A white sign with black text

Description automatically generatedA black and red square with a black background

Description automatically generated

**CONCEPT DOCUMENT**

**CUSTOMER SUPPORT SYSTEM**

**Confidentiality:**

This document is owned by New Street Technologies Private Limited. The contents of this document are strictly Private & Confidential. The reader of this document acknowledges and agrees that all contents of this document belong to the owner of this document or his/her nominees and hereby expressly agrees not to use any of the contents in part or whole thereof for any other purpose than intended by the owner of the document. Any unintended recipient of this document is obliged to destroy any soft and or hard copies of this document and notify its owner at the earliest.

**DOCUMENT CONTROL**

**Distribution**

When finalised, this document should be distributed and signed off by the following stakeholders:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Company** | **Function** | **Email ID** | **Sign off Provided (Y/N)** |
| Sajeev Viswanathan | New Street | Product | sajeev@newstreettech.com |  |
| Shiyas Babu | New Street | Product | shiyas.babu@newstreettech.com |  |
| Shrish Lal | New Street | Business Development | shrishl@newstreettech.com |  |
| Vincent Lukas | New Street | Business Development | vincent.lukas@newstreettech.com |  |
| Sandeep Mina | Shecommerz | Client | sandeep.mina@shecommerz.com |  |
| Kishore Gowtham | Shecommerz | Client | kishore@newstreettech.com |  |

**CHANGE HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Change Summary | Change description |
| 20-03-2025 | v1.0 | Initial Document |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[**1. Introduction** 2](#_Toc193391373)

[**1.1 Purpose** 2](#_Toc193391374)

[**1.2 Scope** 2](#_Toc193391375)

[**2. Business Objectives** 3](#_Toc193391376)

[**3. Key Stakeholders** 3](#_Toc193391377)

[**4. Functional Requirements** 3](#_Toc193391378)

[**4.1 Dashboard for Failed APIs** 3](#_Toc193391379)

[**4.2 Search & Filter** 3](#_Toc193391380)

[**4.3 Failure Details** 3](#_Toc193391381)

[**4.4 Actions & Resolution** 3](#_Toc193391382)

[**4.5 Notifications & Alerts** 4](#_Toc193391383)

[**4.6 Reporting & Analytics** 4](#_Toc193391384)

[**5. Non-Functional Requirements** 4](#_Toc193391385)

[**6. Assumptions & Constraints** 4](#_Toc193391386)

[**7. Success Metrics** 5](#_Toc193391387)

[**8. Future Enhancements with AI/ML** 5](#_Toc193391388)

**Customer Support Management System for Failed APIs**

# **1. Introduction**

## **1.1 Purpose**

The purpose of this document is to define the business requirements for a Customer Support Management System designed to track and take action on failed API transactions for each processed customer. This system will enable customer support teams to view API failures, analyze error details, and initiate appropriate resolutions.

## **1.2 Scope**

This system will focus on monitoring API failures, providing real-time insights, and enabling corrective actions. Key functionalities include viewing failure logs, retrying failed requests, notifying relevant teams, and tracking resolution status.

# **2. Business Objectives**

* Improve response time for handling failed API transactions.
* Reduce customer complaints related to unresolved API failures.
* Provide transparency in error tracking and resolution.
* Enhance operational efficiency of customer support teams.

# **3. Key Stakeholders**

* **Customer Support Representatives** – Monitor and act on failed APIs.
* **Technical Support/Developers** – Analyze issues and resolve complex failures.
* **IT Operations** – Maintain system health and ensure uptime.
* **Customers (Indirectly Affected)** – Impacted by unresolved failures.

# **4. Functional Requirements**

## **4.1 Dashboard for Failed APIs**

* A real-time list of failed API transactions categorized by severity (Critical, High, Medium, Low).
* Quick view of failure trends and pending actions.
* **Individual API Level Information** – Display failure counts per API, categorized by failure type (e.g., timeout, authentication error, data mismatch, etc.).
* **Historic Analysis & Trend Visualization** – Track failure patterns over time for each API, including graphs and charts to visualize trends.
* **Comparison Metrics** – Show percentage increase/decrease in failures compared to previous periods.

## **4.2 Search & Filter**

* Ability to search by **customer ID, API name, error type, timestamp, status**.
* Filters for severity, resolution status, and assigned support agent.

## **4.3 Failure Details**

* View detailed error messages, timestamps, request and response logs.
* Drill-down option for analyzing root cause.
* **Copy & Download API Payload** – Users must have the option to copy and download the API request and response payloads for further analysis.

## **4.4 Actions & Resolution**

* **Retry API Call** – Manual and automated retry options.
* **Trigger Enquiry** – Initiate a request to update the API response.
* **Notify Developers** – Raise tickets to the technical team.
* **Raise Ticket to Concerned Stakeholder** – Create and assign a ticket for resolution by the appropriate team.
* **Manually Resolve Issue** – Mark failures as resolved with notes.
* **Assign to Support Agent** – Allocate failed APIs to specific personnel.

## **4.5 Notifications & Alerts**

* Automated alerts for critical API failures.
* Email/SMS notifications for assigned support agents and developers.
* Escalation alerts if an issue is not addressed within a defined SLA.
* **API Rate Limit Monitoring** – The system should track and alert users when API rate limits (daily or hourly) are nearing or have been exceeded.

## **4.6 Reporting & Analytics**

* Track failure trends and resolution time.
* Generate reports on API performance and failure causes.
* **Historical Trend Analysis** – View failure trends over weeks, months, and quarters.
* **API-Specific Reports** – Breakdown of failures per API with detailed analysis.
* Export reports in multiple formats (CSV, PDF, etc.).

# **5. Non-Functional Requirements**

* **Scalability** – Handle large volumes of API failures efficiently.
* **Security & Access Control** – Role-based access for customer support and technical teams.
* **Audit & Logging** – Maintain a history of API failures and resolution actions.
* **Performance** – Ensure real-time updates and minimal system lag.

# **6. Assumptions & Constraints**

* The system will integrate with existing API gateways and monitoring tools.
* API error logs will be stored securely with retention policies.
* The solution should comply with data protection regulations.

# **7. Success Metrics**

* Reduction in unresolved API failures by at least 30%.
* Decrease in average resolution time by 40%.
* Increased customer satisfaction due to faster issue resolution.
* Improved support team efficiency and reduced manual interventions.

# **8. Future Enhancements with AI/ML**

To further enhance the efficiency and intelligence of the system, future iterations may incorporate AI/ML capabilities, including:

* **Automated Decision-Making** – AI-driven recommendations on the appropriate resolution actions for failed APIs based on historical patterns and failure types.
* **Predictive Analysis** – Forecast potential API failures before they occur and suggest proactive resolutions.
* **Smart Alerts & Escalations** – Machine-learning-based prioritization of critical issues and automated escalation workflows.
* **Anomaly Detection** – Identify unusual failure patterns and proactively suggest corrective actions.
* **Auto-Triage & Resolution** – AI-powered categorization of failures and automated initiation of corrective measures.