First: Final documentation (summary of all the previous work)

1. Project Overview

SkillSync is a career guidance platform designed to assist university students and fresh graduates in enhancing their job readiness. It combines resume analysis, skill evaluation, role-based mentoring, and administrative tools into a centralized backend system. The backend is built using Node.js, Express.js, and MongoDB with an MVC architecture and several software design patterns to ensure scalability and maintainability.

---

2. System Architecture

The project follows a structured **MVC (Model-View-Controller)** design and incorporates key software design patterns:

- **Singleton Pattern**: Applied in `AuthService` to ensure a single reusable instance for authentication logic.
- **Factory Pattern**: Dynamically selects between `KeywordResumeAnalyzer` and `AIResumeAnalyzer` based on the user's subscription type.
- **Observer Pattern**: Used for sending notifications after resume analysis (e.g., email notifications).
- **Role-Based Middleware**: Ensures only authorized roles (student, mentor, admin) can access protected features.

Folder structure includes:

- `controllers/`, `models/`, `routes/`, `services/`, `middleware/`, `observers/`, `factories/`, `config/`

---

3. Features & Functionalities

Core Functionalities:

- User Registration & Login (`JWT` authentication)

- Resume Upload & Analysis (`PDF/DOCX`)
- Resume Scoring & Skill Extraction
- Admin User Management (View/Delete Users)
- Notification System (Observer Pattern)
- Role-Based Access Control (Student, Mentor, Admin)

Future/Planned:

- Skill Gap Analysis
- Career Suggestions
- Progress Tracking
- Mentor–Student Communication

---

4. Database Design

The system uses MongoDB (with Mongoose ODM). Collections include:

- `users`: all system users with roles and subscription
- `resumes`: uploaded resume records linked to users
- `observers`: internal system notifications (pattern-based)

Schemas enforce field validation, referential integrity, and timestamp tracking.

---

5. Testing Summary

Manual tests were performed via Postman. Unit test scenarios included:

- TC01: Successful Login
- TC02: User Registration

- TC03: Resume Upload (Student)
- TC04: Resume Upload (Mentor – denied)
- TC07: Resume Upload with Missing Token – fails with 401

All core flows were tested and confirmed working. Role-based errors (403) and auth errors (401) were also handled and verified.

---

6. Security & Performance Highlights

Security:

- Passwords hashed with `bcrypt`
- JWT-based auth with expiration
- Route-level role restrictions
- Environment variables stored securely
- CORS and file upload validation

Performance:

- Lightweight Express middleware
- Async file parsing (non-blocking)
- Indexed MongoDB queries
- Fast startup (<300ms on Render)

---

7. Deployment Instructions

SkillSync backend can be deployed on Render, locally, or on cloud VMs. Setup includes:

- Node.js installation

- MongoDB local or cloud connection
- Environment file (`.env`) setup with `MONGO_URI`, `JWT_SECRET`, and `PORT`
- `npm run dev` to start server
- Verified connection via browser or Postman

See full installation instructions and configuration guide in the Deployment Plan.

---

8. Lessons Learned

- **Modularity pays off**: Breaking logic into services and patterns makes debugging and testing easier.
- **Real-world testing reveals real-world problems**: Errors such as missing imports, incorrect token usage, and broken middleware only appeared during Postman testing.
- **Security must be layered**: Role checks, token validation, and structured error handling were all essential.
- **Design patterns guide scalability**: The project can easily grow thanks to the Factory and Observer architecture.