds

8. Additional Features

The features have been added based on feedback from different stakeholders after they have reviewed the initial draft.

- By default, the organisation can have its own sender domain. For example, if CAMS RTA is an organisation, then the sender domain will be @camsonline.com.
- The sub-organisations should have their own sender domain. For example, if CAMS RTA is an organisation, HDFC Mutual Fund and SBI Mutual Fund are sub-organisations, then HDFC Mutual Fund and SBI Mutual Funds should be able to send emails through the Notification system with sender domains @hdfcmutualfund.com and @sbimutualfund.com.
- The attachment must have lossless compression
- The message should have option to include buttons and other interactive elements like feedback ratings via stars. The data from the interactive elements should be saved in the notification system.
- Notification system should be able to set passwords for attachments based on logic defined within the notification system or added along with recipient and other details in the template level. For example, if customer PAN is added as a field while recipient details are uploaded, then customer first name and PAN fields can be made as a combo for password. The password could be- ashokCJPRE71328

- The application must also be able to receive inbound communication through SMS, Email and other channels, be able to parse the information and share it to external systems or channels through an API call.
- The SMS and Whatsapp whitelisting APIs for particular message should be triggered only after the message is whitelisted. The status of whitelisting, reasons for rejection should be shared with the approver and also the message creator.
- Retry logic- The notification system can trigger a re-try only if it has received the status of a particular notification as 'Failed'. Until then, retry cannot be attempted.
- Hierarchical Routing Logic- Only one message will be shared with the recipient out of all messages that are present on the template based on the chosen routing logic.
- There should be two messaging pipelines on each channel. One for critical messages like OTP and one for promotional notifications. OTP messages need to reach the recipient as early as possible (less than 3 seconds) whereas promotional notifications can be put in a queue and processed. Ideally, the system should be designed in such a way that all the notifications reach the recipient in less than 3 seconds.
- The notification system should not send more than one notification through a particular channel to a particular investor for a single 'send' operation.
- Default number of fail-overs is 2.
- System should be able to predict and share the approximate cost to the admin before they initiate or while they set up a promotional notification. In the case of transactional notifications, the admin will be able to see the approximate cost per transaction.
- System should also have the ability to let users estimate costs for 'n' notifications (for both transactional and promotional notifications) using a real-time calculator