**📋 Updated To-Do List for Student Project**

**Phase 1 – Kickoff & Repo Setup**

1. **Create GitHub repository**
   * Structure:

/frontend → Flutter web app

/backend → API (FastAPI/Node)

/infra → K8s manifests, Helm charts, Terraform (later)

/docs → ADRs, NFRs, diagrams

* + Add README with: project description, functional + non-functional requirements.

✅ LO1 (Professional standard).

1. **Set up Git branching & CI/CD skeleton**
   * GitHub Actions workflow:
     + Run tests (backend & Flutter).
     + Lint/format checks.
     + Build Docker images.
     + Push to GitHub Container Registry (GHCR).

✅ LO4 (DevOps).

**Phase 2 – Core Infrastructure (Local Kubernetes)**

1. **Install local Kubernetes environment**
   * Use **Minikube** or **k3d** (runs inside Docker).
   * Install kubectl + helm.

✅ LO5 (Cloud Native).

1. **Deploy base infrastructure with Helm**
   * PostgreSQL (Helm chart).
   * MinIO (Helm chart for CV uploads & employer logos).
   * Keycloak (Helm chart for auth).
   * Ingress controller (NGINX).

✅ LO3 (Scalable Arch), LO6 (Security), LO7 (Distributed Data).

**Phase 3 – Authentication with Keycloak**

1. **Configure Keycloak**
   * Create realm + client for your app.
   * Create roles: Employee, Employer.
   * Enable **MFA (Google Authenticator)**.
   * Configure email verification (SMTP, Mailhog for local).

✅ LO6 (Security by Design).

1. **Integrate Flutter with Keycloak (OIDC)**
   * Use PKCE login flow.
   * Save JWT tokens.
   * Protect backend APIs by verifying JWT.

✅ LO6 (AuthN/Z).

**Phase 4 – Database Design**

1. **Design schema in Postgres**
   * Tables:
     + users (Employee/Employer).
     + companies.
     + profiles (availability, hobbies, skills, CV link).
     + jobs (type, description, image, employer\_id).
     + applications (employee\_id → job\_id).
   * Generate ERD (dbdiagram.io).

✅ LO3 (Scalable Arch), LO7 (Data).

1. **Add migrations**
   * Use Alembic (Python) or Prisma (Node).
   * Store under /backend/migrations.

**Phase 5 – Employee Features**

1. **Employee dashboard (Flutter)**
   * Profile form (name, hobbies, skills).
   * Availability toggle (Open/Closed).
   * Upload CV (PDF/DOCX → stored in MinIO).
2. **Backend CV processing**

* Store file in MinIO.
* Parse text (Apache Tika/textract).
* Save structured data in Postgres.

✅ LO1, LO7.

**Phase 6 – Employer Features**

1. **Employer registration**

* Must use company email (checked in Keycloak).
* Email verification code before activation.

1. **Job advertisements**

* Post job (Internship, Graduation Internship, Job).
* Upload image/logo (stored in MinIO).
* Store job info in Postgres.

✅ LO1, LO6.

**Phase 7 – Matching Engine**

1. **MVP Matching**

* Use Postgres full-text search (tsvector).
* Filter by job type + employee availability.
* Return ranked list of employees.

1. **Optional Upgrade**

* Deploy OpenSearch (Helm).
* Store CVs & JDs → BM25 ranking.
* Later: add embeddings with pgvector.

✅ LO3 (Scalable Arch), LO7 (Distributed Data).

**Phase 8 – Notifications**

1. **Employee reactions**

* Employee clicks "Apply" → Employer gets email (profile + CV).

1. **Employer offers**

* Employer selects employee → Employee gets email with job offer.

**How:** Mailhog for local SMTP → easy to demo.

✅ LO1 (Professional), LO6 (Security).

**Phase 9 – DevOps & Monitoring**

1. **CI/CD full pipeline**

* GitHub Actions:
  + Build & push Docker images.
  + Run tests.
  + Deploy to local K8s with kubectl.

1. **Observability stack**

* Prometheus (metrics).
* Grafana (dashboards).
* Loki (logs).

✅ LO4 (DevOps), LO5 (Cloud Native).

**Phase 10 – Security & GDPR Compliance**

1. **Secure everything**

* TLS on Ingress (self-signed cert).
* Enforce Keycloak JWT validation.
* Limit CV size (≤ 10 MB) + virus scan.

1. **GDPR features**

* "Delete my account" → delete profile + CV + jobs.
* "Export my data" → JSON + CV file.

✅ LO6 (Security), LO7 (Data).

**Phase 11 – Final Demo & Documentation**

1. **Demo flow**

* Employee registers, uploads CV, sets availability.
* Employer registers (verified email), posts a job.
* Employer pastes JD → system finds matching employees.
* Both sides exchange via email.

1. **Evidence pack**

* Docs: ADRs, ERD, NFRs, CI/CD pipeline screenshots.
* Security report (Keycloak, MFA, GDPR flows).
* Monitoring screenshots (Grafana).
* Demo video.

✅ Shows **all LOs (1–7)**.

**🟢 Priority Order (Do First)**

1. GitHub repo + CI/CD skeleton.
2. Kubernetes base infra (Postgres, MinIO, Keycloak, Ingress).
3. Authentication with Keycloak + MFA.
4. Database schema.
5. Employee profile + CV upload.
6. Employer job ads.
7. Matching engine MVP.
8. Notifications (emails).
9. CI/CD + monitoring polish.
10. Security & GDPR.
11. Final demo & documentation.

👉 This is the **doable roadmap for a student with no budget**:

* **Local Kubernetes (Minikube/k3d)** → free.
* **Keycloak + MFA** → free.
* **Postgres + MinIO + Mailhog** → free.
* **GitHub Actions** → free.
* **Prometheus + Grafana** → free.