Raffle Winner Automation – Solution Proposal

Candidate: Djordje Mladenovic | **Role:** Product Owner / Solutions Owner

1. Problem Overview

The merchant runs a raffle campaign where customers are progressively marked as winners by adding a Shopify customer tag **rafflewinner**. Once a customer becomes a winner, two automated actions must occur reliably and only once per customer: **(1)** applying a 10% discount to all Recharge subscriptions and **(2)** notifying internal staff via email.

2. Scope Clarification Questions

• Should the automation execute only once per customer even if the tag is re-applied?
• Should the discount apply to active subscriptions only, or also paused ones?
• How should existing subscription discounts be handled?
• Is the discount permanent or time-bound?
• Should future subscriptions created by winners also receive the discount?
• Who should receive the internal notification email and which customer details are required?
• Is Shopify Flow with HTTP request enabled and is a Recharge API token available?

3. Proposed Solution

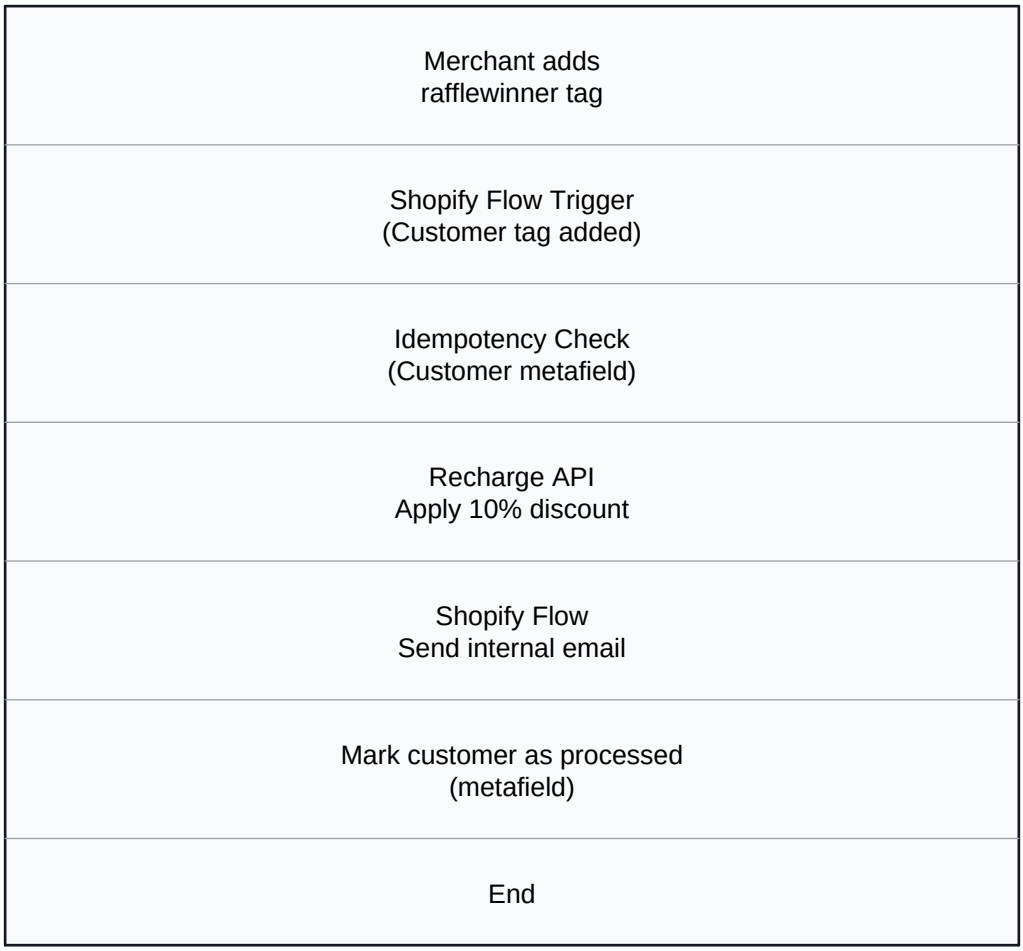
The solution uses **Shopify Flow** as the central orchestration layer to ensure simplicity, reliability, and merchant maintainability. The workflow triggers when the **rafflewinner** customer tag is added and sends a secure HTTP request to the Recharge API to apply a 10% subscription-level discount, avoiding the need for custom applications.

4. High-Level Workflow

1. Merchant adds the **rafflewinner** tag to a customer in Shopify.
2. Shopify Flow triggers on customer tag addition.
3. Flow checks a customer metafield to ensure the customer has not already been processed.
4. Flow sends an HTTP request to Recharge to apply a 10% discount to the customer’s subscriptions.
5. Flow sends an internal notification email to staff.
6. Flow marks the customer as processed using a metafield and safely terminates.

5. High-Level Flow Diagram

The diagram below represents the end-to-end automation flow from tagging a customer as a raffle winner to applying the discount and notifying internal staff.



6. Edge Cases & Safeguards

• Idempotency is enforced via a customer metafield to prevent duplicate discount application.
• Infinite loops are avoided by not modifying customer tags within the same workflow.
• Partial failures can be surfaced through internal email notifications for manual follow-up.
• Existing subscription discounts are handled according to merchant-defined rules.

7. Rationale & Benefits

This approach prioritizes clarity, reliability, and long-term maintainability. By relying exclusively on Shopify Flow and publicly documented Recharge APIs, the merchant avoids unnecessary custom development while retaining full operational visibility and control.