SwiftKart - Business Requirements Document (BRD)

Project: Reduce Failed Deliveries in Bangalore
Date: 18-Aug-2025
Prepared by: Sajit (Business Analyst)
Version: v1.0
Status: Draft

BRD Summary

SwiftKart's Bangalore quick-commerce operations face a failed delivery rate of 11% and an on-time performance of only 82%. Failed deliveries result from poor address quality, unreachable customers, and the absence of a structured rescue process. These issues lead to revenue loss, high refund costs, and a negative customer experience.

The objective of this initiative is to reduce failed deliveries to 4% and increase on-time deliveries to 95% within 60 days. The improvement will begin with three high-volume Bangalore pincodes: 560034 (Koramangala), 560038 (Indiranagar), and 560102 (HSR).

The project will focus on five priority areas:

- Address verification at checkout
- Pre-arrival rider contact
- Rescue and reattempt rules
- Standardized failure proofing
- Audit compliance

Pricing, promotions, catalog, and warehouse design are excluded from this scope.

Success Criteria: Sustained results in a two-week pilot — failure rate ≤4%, on-time ≥95%, at least 90% of high-rise orders with gate codes captured, and complete auditable logs.

Sponsor: City Operations Manager (Bangalore)

Stakeholders & RACI

Key Stakeholders

- **Sponsor / Accountable**: City Ops Manager (Bangalore)
- **Core Team**: Product Manager, Engineering Lead, Dispatch Supervisor, Rider Ops Lead/Trainer, CX Lead/QA
- **Supporting**: Legal/Privacy, Finance
- **Informed**: Store/Hub Leads (pilot hubs), Rider Partners (pilot areas)

Role	City Ops Manager	Product Manager	Engineering Lead	Dispatch Supervisor	Rider Ops Lead	CX Lead	Legal/Privacy	Finance	Store/Hub Leads	Riders
Overall Project	A	R	R	C	C	C	I	I	I	I
Business Rules	A	R	C	R	C	C	C	I	I	I
Process Changes	A	C	C	R	R	C	I	I	I	I
UAT Execution	A	C	C	R	R	R	I	I	I	R
Sign-off	A	C	C	C	C	C	I	I	I	I

Scope & Boundaries

In-Scope

- Address verification at checkout (capture flat no., gate code, landmark).
- Pre-arrival rider contact (IVR, masked call/SMS).
- Rescue and reattempt rules (structured workflow for failed attempts).
- Standardized failure proof (photo, call log, reason codes).
- Audit compliance (digital logs retrievable and verifiable).
- Pilot in three Bangalore pincodes: 560034 (Koramangala), 560038 (Indiranagar), 560102 (HSR).

Out-of-Scope

- Pricing, promotions, discounts.
- Product catalog and assortment.
- Warehouse / dark store design.
- Expansion outside Bangalore pilot areas during initial phase.
- Long-term loyalty/rewards programs.

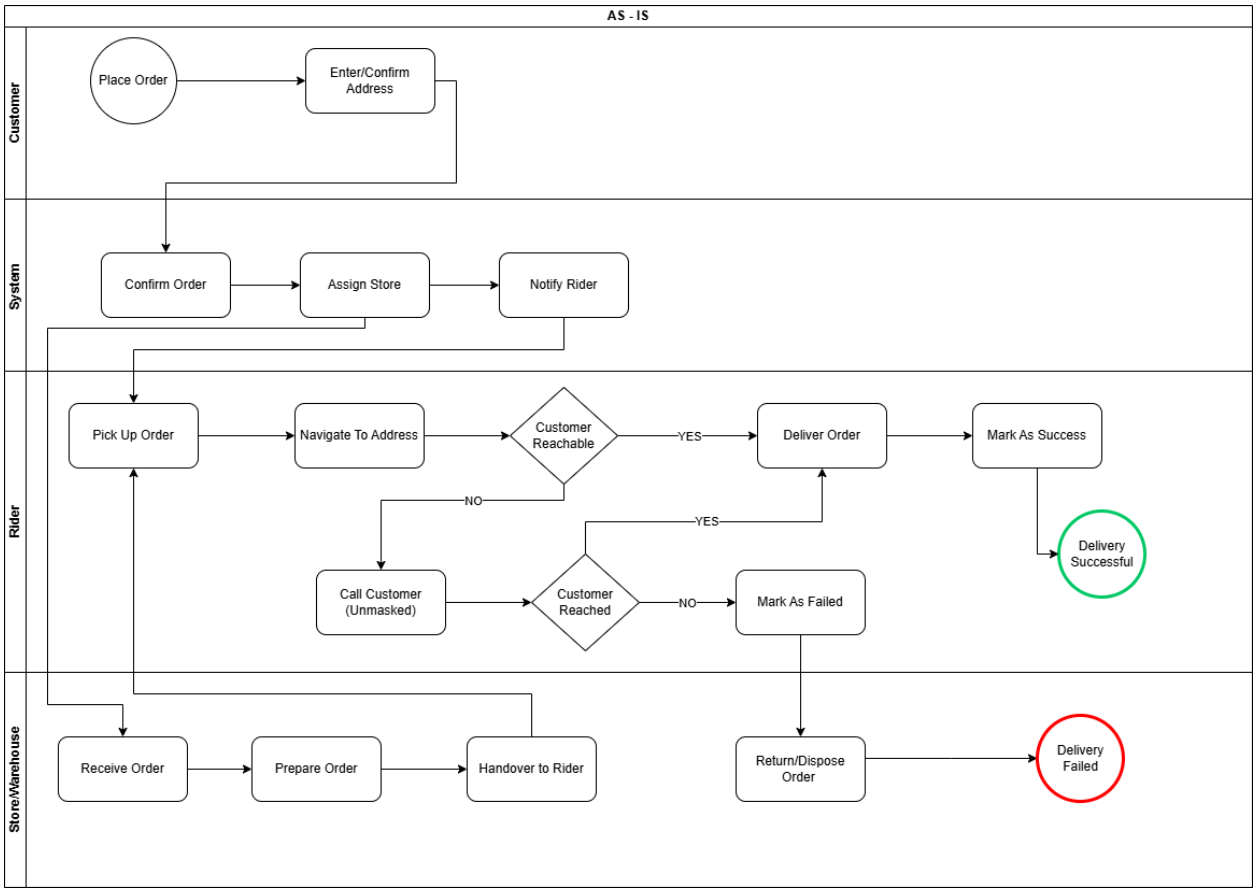
Scope Statement

This project focuses exclusively on reducing failed deliveries and improving on-time performance in selected Bangalore pilot areas. It does not extend to pricing, catalog, or warehouse operations, which are handled by separate initiatives.

Business Process Overview (As-Is)

SwiftKart's current last-mile delivery process in Bangalore operates as follows:

1. **Order Placement** – Customer places an order in the SwiftKart app.
2. **Stock Check & Confirmation** – System verifies product availability.
3. **Order Fulfillment** – Warehouse staff pick, pack, and hand over the order.
4. **Rider Assignment** – System allocates a rider to deliver the order.
5. **Delivery Attempt** – Rider travels to the customer's address.
6. **Completion** – If successful, the order is closed.
7. **Failure Handling** – If unsuccessful, the order is marked as failed with no structured rescue or reattempt rules.



Root Cause Analysis

SwiftKart's failed delivery rate of 11% and on-time performance of only 82% are driven by the following root causes:

1. **Poor Address Quality** – Incomplete or inaccurate customer inputs (missing flat number, gate code, or landmark).

2. **Customer Unreachable** – No structured pre-arrival contact; riders often cannot connect with customers.

3. **No Rescue Flow** – Failed attempts are immediately marked as failed with no retry or reassignment process.

4. **Inconsistent Proofs** – Lack of standardized failure documentation (photo, call log, or reason code).

5. **Audit Gaps** – Inability to retrieve and verify failure logs consistently for compliance.

Business Process Overview (To-Be)

SwiftKart's improved last-mile delivery process will operate as follows:

1. **Order Placement** – Customer places an order in the SwiftKart app.

2. **Stock Check & Confirmation** – System verifies product availability.

3. **Order Fulfillment** – Warehouse staff pick, pack, and hand over the order.

4. **Rider Assignment** – System allocates a rider to deliver the order.

5. **Pre-Arrival Contact** – Automated IVR/Masked call or SMS sent to confirm accessibility before rider leaves hub.

6. **Delivery Attempt** – Rider travels to the customer's address.

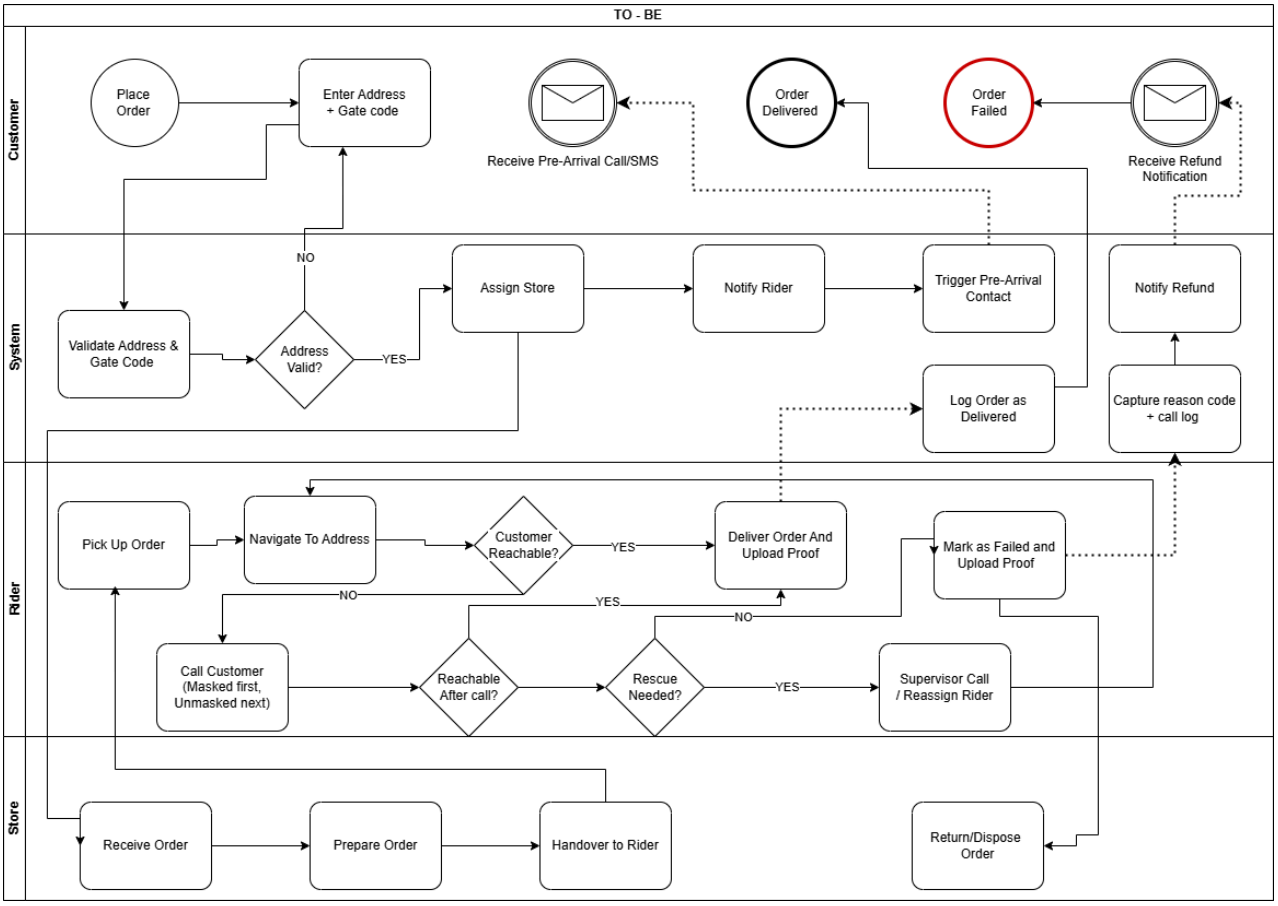
7. **Rescue Flow** – If first attempt fails, structured reattempt/rescue rules apply (e.g., second call, escalation to supervisor, or reassignment).

8. **Completion** –

    - If successful, the order is marked as **Delivered**.

    - If unsuccessful, order is marked as **Failed** with photo, call log, and reason code captured.

9. **Audit & Compliance** – All delivery outcomes are digitally logged, auditable, and retrievable.



Business Rules & Requirements

SwiftKart's improved delivery process will be guided by the following business rules and requirements:

Business Rules

1. **Address Validation** – Orders will not be assigned until the system confirms the address and gate code provided by the customer.

2. **Pre-Arrival Contact** – A masked call or SMS must be sent to the customer before the rider reaches the delivery point.

3. **Customer Contact Attempts** – If a customer is not reachable, the rider must try at least twice (masked call first, then unmasked call).

4. **Rescue Escalation** – If the customer still cannot be reached, the rider must escalate through the app to trigger a supervisor call or rider reassignment before marking the order as failed.

5. **Proof of Delivery/Failure** – Riders must upload proof for every outcome: Delivered: photo. Failed: door/gate photo, call log, or reason code.

6. **Refund & Customer Notification** – Failed orders must automatically trigger a refund process, with the customer receiving an SMS/app notification.

7. **Audit & Compliance** – Every delivery outcome (success or failure) must be logged with timestamp, rider ID, and proof file for compliance tracking.

Functional Requirements

1. System validates customer address and gate code before allocating orders.

2. System automatically sends pre-arrival SMS/IVR once rider is en route.

3. Rider app provides both masked and unmasked call options.

4. Rider app enforces proof upload before allowing completion status.

5. Supervisor dashboard enables quick reassignment of riders in rescue cases.

6. On failure, system triggers automatic refund and notifies the customer.
7. All proofs, call attempts, and reason codes are stored in a retrievable audit log.

Non-Functional Requirements

1. Proof uploads and call attempt logs must sync within 10 seconds.
2. System uptime must be at least 99.5% during delivery hours.
3. Compliance records must be tamper-proof and stored for 6 months.
4. Notifications must reach customers within 1 minute of a status update.

User Stories + NFRs + Data Needs

User Stories

Customer

1. As a customer, I want my address validated during checkout so that deliveries don't fail due to wrong details.
2. As a customer, I want to receive a call/SMS before delivery so that I can be ready to accept the order.
3. As a customer, I want to get notified quickly if my order fails so that I know when the refund is processed.

Rider

1. As a rider, I want to see the customer's address clearly validated so I don't waste time searching.
2. As a rider, I want to make both masked and unmasked calls through the app so I can contact the customer if needed.
3. As a rider, I want to upload proof (photo, signature, call log) so that the system records my delivery attempt properly.

4. As a rider, I want an option to escalate to a supervisor if I can't reach the customer so that the delivery doesn't fail immediately.

Operations Manager

1. As an operations manager, I want to reassign a delivery quickly when a rider escalates so that the order still has a chance of success.

2. As an operations manager, I want visibility of failed deliveries and reasons so that I can monitor rider performance.

Customer Support

1. As a customer support agent, I want the system to automatically trigger refunds so that customers are compensated without delays.

2. As a customer support agent, I want to see the rider's proof and call logs so that I can resolve disputes quickly.

Compliance

1. As a compliance officer, I want every delivery attempt logged with timestamp, rider ID, and proof so that I can perform audits later.

Non-Functional Requirements (NFRs)

1. Address validation and pre-arrival notifications must complete within 10 seconds.

2. Rider app proof uploads must sync even in low network conditions.

3. Refund processing and notifications must occur within 1 minute of failure logging.

4. Compliance data (proofs, call logs) must be secure, tamper-proof, and stored for 6 months.

Data Needs

1. Customer Data: name, phone (masked/unmasked), address, gate code.

2. Order Data: order ID, items, value, store ID, timestamps.
3. Delivery Data: rider ID, navigation data, delivery status, reason codes, escalation details.
4. Proof Data: delivery photo, e-signature, failure photo, call logs, refund details.
5. Audit Data: compliance logs, timestamps, supervisor intervention notes.

MoSCoW Prioritization

Must Have

1. Address validation before order assignment
2. Pre-arrival customer contact (call/SMS)
3. Rider app must enforce proof upload for both successful and failed deliveries
4. Refund must be auto-triggered on failure with customer notification
5. Supervisor escalation flow for unreachable customers
6. Compliance log capturing timestamp, rider ID, proof, and reason codes

Should Have

1. Dual call option (masked and unmasked) in rider app
2. Operations dashboard to monitor delivery failures and reasons
3. Customer support access to proofs and call logs for dispute resolution
4. Notifications delivered within 1 minute of status change

Could Have

1. Rider navigation support with live traffic integration

2. AI-based prediction of high-risk deliveries (bad address clusters, frequent unreachable customers)

3. Customer ability to update delivery instructions in-app post-order placement

Won’t Have (for this release)

1. Pricing or promotions linked to delivery performance

2. Warehouse design or expansion changes

3. Integration with third-party logistics partners

Roadmap & RAID Log

Roadmap (60-Day Plan)

Week 1–2

1. Finalize requirements and business rules

2. Design updated Rider app workflows (proof upload, contact attempts, escalation)

3. Address validation logic design

Week 3–4

1. Develop address validation and pre-arrival contact modules

2. Build proof upload feature in Rider app

3. Backend changes for compliance logging

Week 5–6

1. Implement supervisor escalation dashboard

2. Integrate auto-refund process with finance systems

3. Internal QA testing of modules

Week 7–8

1. User Acceptance Testing (UAT) with riders and operations team
2. Pilot rollout in selected Bangalore zones
3. Monitor KPIs (failed deliveries %, on-time delivery %)

RAID Log

Risks

1. Riders may resist new proof upload requirements (adoption risk)
2. Address validation could slow down order placement if not optimized
3. Refund system integration may face delays

Assumptions

1. Customers will respond to at least one pre-arrival contact attempt
2. Supervisors are available to handle escalations during delivery hours
3. Current Rider smartphones support photo upload and app updates

Issues

1. High variability in customer network availability could still cause unreachable cases
2. Legacy finance systems may not support automated refunds easily

Dependencies

1. Rider app development team must deliver proof upload and call features on time

2. Customer support system must integrate with audit logs
3. Finance system must support real-time refund initiation

UAT Plan

Objectives

1. Validate that the new delivery process reduces failed deliveries and increases on-time deliveries.
2. Ensure all business rules and requirements are functioning as designed.
3. Confirm that the system, rider app, and customer experience work smoothly end-to-end.

Scope

- Address validation flow
- Pre-arrival contact (masked/unmasked calls, SMS)
- Proof upload (delivery success and failure)
- Rescue escalation (supervisor reassignment)
- Refund and customer notification
- Compliance logging

UAT Scenarios & Acceptance Criteria

Customer

1. Enter an invalid address → System prompts correction until valid (Pass if system blocks assignment until valid).

2. Place order → Receive pre-arrival SMS/IVR within 1 minute (Pass if message is received).

3. Failed delivery → Refund notification received (Pass if SMS/app notification is delivered within 1 minute).

Rider

1. Pickup order → App allows only validated addresses (Pass if invalid addresses are blocked).

2. Customer unreachable → App enforces 2 attempts (masked + unmasked) (Pass if blocked until both attempts logged).

3. Complete delivery → Proof upload required before marking status (Pass if app prevents closure without proof).

4. Escalation triggered → Supervisor notified and reassignment possible (Pass if escalation workflow is visible).

Operations / Support / Compliance

1. Failed delivery → Refund auto-initiated in finance system (Pass if refund triggered without manual action).

2. Supervisor dashboard → Delivery reassignment visible (Pass if supervisor can reassign within 2 minutes).

3. Compliance review → Proofs and logs accessible for audit (Pass if timestamp, rider ID, proof are correctly stored).

Sign-Off Criteria

1. All Must-Have requirements pass UAT.

2. No critical or high-severity defects open.

3. Business sponsor (City Ops Manager) and key stakeholders approve test results.

Change Management (ADKAR)

Awareness

1. Communicate the problem (11% failed deliveries, 82% on-time).
2. Share goals with all stakeholders (target: reduce failures to 4%, increase on-time to 95% in 60 days).
3. Present benefits: higher customer trust, better rider efficiency, reduced refund costs.

Desire

1. Motivate riders with incentives for successful, proof-compliant deliveries.
2. Highlight benefits for supervisors (easier escalations, faster resolution).
3. Show store teams how fewer failures improve order turnaround.

Knowledge

1. Train riders on new app features (proof upload, masked/unmasked calls, escalation).
2. Provide supervisors training on dashboard for reassignment.
3. Share updated SOPs for customer support and compliance logging.

Ability

1. Conduct pilot runs in selected zones with real orders.
2. Provide quick reference guides and in-app tooltips for riders.
3. Set up helpline for live troubleshooting during rollout.

Reinforcement

1. Monitor KPIs weekly and celebrate improvements.

2. Provide recognition and bonuses to top-performing riders.

3. Continuous refresher training for new riders and support staff.

Pilot & Go-Live Plan

Pilot Plan

1. **Scope** – Run pilot in 2 Bangalore zones with ~200 orders/day.

2. **Duration** – 2 weeks.

3. **Participants** – Selected riders, supervisors, store managers, customer support team.

4. **Features Tested** – Address validation, pre-arrival contact, proof upload, rescue escalation, refund automation, compliance logging.

5. **Success Criteria** –

    - Failed deliveries reduced from 11% → ≤6% in pilot zones.

    - On-time deliveries improved from 82% → ≥92%.

    - 95% rider compliance with proof uploads.

    - Refunds processed within 1 minute of failure.

Go-Live Plan

1. **Phase 1 (Weeks 1–2)** – Expand to all Bangalore central zones (approx. 20% of total orders).

2. **Phase 2 (Weeks 3–4)** – Full Bangalore rollout across all hubs and stores.

3. **Phase 3 (Week 5 onward)** – Monitor KPIs daily for 30 days, stabilize, and hand over to BAU (Business as Usual).

4. **Support Structure** –

    - Dedicated rider support line during first 2 weeks.
    - Daily stand-ups with Ops Managers and Tech team during rollout.
    - Weekly performance report to Sponsor (City Ops Manager).
5. **Exit Criteria for Go-Live** –
    - Failed deliveries reduced to ≤4%.
    - On-time deliveries consistently ≥95%.
    - No critical system issues open.

Benefits Tracking & Post-Implementation Review

KPIs to Track

1. Failed deliveries reduced from 11% → ≤4% within 60 days.
2. On-time deliveries improved from 82% → ≥95%.
3. Average refund initiation time ≤1 minute.
4. Rider compliance with proof uploads ≥95%.
5. Customer satisfaction score (CSAT) improved by 10%.
6. Reduction in refund costs by at least 20%.

Tracking Mechanism

1. Daily automated reports from delivery system on success/failure rates.
2. Weekly KPI dashboard reviewed by City Ops Manager.
3. Monthly compliance audits of proof logs and reason codes.

4. Customer feedback surveys (in-app rating + post-delivery SMS).

5. Finance reports to track refund amounts vs baseline.

Post-Implementation Review (PIR)

1. Conduct PIR meeting at Day 60 with Sponsor and key stakeholders.

2. Compare KPIs against baseline and project goals.

3. Document lessons learned (process, tech, adoption challenges).

4. Recommend follow-up actions (further automation, AI predictions for risky deliveries, expanded training).

5. Archive all process documents and update SOPs to reflect new workflow.