SRS

Technical system requirements (Software Requirements Specification)

The comprehensive technical document for architectural design and code structure

Engineering & Operations Team (WSS)

Technical Application Profile

Project : NotificationPublisher Ecosystem

Paths (Endpoints) : Email (Single/Bulk), SMS, WhatsApp, FCM

Data protection (Secrets) : HashiCorp Vault KV v2

Implementation stage : Production - OKE Cluster

Core Tech : ASP.NET Core 8.0, RabbitMQ

1. API Ecosystem Requirements

The functional requirements are detailed to cover five NotificationPublisher service pipelines, configured to run asynchronously via RabbitMQ.

- Email path: To support sending a single email with attachments (Attachment Base64) and specifying Priority, CC, and BCC.
- Emails path: to support bulk operations by receiving the List<EmailPublisher> array
- SMS Path: Receive a comma separated phone group as Receiver with support for automatic segmentation and cost calculation
- WhatsApp Path: Sends Templates via Meta Business interfaces with support for attachments via S3
- FCM Path: Send smart device notifications via Firebase Cloud Messaging

2. Security & Authentication system

All requests must be authorized using the [Authorize] header. No summons works anonymously.

Level 1: API authentication (for external systems)

- Required data: Username:Password:AppId (three parts separated by colons)
- Encoding: Convert text to Base64
- Header: Authorization: Basic [Token]
- Also supports: X-Authorization: Basic [Token] (for environments that rewrite the Authorization header)
- Data is verified against the database via IUserService.Authenticate()

Level 2: Control Panel Authentication (UI Admin)

- Required data: Username:Password (two parts only)
- Verification is done against credentials stored in appsettings.json
- Grants full Admin privileges with ClaimTypes.Role = Admin

```
// BasicAuthenticationHandler.cs
[Route("api/v{version:apiVersion}/[controller]")]
[ApiController]
[Authorize]
public class NotificationController : ControllerBase {
    private readonly IPublisherOperations publisherOperations;
    public NotificationController(IPublisherOperations ops) {
        this.publisherOperations = ops;
    }
}
```

3. Per-Channel Functional Requirements

Email channel

- Recipient: Receiver (mandatory), EmailCC (carbon copy), EmailBCC (Bcc)
- Content: Subject (mandatory), Message (mandatory, supports full HTML)
- Priority: MsgPriority (integer for SMTP headers)
- Attachment: An array of MsgAttachment containing Base64, filename and MIME type
- Batch mode: /Emails dot accepts an EmailPublisher array with independent processing for each message

SMS channel

- Replace Sender: If Sender is empty, the sender name registered in the database is used
- Multiple numbers: Comma-separated numbers that are automatically split and verified
- Message split: SMS = 70 characters. Longer messages are divided into 67-character segments
- Carrier Factor: Specific carrier numbers (57) carry a cost factor of 4 times
- Quota check: Cost = number of parts x carrier factor x number of receivers

WhatsApp channel

- Meta Templates: When using WhatsAppProviderId=Meta, Meta Business API compatible JSON is built
- Header Components: Supports Text, Document, Image and Video
- Text variables: BodyParams (comma separated) bound to placeholders (1, 2...)
- URLButtons: UrlParam is injected into button index 0
- File protection: Attachments are uploaded to S3 and served via signed HMAC links

FCM Notices Channel

- Payload: Standard Firebase Cloud Messaging architecture
- No quota check - Firebase manages rate control internally
- Setup code: Returns PushNotificationSettingNotActive if the channel is disabled

4. Response Codes

The system returns a unified ServiceResponse code showing the status of the process:

Description	The constant	Code
Submitted successfully and placed in queue	Success	100
The quota is not enough	ExpireQuoata	11
The application is not working	AppNotActive	15
SMS setting is not working	SMSSettingNotActive	16
Mail setting is inactive	EmailSettingNotActive	17

WhatsApp setting is not working	WhatsAppSettingNotActive	18
Notification setting is inactive	PushNotificationSettingNotActive	19
Internal system error	Exception	-1
Failed to save message	FailDBSaved	-2
The application does not exist	NotFound	-3
Status update failed	FailDBUpdate	-4
Failed to send to RabbitMQ	PushFail	-5
There is no connection to the message server	ConnectionNotExists	-6
The order data is incorrect	BadParameter	-7

5. Excel Bulk Import Requirements

- Template download: .xlsx files pre-formatted for each channel type
- Salutation logic: If the client does not specify a greeting, the system queries the database
- Default Arabic: Dear/Dear | English default: Dear
- Row-by-row processing: Invalid rows log an error and do not stop the batch
- Import Report: Returns TotalRows, SuccessCount, and FailedCount