Progressive Rollout Playbook – Canary Routing Model

This playbook outlines the step-by-step actions to enable a controlled canary rollout for API partners using Apigee header-based routing and developer app attributes.

# Step-by-Step Checklist

## Partner Preparation

* ☐ Inform partner about upcoming feature rollout and testing opportunity.
* ☐ Confirm partner readiness to send 'X-Feature-Variant: new' in request headers.

## Apigee Developer App Setup (Synchrony-owned)

* ☐ Navigate to Apigee Developer App.
* ☐ Set custom attribute 'canary\_access\_enabled=true' for selected partners.
* ☐ Leave attribute as 'false' or unset for non-participating partners.

## Canary App Deployment

* ☐ Deploy new version of backend service to PCF as 'paylater-canary' or versioned route.
* ☐ Verify canary app health, logging, and monitoring.
* ☐ Ensure separate PCF route or hostname exists for canary app.

## Apigee Proxy Configuration

* ☐ Add JavaScript policy to read 'X-Feature-Variant' and 'canary\_access\_enabled'.
* ☐ Implement routing logic to direct traffic to main or canary based on rules.
* ☐ Inject response headers for tracing (e.g., X-Target-App).
* ☐ Test routing with different combinations (true/false/new/old).

## Validation & Monitoring

* ☐ Enable partner traffic to hit new flow (X-Feature-Variant: new + dialup=true).
* ☐ Monitor API logs (e.g., Splunk) and canary app behavior.
* ☐ Gather partner feedback and metrics.

## Promotion to Main App

* ☐ Once canary is validated, deploy same code to 'main' app (paylater-main).
* ☐ Remove or ignore 'X-Feature-Variant' header in routing logic.
* ☐ Update Apigee routing if needed to always direct to main app.
* ☐ Optionally retire the canary app after validation window.