HLD

Advanced architectural design (High Level Architecture)

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. Microservices Architecture

Routing responsibilities (Notification-Api) are separated from shipping functions (EmailConsumer, SMSConsumer, WhatsAppConsumer). The RabbitMQ broker connects each layer.

Architectural services roles

Notification-Api (Publisher): The single entry point for the REST API. It receives requests, authenticates clients, and sends to RabbitMQ

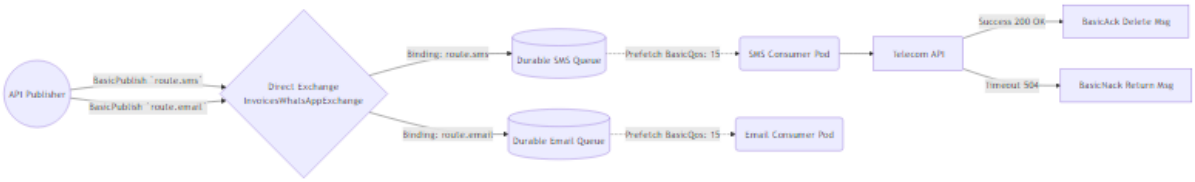
- Background Consumers: .NET 8.0 Worker Service services are standalone without open HTTP ports
- WhatsAppConsumer: Pulls from WhatsApp queue and communicates with Meta Business API
- SMSConsumer: Pulls from the SMS queue and communicates with the SMS gateway
- EmailConsumer: Pulls from mail queue and sends via SMTP
- PushConsumer: Pulls from FCM queue and sends via Firebase

API Controllers Inventory

Responsibility	Track	Controller
Basic message dissemination	/api/v1/Notification	NotificationController
Manage applications, providers and templates	/api/v1/Settings	SettingsController
Message history and statistics	/api/v1/Messages	MessagesController
Bulk import via Excel	/api/v1/Excel	ExcelController
Secure proxy to download S3 files	/api/FileDownload	FileDownloadController
Post public messages	/api/v1/Publisher	PublisherController
Control panel authentication	/api/v1/UIAuth	UIAuthController

2. RabbitMQ (Message Broker Architecture)

To ensure reliable, loss-free routing even when consumers are offline:

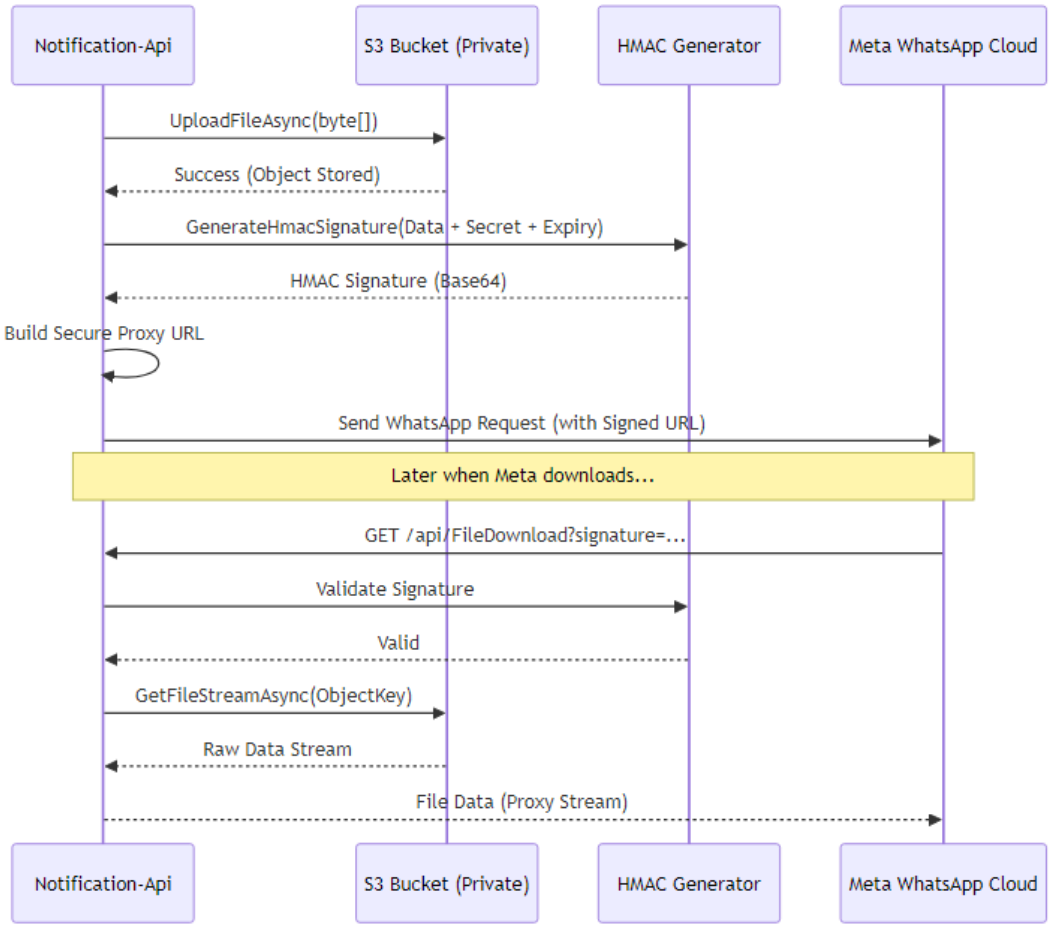


RabbitMQ routing scheme with Direct Exchange

- Exchange Strategy: Direct Exchange - directs the message precisely to the designated queue
- Queues are durable: they survive a server restart
- Lazy Queue mode: Messages are transferred to disk immediately to reduce memory consumption
- BasicQos: Each consumer pulls a maximum of 15 messages at a time
- Publisher Confirms: Publisher confirmation is waited for 5 seconds before updating the database
- Ack/Nack: The message is deleted only after success is confirmed. When you fail, you return to the queue

3. S3 cloud storage architecture (Object Storage)

The system manages large binary assets (PDF, images, video) via Oracle Cloud S3-compatible Object Storage.



S3 Storage Lifecycle: Upload -> HMAC Signing -> Secure Download

HMAC Signature Security Model

- Signature: WhatsAppService.GetFileLink() calculates HMAC-SHA256(path + ?expires= + timestamp, SecretKey)
- Safe encoding: Base64 with + replaced by - and / with _ and = removed
- Verification: FileDownloadController recalculates HMAC and compares. Mismatch = 403 Forbidden
- Expiry: If the current time exceeds expires returns 410 Gone

4. Data protection via HashiCorp Vault

The design completely prohibits the presence of manual appsettings.json files in the code. Vault Agent Sidecar is implemented.

- Sidecar Intercept: When a Pod is started, a temporary Vault Agent (Init Container) is deployed before the .NET container
- Retrieving keys: vault kv get -mount=notifications-gateway api/dev
- Empty volume: Creates appsettings.Development.json in a shared memory volume (emptyDir)
- .NET consumption: The application container reads the injected file without knowing its source

```
entrypoint: >
  sh -c 'apk add --no-cache jq &&
  vault kv get -mount=notifications-gateway \
      -format=json notification-api/development \
      | jq -r .data.data.config \
      > /api-configs/appsettings.Development.json'
```

5. API Versioning & Swagger

- URL-Segment Versioning: /api/v{version}/[controller]
- Default version: v1 with AssumeDefaultVersionWhenUnspecified = true
- ReportApiVersions: Reports supported versions in response headers
- Swagger protected: SwaggerBasicAuthMiddleware forces authentication before access
- XML Comments: An XML documentation file is loaded to display the description of the Endpoints in Swagger

6. Thread Safety & Connection Pooling

PublisherCore maintains a static connection to RabbitMQ shared between all threads:

```
private static readonly object _lock = new object();
private static IModel _channel;
private static IConnection _connection;

private bool ConnectionExists() {
    lock (_lock) {
        // 1. Hot Path: both alive -> return
        // 2. Channel Recovery: recreate channel only
        // 3. Full Reconnect: new connection + channel
    }
}
```

- Fast Track: Both alive -> Return immediately
- Recover Channel: Connection is alive but channel has been closed -> Just create a new channel
- Full reconnection: with AutomaticRecoveryEnabled = true and a timeout of 10 minutes