1. Overview

Objective: Build an MVP of Econlytics, a unit economics analytics platform, to track customer-level profitability via Stripe (revenue) and Zendesk (cost) integrations.
Success Criteria:

- Ingest customer transaction and support ticket data.
- Calculate per-customer profit (revenue - fees - support costs).
- Display profitability dashboards and cohort analysis.
- Reconcile aggregated data with company financials (±5% accuracy).

---

2. MVP Scope

In Scope

- **Integrations**:
    - **Stripe**: Track revenue, fees, and customer IDs.
    - **Zendesk**: Track support ticket duration and associate costs to customers.
- **Core Features**:
    - Customer-level P&L calculation.
    - Cohort filtering (e.g., acquisition channel, support ticket count).
    - CSV export of profitability data.
    - Reconciliation report vs. company financials.
- **Performance**:
    - Handle 100K customer records with <15-minute data latency.

Out of Scope

- Third-party integrations beyond Stripe/Zendesk (e.g., AWS, Salesforce).
- User authentication or multi-tenant support.
- Predictive analytics or ML.

---

3. Technical Architecture

System Diagram

Simplified flow: Events → Ingestion → Processing → Storage → Dashboard

Components

1. **Event Ingestion**

- **Stripe**: Webhook endpoint to capture charge.succeeded, refund, and application_fee events.
- **Zendesk**: API polling (every 15 mins) to fetch tickets and calculate costs (duration × hourly rate).

2. **Data Pipeline**
   - **Processing**: Python scripts (AWS Lambda) to transform raw data.
   - **Storage**: PostgreSQL (RDS) for structured data (transactions, support costs).
3. **Cost Allocation Engine**
   - **Formula**: Customer Profit = (Stripe Revenue - Stripe Fees) - (Zendesk Cost per Ticket × Ticket Count).
   - **Reconciliation**: Daily job to match total revenue/cost with financial statements.
4. **Dashboard**
   - **Tool**: Retool (low-code frontend) connected to PostgreSQL.
   - **Features**: Table view, cohort filters, CSV export.

---

4. Data Model

PostgreSQL Schema

Key Data Flows

1. **Stripe Webhook → PostgreSQL**
   - Trigger: Every successful charge.
   - Fields: customer_id, revenue, fee, timestamp.
2. **Zendesk API → PostgreSQL**
   - Trigger: Poll every 15 minutes.
   - Fields: customer_id, duration, cost, timestamp.

---

5. API & Integration Specifications

Stripe Webhook (Python/Flask)

Zendesk API Client (Python)

---

6. Dashboard Requirements

Retool Configuration

1. **Customer Profitability Table**
    - Columns: Customer ID, Total Revenue, Total Fees, Support Costs, Net Profit.
    - Filter: Cohort, signup date, min/max profit.
2. **Cohort Analysis**
    - Group by: cohort, acquisition_month.
    - Metrics: Avg. Profit per Customer, Total Profit.
3. **Reconciliation Report**
    - Compare: System Total Revenue vs. QuickBooks Revenue.

---

7. Testing Plan

Unit Tests

- Verify Stripe webhook correctly parses and stores charges.
- Ensure Zendesk cost calculation uses the correct hourly rate.
- Validate reconciliation SQL matches totals within 5%.

Integration Tests

- End-to-end test: Simulate 1K transactions + 100 tickets → check P&L.
- Latency test: Ensure data appears in Retool within 15 minutes.

Load Test

- Ingest 100K dummy customer records → validate dashboard performance.

---

8. Deployment

Infrastructure

- **AWS Services**: Lambda (Python runtime), RDS (PostgreSQL), S3 (backups).
- **Retool**: Cloud-hosted instance connected to RDS.

Security

- Encrypt RDS at rest (AES-256).
- Store API keys in AWS Secrets Manager.

---

9. Milestones & Timeline

Milestone	Deadline	Owner
Stripe webhook live	Day 7	
Zendesk API polling implemented	Day 14	
MVP dashboard in Retool	Day 21	
First beta partner onboarded	Day 30	

---

10. Risks & Mitigations

Risk	Mitigation
Stripe/Zendesk API rate limits	Implement exponential backoff for retries.
Data discrepancies in reconciliation	Allow manual adjustment of formulas in Retool.
High AWS costs for Lambda	Optimize code + set budget alerts.

---

11. Acceptance Criteria

The MVP is complete when:

- Beta partner can view customer-level profitability.
- Cohort filtering (e.g., “Facebook Ads users”) works in Retool.
- Reconciliation report shows ≤5% variance from financial statements.
- Data latency ≤15 minutes.