Blood Test Comparison Application: Architecture, Cost, and Testing Strategy

# 1. Application Overview

Functionality: Upload, process, compare, summarize, email, multilingual support.

Target Users: Patients, potential healthcare professionals.

# 2. Architecture

## 2.1. Frontend (Client-Side)

Technology: React.js with Material-UI (MUI)

Key Features: Intuitive UI, responsive design, state management.

Deployment: CDN (Cloudflare, AWS CloudFront).

## 2.2. Backend (Server-Side)

Technology: Python (Flask or FastAPI).

Key Features: RESTful API, PDF processing, data extraction, Gemini API integration, email service.

Deployment: Serverless (AWS Lambda/GCP Functions) or Containerized (Docker, ECS/Cloud Run).

Database (Optional): PostgreSQL or MongoDB.

## 2.3. Cloud Hosting

Platform: AWS or Google Cloud Platform.

Key Considerations: Scalability, cost optimization, security.

## 2.4. Scalability and Performance

Load Balancing, Caching, CDN, Database Optimization.

## 2.5. Internationalization (i18n)

Frontend: React-i18next. Backend: Translations in database or JSON. Language Detection.

## 2.6. Cost Considerations

Compute, Storage, Data Transfer, AI Model Costs, Email Service, Database Costs, Monitoring.

## 2.7. Deployment

CI/CD pipelines, automated deployments.

## 2.8. Security Considerations

HTTPS, authentication/authorization, data privacy, input validation.

# 3. Testing Strategy

## 3.1. Functional Testing

Technologies: Jest, React Testing Library, Cypress (Frontend), Pytest, Unittest, Postman/Insomnia (Backend).

Test Cases: File upload, data extraction, comparison logic, Gemini API, email, multilingual, form validation, error handling.

## 3.2. Performance Testing

Technologies: JMeter, Locust, k6 (Load Testing), CloudWatch, Google Cloud Monitoring, Prometheus.

Test Cases: Response times, concurrent users, memory/CPU usage, scalability, PDF processing time, API rate limiting.

## 3.3. Security Testing

Technologies: SonarQube, Bandit, ESLint (SAST), OWASP ZAP, Burp Suite (DAST), Nessus Essentials, OpenVAS.

Test Cases: Input validation, authentication/authorization, data encryption, API security, file upload security, CORS, regular audits.

## 3.4. Vulnerability Testing

Technologies: OWASP Dependency-Check, Snyk.

Test Cases: Dependency scanning, regular updates, security hardening.

## 3.5. Usability Testing

Methods: User interviews, surveys, A/B testing, heuristic evaluation.

Test Cases: Ease of use, clarity of results, navigation, accessibility.

## 3.6. Integration Testing

Technologies: Mocking libraries, integration testing frameworks.

Test Cases: API integration, email service, database integration, frontend-backend communication.

## 3.7. Regression Testing

Automation: Automated test suites. Execution: Run tests after code changes.

# 4. Continuous Integration/Continuous Deployment (CI/CD)

Automation: Automated testing and deployment pipelines. Tools: Jenkins, GitLab CI, GitHub Actions.

# 5. Monitoring and Logging

Tools: CloudWatch, Google Cloud Monitoring, Prometheus, ELK stack. Alerts: Critical errors, performance degradation, security incidents.

# 6. Documentation

Types: API documentation, user manuals, deployment guides.

# 7. Regular Maintenance

Updates: Keep dependencies up to date. Security Patches: Apply patches promptly. Performance Tuning: Regularly optimize.

# 8. Cost Estimation (Initial Stage - Rough)

Serverless compute: $10-$50 per month. Object storage: $5-$20 per month. CDN: $5-$15 per month. Gemini API: Variable. Email Service: $10-$30 per month. Total: $30-$115+ per month (excluding database costs).