**Phase A → tiny chunks**

**Chunk 0 — Project bootstrap (30–60 min)**

**Goal:** repo + env that runs “Hello FastAPI”.

* **Deliverables:** pyproject.toml, app/main.py, app/config.py, app/db.py, alembic init, .env.example, docker-compose.yml (Postgres).
* **DoD:** GET /health returns {status:"ok"} and DB ping passes.
* **Quick test:** run uvicorn app.main:app --reload and hit /docs.

**Chunk 1 — Auth + RBAC core (2–4 hrs)**

**Goal:** login, JWT, current user, role checks.

* **Tables:** users(id, email, hashed\_password, first\_name, last\_name, role, school\_id, is\_active, last\_login\_at).
* **Endpoints:** POST /auth/login, GET /auth/me.
* **Guards:** require\_role("admin"|"teacher"|…).
* **DoD:** 401 for bad token, 403 for wrong role, success for correct.
* **Quick test:** hit /auth/me with/without token.

**Chunk 2 — Schools (tz rules) (1–2 hrs)**

**Goal:** manage schools with **US-only** time zones.

* **Tables:** schools(id, name, address, tz).
* **Endpoints:** POST /schools, GET /schools/{id}, PATCH /schools/{id} (admin only).
* **Rules:** validate tz ∈ {America/New\_York,…,Pacific/Honolulu}. Store datetimes UTC; school tz used only for display.
* **DoD:** invalid tz rejected; create/read/update works; RBAC enforced.

**Chunk 3 — Students (2–3 hrs)**

**Goal:** CRUD students (minimal PII), search.

* **Tables:** students(id, school\_id, first\_name, last\_name, student\_number UNIQUE, grade\_level, enrollment\_status, guardians\_summary jsonb).
* **Endpoints:** POST /students, GET /students?q=&grade\_level=, GET /students/{id}, PATCH /students/{id}.
* **DoD:** unique student\_number enforced; pagination; teachers can read only students they teach (temp allow all until classes chunk lands).

**Chunk 4 — Classes & Enrollments (2–4 hrs)**

**Goal:** sections + roster management.

* **Tables:** class\_sections(id, school\_id, name, grade\_level, subject, teacher\_id, term\_label), enrollments(student\_id, class\_id, enrolled\_date, status).
* **Endpoints:**
  + Classes: POST /classes, GET /classes?teacher\_id=, GET /classes/{id}, PATCH /classes/{id}
  + Enrollments: POST /classes/{class\_id}/enrollments (bulk add), DELETE /classes/{class\_id}/enrollments/{student\_id}, GET /classes/{class\_id}/students
* **RBAC:** teacher can CRUD their own classes; admin full.
* **DoD:** duplicate enrollment blocked; teacher can list roster of their class.

**Chunk 5 — Calendar Blocks (1–2 hrs)**

**Goal:** schedule meeting times per class.

* **Tables:** calendar\_blocks(id, class\_id, start\_at UTC, end\_at UTC, label, room).
* **Endpoints:** POST /classes/{class\_id}/calendar-blocks, GET /classes/{class\_id}/calendar-blocks?from=&to=, PATCH /calendar-blocks/{id}, DELETE /calendar-blocks/{id}.
* **Rules:** end\_at > start\_at; all UTC; no hard conflict logic yet.
* **DoD:** teacher can CRUD blocks for their class; date filters work.

**Chunk 6 — Attendance (2–3 hrs)**

**Goal:** fast daily attendance with uniqueness.

* **Tables:** attendance\_records(id, student\_id, class\_id, date, status, notes, recorded\_by, recorded\_at); UNIQUE(student\_id, class\_id, date).
* **Endpoints:** GET /classes/{class\_id}/attendance?date=YYYY-MM-DD, POST /classes/{class\_id}/attendance (bulk), PUT /attendance/{id}.
* **Rules:** default Present implied in UI (API: only writes exceptions); statuses: present/absent\_excused/absent\_unexcused/tardy\_excused/tardy\_unexcused.
* **DoD:** bulk upsert works; uniqueness protected; teacher can’t mark other teacher’s class.

**Chunk 7 — Gradebook baseline (3–5 hrs)**

**Goal:** basic categories, assignments, and grades.

* **Tables:**
  + grade\_categories(id, class\_id, name, weight>=0, drop\_lowest int default 0, is\_active)
  + assignments(id, class\_id, title, max\_points>0, due\_at UTC, category\_id?, is\_published)
  + grades(id, assignment\_id, student\_id, points\_earned>=0, status, comment, graded\_by, graded\_at); UNIQUE(assignment\_id, student\_id)
* **Endpoints:**
  + Categories: GET/POST/PATCH /classes/{class\_id}/grade-categories
  + Assignments: GET/POST/PATCH /classes/{class\_id}/assignments, /assignments/{id}
  + Grades: GET /assignments/{id}/grades, POST /assignments/{id}/grades/bulk, PATCH /grades/{id}
* **DoD:** bulk grade entry; simple percent calc service for sanity; RBAC scoped to teacher’s classes.

**Chunk 8 — Audit logs + seeds + smoke tests (2–4 hrs)**

**Goal:** compliance-ready logging & demo data.

* **Tables:** audit\_logs(id, user\_id, entity\_type, entity\_id, action, payload jsonb, ip, ua, at).
* **Middleware:** record write ops.
* **Seeds:** 1 school (America/Chicago), 1 admin, 2 teachers, 50 students, 2 classes, rosters, a week of blocks, sample categories/assignments/grades.
* **Tests:** pytest for auth, RBAC on classes, enrollments duplicate guard, attendance uniqueness, grade bulk API happy path.
* **DoD:** pytest green; POST /auth/login works with seed users; demo tour runs end-to-end.

**How we’ll work each chunk**

* **Artifacts:** models → schemas → router → service (if needed) → Alembic migration → minimal tests → “how to run”.
* **Branching:** feature/chunk-##-name → PR → tag phase-a-ck##.
* **Runtime loop:** run DB via docker, app via uvicorn; use httpx tests.

**Suggested order to start this week**

1. Chunk 0 → 1 → 2 (you’ll have auth + schools in ~half a day)
2. Chunk 3 (students)
3. Chunk 4 (classes/enrollments)
4. Chunk 6 (attendance) **or** 7 (gradebook), depending on priority
5. Chunk 8 (audit + seeds + tests)

If you want, I can start with **Chunk 0** right now and hand you the exact files to paste.