9. TESTING

9.1 Testing Strategy

The project implements comprehensive testing through automated CI/CD pipeline, with all tests executing on every pull request. The testing pyramid prioritizes unit tests (70%), integration tests (20%), and end-to-end tests (10%).

9.2 Types of Testing

- **Unit Testing:** Go (testify) - 156 tests covering services, handlers, utilities; React (Jest/RTL) - 89 component and hook tests
- **Integration Testing:** API tests with real PostgreSQL/Redis via Docker; database relationship and constraint verification
- **End-to-End Testing:** Playwright browser automation for critical user journeys
- **Security Testing:** Authentication/authorization verification, input validation (SQL injection, XSS, path traversal), dependency scanning
- **Performance Testing:** Load testing with k6 (100-1000 concurrent users), API response time benchmarks, database query optimization

9.3 Test Cases Summary

Test Suite	Cases	Coverage Area
Authentication	12	Register, Login, JWT, Password reset
Academic Hierarchy	23	Universities, Courses, Subjects, Relationships
Documents & Syllabus	28	Upload, Validation, Extraction, Storage
Chat & AI	14	Sessions, Messages, Streaming, Citations
Admin	9	User management, Settings, Audit logs

9.4 E2E User Journeys

User Journey	Steps	Duration	Pass Rate
User Registration & Login	7	15s	98%
Course Enrollment	10	22s	96%
Document Upload & Processing	12	45s	94%
AI Chat Interaction	8	18s	97%
Admin User Management	9	20s	99%

9.5 Performance Benchmarks

Test Type	Virtual Users	Duration	Result
Baseline Load	100	5 min	95% requests < 2s
Spike Test	50 -> 500	10 min	No crashes
Soak Test	200	30 min	Stable memory

9.6 Test Results Summary

Metric	Target	Current	Status
Unit Test Coverage	70%	76%	Pass
Integration Test Coverage	60%	68%	Pass
E2E Coverage (Critical Paths)	100%	100%	Pass
API Response Time (p95)	< 2s	1.2s	Pass
Security Vulnerabilities (High/Critical)	0	0	Pass
CI Pipeline Duration	< 15 min	10 min	Pass
Overall Test Pass Rate	> 95%	97.8%	Pass