now give me final report of the project with all the details

Here is the final project report for HireWrite, structured for a web-first build and then mobile, with simple, practical detail for features, technologies, and databases.

Executive summary

HireWrite is a free, multi-platform recruiting suite connecting candidates and companies with a shared backend and unified accounts. The web app launches first with Candidate, Company, and Admin areas; later, native Android (Kotlin) and iOS (SwiftUI) mirror the same features. Core capabilities include resume upload and AI-assisted resume generation, job posting, applications with fit scoring, pipelines, and admin approvals/moderation. All components target free, open-source technologies and self-hosting.

Scope and users

- Candidate: Create profile, upload/build resumes, search jobs, apply, get fit score and suggestions, track status, and use the browser extension later.
- Company: Create organization, add teammates, post and manage jobs, review applicants, move stages, add notes, and export data.
- Admin: Approve companies, monitor jobs and users, remove fake accounts, and view audit logs and simple metrics.

Platforms and rollout

- Phase 1: Web app for Candidate, Company, Admin.
- Phase 2: Android app (Kotlin) and iOS app (SwiftUI), reusing the web APIs.
- Extension: Integrated after the web MVP, tied to the same account.

Features by side

Candidate

- Accounts and profile
  - Sign up/login, role = Candidate.
  - Profile: name, contact, location, skills, links, education, experience.
- Resumes

    - Upload PDF/DOCX; auto-extract details; manual editing.
    - Multiple versions; set default; export to PDF/DOCX.
    - AI resume generation: generate from profile fields and optional job description; allows edits and saves as a new version.
- Jobs and applications
    - Jobs list with filters: title, skills, location type, experience, job type.
    - Job details page.
    - Apply with selected resume; optional cover letter note.
    - Fit score and short explanation (keyword-based initially).
    - Application tracker: Applied, Shortlisted, Interview, Offer, Hired, Rejected; notifications.

Company

- Organization and team
    - Create organization; profile (name, logo, website).
    - Domain/email verification status (Pending, Approved, Suspended).
    - Invite teammates; roles: Owner, Recruiter, Viewer.
- Jobs and applicants
    - Post job: title, experience range, location type, employment type, required and preferred skills, description.
    - Manage jobs: open/close/edit/duplicate; basic analytics (views, applicants).
    - Applicant pipeline: list candidates with resume preview, match score, notes, tags, stage changes, CSV export.
- AI helpers (optional, free)
    - Generate job description draft from structured inputs for easy edits.

Admin

- Approvals and moderation
    - Company approvals (review domain email or uploaded docs); approve/suspend.
    - Job moderation: flag suspicious jobs, take action.
- Users and audit
    - View users, disable fake/abusive accounts.
    - Audit log for all sensitive actions.
- Metrics (basic)
    - Counts of users, orgs, jobs, applications; simple trend lines.

Web-first development plan

Phase 1 (Foundation)

- Auth and roles: Candidate, CompanyUser, Admin.
- Database setup and migrations.
- File storage setup for resumes.
- Basic admin access gating.

Phase 2 (Candidate)

- Profile edit screens.
- Resume upload → parse → edit → versions.
- Jobs list + filters; job detail; apply flow.
- Applications list + status timeline.

Phase 3 (Company)

- Organization create/edit; verification status.
- Team invites and role management.
- Post/manage jobs; applicants per job; stage changes; notes.

Phase 4 (Matching and Admin)

- Keyword-based fit scoring (required vs preferred skills).
- Fit/Not Fit with short explanation stored on the application.
- Admin approvals/moderation, users list, job moderation, audit logs.

Phase 5 (AI Assist)

- AI resume generation endpoint: create resume drafts from profile + optional JD.
- AI JD generation endpoint: draft job descriptions from structured fields.

Technologies (free/open-source)

Frontend (Web)

- Next.js (React, TypeScript) for a single web codebase hosting Candidate, Company, and Admin areas with role-protected routes.
- Forms and validation: React Hook Form + Zod.
- UI: Tailwind CSS or a lightweight component library.
- State: Query-based (React Query) for caching and loading states.

Backend (choose one)

- Option A: FastAPI (Python), Pydantic models, JWT auth, SQLAlchemy async, OpenAPI docs.
- Option B: NestJS (Node.js), class-validator DTOs, JWT guards, Prisma or TypeORM.

Storage and infrastructure

- Database: PostgreSQL for all structured data.
- Object storage: MinIO (S3-compatible) for resume files and exports.
- Queue: Redis for background jobs (parsing, scoring, AI generation, emails).
- Reverse proxy: Caddy or Traefik with TLS; deploy via Docker Compose on a VPS.
- Optional search: Meilisearch for fast job/candidate filters (can be added later).

AI and parsing (local, free)

- LLM runtime: Local models via a self-hosted inference server (e.g., Llama/Mistral class) for AI resume and JD generation.
- Embeddings (later): Local sentence-transformers for semantic matching.
- Resume parsing: Open-source PDF/DOCX parsers to extract text and sections; manual correction UI for accuracy.
- Rendering: HTML templates → PDF via wkhtmltopdf/WeasyPrint; DOCX via docx libraries.

API design (shared for web and mobile)

- Auth: register, login, refresh, logout.
- Candidate:
  - GET/PUT profile.
  - POST resumes (upload), POST resumes/parse, GET resumes, GET resumes/:id.
  - POST resumes/generate (AI).
- Jobs:
  - GET jobs?filters, GET jobs/:id.
  - POST orgs/:orgId/jobs, PUT jobs/:id, PATCH jobs/:id/status.
- Applications:
  - POST jobs/:id/applications (candidate).
  - GET me/applications (candidate).
  - GET orgs/:orgId/jobs/:id/applications (company).
  - PATCH applications/:id/stage (company).
- Matching:
  - POST applications/:id/score → {keyword_score, fit_flag, explanation}.

- Admin:
  - GET admin/companies?status=pending; PATCH admin/companies/:id/approve|suspend.
  - GET admin/jobs?flagged=true; PATCH admin/jobs/:id/moderate.
  - GET admin/users; PATCH admin/users/:id/status.
  - GET admin/audit.

Database schema (plain-language tables)

- users
  - id, email, hashed_password, role [candidate|company|admin], status, created_at.
- organizations
  - id, name, website, verification_status [pending|approved|suspended], created_at.
- org_users
  - id, org_id, user_id, org_role [owner|recruiter|viewer], created_at.
- candidate_profiles
  - user_id (PK/FK), name, city, summary, skills_json, education_json, experience_json, links_json, updated_at.
- resumes
  - id, user_id, storage_path, parsed_json, template_id, source [uploaded|generated], version_label, active (bool), created_at.
- jobs
  - id, org_id, title, location_type [remote|hybrid|onsite], employment_type [full-time|part-time|intern|contract], experience_min, experience_max, skills_required_json, skills_preferred_json, jd_text, status [open|closed|draft], created_at.
- applications
  - id, job_id, user_id, resume_id, stage [applied|shortlisted|interview|offer|hired|rejected], decision [null|offer|reject], created_at, updated_at.
- scores
  - application_id (PK/FK), keyword_score (0–100), semantic_score (optional), fit_flag (bool), explanation (short text).
- notes
  - id, application_id, author_user_id, text, created_at.
- admin_audit
  - id, actor_user_id, action, target_type [user|org|job|application], target_id, payload_json, created_at.
- templates (optional, for resume rendering)
  - id, name, html_template_path, css_path, status [active|inactive].

Security and privacy

- JWT with short-lived access tokens and refresh tokens; roles in claims.
- Signed URLs for file download; per-object access control.
- Validation and rate limiting on sensitive endpoints (apply, parse, generate).
- PII handling: no sensitive data in logs; audit logs for admin/company actions.
- User rights: data export and delete endpoints.

Quality and testing

- E2E tests on web flows (auth, upload/parse, apply, post job, stage changes).
- Backend unit and integration tests for models and endpoints.
- Seed data for demo orgs, jobs, and candidate profiles.
- Accessibility checks on web UI; responsive design.

Delivery plan and timeline

- Weeks 1–2: Auth + roles, DB, file storage, base layouts; Candidate profile + resume upload/parse.
- Weeks 3–4: Jobs list/detail, Apply flow, Applications tracker.
- Weeks 5–6: Company profile, team invites, post/manage jobs, applicants list and stage changes.
- Weeks 7–8: Matching v1 (keyword fit), Admin approvals/moderation, audit log, basic metrics.
- Weeks 9–10: AI resume generation and optional AI JD generation; export to PDF/DOCX.
- Post-MVP: Browser extension integration; semantic matching; mobile apps (Kotlin/SwiftUI) using the same APIs.

Mobile follow-on (after web)

- Android (Kotlin): Native app with the same flows (Profile, Resumes, Jobs, Apply, Tracker; Company dashboard); Room for offline caches; secure token storage.
- iOS (SwiftUI): Same screens and endpoints; Keychain for tokens; PDF preview for resumes; share sheet for exports.

Risks and mitigations

- Resume parsing accuracy: Provide manual correction UI; keep parsers modular; allow multiple resume versions.
- Site scraping changes (extension): Maintain selectors and fallbacks; provide a manual paste JD option.
- Performance: Offload parsing/scoring/generation to background jobs; cache results; paginate lists.

- Abuse/fraud: Company approvals, job moderation, rate limiting, user verification, audit logs.

Success metrics

- Candidate: profile completion rate, resumes generated, applications per week, interviews scheduled.
- Company: time to post job, qualified applicants per job, pipeline throughput.
- Platform: approval turnaround time, moderation actions, uptime, average API response time.

This report captures the final scope, plan, technologies, and database for launching HireWrite on the web first, then extending to Android (Kotlin) and iOS (SwiftUI), while keeping the entire system free and open-source friendly.

⁂

1. https://www.rnbglobal.edu.in/assets/pdfs/1.3.4/1.3.4 Sample Project Report.pdf
2. https://www.projectmanager.com/blog/5-free-project-report-templates
3. https://www.scribd.com/document/213508424/project-report-on-recruitment-and-selection
4. https://www.indeed.com/career-advice/career-development/recruitment-report
5. https://www.smartsheet.com/content/project-report-templates
6. https://www.orangescrum.com/project-template/recruitment
7. https://www.template.net/edit-online/464746/recruitment-report-format-for-workers
8. https://www.slideshare.net/slideshow/a-project-report-on-recruitment-and-selection/259667851