TourLens — Architecture Design (Option 2)

Table of Contents

1. System Context (C4 Level 1)

Shows TourLens as a black box — who uses it and what it depends on.

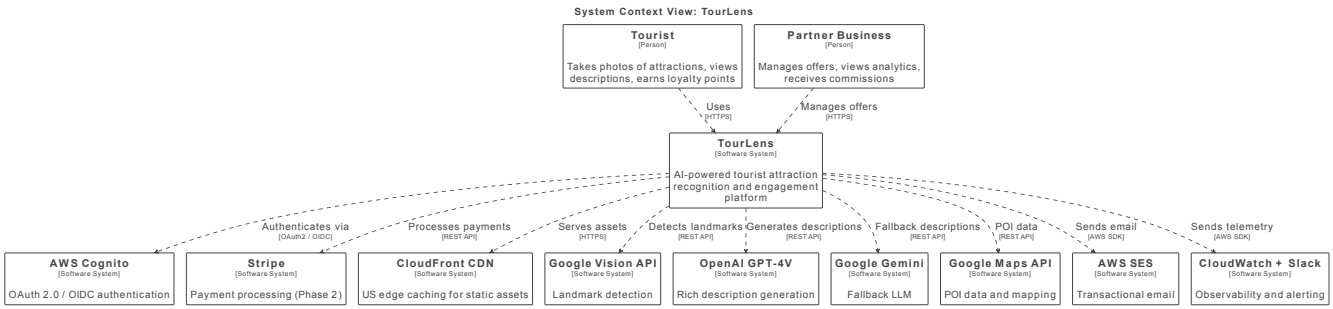

Figure 1. System Context — TourLens

2. Container Diagram (C4 Level 2)

Shows the applications, data stores, and message brokers inside the TourLens system boundary.

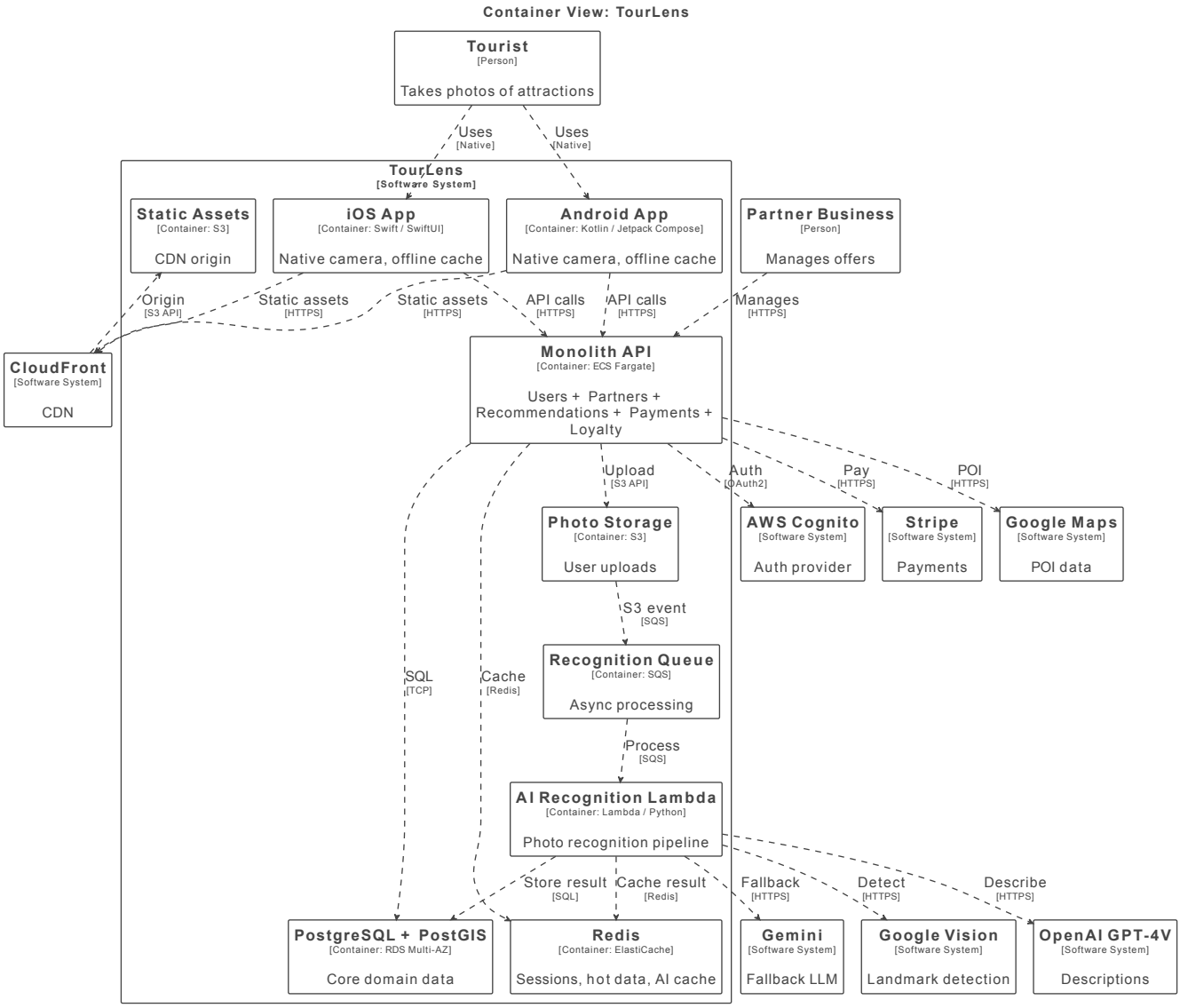

Figure 2. Container View — TourLens

3. Deployment Diagram (C4 Level 3)

Shows how containers map to AWS infrastructure nodes.

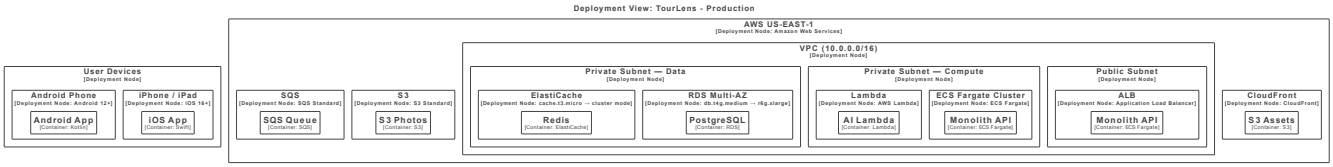

Figure 3. Deployment View — TourLens Production (US-EAST-1)

4. Component Architecture

Internal services, data stores, and connections within the TourLens monolith.

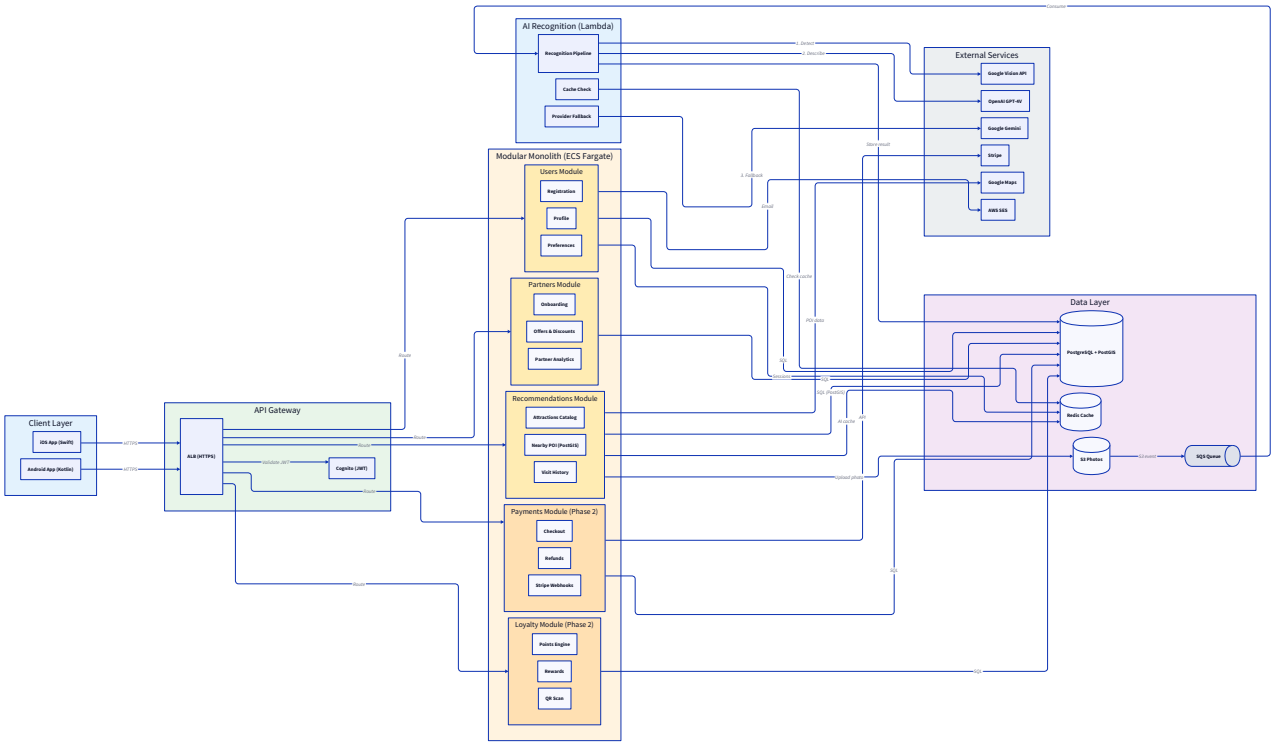

Figure 4. Component Architecture — TourLens Monolith

5. Data Model

Entity-relationship diagram for the core domain.

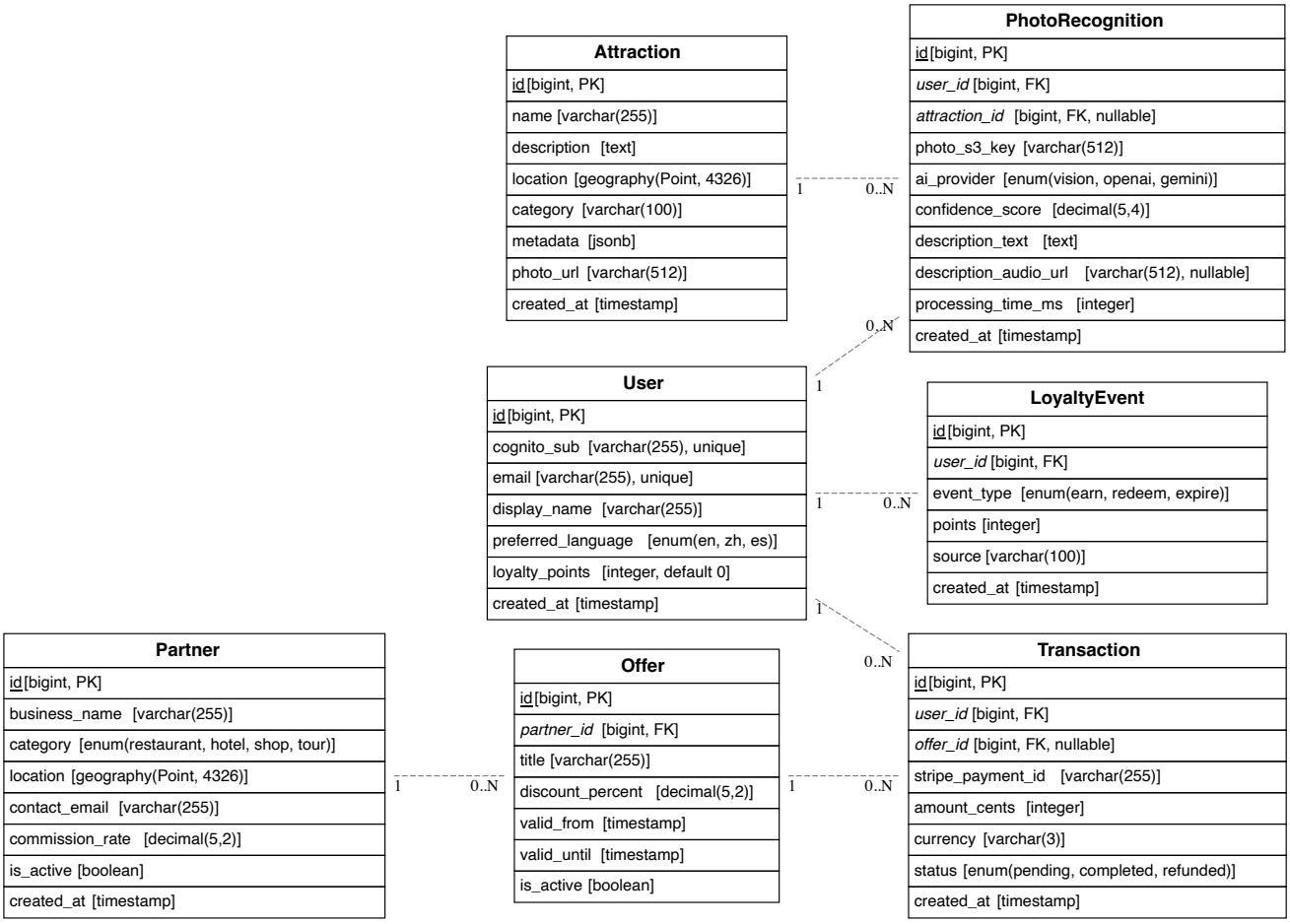

Figure 5. Data Model — TourLens

6. Key Flows

6.1. Happy Path — Photo Recognition

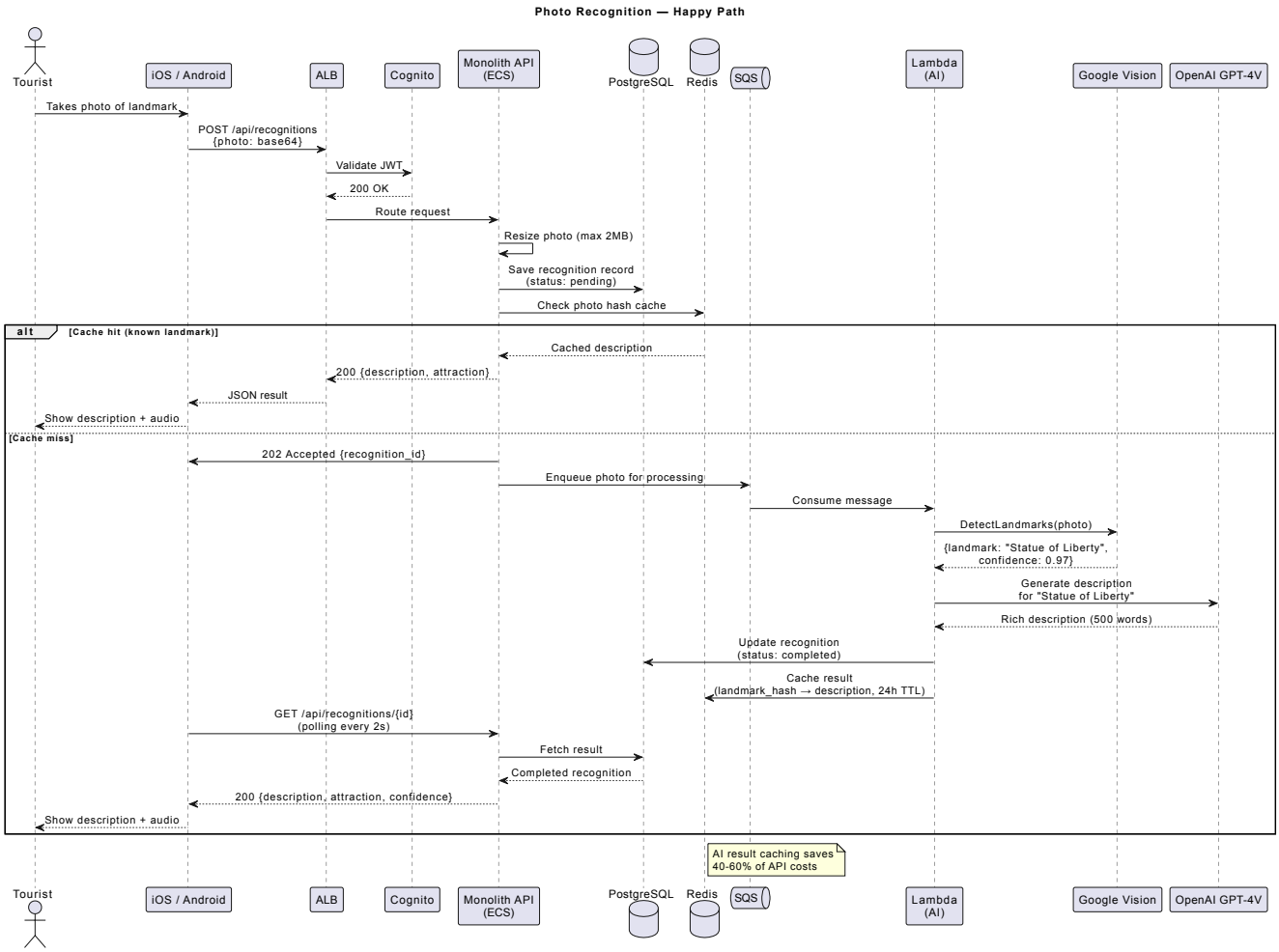

Figure 6. Core Flow — Tourist Takes Photo of Landmark

6.2. Error Flow — AI Provider Fallback Chain

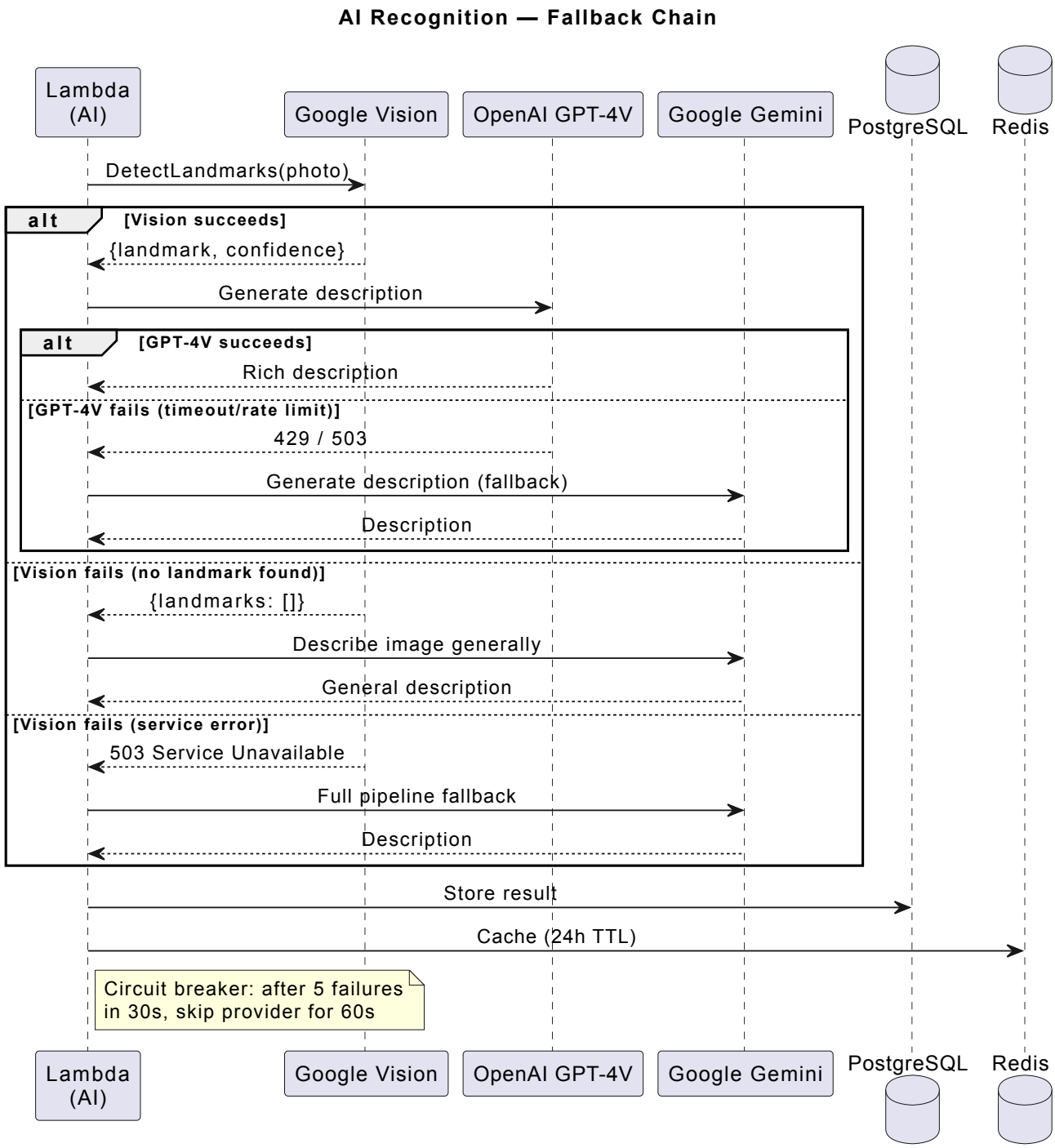

Figure 7. AI Fallback — Provider Failure Handling

7. Infrastructure Topology

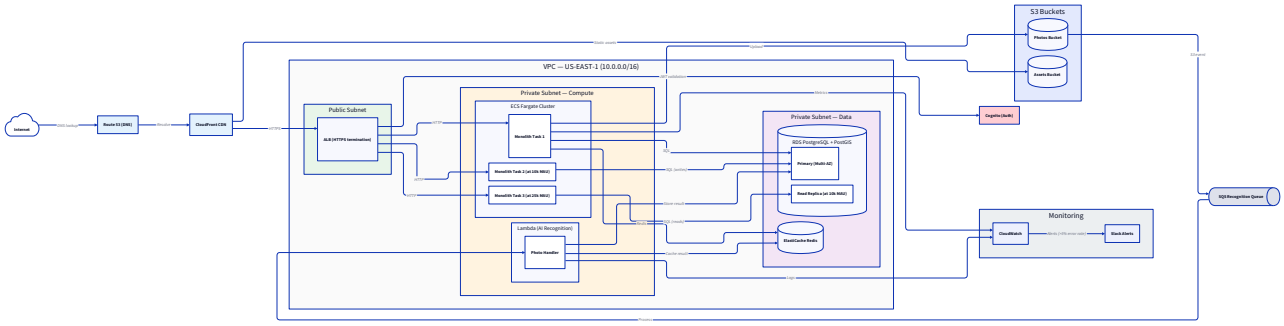

Figure 8. Infrastructure — TourLens Production (US-EAST-1)

Grayed-out components (Task 2, Task 3, Read Replica) represent future scaling steps that are added as MAU grows. At launch (1k MAU), only Task 1 and the Primary DB are active.

8. Scalability Architecture

Shows how the architecture evolves from 1 ECS task to multiple tasks with read replicas as traffic grows. All scaling happens within the same monolith codebase — no architectural changes needed until 500k+ MAU.

8.1. Scaling Stages

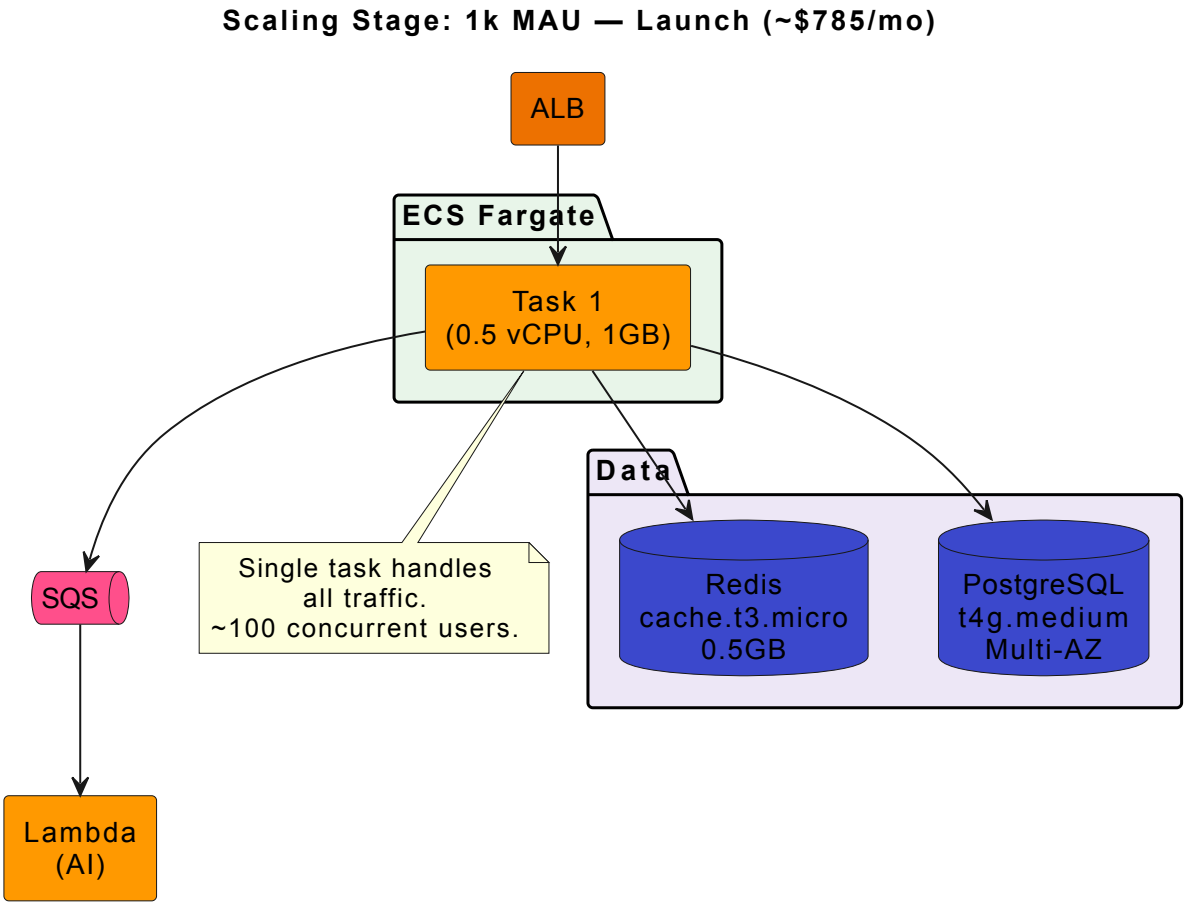

Figure 9. Scaling — 1k MAU (Launch)

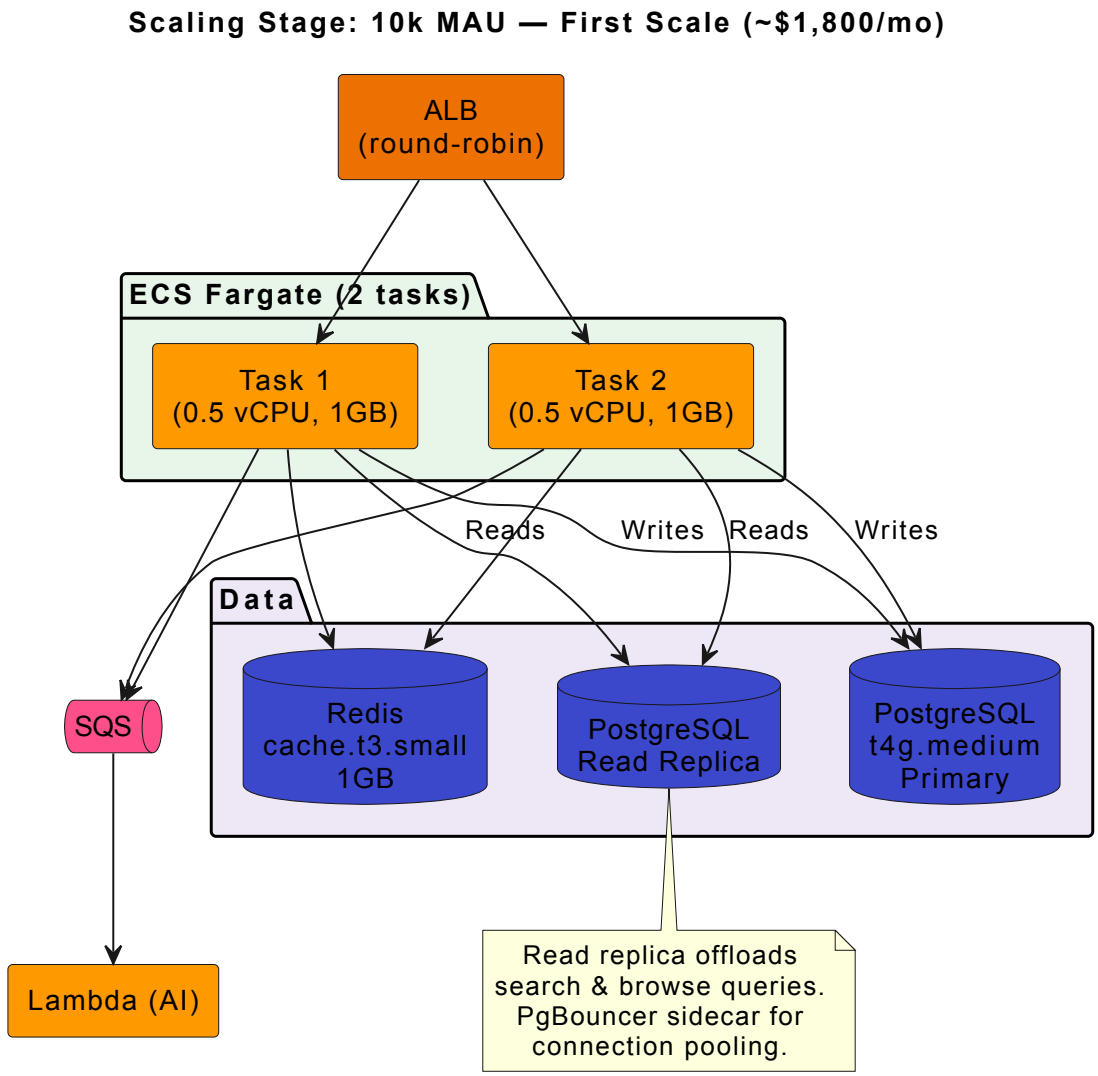

Figure 10. Scaling — 10k MAU (First Scale)

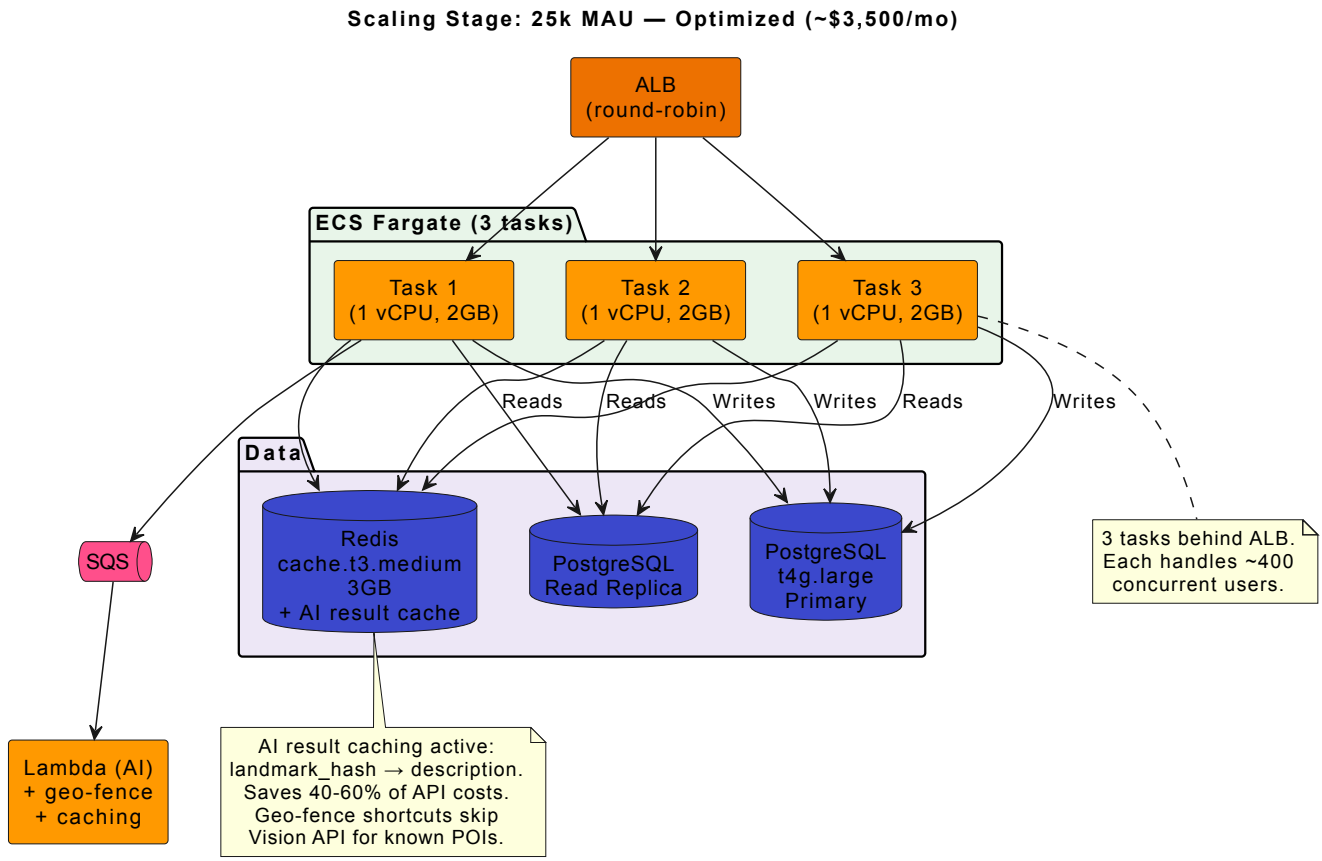

Figure 11. Scaling — 25k MAU (Optimized)

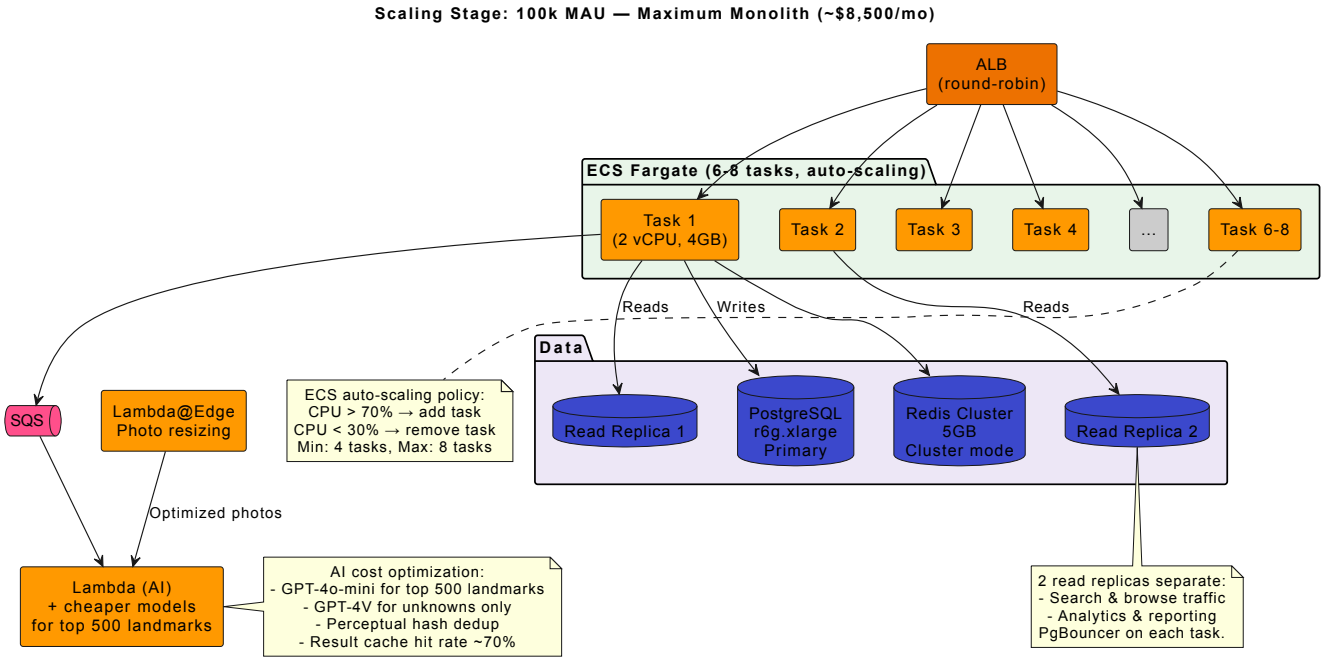

Figure 12. Scaling — 100k MAU (Maximum Monolith)

8.2. Scaling Summary

MAU	ECS Tasks	DB Config	OPEX/mo	Key Change
1k	1	t4g.medium, Multi-AZ	~$785	Launch config — handles everything
5k	1	t4g.medium, Multi-AZ	~$1,200	Upgrade Redis only
10k	2	t4g.medium + read replica	~$1,800	First horizontal scale + PgBouncer
25k	3	t4g.large + read replica	~$3,500	AI result caching (first code change)
50k	4-5	t4g.large + read replica	~$5,500	Redis cluster mode, photo dedup
100k	6-8	r6g.xlarge + 2 read replicas	~$8,500	ECS auto-scaling, Lambda@Edge, cheaper AI models
250k	8-12	Aurora PostgreSQL	~$15-20k	Aurora migration, custom ML model
500k+	→ EKS	Aurora Global DB	~$50k+	Microservices, Kafka, multi-region

9. Observability Stack

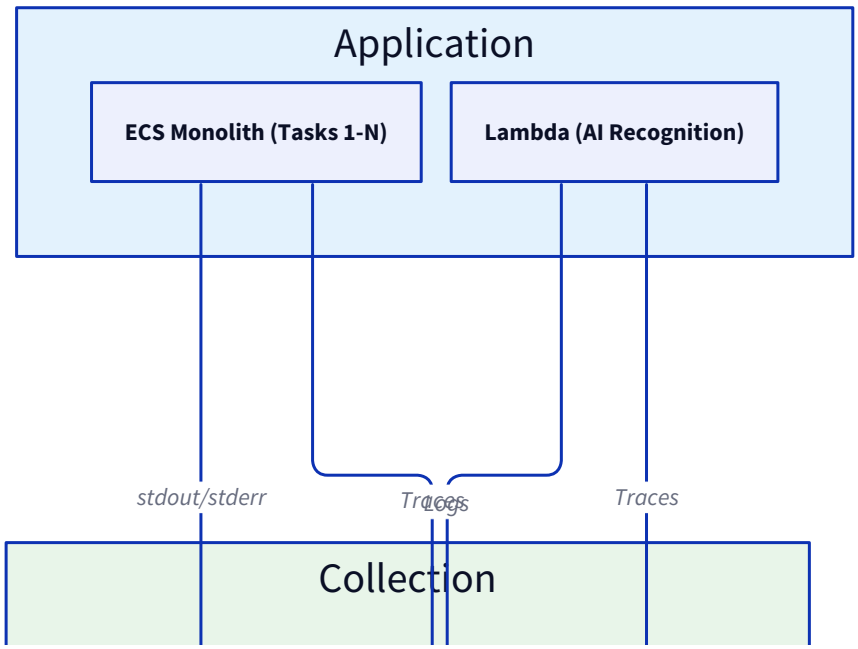

Figure 13. Observability — TourLens

9.1. SLOs

Objective	Indicator (SLI)	Target	Alert
Availability	Successful requests / total	99.9% monthly	< 99.5% over 5 min
API Latency	P95 response time	< 200ms	> 500ms over 5 min
AI Recognition	P95 processing time	< 5 seconds	> 10 seconds over 5 min
Error Rate	5xx / total responses	< 0.1%	> 1% over 5 min
AI Cache Hit Rate	Cache hits / total recognitions	> 60% (at 25k+ MAU)	< 40% over 1 hour

10. Business Process Flow

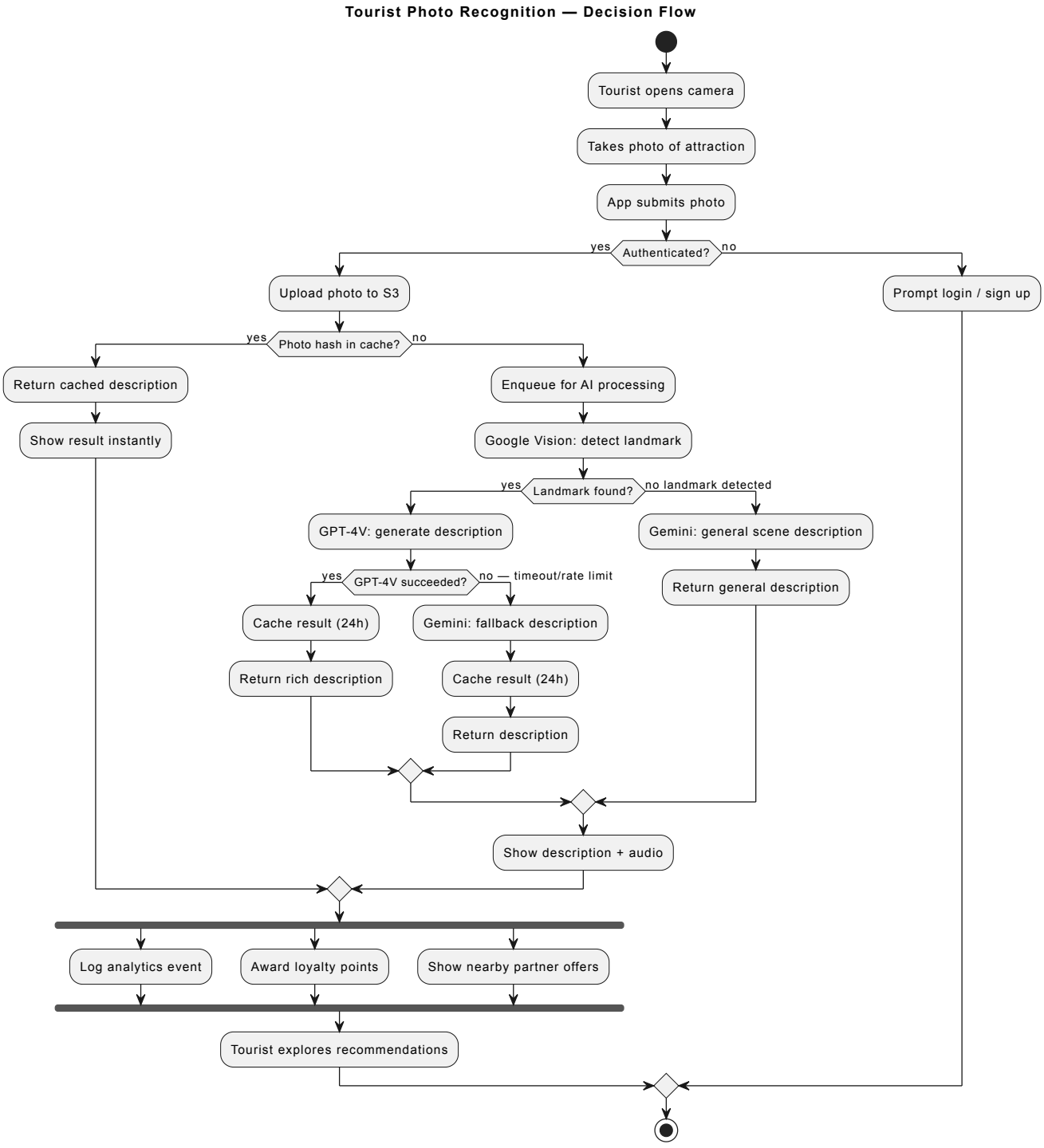

Figure 14. Tourist Photo Recognition — Decision Flow

11. Feature Breakdown

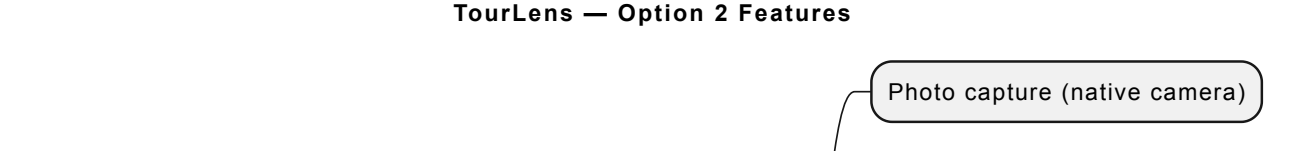

Figure 15. Feature Mind Map — TourLens Option 2

12. Work Breakdown Structure