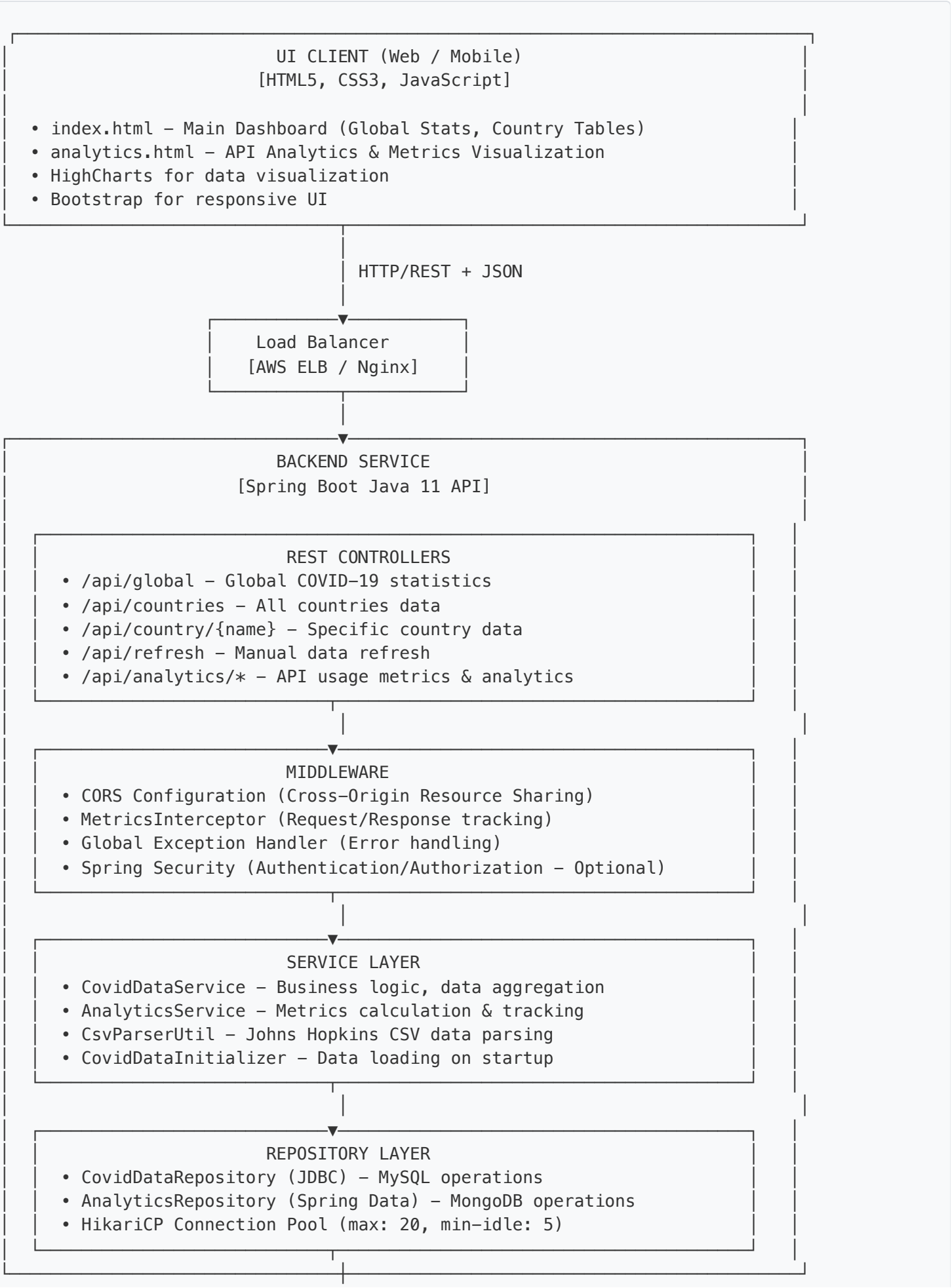
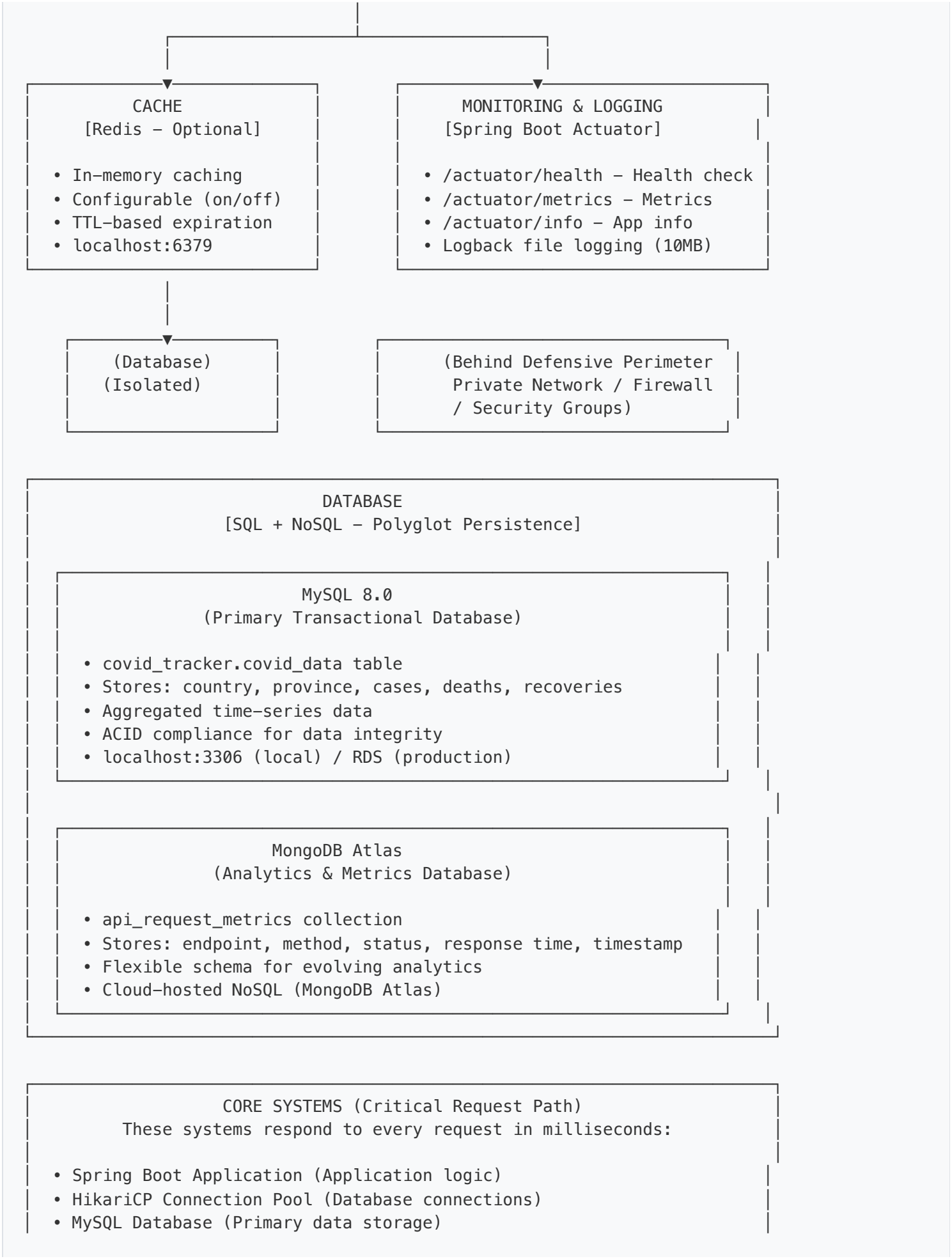
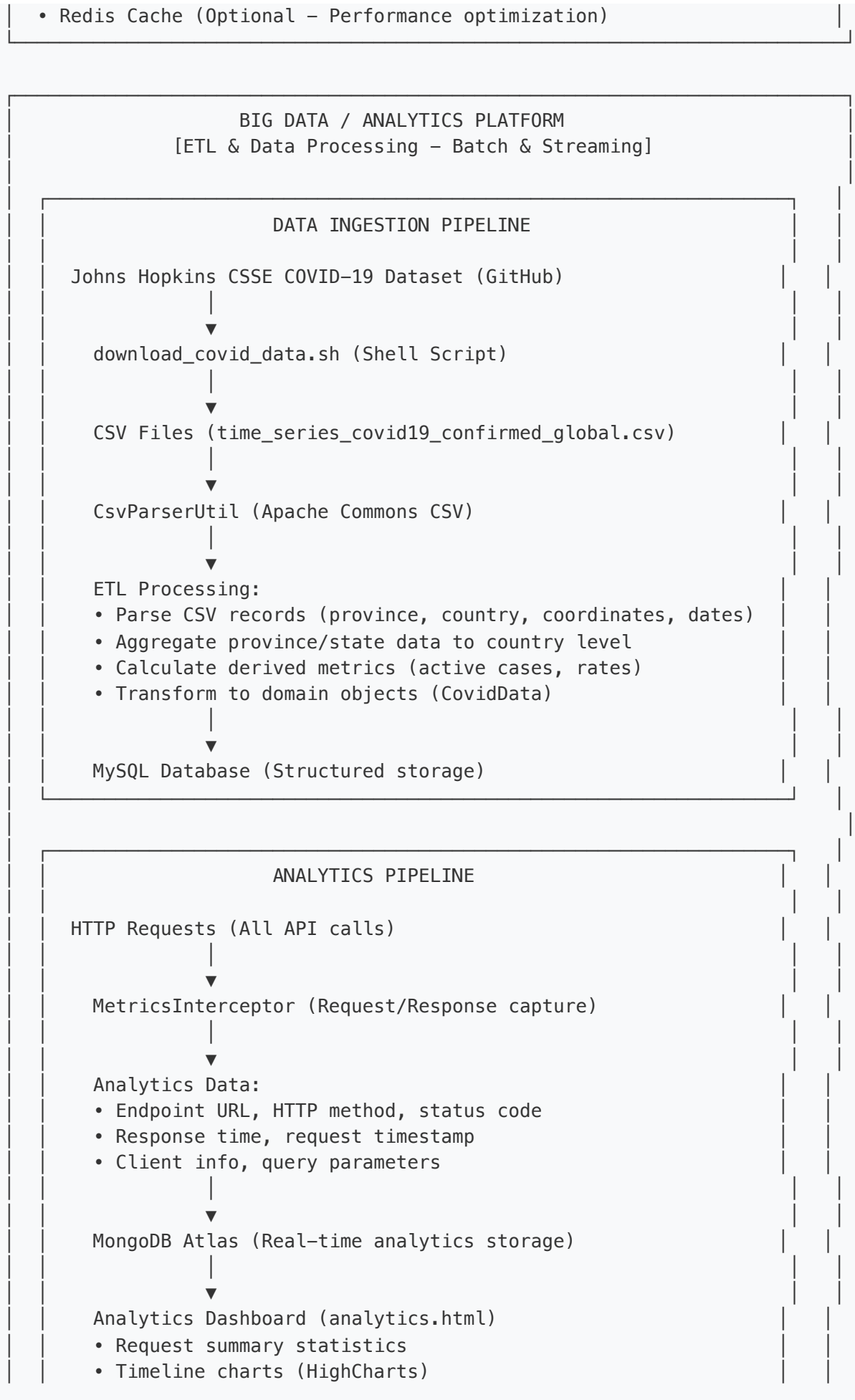
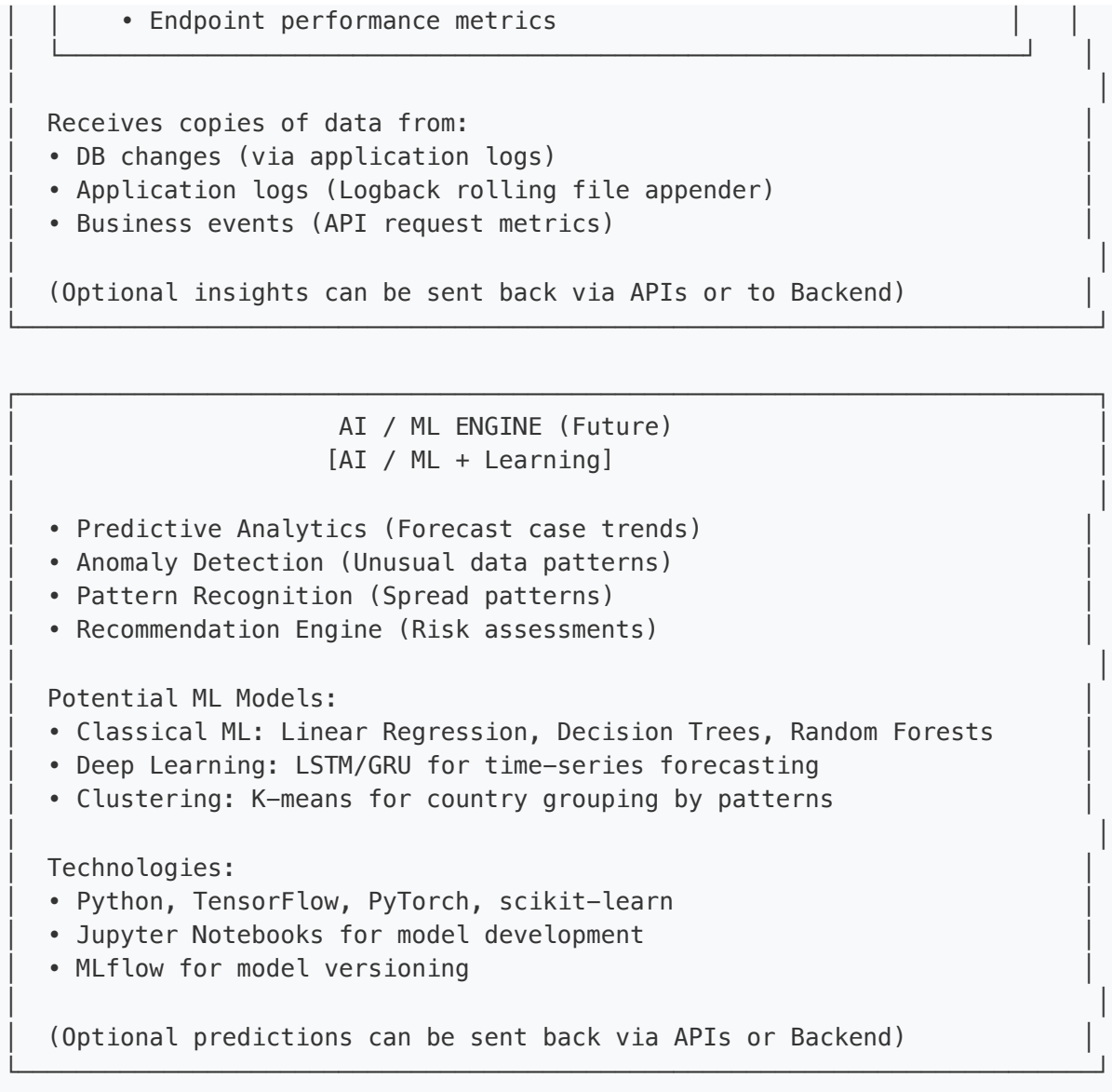
COVID-19 Tracker - System Architecture Diagram

High-Level System Design


UI CLIENT (Web / Mobile)
[HTML5, CSS3, JavaScript]
• index.html – Main Dashboard (Global Stats, Country Tables)
• analytics.html – API Analytics & Metrics Visualization
• HighCharts for data visualization
• Bootstrap for responsive UI
HTTP/REST + JSON
Load Balancer
[AWS ELB / Nginx]
BACKEND SERVICE
[Spring Boot Java 11 API]
REST CONTROLLERS
• /api/global – Global COVID-19 statistics
• /api/countries – All countries data
• /api/country/{name} – Specific country data
• /api/refresh – Manual data refresh
• /api/analytics/* – API usage metrics & analytics
MIDDLEWARE
• CORS Configuration (Cross-Origin Resource Sharing)
• MetricsInterceptor (Request/Response tracking)
• Global Exception Handler (Error handling)
• Spring Security (Authentication/Authorization – Optional)
SERVICE LAYER
• CovidDataService – Business logic, data aggregation
• AnalyticsService – Metrics calculation & tracking
• CsvParserUtil – Johns Hopkins CSV data parsing
• CovidDataInitializer – Data loading on startup
REPOSITORY LAYER
• CovidDataRepository (JDBC) – MySQL operations
• AnalyticsRepository (Spring Data) – MongoDB operations
• HikariCP Connection Pool (max: 20, min-idle: 5)

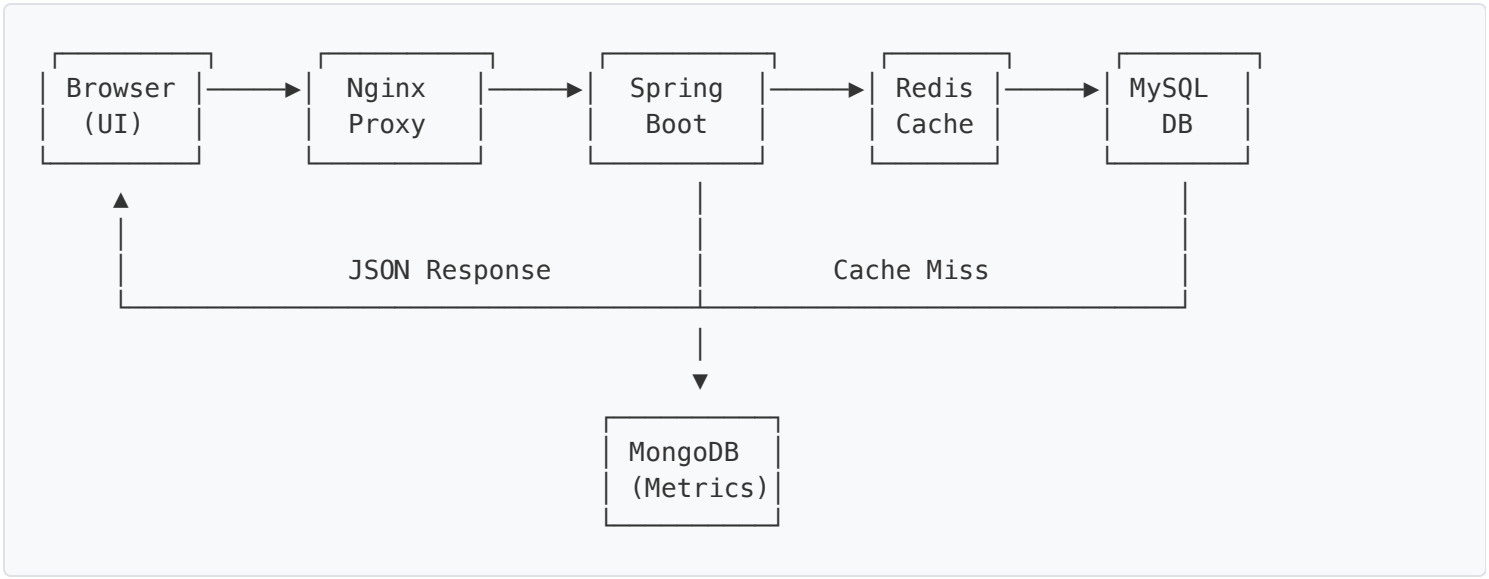

CACHE
[Redis - Optional]
• In-memory caching
• Configurable (on/off)
• TTL-based expiration
• localhost:6379
MONITORING & LOGGING
[Spring Boot Actuator]
• /actuator/health - Health check
• /actuator/metrics - Metrics
• /actuator/info - App info
• Logback file logging (10MB)
(Database)
(Isolated)
(Behind Defensive Perimeter
Private Network / Firewall
/ Security Groups)
DATABASE
[SQL + NoSQL - Polyglot Persistence]
MySQL 8.0
(Primary Transactional Database)
• covid_tracker.covid_data table
• Stores: country, province, cases, deaths, recoveries
• Aggregated time-series data
• ACID compliance for data integrity
• localhost:3306 (local) / RDS (production)
MongoDB Atlas
(Analytics & Metrics Database)
• api_request_metrics collection
• Stores: endpoint, method, status, response time, timestamp
• Flexible schema for evolving analytics
• Cloud-hosted NoSQL (MongoDB Atlas)
CORE SYSTEMS (Critical Request Path)
These systems respond to every request in milliseconds:
• Spring Boot Application (Application logic)
• HikariCP Connection Pool (Database connections)
• MySQL Database (Primary data storage)


• Redis Cache (Optional - Performance optimization)
BIG DATA / ANALYTICS PLATFORM
[ETL & Data Processing - Batch & Streaming]
DATA INGESTION PIPELINE
Johns Hopkins CSSE COVID-19 Dataset (GitHub)
download_covid_data.sh (Shell Script)
CSV Files (time_series_covid19_confirmed_global.csv)
CsvParserUtil (Apache Commons CSV)
ETL Processing:
• Parse CSV records (province, country, coordinates, dates)
• Aggregate province/state data to country level
• Calculate derived metrics (active cases, rates)
• Transform to domain objects (CovidData)
MySQL Database (Structured storage)
ANALYTICS PIPELINE
HTTP Requests (All API calls)
MetricsInterceptor (Request/Response capture)
Analytics Data:
• Endpoint URL, HTTP method, status code
• Response time, request timestamp
• Client info, query parameters
MongoDB Atlas (Real-time analytics storage)
Analytics Dashboard (analytics.html)
• Request summary statistics
• Timeline charts (HighCharts)

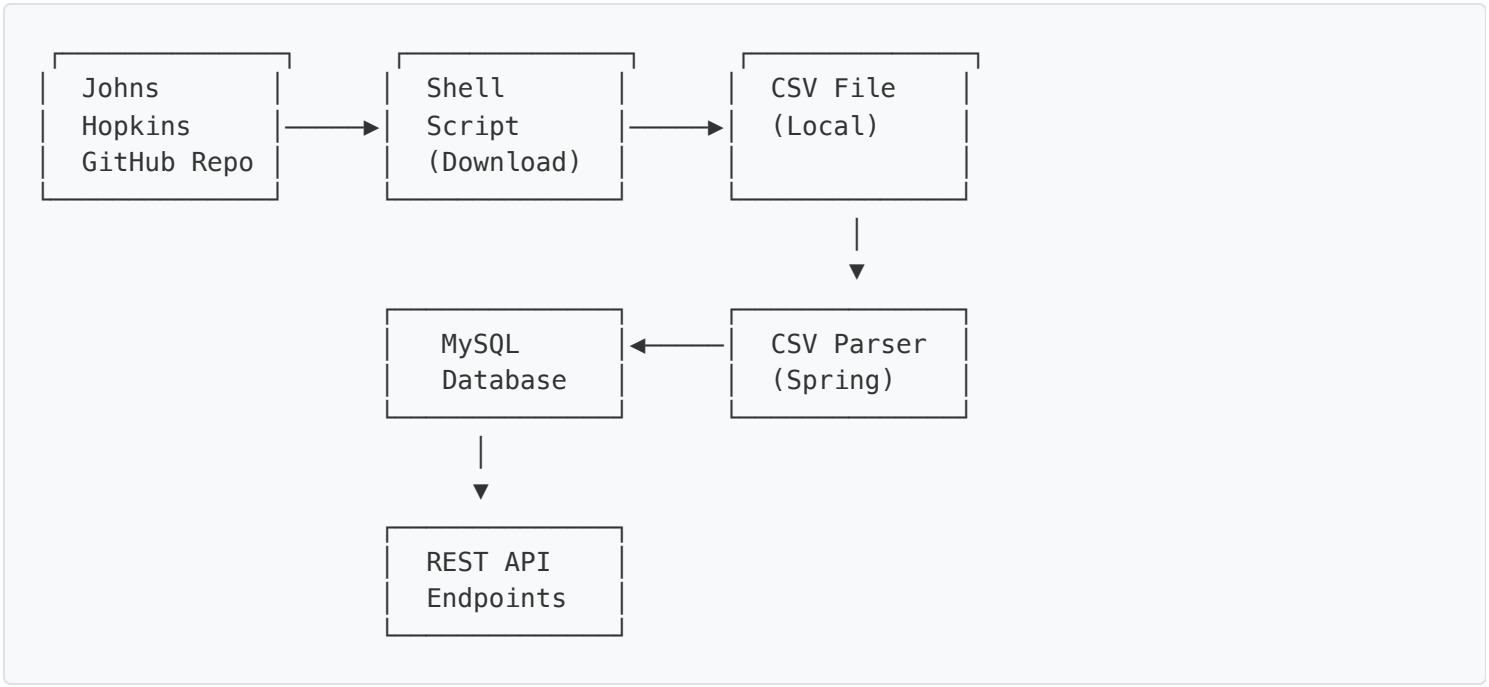


Data Flow Diagrams

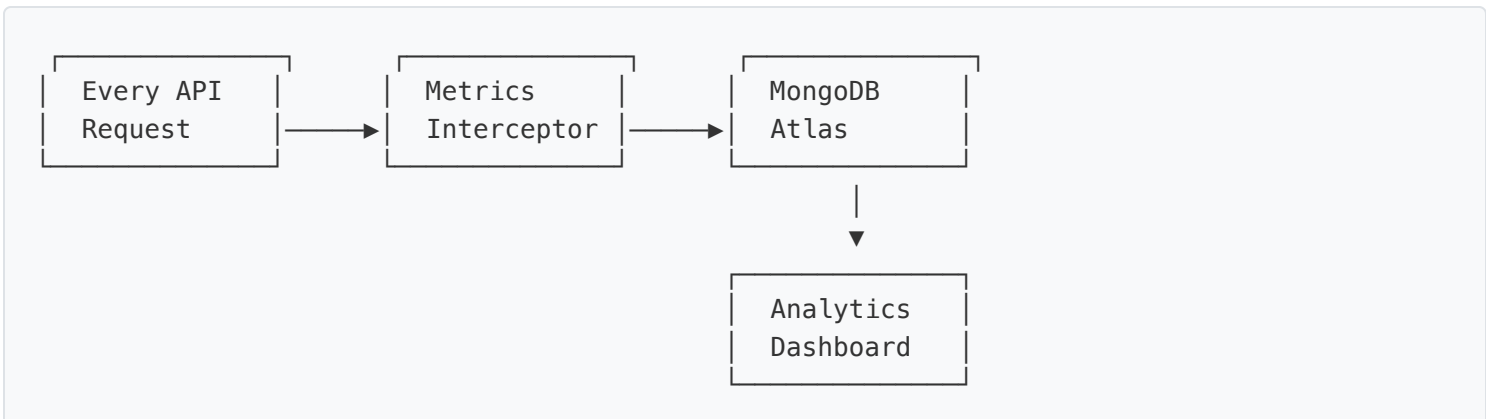
1. Primary Data Flow (User Request)



2. Data Ingestion Flow (ETL Pipeline)



3. Analytics Flow



Technology Stack by Layer

Frontend Layer

```
┌─────────────────────────────────────────────────────────┐
│  • HTML5 (Semantic markup)                              │
│  • CSS3 (Responsive design, Bootstrap 5)                │
│  • JavaScript (ES6+, Vanilla JS)                        │
│  • HighCharts.js (Data visualization)                   │
│  • Fetch API (HTTP requests)                            │
└─────────────────────────────────────────────────────────┘
```

Application Layer

```
┌─────────────────────────────────────────────────────────┐
│  • Java 11                                              │
│  • Spring Boot 2.7.14                                   │
│  • Spring Web MVC (REST Controllers)                    │
│  • Spring JDBC (Database access)                        │
│  • Spring Data MongoDB (Analytics)                      │
│  • Spring Data Redis (Caching)                          │
│  • Spring Boot Actuator (Monitoring)                    │
│  • SpringDoc OpenAPI (Swagger documentation)            │
│  • Apache Commons CSV (CSV parsing)                     │
│  • HikariCP (Connection pooling)                        │
│  • Logback (Logging)                                    │
│  • Maven (Build & dependency management)                │
└─────────────────────────────────────────────────────────┘
```

Data Layer

```
┌─────────────────────────────────────────────────────────┐
│  • MySQL 8.0 (Primary relational database)              │
│    - InnoDB storage engine                              │
│    - ACID transactions                                  │
│    - B-Tree indexes                                     │
│                                                         │
│  • MongoDB Atlas (Analytics NoSQL database)             │
│    - Document-based storage                             │
│    - Flexible schema                                    │
│    - Cloud-hosted                                       │
│                                                         │
│  • Redis (Optional in-memory cache)                     │
│    - Key-value store                                    │
│    - TTL-based expiration                               │
│    - localhost:6379                                     │
└─────────────────────────────────────────────────────────┘
```

Infrastructure Layer

```
┌─────────────────────────────────────────────────────────┐
│  • AWS EC2 (Compute instances)                          │
│  • Nginx (Reverse proxy, load balancing)                │
│  • Systemd (Service management)                         │
│  • Cron (Scheduled data refresh)                        │
│  • AWS Elastic IP (Static IP addressing)                │
│  • AWS Security Groups (Firewall rules)                 │
│  • SSH (Secure remote access)                           │
└─────────────────────────────────────────────────────────┘
```

Key Architecture Patterns

1. Layered Architecture (N-Tier)

```
Controller ──▶ Service ──▶ Repository ──▶ Database
  (REST)      (Business)   (Data Access)   (Storage)
```

2. Polyglot Persistence

```
MySQL (Transactional) + MongoDB (Analytics) + Redis (Cache)
```

3. Caching Strategy

```
Request ──► Check Cache ──► Cache Hit? ──► Return
                  │
                  ▼ (Cache Miss)
          Query Database ──► Store in Cache ──► Return
```

4. MVC Pattern (Frontend)

```
Model (Data) ──► View (HTML) ──► Controller (JavaScript)
```

5. Repository Pattern (Backend)

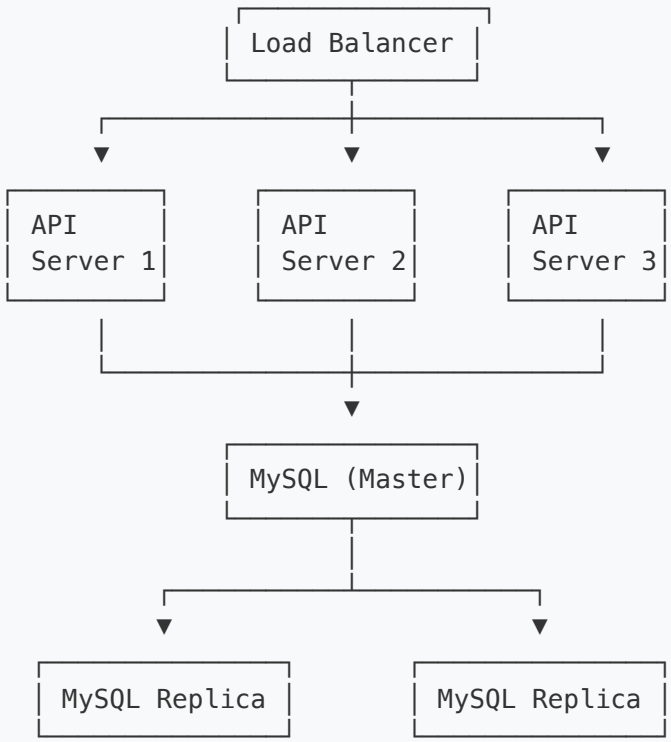
```
Service ──► Repository Interface ──► Repository Implementation ──► Database
```

Security Considerations

```
┌──────────────────────────────────────────────────────────┐
│  • CORS Configuration (Restricted origins)               │
│  • AWS Security Groups (Port restrictions)               │
│  • Environment-based configuration (secrets)             │
│  • Connection pooling limits (DoS prevention)            │
│  • Actuator endpoint security                            │
│  • Database credentials management                       │
│  • HTTPS/SSL (Production - via AWS ELB/Nginx)            │
└──────────────────────────────────────────────────────────┘
```

Scalability & Performance

Horizontal Scaling



Caching Strategy

- **Redis Cache**: Optional in-memory caching for frequently accessed data
- **HikariCP Pool**: Connection reuse (5-20 connections)
- **HTTP Caching**: Browser-level caching headers

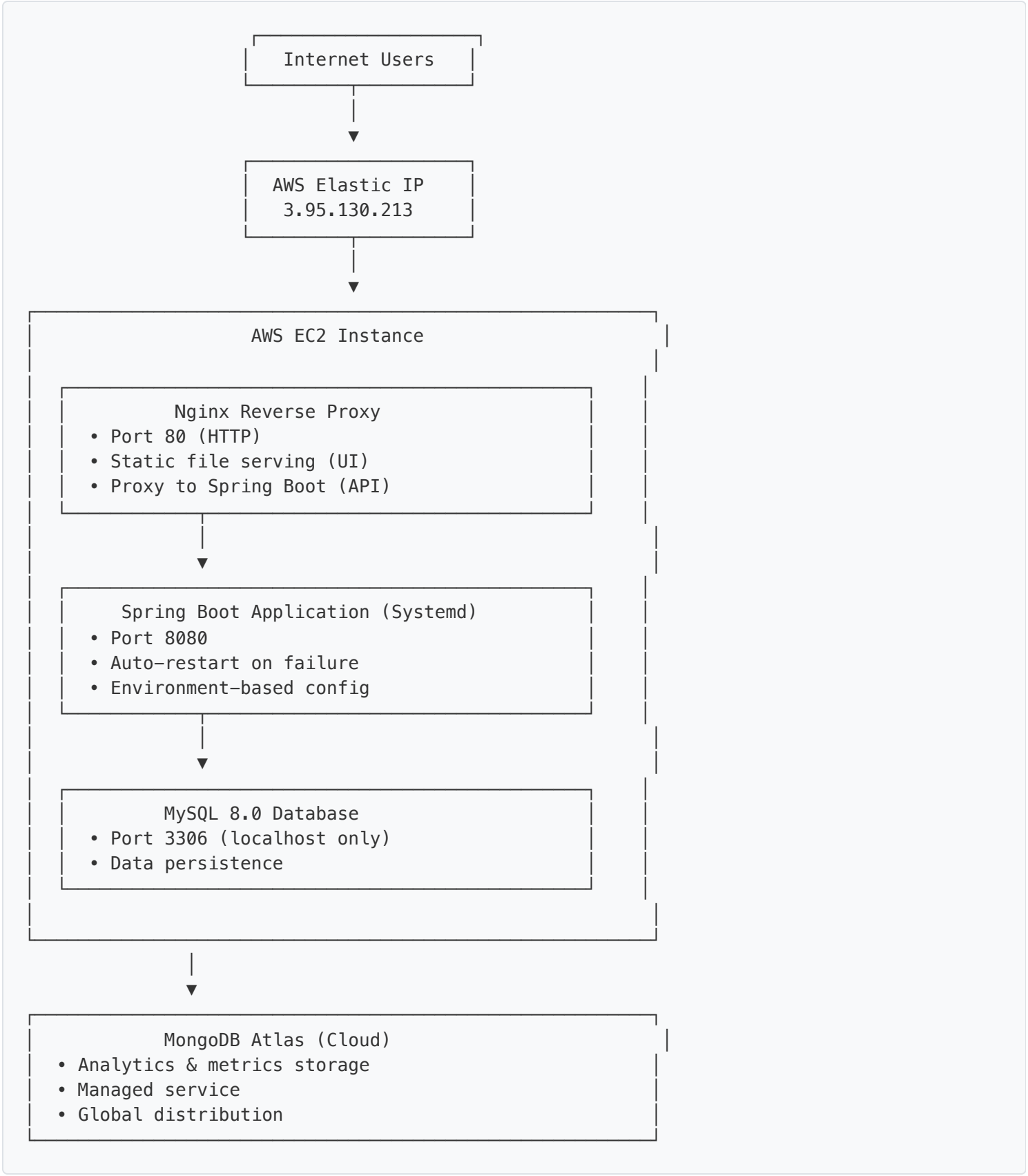
Database Optimization

- **Indexes**: On country, province, date columns
- **Connection Pooling**: HikariCP with configurable limits
- **Query Optimization**: Efficient aggregation queries

Monitoring & Observability

```
┌──────────────────────────────────────────────────────────┐
│  Spring Boot Actuator Endpoints:                         │
│  • /actuator/health – Application health status          │
│  • /actuator/metrics – JVM & application metrics         │
│  • /actuator/info – Application metadata                 │
│                                                          │
│  Logging:                                                │
│  • Logback with rolling file appender                    │
│  • 10MB file size limit                                  │
│  • 10 files retention                                    │
│  • Gzip compression for archived logs                    │
│                                                          │
│  Analytics Dashboard:                                    │
│  • Real-time API usage metrics                           │
│  • Request/response time tracking                        │
│  • Endpoint performance visualization                    │
└──────────────────────────────────────────────────────────┘
```

Deployment Architecture (AWS)



API Endpoints Summary

COVID Data Endpoints

- `GET /api/global` - Global statistics
- `GET /api/countries` - All countries data
- `GET /api/country/{name}` - Specific country data
- `POST /api/refresh` - Manual data refresh

Analytics Endpoints

- `GET /api/analytics/summary` - Request summary statistics
- `GET /api/analytics/timeline` - Timeline metrics

Monitoring Endpoints

- `GET /actuator/health` - Health check
- `GET /actuator/metrics` - Application metrics
- `GET /actuator/info` - Application info

Documentation

- `GET /swagger-ui/index.html` - Interactive API documentation

Created for: COVID-19 Tracker Full Stack Application
Version: 1.0.0
Last Updated: January 2026