**گزارش پیشرفت تا این لحظه**

* **وضعیت کلی**
* مخزن آماده توسعه با Backend پایه، پایگاه داده، تست‌ها، و CI.
* توسعه به‌صورت قدم‌به‌قدم با پیام‌های کامیت کنوانشنال (feat, chore, ci, test).
* **Backend (FastAPI)**
* اپلیکیشن: backend/app/main.py
* Endpoint سلامت: GET /health
* ثبت روت‌ها: auth, users, students, courses, enrollments, grades
* ساخت جداول در استارتاپ (برای SQLite dev)
* **مدل‌های داده (SQLAlchemy) + مهاجرت‌ها (Alembic)**
* User: email(unique), username, role, password\_hash
* مهاجرت: 0002\_create\_users
* Student: student\_no(unique), entry\_year, full\_name
* مهاجرت: 0001\_create\_students
* Course: code(unique), title, credits, department
* مهاجرت: 0003\_create\_courses
* Enrollment: student\_id, course\_id, term, grade
* مهاجرت: 0004\_create\_enrollments
* Grade: enrollment\_id, value
* مهاجرت: 0005\_create\_grades
* تنظیمات Alembic: backend/alembic.ini, env.py, اسکریپت قالب، نسخه‌ها
* **APIها (CRUD)**
* Auth: /api/v1/auth/register (JSON), /api/v1/auth/login (OAuth2 Password)
* توکن JWT با HS256 و انقضا قابل‌پیکربندی
* Users: /api/v1/users (List/Create/Get/Patch/Delete)
* Students: /api/v1/students (List/Create/Get/Patch/Delete)
* Courses: /api/v1/courses (List/Create/Get/Patch/Delete)
* Enrollments: /api/v1/enrollments (List/Create/Get/Patch/Delete)
* اعتبارسنجی وجود Student/Course و جلوگیری از تکرار
* Grades: /api/v1/grades (List/Create/Get/Patch/Delete)
* اعتبارسنجی وجود Enrollment
* اسکیماها: Pydantic v2 (from\_attributes=True)
* **احراز هویت و امنیت**
* ثبت‌نام و ورود (JWT)، هش گذرواژه با passlib[bcrypt]
* تنظیمات JWT: Secret/Algorithm/Expiry در core/config.py
* **تست‌ها (pytest + httpx)**
* test\_health.py: سلامت سرویس
* test\_auth.py: ثبت‌نام و ورود
* test\_students.py: CRUD دانشجو
* test\_courses.py: CRUD درس
* test\_enrollments.py: CRUD ثبت‌نام
* test\_grades.py: CRUD نمره
* هر تست با DB موقت SQLite و ایزوله اجرا می‌شود
* **CI (GitHub Actions)**
* Workflow: .github/workflows/ci.yml
* Python 3.13، کش pip
* نصب requirements.txt و requirements-dev.txt
* اجرای pytest -q backend/tests
* چک import (قبلاً) → فعلاً تمرکز روی تست‌ها
* **Dev/Run**
* اجرای محلی:
* python -m venv .venv && .venv\Scripts\pip install -r backend/requirements.txt
* cd backend && python -m app یا uvicorn app.main:app --reload
* API: http://127.0.0.1:8000
* Docker Compose موجود است؛ (روی ویندوز نیاز به Docker Desktop فعال دارد)
* **مستندات**
* شاخص: docs/README.md
* RFP, SRS, HLA, Architecture, ERD, OpenAPI (اسکلت), TestPlan, SecurityPrivacy
* README ریشه: مقدمه و Quickstart
* **کامیت‌های کلیدی اخیر (نمونه)**
* feat(db): Alembic setup + migrations (students/users/courses/enrollments/grades)
* feat(auth): JWT settings, token utils, register/login, tests
* feat(student/course/enrollment/grade): مدل‌ها، API، تست‌ها
* ci: run pytest

**برنامه ادامه (گام‌های آتی پیشنهادی)**

* کوتاه‌مدت (Sprint بعدی)
* RBAC ساده روی Endpointهای حساس (ثبت نمره/مدیریت) بر اساس role (student/faculty/admin)
* Pagination/Filtering/Ordering برای لیست‌ها
* Validation تکمیلی و پاسخ‌های خطای یک‌دست
* Seed Data (نمونه دانشجو، درس، ثبت‌نام، نمره)
* Alembic migrations در CI (چک dry-run)
* Lint/Type-check در CI: اضافه‌کردن ruff و mypy
* پوشش تست افزایشی (negative cases)
* میان‌مدت (بر اساس RFP/SRS)
* Scheduling v0: پیاده‌سازی حل‌گر Greedy/Heuristic، ورودی constraints JSON، خروجی برنامه ترم
* Analytics v1 (At-Risk): Pipeline ساده با scikit-learn، endpoint POST /analytics/at\_risk/run، ذخیره risk\_scores
* Recommender v1: توصیه درس (Popularity/Item-based ساده)، endpoint و تست‌ها
* Reporting: خروجی CSV/PDF برای گزارش‌های پایه؛ Endpoint دانلود
* Observability اولیه: لاگ ساختاریافته + health/metrics endpoints؛ در فاز بعد ELK/Grafana
* Integration آماده‌سازی: اسکلت Connectorهای LMS/SIS (interface و mock)
* Frontend
* Scaffold React + Vite + Tailwind + React Query + i18n (FA/EN)
* صفحات: Login/Register، Dashboard، لیست‌ها (Students/Courses/Enrollments/Grades)
* چارت‌های اولیه (KPIهای ساده)
* DevOps
* Docker images استاندارد؛ Registry
* مسیر استقرار: VM (مرحله اول) → K8s (مرحله بعد)
* GitHub Actions: ماتریس نسخه پایتون، build تصاویر، امنیت (Trivy/Snyk)
* Secrets/Config: جداسازی envها، مدیریت امن secretها
* امنیت و حریم خصوصی
* Rate limiting، CORS، headers امنیتی
* Audit logging برای رویدادهای حساس (به‌ویژه Grade)
* سیاست‌های PII (masking/redaction در لاگ‌ها)
* آماده‌سازی برای SSO/Keycloak (فاز دوم)
* داده و پژوهش
* Data dictionary کامل‌تر و schema.sql نهایی
* آماده‌سازی داده‌های نمونه (anonymized)
* مستندسازی baseline‌های ML و قابل‌تکرار بودن نتایج
* Roadmap پیشنهادی (High-level)
* Sprint 1-2: RBAC، کیفیت API، تست‌ها، Frontend scaffold
* Sprint 3: Scheduling v0، Reporting، بهبود CI
* Sprint 4: Analytics v1، Recommender v1، Observability
* Sprint 5: Integrations (LMS/SIS)، استقرار پایدار
* Sprint 6: polish, docs, demo

اگر تأیید می‌کنی، با افزودن RBAC ساده روی Endpointهای نمره و ثبت‌نام (role=faculty/admin) شروع کنم، سپس Pagination/Filtering را اضافه کنیم.