Complete Architecture Documentation

Medical Document Chatbot - System Architecture & Integration

Version: 1.0

Date: October 15, 2025

Status: Production Architecture

Table of Contents

Architecture Overview

Architecture Style

- **Pattern:** Microservices + Event-Driven Architecture
- **Deployment:** Cloud-Native on Microsoft Azure
- **Scale:** Auto-scaling, globally distributed
- **Security:** Defense in depth, zero trust model

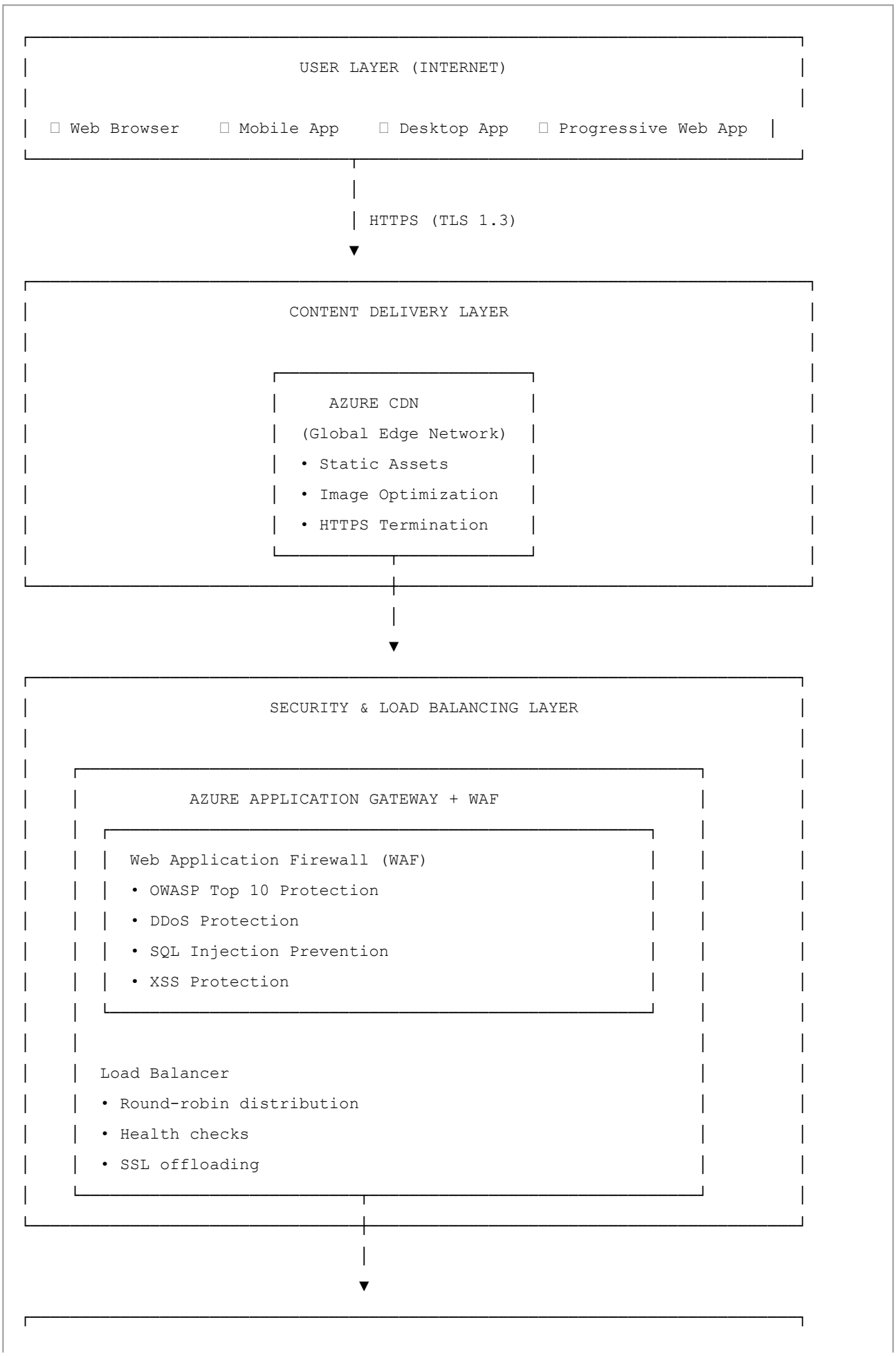
Key Characteristics

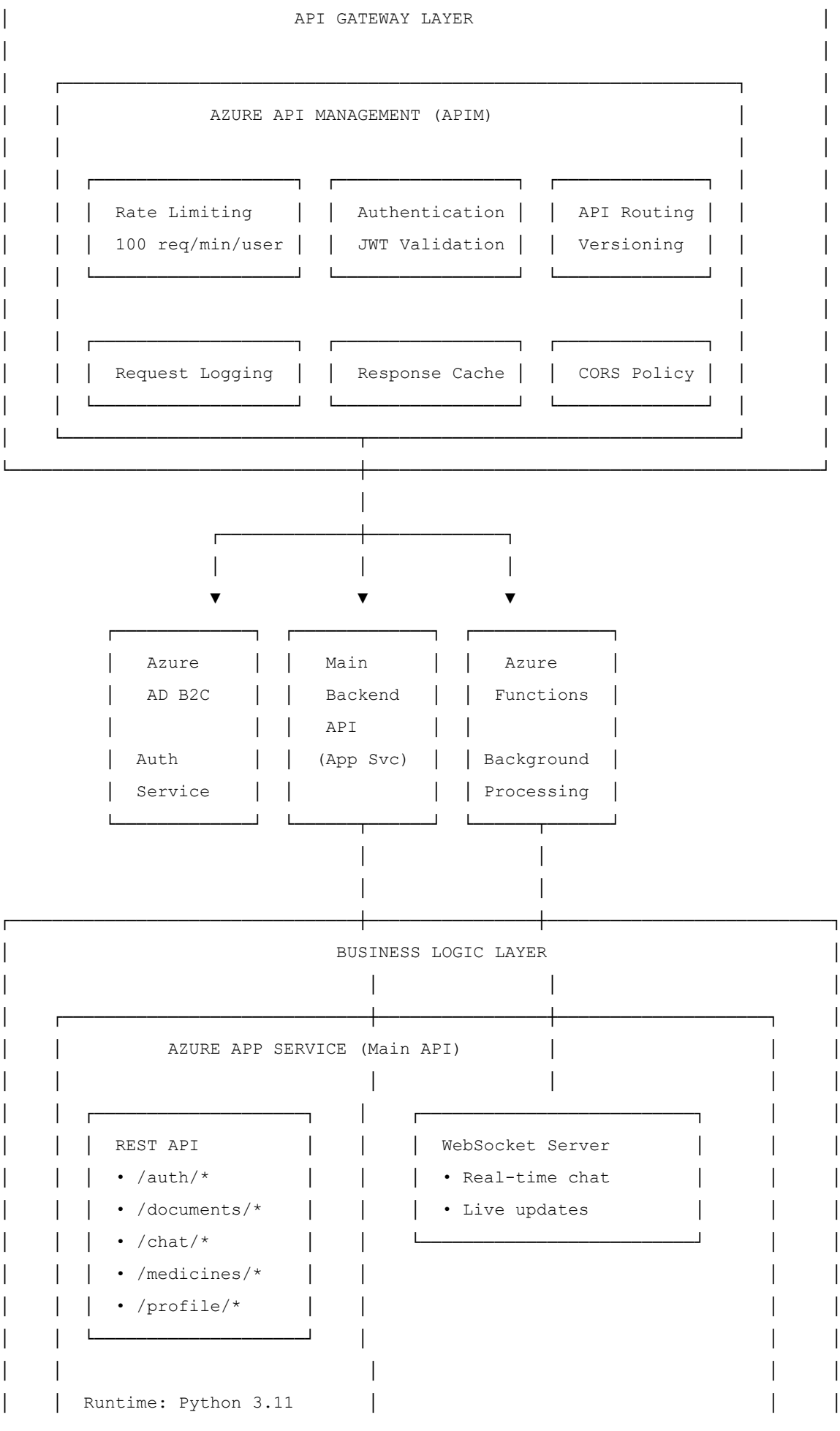
- □ **Highly Available:** 99.9% uptime SLA
- □ **Scalable:** Auto-scales from 2 to 100+ instances

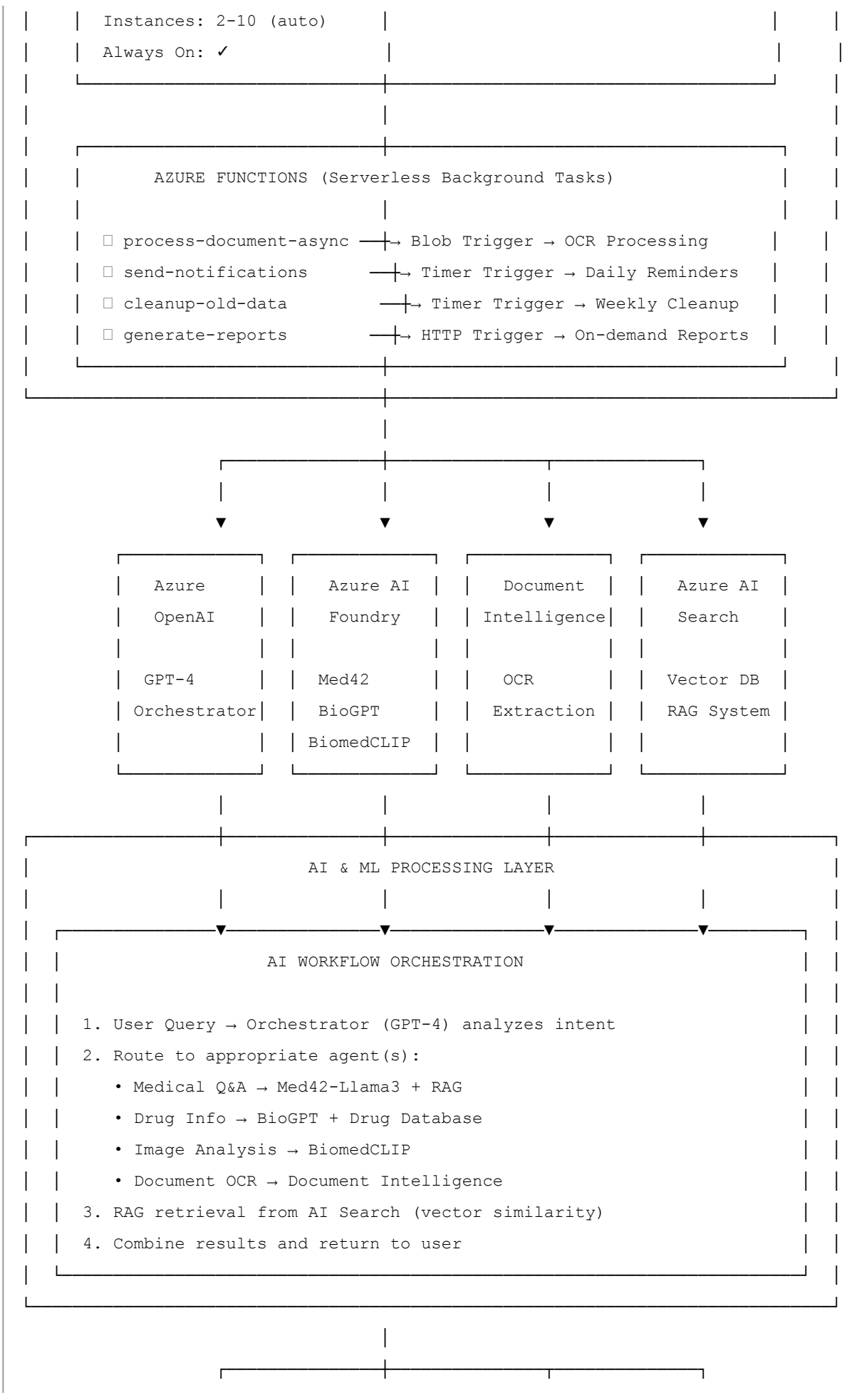
- □ **Secure:** Multi-layer security (WAF, APIM, encryption)
- □ **Performant:** <2s response time with caching
- □ **Observable:** Real-time monitoring and alerts
- □ **Cost-Optimized:** Pay-per-use, auto-shutdown

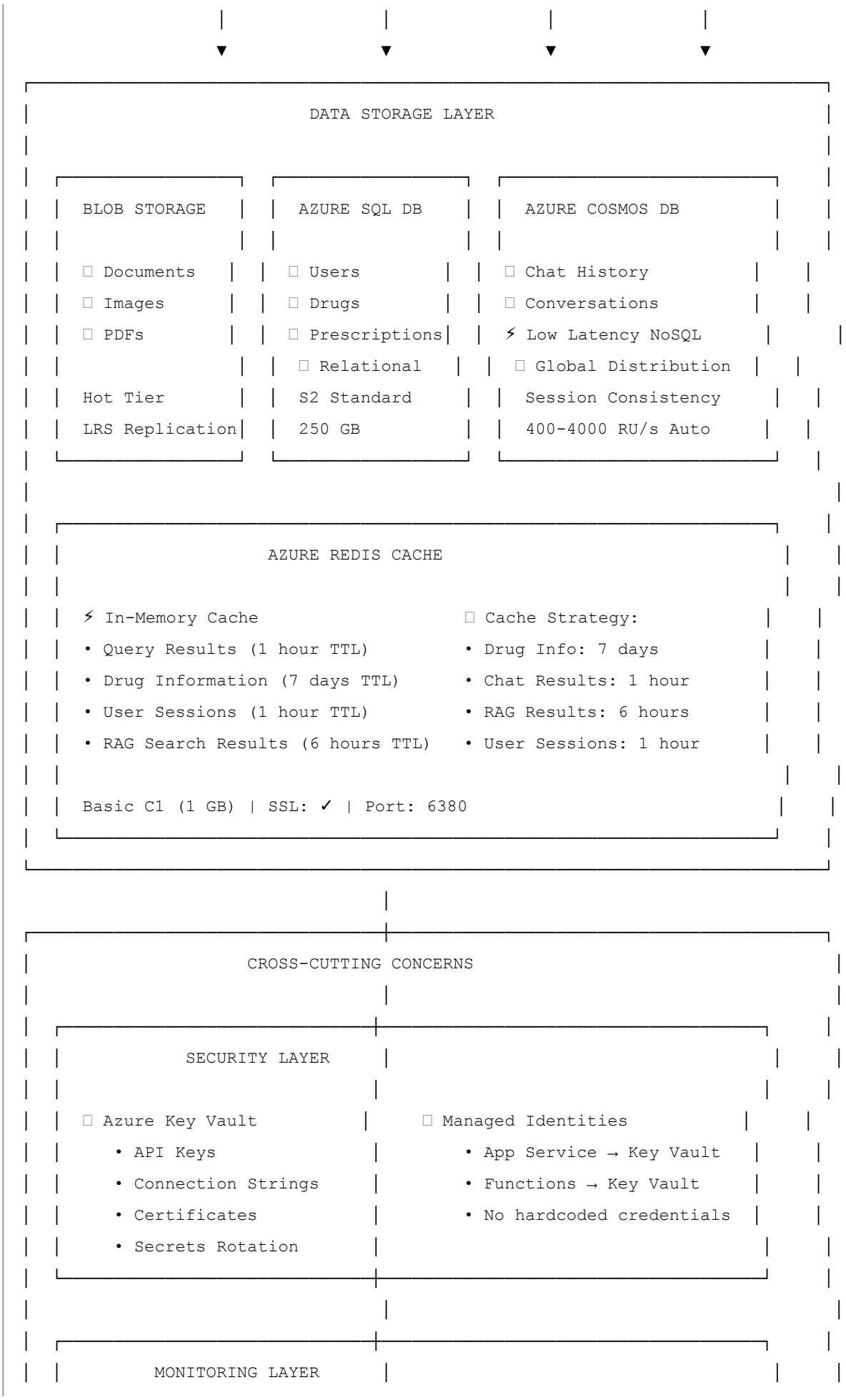
High-Level Architecture Diagram

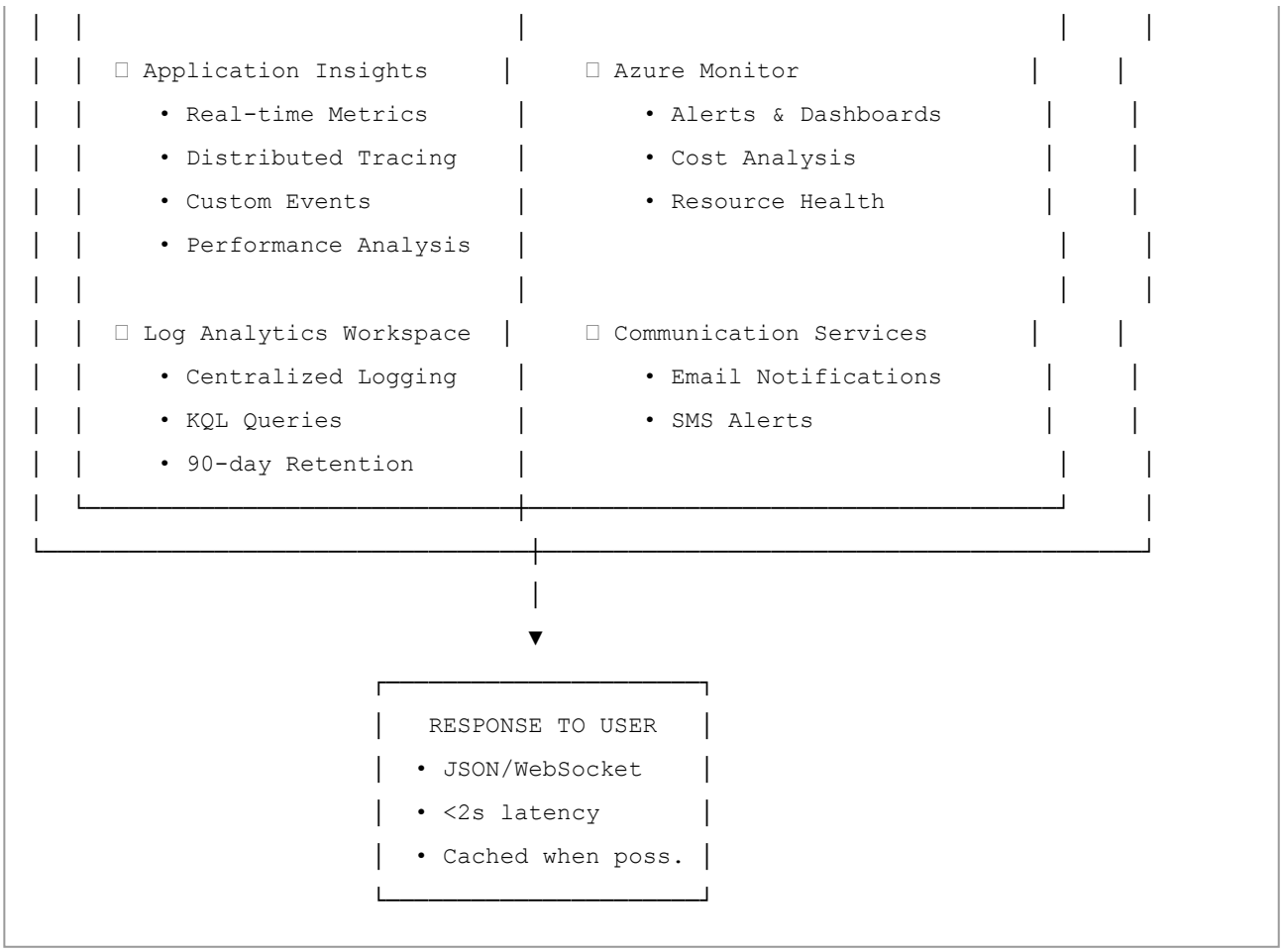
Complete System Architecture


USER LAYER (INTERNET)
Web Browser
Mobile App
Desktop App
Progressive Web App
HTTPS (TLS 1.3)
CONTENT DELIVERY LAYER
AZURE CDN
(Global Edge Network)
• Static Assets
• Image Optimization
• HTTPS Termination
SECURITY & LOAD BALANCING LAYER
AZURE APPLICATION GATEWAY + WAF
Web Application Firewall (WAF)
• OWASP Top 10 Protection
• DDoS Protection
• SQL Injection Prevention
• XSS Protection
Load Balancer
• Round-robin distribution
• Health checks
• SSL offloading


API GATEWAY LAYER
AZURE API MANAGEMENT (APIM)
Rate Limiting
100 req/min/user
Authentication
JWT Validation
API Routing
Versioning
Request Logging
Response Cache
CORS Policy
Azure
AD B2C
Auth
Service
Main
Backend
API
(App Svc)
Azure
Functions
Background
Processing
BUSINESS LOGIC LAYER
AZURE APP SERVICE (Main API)
REST API
• /auth/*
• /documents/*
• /chat/*
• /medicines/*
• /profile/*
WebSocket Server
• Real-time chat
• Live updates
Runtime: Python 3.11

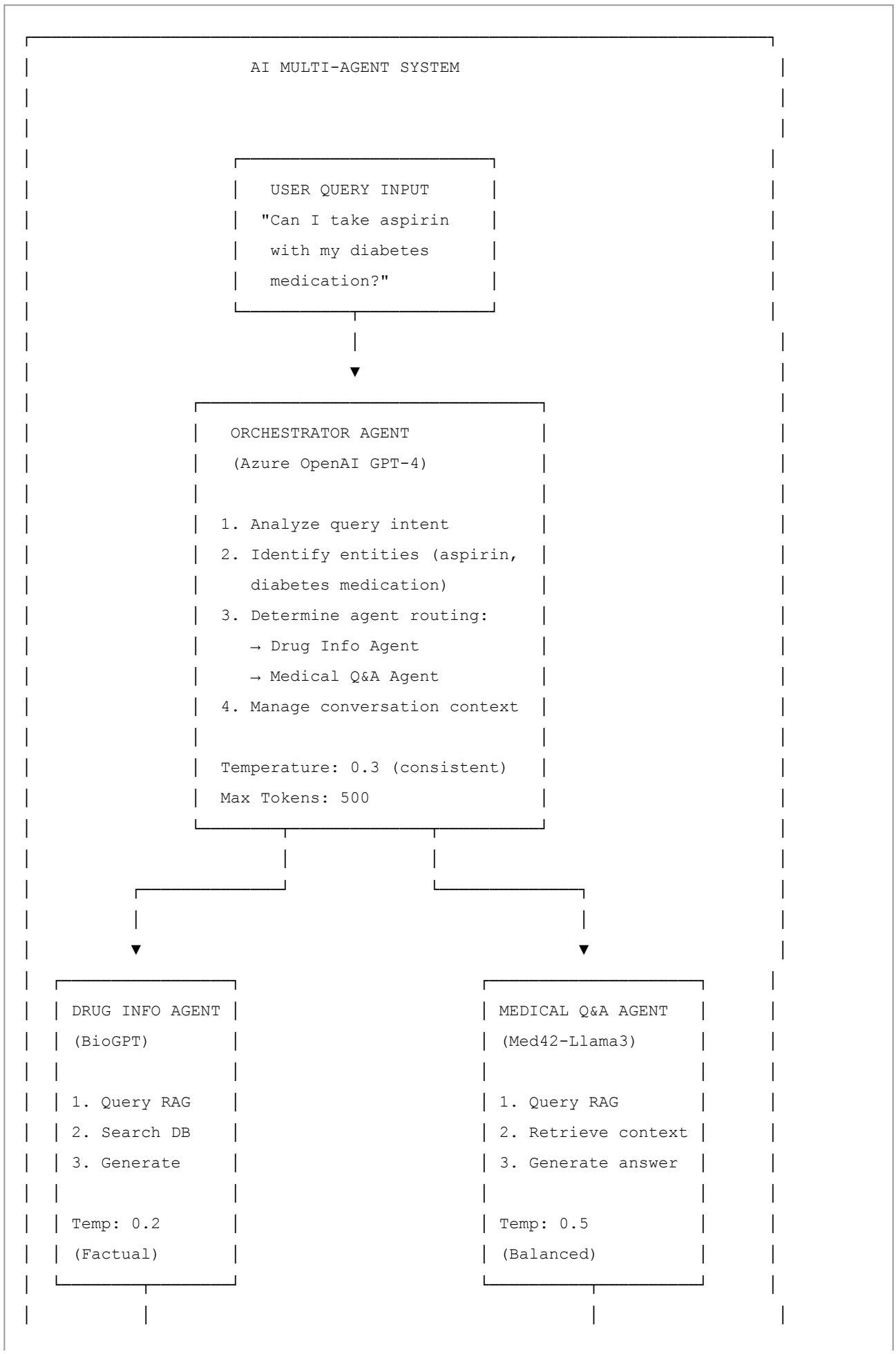

Instances: 2-10 (auto)
Always On: ✓
AZURE FUNCTIONS (Serverless Background Tasks)
process-document-async → Blob Trigger → OCR Processing
send-notifications → Timer Trigger → Daily Reminders
cleanup-old-data → Timer Trigger → Weekly Cleanup
generate-reports → HTTP Trigger → On-demand Reports
Azure OpenAI
GPT-4 Orchestrator
Azure AI Foundry
Med42 BioGPT BiomedCLIP
Document Intelligence
OCR Extraction
Azure AI Search
Vector DB RAG System
AI & ML PROCESSING LAYER
AI WORKFLOW ORCHESTRATION
1. User Query → Orchestrator (GPT-4) analyzes intent
2. Route to appropriate agent(s):
• Medical Q&A → Med42-Llama3 + RAG
• Drug Info → BioGPT + Drug Database
• Image Analysis → BiomedCLIP
• Document OCR → Document Intelligence
3. RAG retrieval from AI Search (vector similarity)
4. Combine results and return to user

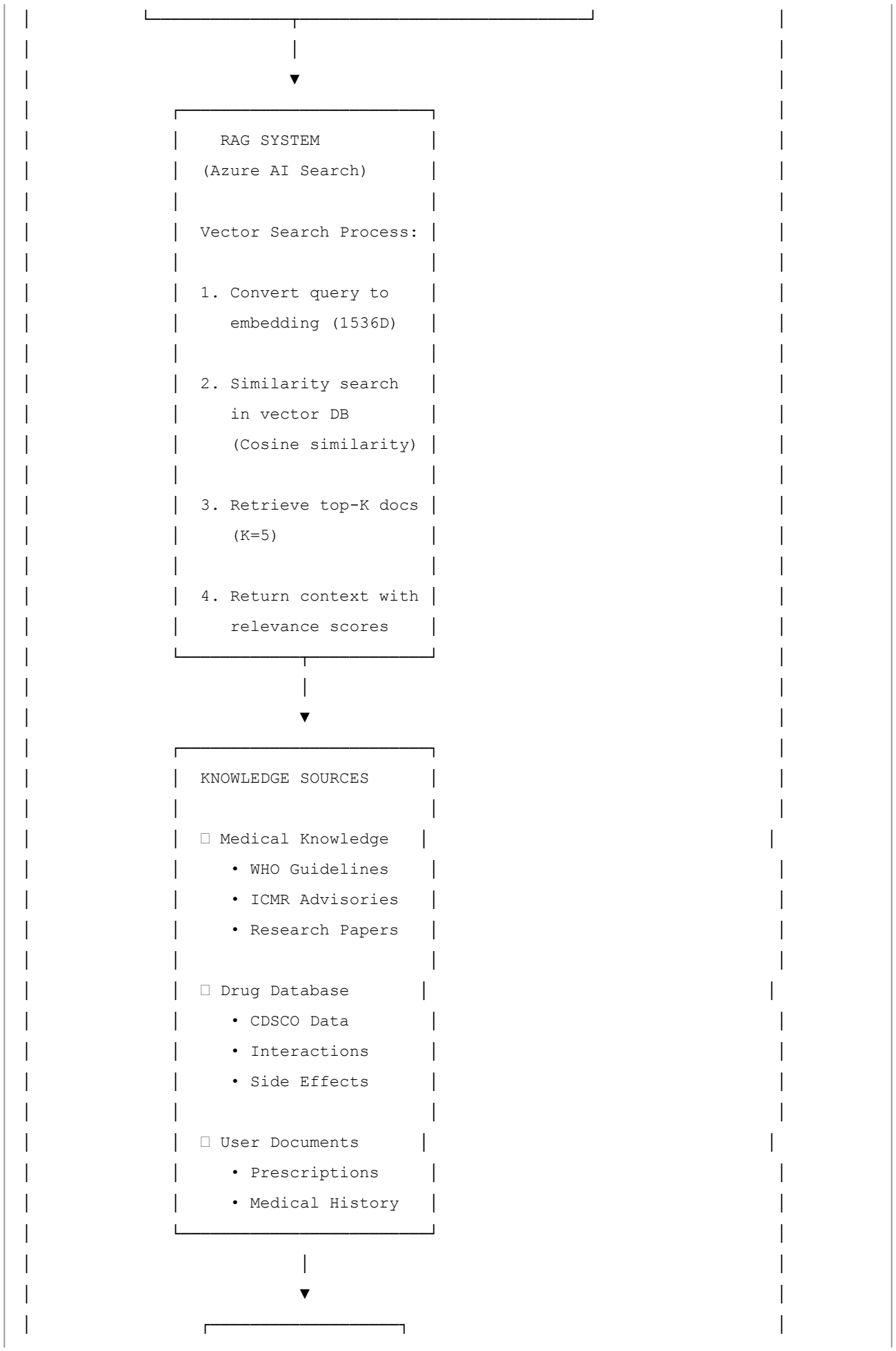

DATA STORAGE LAYER
BLOB STORAGE
Documents
Images
PDFs
Hot Tier
LRS Replication
AZURE SQL DB
Users
Drugs
Prescriptions
Relational
S2 Standard
250 GB
AZURE COSMOS DB
Chat History
Conversations
Low Latency NoSQL
Global Distribution
Session Consistency
400-4000 RU/s Auto
AZURE REDIS CACHE
In-Memory Cache
• Query Results (1 hour TTL)
• Drug Information (7 days TTL)
• User Sessions (1 hour TTL)
• RAG Search Results (6 hours TTL)
Cache Strategy:
• Drug Info: 7 days
• Chat Results: 1 hour
• RAG Results: 6 hours
• User Sessions: 1 hour
Basic C1 (1 GB) | SSL: ✓ | Port: 6380
CROSS-CUTTING CONCERNS
SECURITY LAYER
Azure Key Vault
• API Keys
• Connection Strings
• Certificates
• Secrets Rotation
Managed Identities
• App Service → Key Vault
• Functions → Key Vault
• No hardcoded credentials
MONITORING LAYER

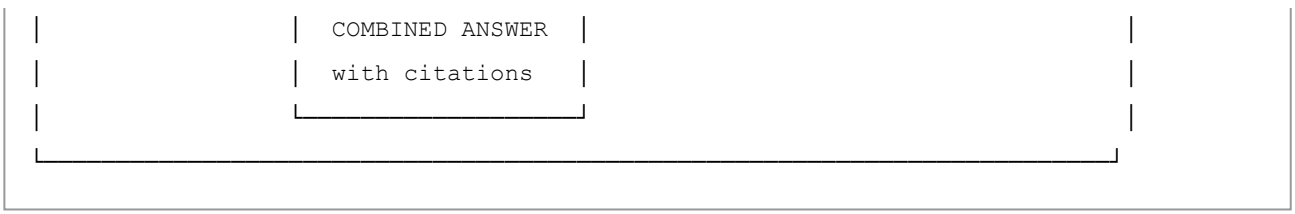


Detailed Component Diagrams

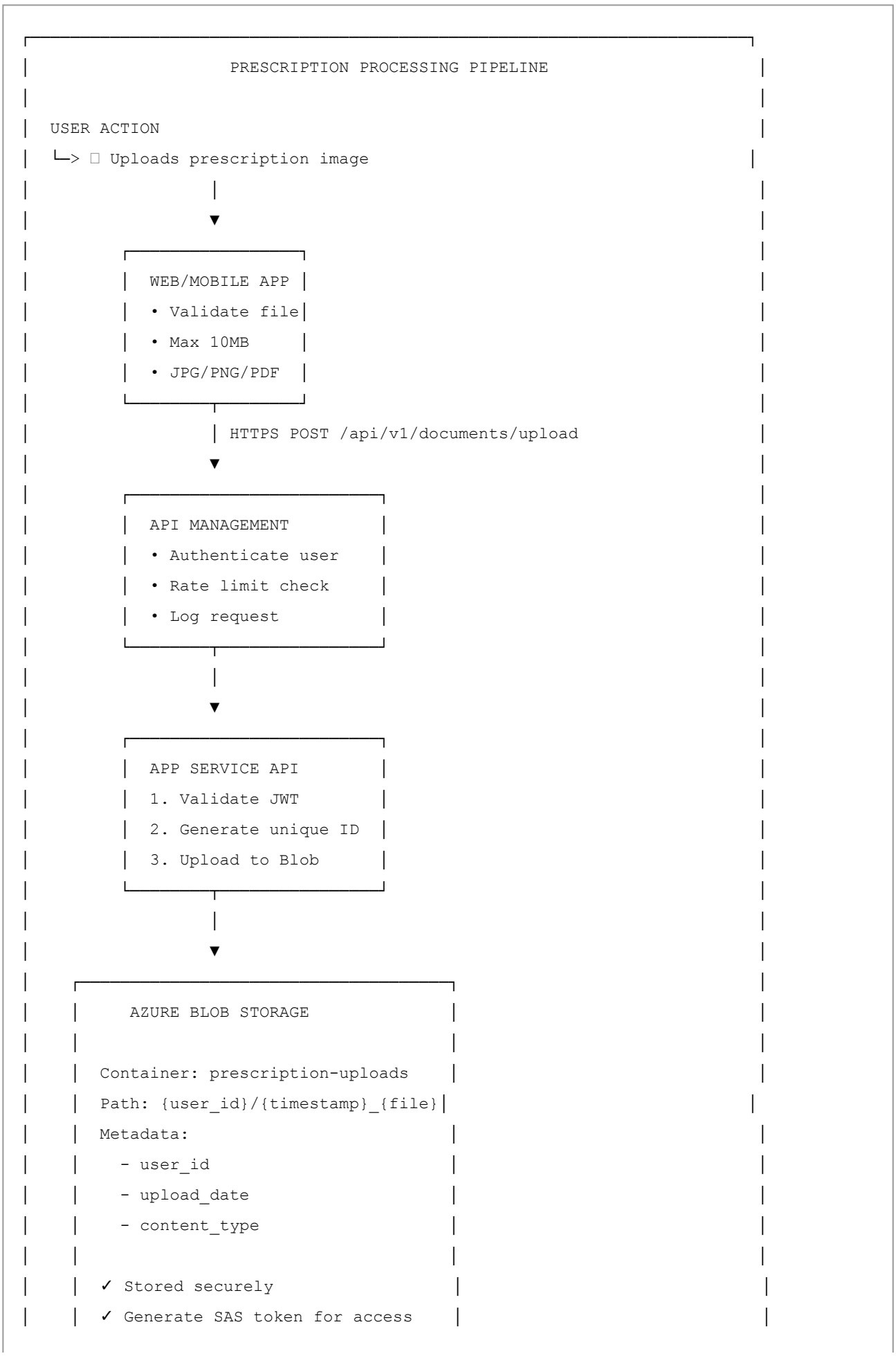
1. AI Processing Architecture

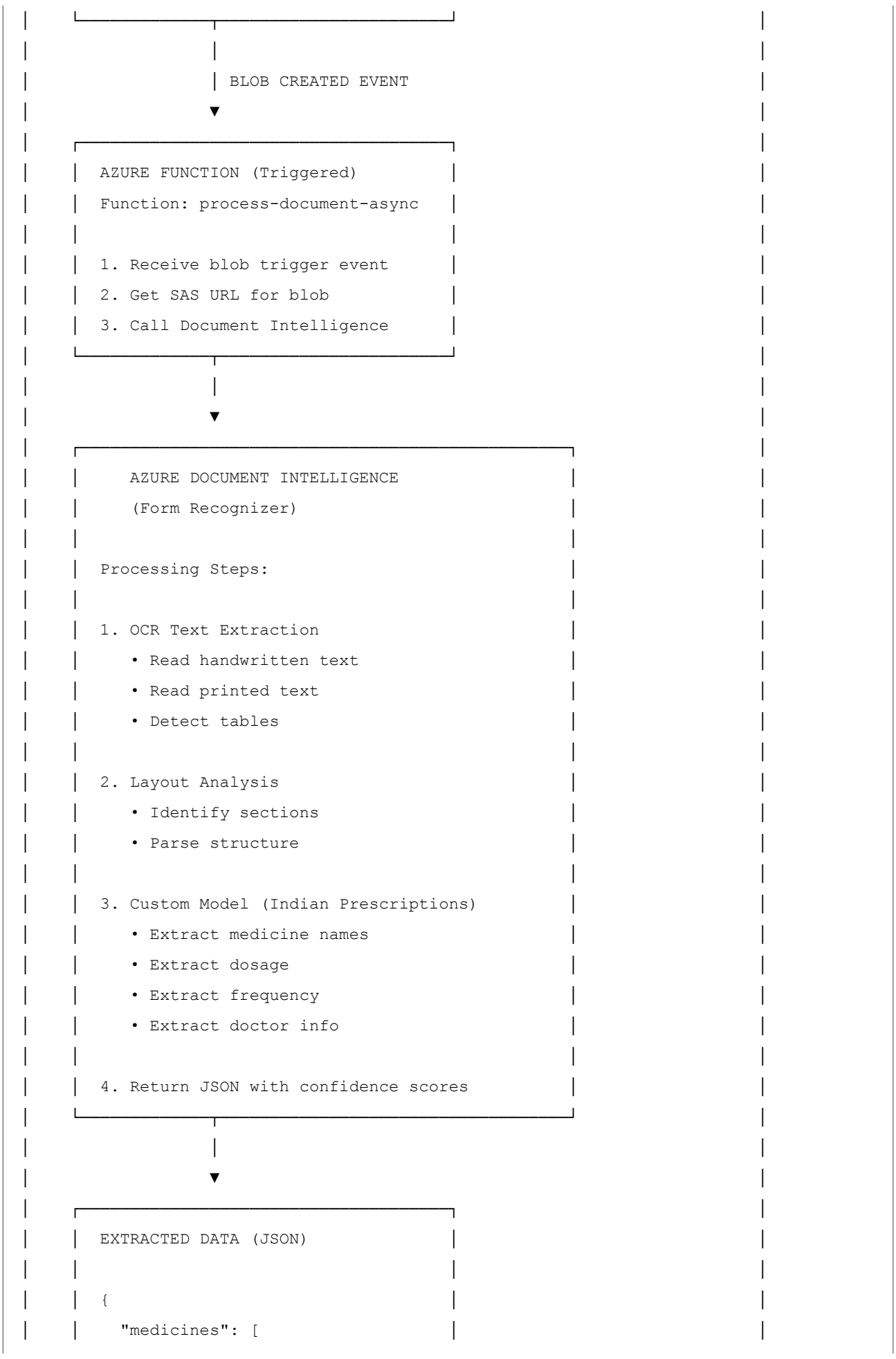

AI MULTI-AGENT SYSTEM
USER QUERY INPUT
"Can I take aspirin with my diabetes medication?"
ORCHESTRATOR AGENT
(Azure OpenAI GPT-4)
1. Analyze query intent
2. Identify entities (aspirin, diabetes medication)
3. Determine agent routing:
→ Drug Info Agent
→ Medical Q&A Agent
4. Manage conversation context
Temperature: 0.3 (consistent)
Max Tokens: 500
DRUG INFO AGENT
(BioGPT)
1. Query RAG
2. Search DB
3. Generate
Temp: 0.2
(Factual)
MEDICAL Q&A AGENT
(Med42-Llama3)
1. Query RAG
2. Retrieve context
3. Generate answer
Temp: 0.5
(Balanced)

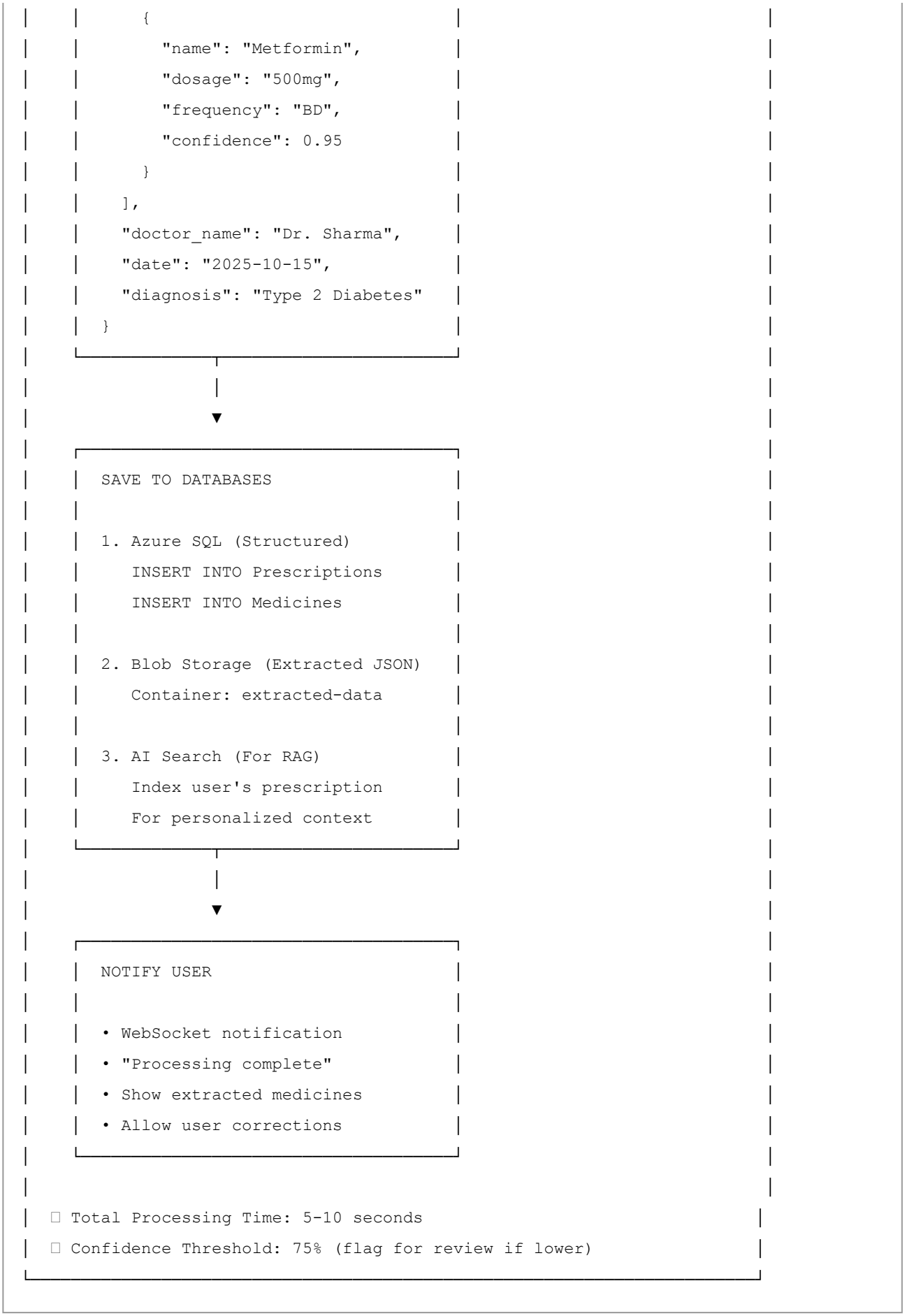

RAG SYSTEM
(Azure AI Search)
Vector Search Process:
1. Convert query to embedding (1536D)
2. Similarity search in vector DB (Cosine similarity)
3. Retrieve top-K docs (K=5)
4. Return context with relevance scores
KNOWLEDGE SOURCES
Medical Knowledge
• WHO Guidelines
• ICMR Advisories
• Research Papers
Drug Database
• CDSCO Data
• Interactions
• Side Effects
User Documents
• Prescriptions
• Medical History



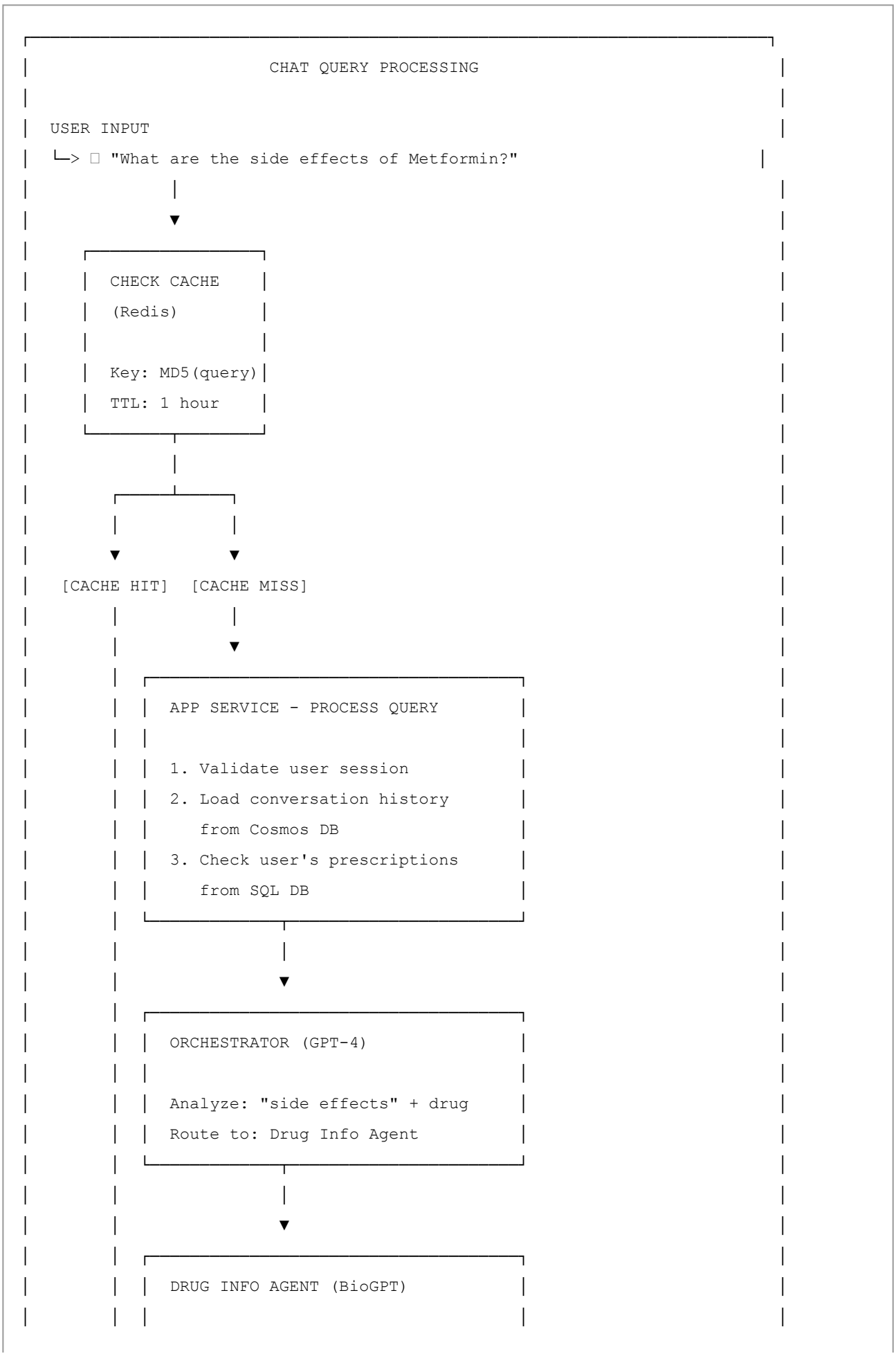
2. Document Processing Flow

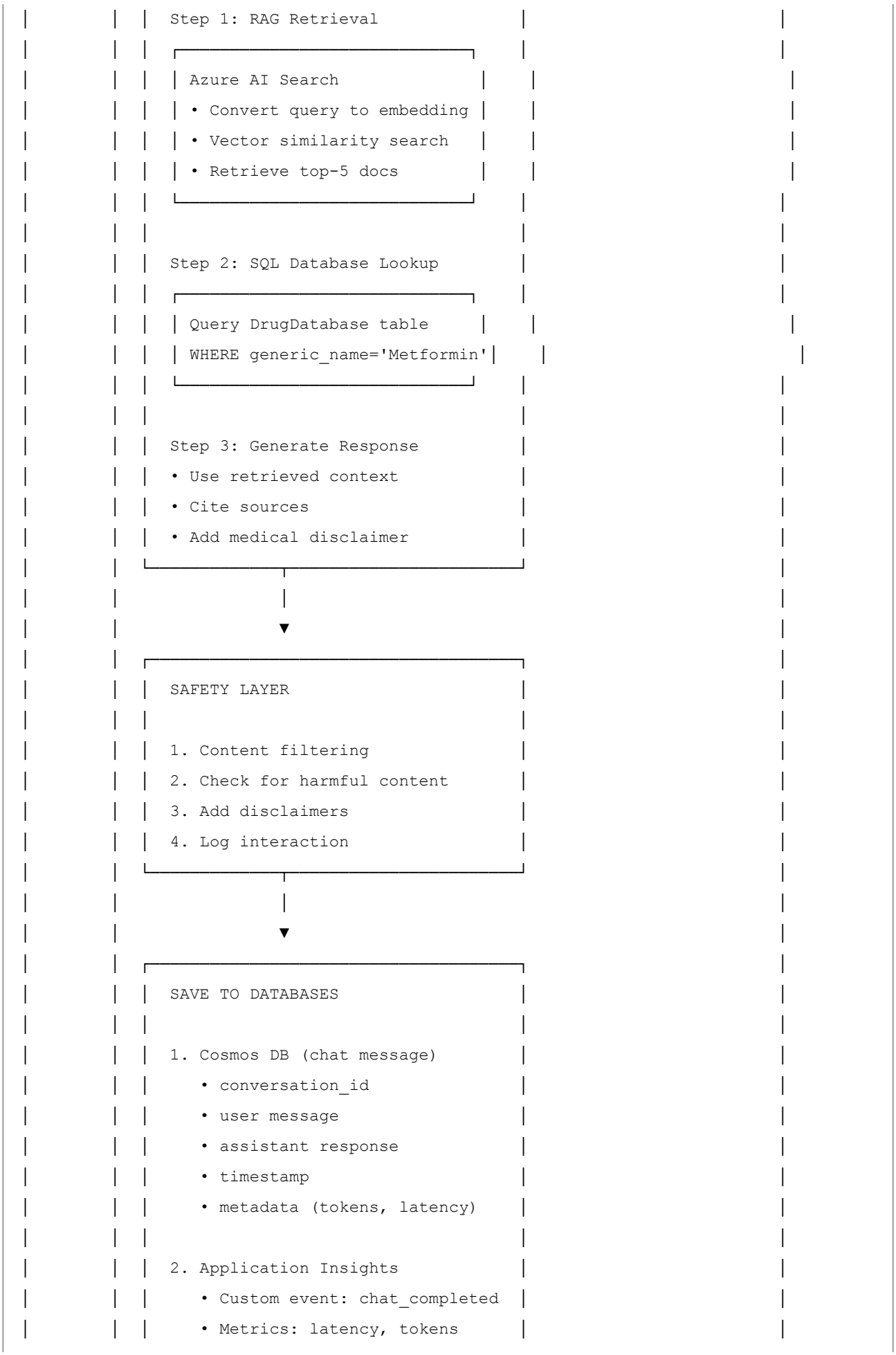

PRESCRIPTION PROCESSING PIPELINE
USER ACTION
└> □ Uploads prescription image
WEB/MOBILE APP
• Validate file
• Max 10MB
• JPG/PNG/PDF
HTTPS POST /api/v1/documents/upload
API MANAGEMENT
• Authenticate user
• Rate limit check
• Log request
APP SERVICE API
1. Validate JWT
2. Generate unique ID
3. Upload to Blob
AZURE BLOB STORAGE
Container: prescription-uploads
Path: {user_id}/{timestamp}_{file}
Metadata:
- user_id
- upload_date
- content_type
✓ Stored securely
✓ Generate SAS token for access

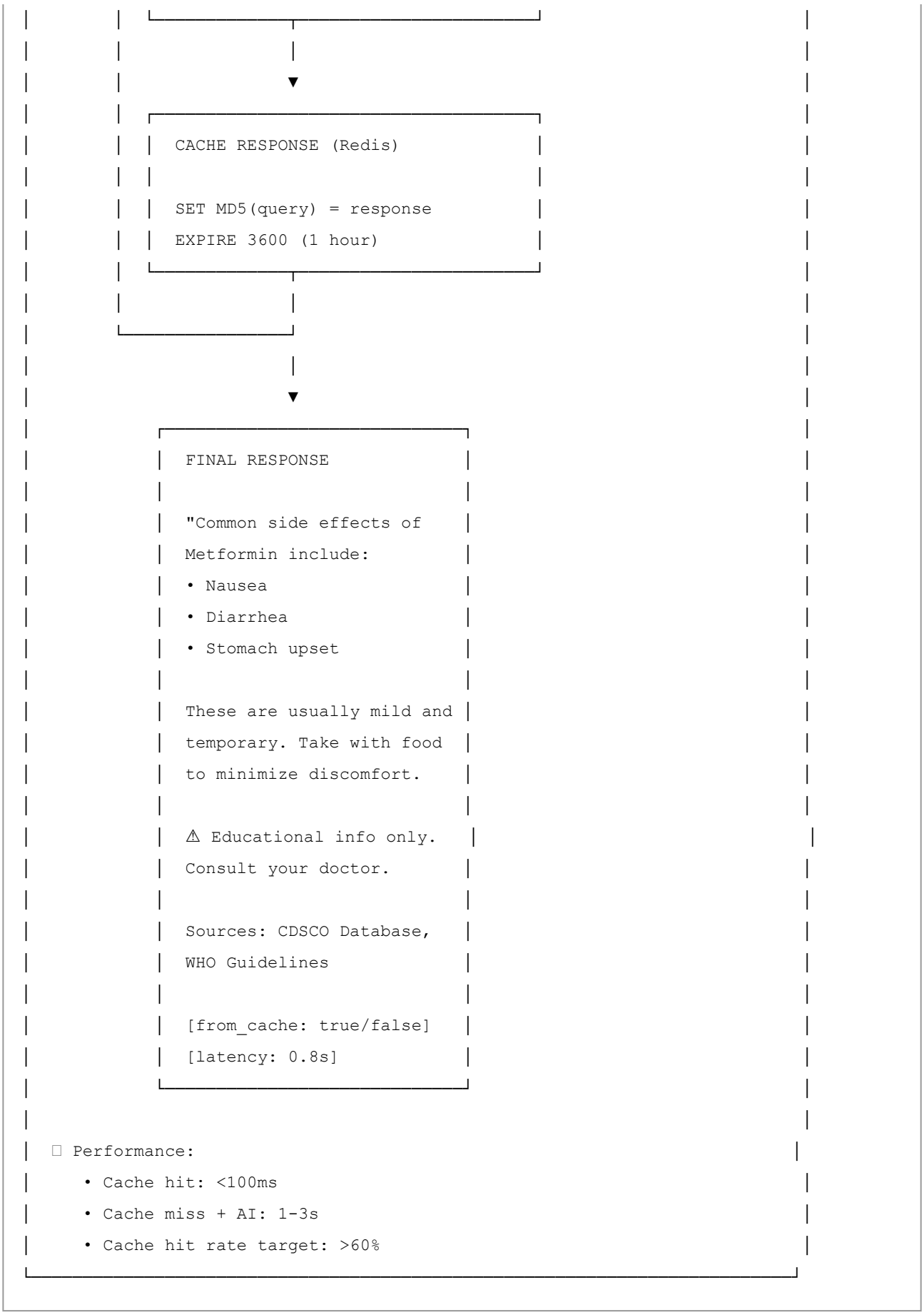

BLOB CREATED EVENT
AZURE FUNCTION (Triggered)
Function: process-document-async
1. Receive blob trigger event
2. Get SAS URL for blob
3. Call Document Intelligence
AZURE DOCUMENT INTELLIGENCE
(Form Recognizer)
Processing Steps:
1. OCR Text Extraction
• Read handwritten text
• Read printed text
• Detect tables
2. Layout Analysis
• Identify sections
• Parse structure
3. Custom Model (Indian Prescriptions)
• Extract medicine names
• Extract dosage
• Extract frequency
• Extract doctor info
4. Return JSON with confidence scores
EXTRACTED DATA (JSON)
{
"medicines": [


{
"name": "Metformin",
"dosage": "500mg",
"frequency": "BD",
"confidence": 0.95
}
],
"doctor_name": "Dr. Sharma",
"date": "2025-10-15",
"diagnosis": "Type 2 Diabetes"
}
SAVE TO DATABASES
1. Azure SQL (Structured)
INSERT INTO Prescriptions
INSERT INTO Medicines
2. Blob Storage (Extracted JSON)
Container: extracted-data
3. AI Search (For RAG)
Index user's prescription
For personalized context
NOTIFY USER
• WebSocket notification
• "Processing complete"
• Show extracted medicines
• Allow user corrections
Total Processing Time: 5-10 seconds
Confidence Threshold: 75% (flag for review if lower)

3. Chat Query Flow

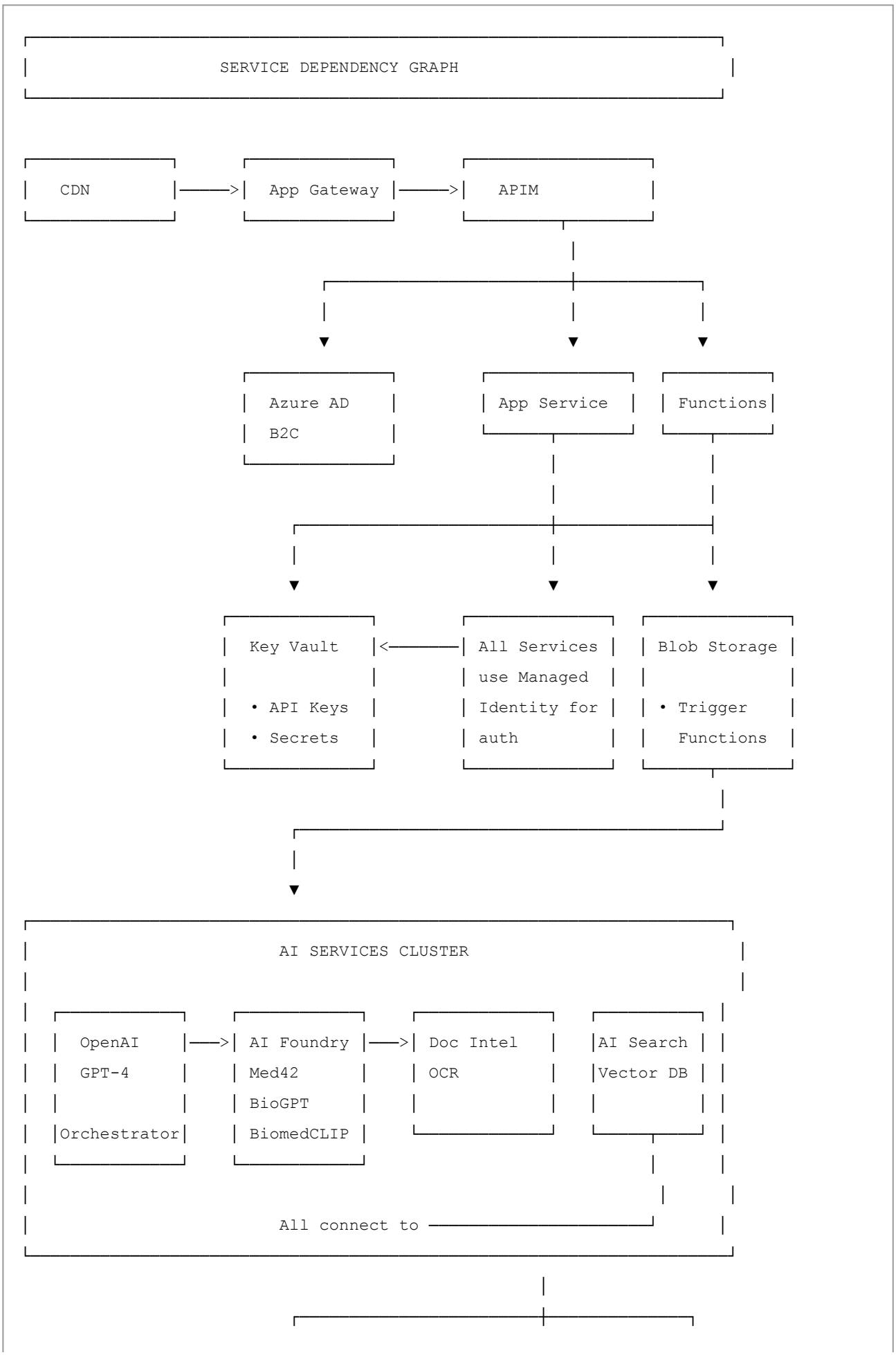

CHAT QUERY PROCESSING
USER INPUT
"What are the side effects of Metformin?"
CHECK CACHE
(Redis)
Key: MD5(query)
TTL: 1 hour
[CACHE HIT]
[CACHE MISS]
APP SERVICE - PROCESS QUERY
1. Validate user session
2. Load conversation history
from Cosmos DB
3. Check user's prescriptions
from SQL DB
ORCHESTRATOR (GPT-4)
Analyze: "side effects" + drug
Route to: Drug Info Agent
DRUG INFO AGENT (BioGPT)

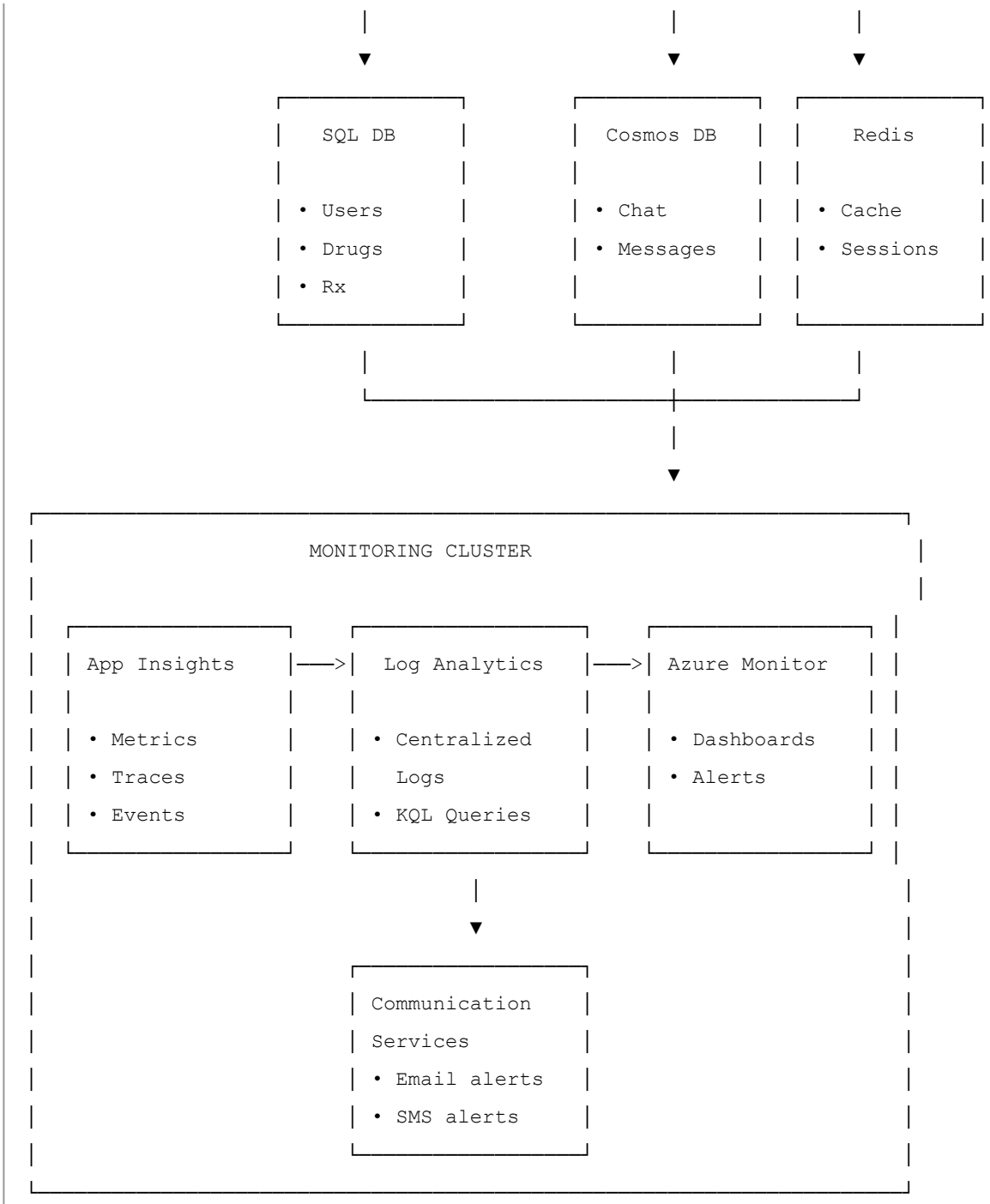

Step 1: RAG Retrieval
Azure AI Search
• Convert query to embedding
• Vector similarity search
• Retrieve top-5 docs
Step 2: SQL Database Lookup
Query DrugDatabase table
WHERE generic_name='Metformin'
Step 3: Generate Response
• Use retrieved context
• Cite sources
• Add medical disclaimer
SAFETY LAYER
1. Content filtering
2. Check for harmful content
3. Add disclaimers
4. Log interaction
SAVE TO DATABASES
1. Cosmos DB (chat message)
• conversation_id
• user message
• assistant response
• timestamp
• metadata (tokens, latency)
2. Application Insights
• Custom event: chat_completed
• Metrics: latency, tokens


CACHE RESPONSE (Redis)
SET MD5(query) = response
EXPIRE 3600 (1 hour)
FINAL RESPONSE
"Common side effects of
Metformin include:
• Nausea
• Diarrhea
• Stomach upset
These are usually mild and
temporary. Take with food
to minimize discomfort.
⚠ Educational info only.
Consult your doctor.
Sources: CDSCO Database,
WHO Guidelines
[from_cache: true/false]
[latency: 0.8s]
Performance:
• Cache hit: <100ms
• Cache miss + AI: 1-3s
• Cache hit rate target: >60%

Service Integration Map

Complete Service Dependencies


SERVICE DEPENDENCY GRAPH
CDN
App Gateway
APIM
Azure AD B2C
App Service
Functions
Key Vault
• API Keys
• Secrets
All Services use Managed Identity for auth
Blob Storage
• Trigger Functions
AI SERVICES CLUSTER
OpenAI GPT-4
Orchestrator
AI Foundry
Med42
BioGPT
BiomedCLIP
Doc Intel
OCR
AI Search
Vector DB
All connect to



KEY:

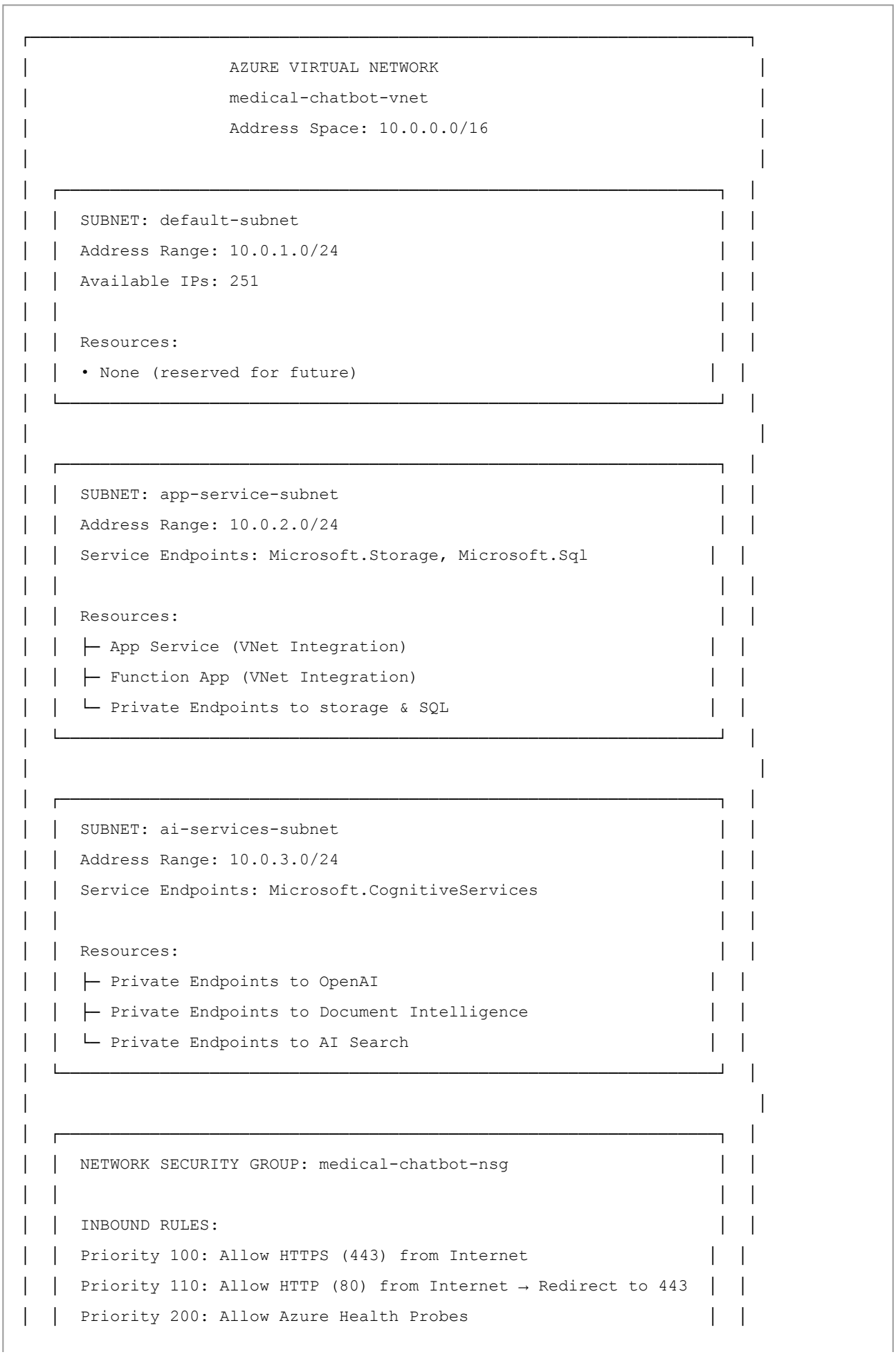
———> Direct dependency (calls service)

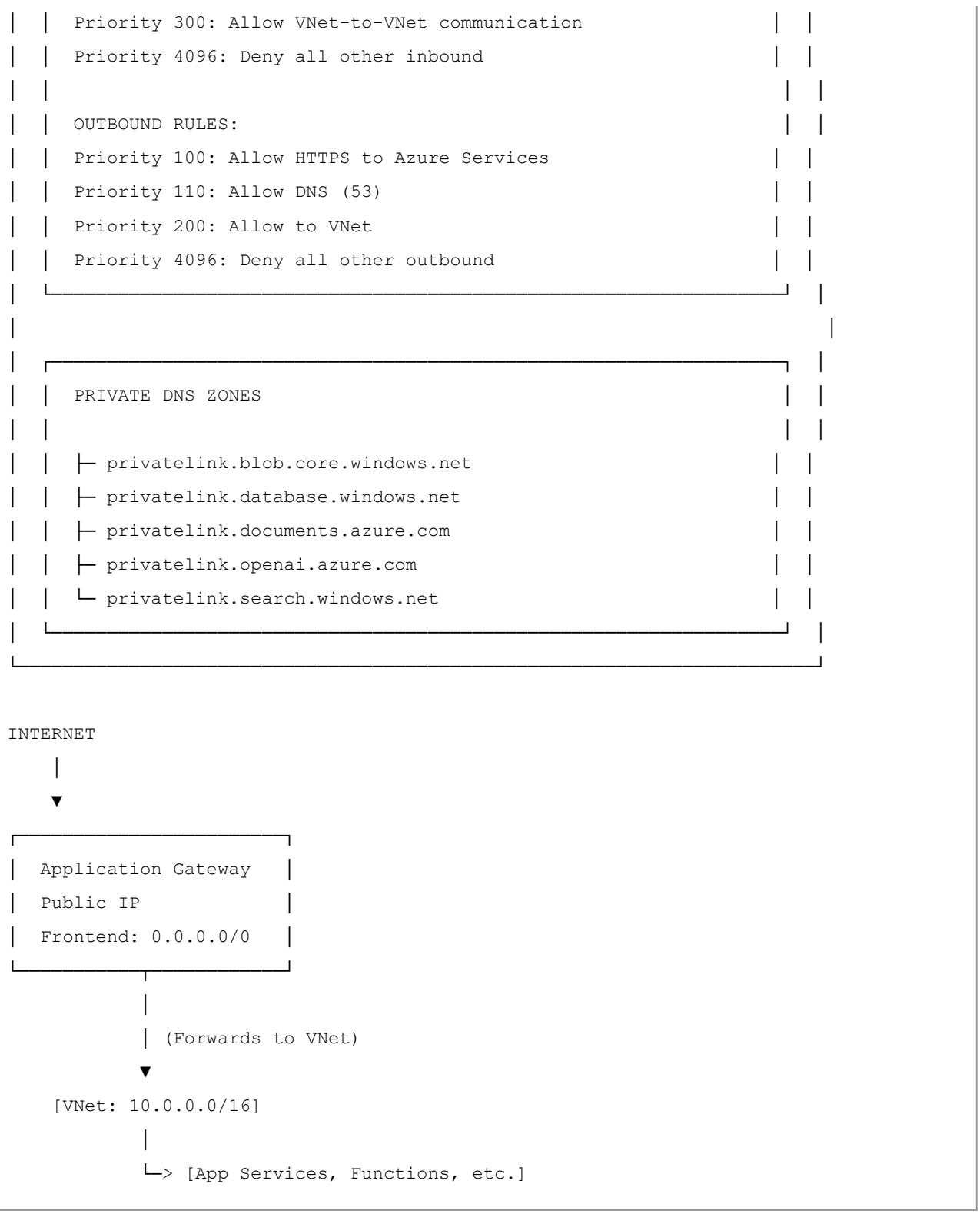
- - -> Indirect dependency (uses data)

<———> Bidirectional (sends & receives)

Network Architecture

Virtual Network Topology


AZURE VIRTUAL NETWORK
medical-chatbot-vnet
Address Space: 10.0.0.0/16
SUBNET: default-subnet
Address Range: 10.0.1.0/24
Available IPs: 251
Resources:
• None (reserved for future)
SUBNET: app-service-subnet
Address Range: 10.0.2.0/24
Service Endpoints: Microsoft.Storage, Microsoft.Sql
Resources:
├ App Service (VNet Integration)
├ Function App (VNet Integration)
└ Private Endpoints to storage & SQL
SUBNET: ai-services-subnet
Address Range: 10.0.3.0/24
Service Endpoints: Microsoft.CognitiveServices
Resources:
├ Private Endpoints to OpenAI
├ Private Endpoints to Document Intelligence
└ Private Endpoints to AI Search
NETWORK SECURITY GROUP: medical-chatbot-nsg
INBOUND RULES:
Priority 100: Allow HTTPS (443) from Internet
Priority 110: Allow HTTP (80) from Internet → Redirect to 443
Priority 200: Allow Azure Health Probes



Security Architecture

Defense in Depth

```
┌─────────────────────────────────────────────────────────────────────┐
│                    SECURITY LAYERS (Defense in Depth)               │
└─────────────────────────────────────────────────────────────────────┘

LAYER 7: USER/APPLICATION
┌─────────────────────────────────────────────────────────────────────┐
│  □ Azure AD B2C Authentication                                      │
│     • Multi-factor authentication (MFA)                             │
│     • Passwordless (phone/email OTP)                                │
│     • Social login (Google, Facebook)                               │
│     • JWT tokens with 1-hour expiry                                 │
│     • Refresh tokens with 30-day expiry                             │
└─────────────────────────────────────────────────────────────────────┘

LAYER 6: EDGE/PERIMETER
┌─────────────────────────────────────────────────────────────────────┐
│  □ Web Application Firewall (WAF)                                   │
│     • OWASP Top 10 protection                                       │
│     • Custom rules for medical data                                 │
│     • SQL injection prevention                                      │
│     • XSS (Cross-Site Scripting) protection                         │
│     • DDoS protection (Application Gateway)                         │
└─────────────────────────────────────────────────────────────────────┘

LAYER 5: API GATEWAY
┌─────────────────────────────────────────────────────────────────────┐
│  □ API Management Security                                          │
│     • Rate limiting: 100 req/min per user                           │
│     • IP whitelisting/blacklisting                                  │
│     • JWT validation before backend                                 │
│     • API key rotation every 90 days                                │
│     • Request/response validation                                   │
│     • CORS policy enforcement                                       │
└─────────────────────────────────────────────────────────────────────┘

LAYER 4: APPLICATION
┌─────────────────────────────────────────────────────────────────────┐
│  □ Application Security                                             │
│     • Managed Identity (no stored credentials)                      │
│     • Input validation and sanitization                             │
│     • Output encoding                                               │
│     • HTTPS only (TLS 1.3)                                          │
│     • Secure headers (HSTS, CSP, X-Frame-Options)                   │
```

```
│     • Session management (Redis with secure cookies)                    │
└──────────────────────────────────────────────────────────────────────────┘
```

LAYER 3: NETWORK

```
┌──────────────────────────────────────────────────────────────────────────┐
│  □ Network Security                                                      │
│     • Virtual Network isolation                                          │
│     • Network Security Groups (NSGs)                                     │
│     • Private Endpoints for Azure services                               │
│     • No public access to databases                                      │
│     • Service Endpoints for trusted services                             │
│     • Private DNS zones                                                  │
└──────────────────────────────────────────────────────────────────────────┘
```

LAYER 2: DATA

```
┌──────────────────────────────────────────────────────────────────────────┐
│  □ Data Security                                                         │
│     • Encryption at rest: AES-256                                        │
│     • Encryption in transit: TLS 1.3                                     │
│     • Field-level encryption for PII                                     │
│     • Soft delete enabled (7-90 days)                                    │
│     • Automated backups (geo-redundant)                                  │
│     • Access auditing and logging                                        │
└──────────────────────────────────────────────────────────────────────────┘
```

LAYER 1: SECRETS MANAGEMENT

```
┌──────────────────────────────────────────────────────────────────────────┐
│  □ Azure Key Vault                                                       │
│     • All secrets stored centrally                                       │
│     • Hardware Security Module (HSM) backed                              │
│     • Access via Managed Identity only                                   │
│     • Secret rotation every 90 days                                      │
│     • Audit logging for all access                                       │
│     • Purge protection enabled                                           │
│     • Soft delete (90 days)                                              │
└──────────────────────────────────────────────────────────────────────────┘
```

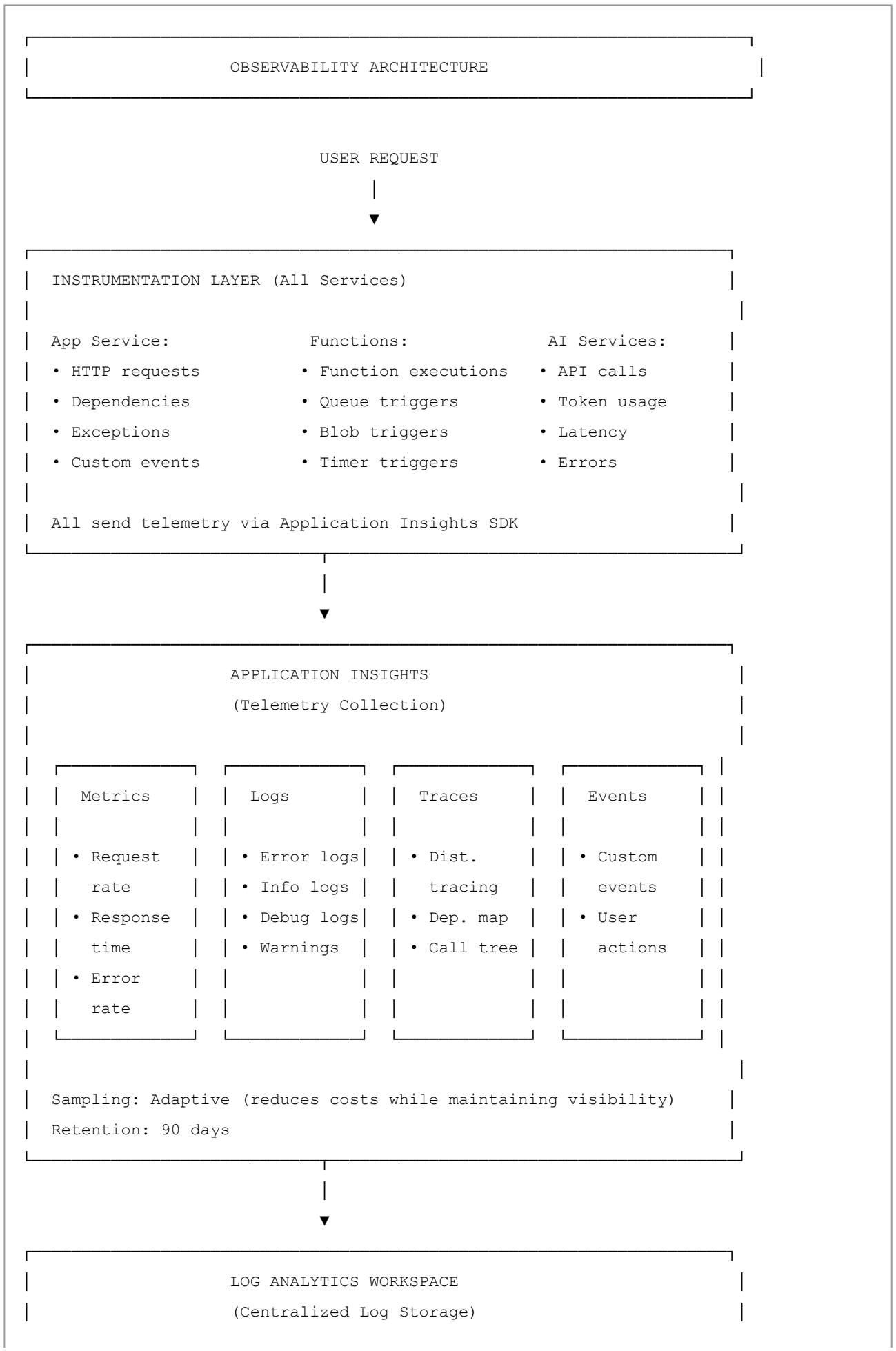
LAYER 0: IDENTITY & ACCESS

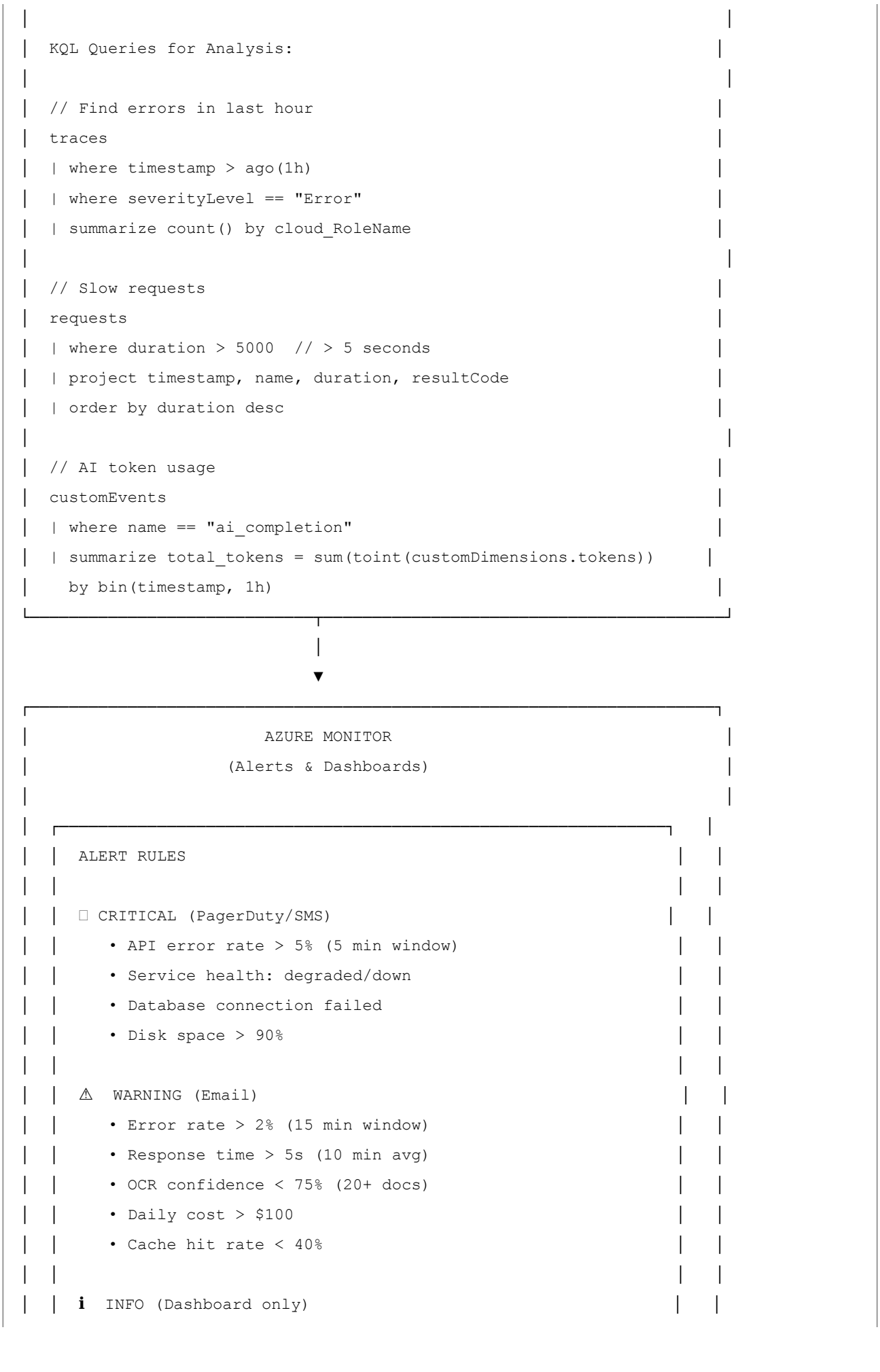
```
┌──────────────────────────────────────────────────────────────────────────┐
│  □ Role-Based Access Control (RBAC)                                      │
│     • Principle of least privilege                                       │
│     • Separate dev/staging/prod access                                   │
│     • Managed identities for service-to-service                          │
│     • No service principals with passwords                               │
```

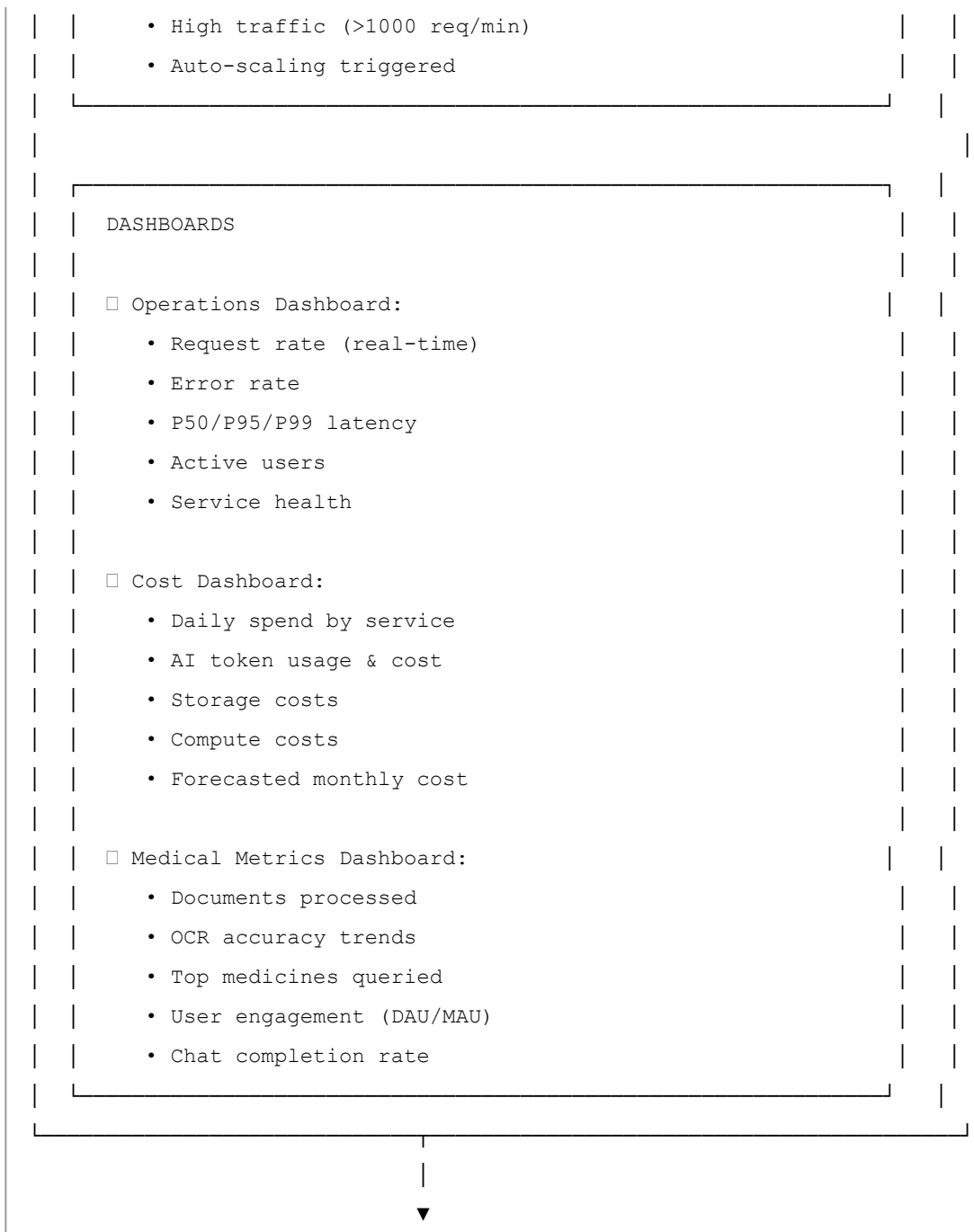
```
│     • MFA required for admin access                                    │
│     • Regular access reviews                                           │
└────────────────────────────────────────────────────────────────────────┘


CROSS-CUTTING SECURITY CONTROLS

┌────────────────────────────────────────────────────────────────────────┐
│  □ Monitoring & Compliance                                             │
│     • Azure Security Center (Defender for Cloud)                       │
│     • Continuous compliance scanning                                   │
│     • Threat detection and alerts                                      │
│     • Vulnerability scanning                                           │
│     • Security score monitoring                                        │
│     • Incident response automation                                     │
│                                                                        │
│  □ Audit & Compliance                                                  │
│     • All actions logged to Log Analytics                              │
│     • 7-year retention for compliance                                  │
│     • DPDPA compliance (India)                                         │
│     • GDPR-ready architecture                                          │
│     • HIPAA-equivalent controls                                        │
│     • Regular security audits                                          │
└────────────────────────────────────────────────────────────────────────┘
```

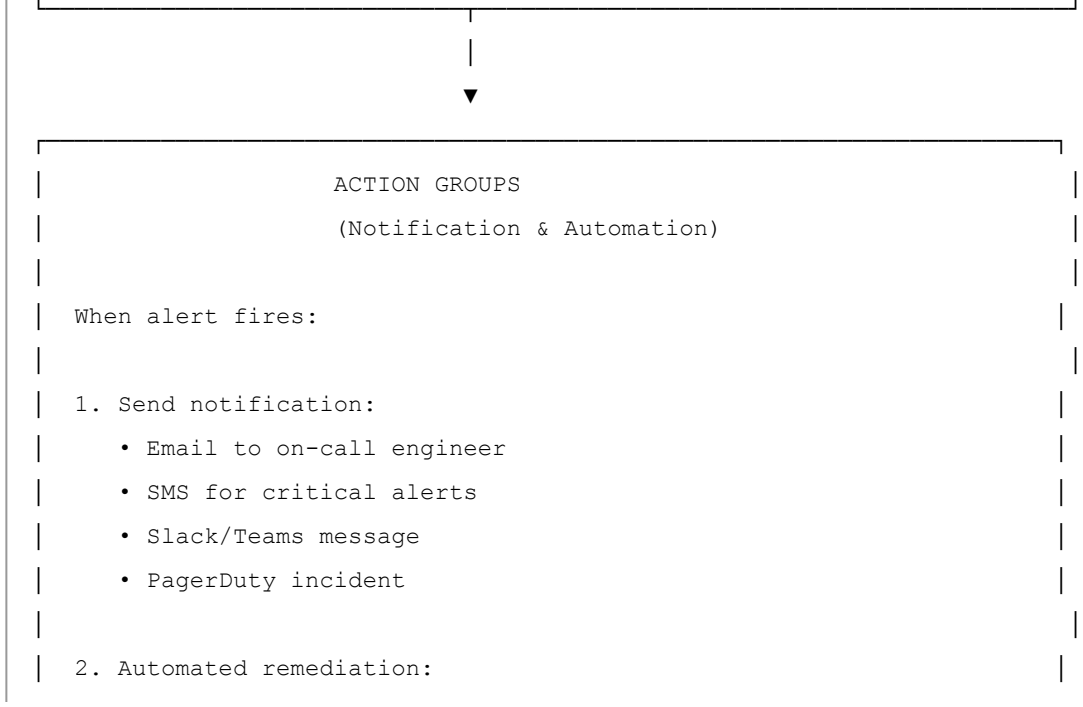
Monitoring & Observability

Observability Stack


OBSERVABILITY ARCHITECTURE
USER REQUEST
INSTRUMENTATION LAYER (All Services)
App Service:
• HTTP requests
• Dependencies
• Exceptions
• Custom events
Functions:
• Function executions
• Queue triggers
• Blob triggers
• Timer triggers
AI Services:
• API calls
• Token usage
• Latency
• Errors
All send telemetry via Application Insights SDK
APPLICATION INSIGHTS
(Telemetry Collection)
Metrics
• Request rate
• Response time
• Error rate
Logs
• Error logs
• Info logs
• Debug logs
• Warnings
Traces
• Dist. tracing
• Dep. map
• Call tree
Events
• Custom events
• User actions
Sampling: Adaptive (reduces costs while maintaining visibility)
Retention: 90 days
LOG ANALYTICS WORKSPACE
(Centralized Log Storage)


KQL Queries for Analysis:
// Find errors in last hour
traces
| where timestamp > ago(1h)
| where severityLevel == "Error"
| summarize count() by cloud_RoleName
// Slow requests
requests
| where duration > 5000 // > 5 seconds
| project timestamp, name, duration, resultCode
| order by duration desc
// AI token usage
customEvents
| where name == "ai_completion"
| summarize total_tokens = sum(toint(customDimensions.tokens))
by bin(timestamp, 1h)
AZURE MONITOR
(Alerts & Dashboards)
ALERT RULES
CRITICAL (PagerDuty/SMS)
• API error rate > 5% (5 min window)
• Service health: degraded/down
• Database connection failed
• Disk space > 90%
WARNING (Email)
• Error rate > 2% (15 min window)
• Response time > 5s (10 min avg)
• OCR confidence < 75% (20+ docs)
• Daily cost > $100
• Cache hit rate < 40%
INFO (Dashboard only)


• High traffic (>1000 req/min)
• Auto-scaling triggered
DASHBOARDS
Operations Dashboard:
• Request rate (real-time)
• Error rate
• P50/P95/P99 latency
• Active users
• Service health
Cost Dashboard:
• Daily spend by service
• AI token usage & cost
• Storage costs
• Compute costs
• Forecasted monthly cost
Medical Metrics Dashboard:
• Documents processed
• OCR accuracy trends
• Top medicines queried
• User engagement (DAU/MAU)
• Chat completion rate


ACTION GROUPS
(Notification & Automation)
When alert fires:
1. Send notification:
• Email to on-call engineer
• SMS for critical alerts
• Slack/Teams message
• PagerDuty incident
2. Automated remediation:

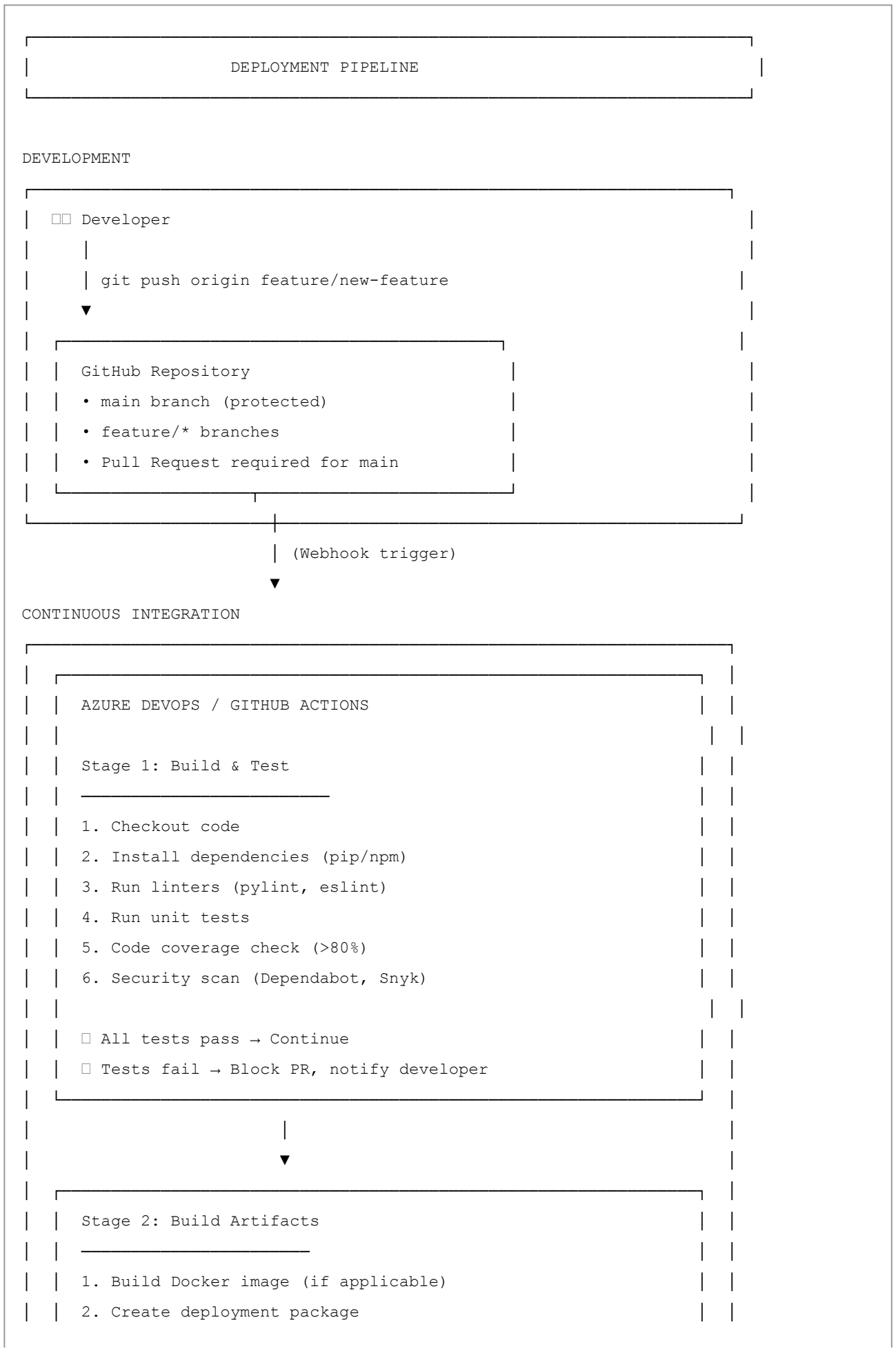
```
│     • Restart unhealthy App Service                                    │
│     • Scale up if CPU > 80%                                            │
│     • Clear cache if memory > 90%                                      │
│     • Run diagnostic scripts                                           │
│                                                                        │
│  3. Create incident ticket:                                            │
│     • Log in ticketing system                                          │
│     • Attach relevant logs                                             │
│     • Assign to team                                                   │
└────────────────────────────────────────────────────────────────────────┘
```

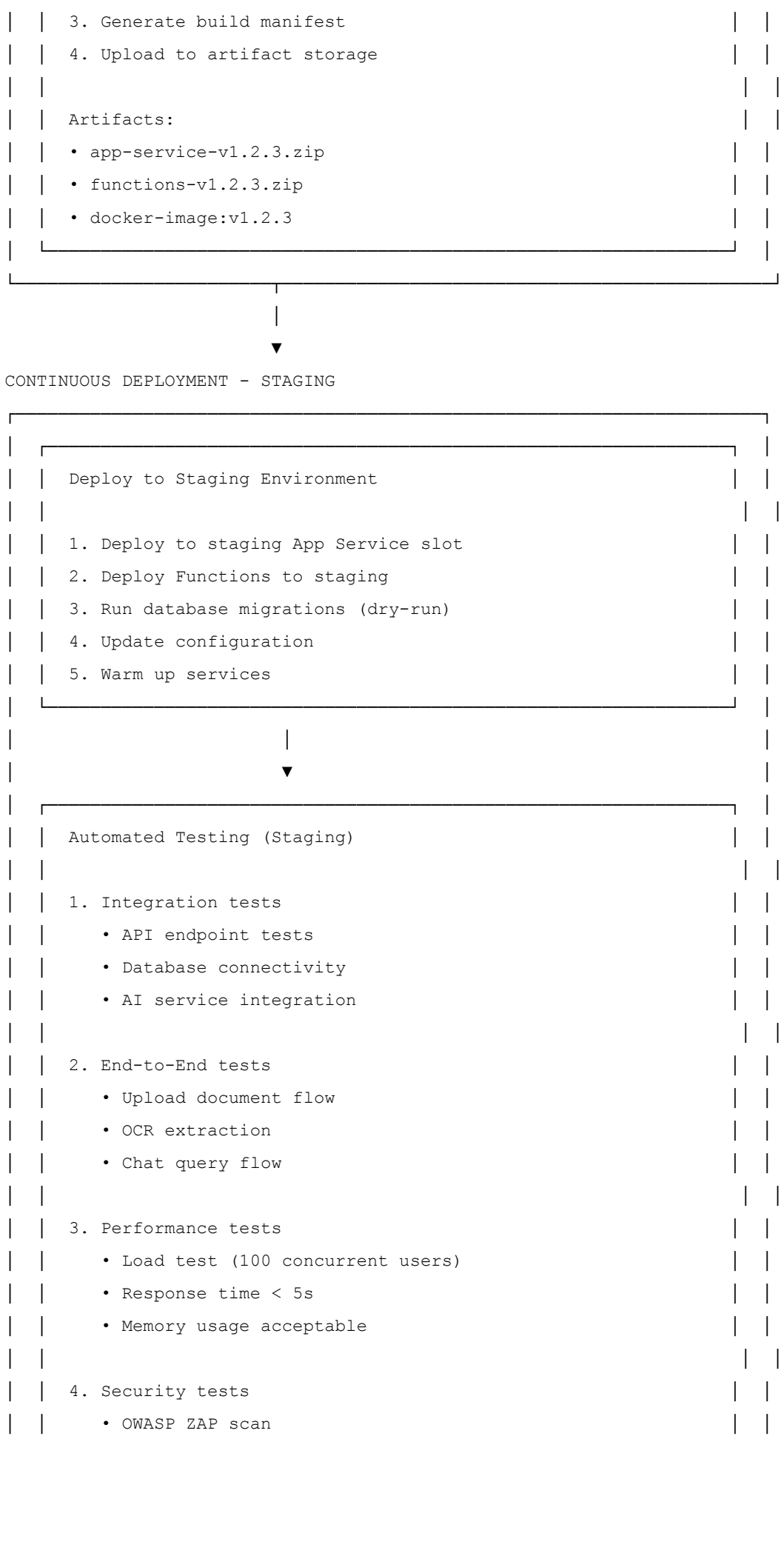
CUSTOM TELEMETRY EVENTS

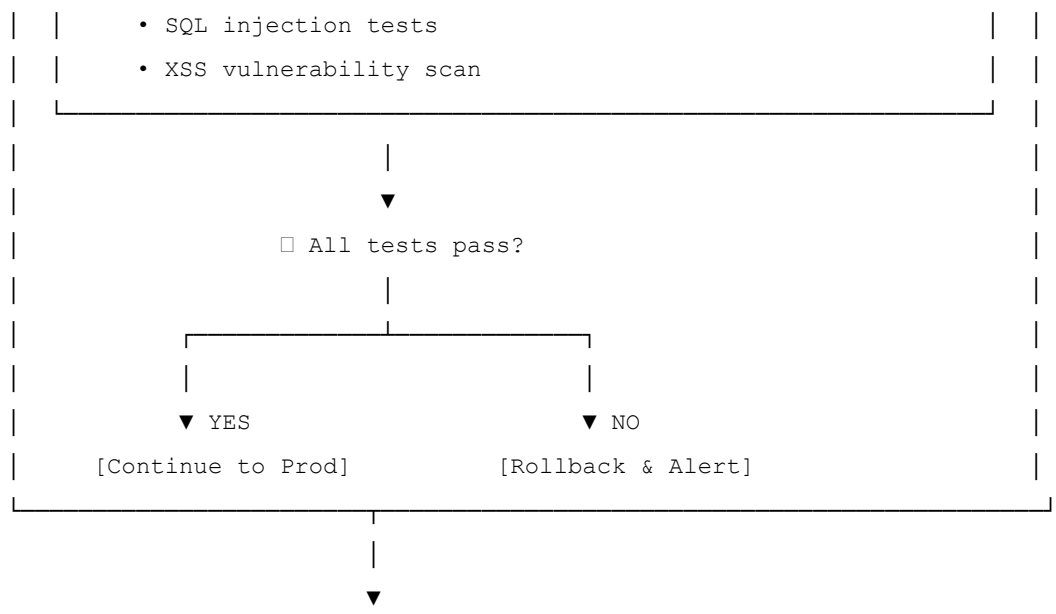
```
┌──────────────────────────────────────────────────────────────────────────┐
│  Event Name              Properties                  Metrics             │
│  ──────────────────────────────────────────────────────────────────────  │
│  document_uploaded       • user_id                   • file_size_mb      │
│                          • file_type                 • upload_time_ms    │
│                          • document_type                                 │
│                                                                          │
│  ocr_completed           • user_id                   • confidence        │
│                          • document_id               • processing_ms     │
│                          • model_version             • field_count       │
│                                                                          │
│  chat_message            • user_id                   • tokens            │
│                          • conversation_id           • latency_ms        │
│                          • agent_used                • cost_usd          │
│                          • cache_hit                                     │
│                                                                          │
│  rag_retrieval           • query                     • num_results       │
│                          • index_name                • search_time_ms    │
│                          • top_score                                     │
│                                                                          │
│  ai_completion           • model                     • tokens_in         │
│                          • deployment                • tokens_out        │
│                          • temperature               • latency_ms        │
│                                                                          │
│  error_occurred          • error_type                • stack_trace       │
│                          • user_id                   • timestamp         │
│                          • service                                       │
└──────────────────────────────────────────────────────────────────────────┘
```

Deployment Architecture

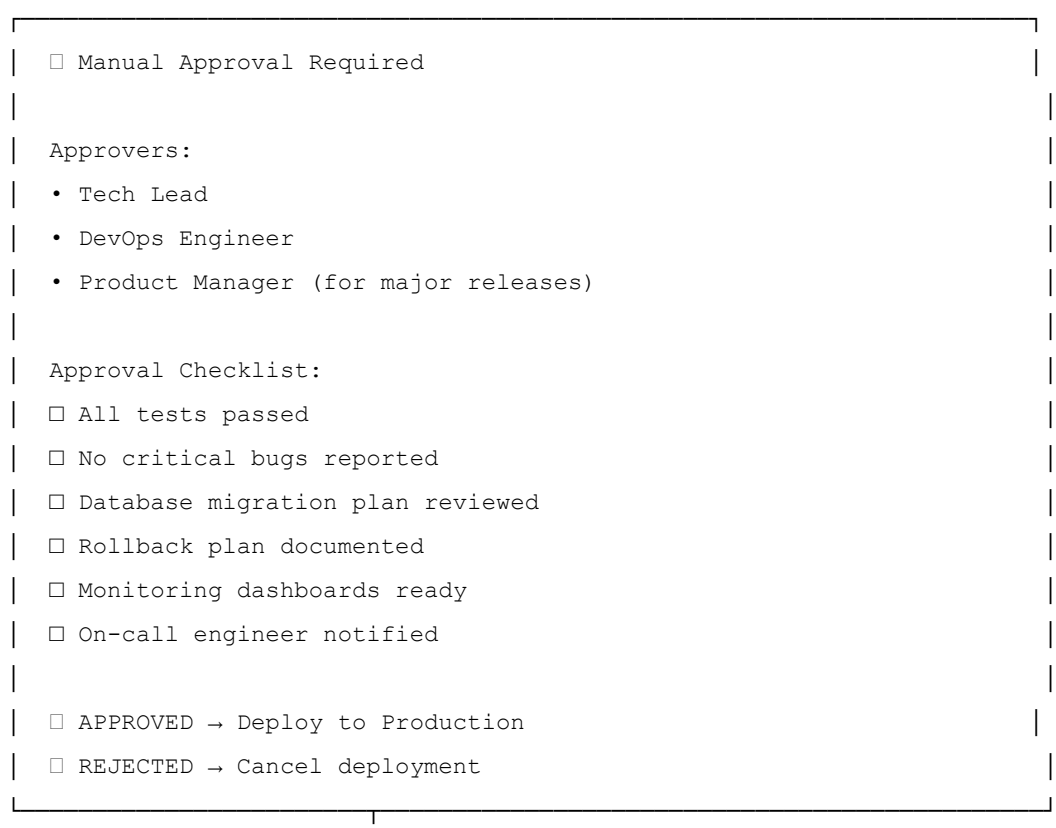
CI/CD Pipeline


DEPLOYMENT PIPELINE
DEVELOPMENT
Developer
git push origin feature/new-feature
GitHub Repository
• main branch (protected)
• feature/* branches
• Pull Request required for main
(Webhook trigger)
CONTINUOUS INTEGRATION
AZURE DEVOPS / GITHUB ACTIONS
Stage 1: Build & Test
1. Checkout code
2. Install dependencies (pip/npm)
3. Run linters (pylint, eslint)
4. Run unit tests
5. Code coverage check (>80%)
6. Security scan (Dependabot, Snyk)
All tests pass → Continue
Tests fail → Block PR, notify developer
Stage 2: Build Artifacts
1. Build Docker image (if applicable)
2. Create deployment package

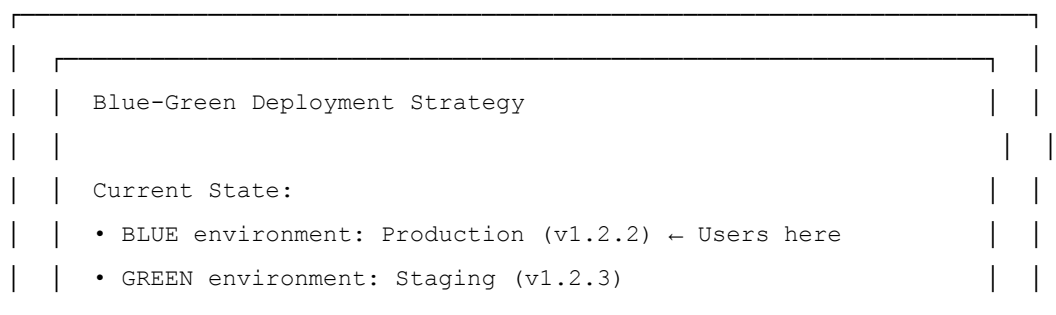

3. Generate build manifest
4. Upload to artifact storage
Artifacts:
• app-service-v1.2.3.zip
• functions-v1.2.3.zip
• docker-image:v1.2.3
CONTINUOUS DEPLOYMENT - STAGING
Deploy to Staging Environment
1. Deploy to staging App Service slot
2. Deploy Functions to staging
3. Run database migrations (dry-run)
4. Update configuration
5. Warm up services
Automated Testing (Staging)
1. Integration tests
• API endpoint tests
• Database connectivity
• AI service integration
2. End-to-End tests
• Upload document flow
• OCR extraction
• Chat query flow
3. Performance tests
• Load test (100 concurrent users)
• Response time < 5s
• Memory usage acceptable
4. Security tests
• OWASP ZAP scan

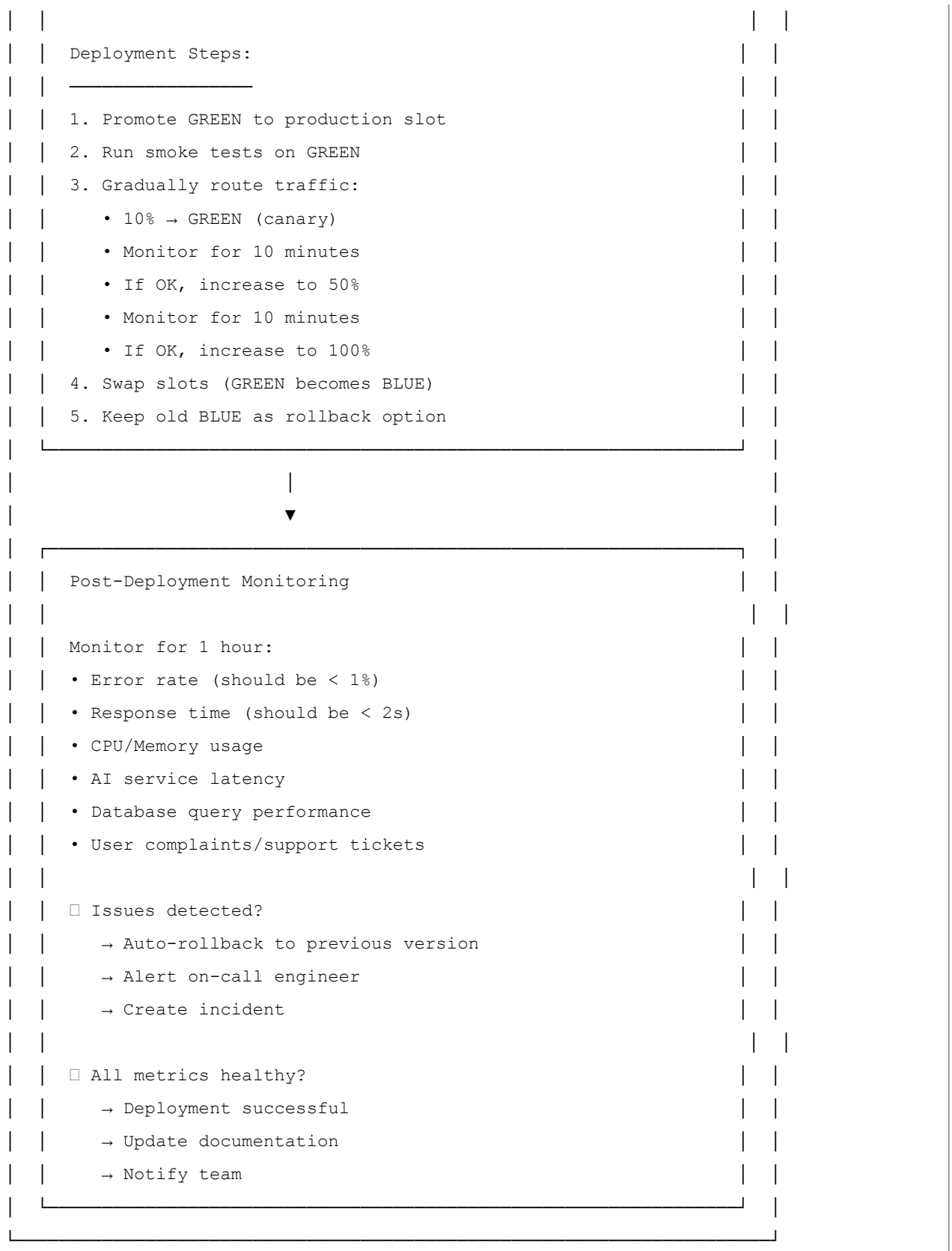


MANUAL APPROVAL (Production Gate)

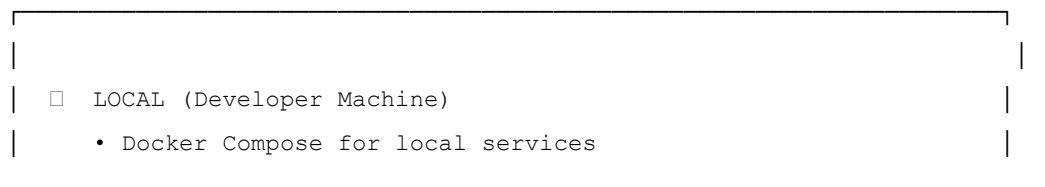


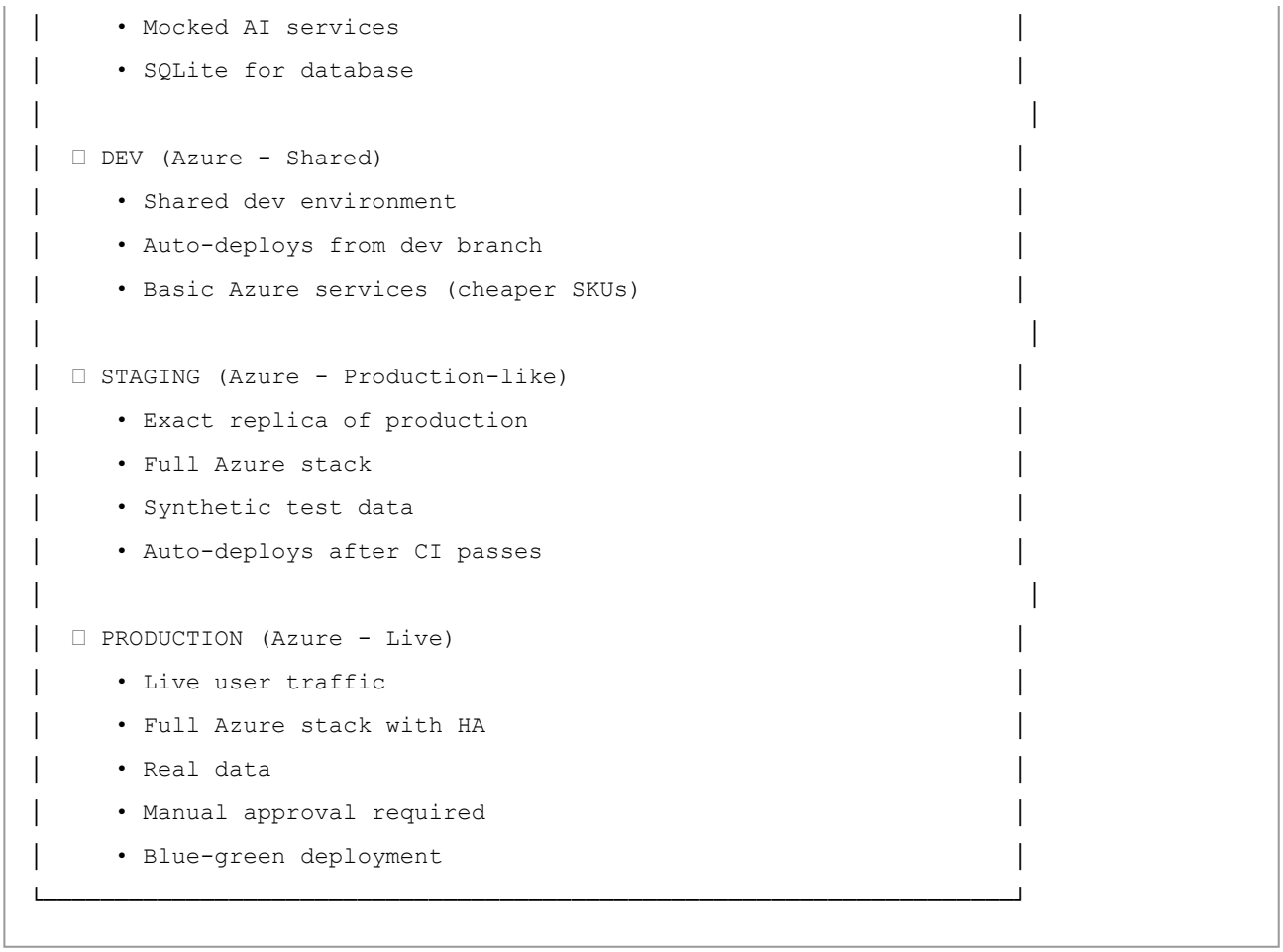
PRODUCTION DEPLOYMENT





ENVIRONMENTS SUMMARY





Summary

Architecture Characteristics

Characteristic	Implementation	Notes
Scalability	Auto-scaling, serverless, CDN	Handles 10x traffic spikes
Availability	Multi-instance, geo-redundancy	99.9% SLA
Security	Defense in depth, zero trust	8 security layers
Performance	Redis cache, CDN, AI Search	<2s response time
Observability	Full telemetry, dashboards	Real-time monitoring
Cost	Pay-per-use, auto-shutdown	$1000-1300/month
Compliance	DPDPA, GDPR-ready	Audit logs, encryption
Maintainability	Microservices, CI/CD	Easy updates, rollbacks

Technology Choices Rationale

Service	Why Chosen	Alternative Considered
Azure OpenAI	Enterprise-ready, GPT-4 access	OpenAI API, AWS Bedrock
Azure AI Search	Native vector search, RAG	Pinecone, Weaviate
Document Intelligence	Best OCR for complex docs	Google Vision, Tesseract

Service	Why Chosen	Alternative Considered
Cosmos DB	Low-latency globally	MongoDB Atlas, DynamoDB
Redis Cache	Industry standard, fast	Memcached
App Service	Managed, auto-scale	AKS, VMs
APIM	Enterprise API gateway	Kong, AWS API Gateway

Document Information

Version: 1.0

Last Updated: October 15, 2025

Maintained By: Medical Chatbot Team

Review Frequency: Quarterly

Related Documents:

- Azure Services Setup Guide
- Azure Configuration Master Sheet
- Medical Chatbot PRD

END OF ARCHITECTURE DOCUMENTATION