Good Software

Technical Specification

Customer 360 Lifecycle Management

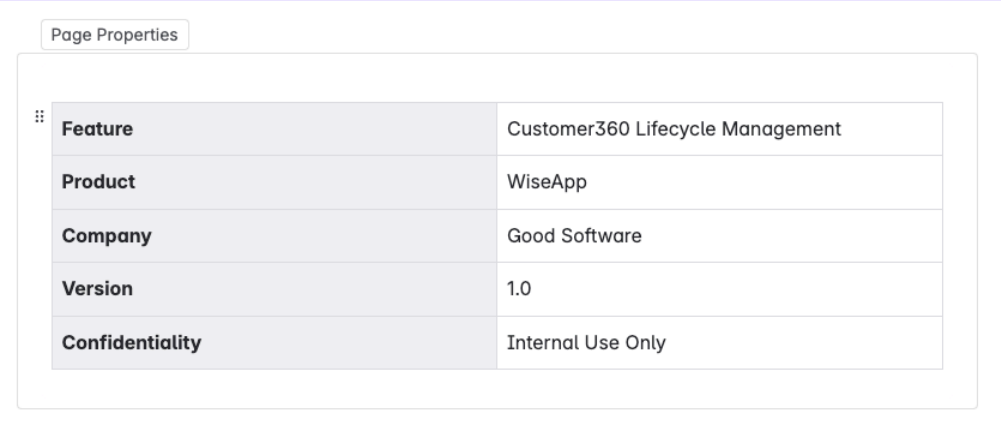
Internal Use Only

Version: 1.0

Last update: July, 2025

This page is not part of the template file and won't affect your exports

- To get the most of this Scroll PDF Exporter template: set up your brand and your logo and brand colors will be used automatically.
- Want to have more control over how your Confluence content is exported? Take a look at Scroll Extensions. With these macros you can insert page breaks, ignore content from your export, and more directly in your Confluence content.
- This template uses custom placeholders that are set by the content/page properties from the Confluence page template with the keys 'Feature', 'Company', 'Confidentiality' and 'Version'. Image attached below.
- Have suggestions on how to improve our templates or a use case we have not covered? Contact us at support@k15t.com



Page properties used inside this template

Table of Contents

Overview

Customer360 Lifecycle Management empowers B2B organizations using WiseApp to manage customer information from onboarding through post-sales activities. It consolidates data from multiple sources into a unified customer profile and tracks key lifecycle stages such as acquisition, engagement, renewal, and support.

Objective

- Provide a centralized and consistent view of each customer.
- Enable seamless transitions across lifecycle stages.
- Facilitate data-driven decision-making for customer success and account teams.
- Ensure compliance with data protection regulations (e.g. GDPR).

Stakeholders

Role	Name	Responsibility
Product Owner	Jane Doe	Feature definition & prioritization
Engineering Lead	Alex Smith	Technical feasibility & architecture
UX Designer	Rina Patel	UI/UX flows
QA Engineer	Carlos Nguyen	Test planning & execution

Requirements

Functional Requirements

ID	Description
FR-01	Sync customer data from CRM, Billing, and Support modules
FR-02	Allow lifecycle status updates via UI and API
FR-03	Display current lifecycle stage and summary metrics in Customer360 tab
FR-04	Support custom lifecycle stage definitions per tenant
FR-05	Log all stage transitions with timestamp and user ID

Non-functional Requirements

- Response time under 500ms for lifecycle summary API.
- System must support 50,000 customer records per tenant.
- Data must be encrypted at rest.
- Full audit logging of stage changes.
- RBAC enforcement for lifecycle updates.

Use Cases

Use Case 1: Customer Success Management

A Customer Success Manager logs into WiseApp and checks which customers are currently "At-Risk" to prioritize outreach.

Use Case 2: Lifecycle Integration via API

An external workflow tool uses the lifecycle update API to automatically mark customers as "Renewed" when their annual contract is signed.

Architecture & API Design

Architecture

Overview

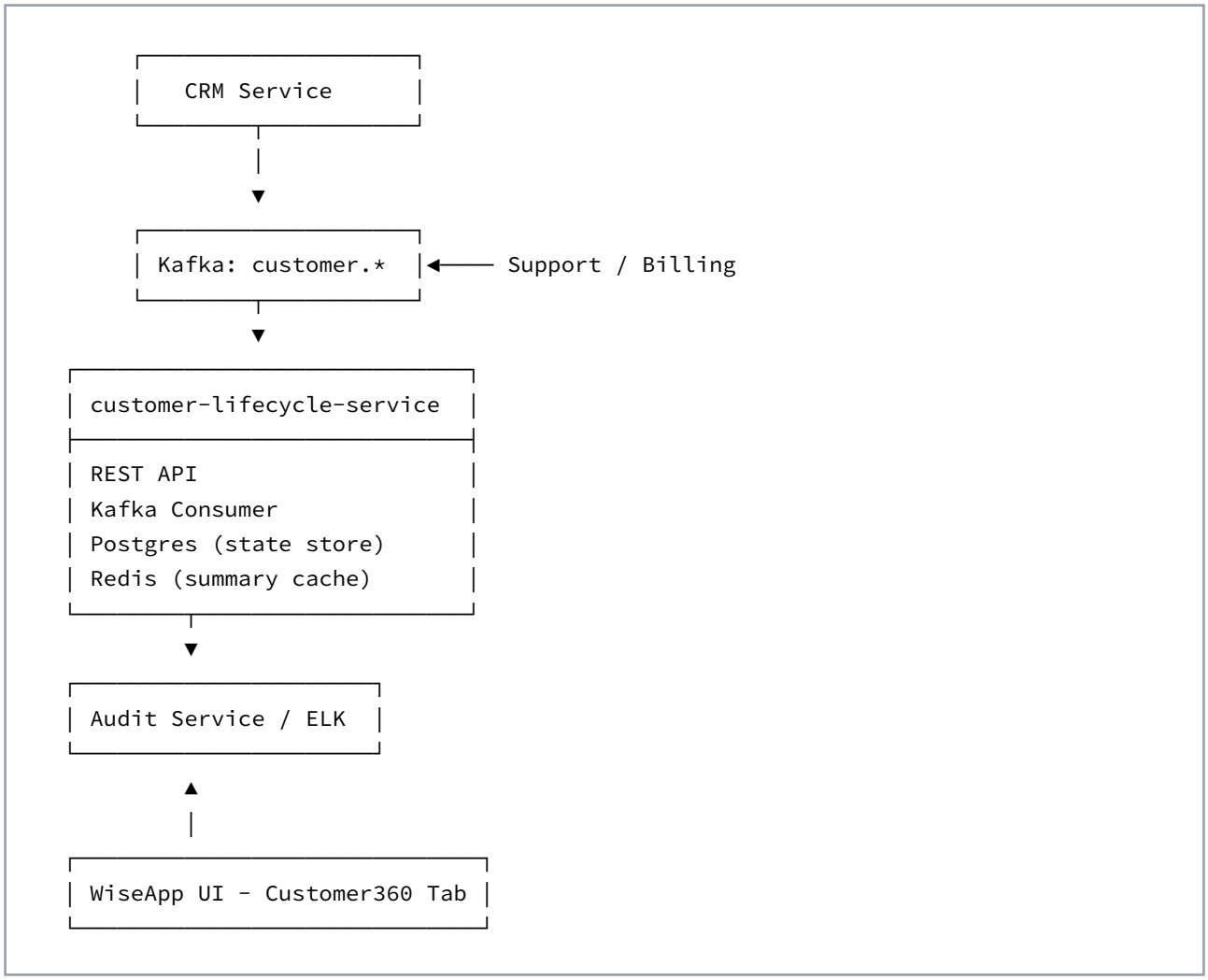
The **Customer360 Lifecycle Management** feature introduces a new microservice that centralizes customer lifecycle tracking across CRM, Billing, and Support systems. The architecture follows an **event-driven**, **modular**, and **secure-by-default** approach.

Components

1. `customer-lifecycle-service` (New Microservice)
    a. Responsible for:
        i. Managing lifecycle state transitions
        ii. Serving API requests related to customer status
        iii. Listening to domain events (e.g. customer created, contract renewed)
        iv. Built with: **Node.js (Express)** or **Java Spring Boot**
        v. Connected to a PostgreSQL database and Redis for caching
2. Message Queue Integration
    a. Uses **Kafka** topics for event-driven communication:
        i. `customer.created`
        ii. `customer.updated`
        iii. `contract.renewed`
        iv. `support.issue.closed`
3. Redis Cache
    a. Stores recent lifecycle data for fast retrieval in UI
    b. TTL: 5 minutes for summary views, 24 hours for full profiles
4. Audit & Logging Layer
    a. Centralized logging using **ELK stack**
    b. Lifecycle changes are sent to an audit.lifecycle.updated Kafka topic

---

Diagram (ASCII-friendly)



API Specifications

Update Lifecycle Stage

Endpoint:
`PATCH /api/v1/customers/{id}/lifecycle`

Request Payload:

```
{
  "stage": "At-Risk",
  "updated_by": "user-42"
}
```

Response:

```
{
  "success": true,
  "updated_at": "2025-07-28T12:34:00Z"
}
```

Errors:

- `400 Bad Request` (invalid stage)
- `403 Forbidden` (insufficient permissions)
- `500 Internal Server Error`

Data Model & Security

Data Model

The **Customer Lifecycle Status** is modeled in a dedicated table to track the current lifecycle stage, historical changes, and related metadata. This structure supports tenant-level customization and full auditability.

Field	Type	Description
`id`	UUID	Unique identifier (primary key)
`customer_id`	UUID	Foreign key to `customers` table
`tenant_id`	UUID	Foreign key to `tenants` table
`current_stage`	ENUM	Current lifecycle stage (custom per tenant)
`stage_entered_at`	TIMESTAMP	Timestamp when the current stage was entered
`updated_by`	UUID	ID of user who performed the last update
`last_updated`	TIMESTAMP	Last update timestamp
`reason_code`	VARCHAR(255)	Optional system or user-defined reason for transition
`is_manual_update`	BOOLEAN	Indicates whether the update was user-driven or system-driven

Customer lifecycle status

> **Note:** A historical table for lifecycle transitions is also maintained for auditing purposes.

Field	Type	Description
`id`	UUID	Primary key
`customer_id`	UUID	Reference to customer
`from_stage`	ENUM	Previous lifecycle stage
`to_stage`	ENUM	New lifecycle stage
`changed_at`	TIMESTAMP	When the transition occurred
`changed_by`	UUID	User or system ID responsible
`reason_code`	VARCHAR(255)	Optional transition reason
`source`	ENUM	'UI', 'API', 'EventListener', etc.

Security & Permissions

Access Control

Role	View Lifecycle	Update Lifecycle	View History	Configure Stages
Customer Viewer		×	×	×
Account Manager				×
Tenant Admin				
Super Admin (Internal)				

Permission Checks

Lifecycle operations are secured using **JWT-based role verification** and **tenant-bound access filters**.

```
function hasLifecyclePermission(user, action) {
  return user.roles.includes('ROLE_ACCOUNT_MANAGER') &&
         user.tenant_id === customer.tenant_id;
}
```

- **Lifecycle updates** must validate:
  - User has permission for the customer's tenant
  - Stage transition is allowed (based on tenant config)
  - Reason (if required) is present

Data Protection

- **Encryption at Rest**: AES-256 encryption for lifecycle tables
- **Encryption in Transit**: All APIs served via HTTPS only
- **Rate Limiting**: API Gateway limits updates to 1000 per user per hour
- **Audit Logging**: All lifecycle transitions are written to an immutable audit store

> **Warning: Lifecycle Status May Trigger Business Processes**
>
> Changing a customer's lifecycle stage may automatically trigger business logic such as:
>
> - Renewal campaign emails
> - Escalation alerts for "At-Risk" customers
> - Churn forecasting updates
>
> Please ensure all updates are intentional and authorized.

Acceptance Criteria & Testing Notes

Acceptance Criteria

- Users with correct permissions can update lifecycle stage via UI and API
- Updated stage reflects in UI within 2 seconds after API call
- Only predefined tenant stages are allowed (no free-text stages)
- Every lifecycle change is logged in `customer_lifecycle_history`
- System gracefully handles simultaneous updates (concurrent modification)
- UI clearly shows the current stage and last change date
- Caching layer is refreshed automatically after lifecycle updates
- Reason field is required when moving to "At-Risk" or "Churned"

Testing Notes

Unit Tests

- Validate stage transitions: allowed vs. disallowed
- Ensure data validation (e.g., ENUM values, UUID formats)
- Test reason code enforcement logic

Integration Tests

- API permission tests for different roles
- Test full lifecycle from CRM event to UI update
- Verify audit trail is recorded correctly

Performance Tests

- Test 100,000 lifecycle transitions in under 5 minutes
- Ensure Redis cache invalidation happens on write
- Simulate burst update traffic to validate rate-limiting

Regression Scenarios

- Updates to lifecycle should not affect unrelated customer modules (CRM, Billing)
- Transition logic should remain consistent after tenant config changes

Appendix & Glossary

Glossary

Term	Definition
Lifecycle Stage	A label describing where a customer stands in their business journey (e.g., "Active", "At-Risk").
Tenant	A customer organization using WiseApp (multi-tenant SaaS model).
Reason Code	A short label explaining why a lifecycle stage was changed (e.g., "Low Engagement").
Manual Update	A change triggered by a user vs. automatic system event.

Custom Lifecycle Configuration (Tenant Admins)

Admins can configure available lifecycle stages via the Admin UI:

```
{
  "stages": [
    "Prospect",
    "Onboarded",
    "Active",
    "At-Risk",
    "Churned"
  ],
  "required_reason_codes": ["At-Risk", "Churned"]
}
```

Sample API Token Permission

```
{
  "token": "abc.def.ghi",
  "roles": ["ROLE_ACCOUNT_MANAGER"],
  "tenant_id": "tenant-789",
  "expires": "2025-12-31T23:59:59Z"
}
```

Wireframe & UX Link

- Figma Link – Customer360 Tab Wireframe

Good Software