System Design - End Term - Yerlan Dias

Github repo link

Task 3.

Functional requirements:

1. User management
    - Student Module
    - Employer module
    - Admin module
2. Internship matching engine: AI/ML-based recommendations
3. Application workflow: internship application platform
4. In-app messaging, video collaboration
5. Document verification engine: OCR & AI based document parsing
6. Analytics and reports: application performance for students, platform health for admins etc.

Non-functional requirements:

1. ≤ 500ms response for registration
2. Atomic operations across microservices
3. Eventual consistency is acceptable for non-critical operations (e.g., notifications, analytics sync)
4. 3 x 250 GB HDD
5. Backups daily, retention for 30 days.
6. 16 GB RAM minimum per server node
7. Support 2,000 concurrent connections
8. Performance <300ms response time for 95% of API requests
9. Security - OAuth2/ JWT based auth, Role-based access control
10. 99% uptime SLA

11. Multi-region deployment - CDN
12. CI/CD pipeline integration
13. Localization - multi-language support

In client layer there are 3 roles: User, Employer and Admin. Admin monitors activity, verify companies etc.

API Gateway acts as the single entry point for all user requests. Can also apply rate limiting, authentication, and request aggregation.

Auth service handles login/signup via OAuth2/ JWT tokens

Profile Management service stores and manages student profiles, resumes, skills, interests.Stores employer company profiles, logos, hiring preferences

Internship Management Service makes CRUD operations on internships and supports advanced filters: location, skills, duration, paid/unpaid. Employees post here and the data is stored in a searchable datastore like PostgreSQL

Matching engine - machine learning model that parses resumes, recommends internships to students, can use NLP + vector similarity for skill-based matching

Application Workflow Orchestrator manages full internship application lifecycle, users event-driven architecture for transitions.

Service Mesh (Istio / Linkerd) handles all inter-service communication