گزارشکار اول پیشرفت پروژه

سیستم هوشمند مدیریت و تحلیل داده های دانشگاهی

ارائه دهنده : نیما مخملی

تاریخ : 02/07/1404

چکیده اجرایی:

این پروژه با هدف یکپارچه سازی داده های آموزشی و پژوهشی دانشگاه و ارائه ی تحلیل های مبتنی بر هوش مصنوعی آغاز شد.

تا این مرحله هسته ی Backend با فریم ورک FastApi پیاده سازی و ماژول های کلیدی دامنه دانشجو – استاد – درس – ثبت نام – نمره به همراه احراز هویت مبتنی بر jwt - کنترل دسترسی نقش محور RBAC – صفحه بندی API - زمانبندی اولیه (Scheduling v0 ) و ماژول تحلیل خطر افت تحصیلی (Analytics v1) تکمیل شده است.

کلیه ی عملکردها با تست های خودکار پوشش داده شده و CI روی Github Actions فعال است.

ادامه ی مسیر شامل توسعه ی داشبورد فرانت اند – گزارشگیری و پیشبرد ماژول های مبتنی بر یادگیری ماشین و بهینه سازی است.

اهداف :

یکپارچه سازی داده های دانشجویان – اساتید – دروس – پژوهش ها

تحلیل آموزشی (Educational Data Mining) و پیش بینی دانشجویان در معرض خطر

سیستم توصیه گر – زمانبندی کلاس ها و داشبورد مدیریتی

ارائه Api امن و قابل اتصال به LMS/SIS , CI/CD و مستندسازی پژوهشی

دامنه ی فعلی در نسخه ی MVP :

مدیریت کاربران و نقش ها – دانشجو – درس – ثبت نام – نمره

احراز هویت و RBAC – صفحه بندی – زمانبندی v0 – تحلیل At-Risk v1

تست خودکار و CI - اسکلت مستندات (RFP/SRS/Architecture/ERD/OpenAPI)

معماری :

لایه ی فرانت اند (فاز بعد) : React + vite + Tailwind + React Query + i18n

لایه ی Api و سرویس ها : FastApi (Python) + WebSocket

لایه ی داده ها (پایگاه داده) : پایگاه داده اصلی PostgreSQL - redis برای کش – SQLite برای تست محلی

مهاجرت ها : Alembic نسخه بندی Schema

هوش مصنوعی : Scikit Learn

Devops : Docker – Github Actions (CI) –

امنیت : JWT (sha256) – نقش ها (student – faculty - Admin)

### وضعیت پیاده سازی :

### اهراز هویت :

### ثبت نام و ورود JWT - هش گذرواژه bcrypt - RBAC با نقش های Student / Faculty / Admin

### دامنه ی اصلی (CRUD) :

### Users : ثبت / ویرایش / حذف کاربر / یکتایی ایمیل

### Students : شماره دانشجویی یکتا / سال ورود / نام کامل

### Courses : کد درس یکتا / عنوان / واحد / دانشکده

### Enrollments : لینک دانشجو / درس / ترم / جلوگیری از تکرارRBAC (ایجاد / ویرایش / حذف / faculty / admin)

### API :

### Pagination : Limit / offset برای لیست ها

### خطاهای معنادار : 404 – 401 – 403 – 409

### Scheduling v0 (Greedy) :

### ورودی : بازه زمانی / ظرفیت کلاس ها / دسترس پذیری مدرس / فهرست سکشن ها

### خروجی : تخصیص (کلاس و بازه) بدون تداخل تا حد امکان + لیست

### Analytics v1 (At-Risk) :

### مدل RandomForest با داده ی هوش مصنوعی آموزشی synthetic برای دمو

### ورودی : وِیژگی های پایه (GPA – حضور – واحد – دروس افتاده – سن - جنسیت)

### خروجی : Risk-Score(0…1) – عوامل موثر – confidence - Risk-Level(Low/High/Medium)

### نیازمند نقش : factuly – Admin

ساختار پوشه‌ها:

* backend/app/api/v1/: روت‌های API (auth, users, students, courses, enrollments, grades, scheduling, analytics)
* backend/app/models/: مدل‌های SQLAlchemy
* backend/app/schemas/: اسکیماهای Pydantic
* backend/app/services/: منطق دامنه (authz, tokens, pagination, scheduling, analytics)
* backend/app/db/: Base و Session و مهاجرت‌ها (alembic/)
* backend/tests/: تست‌های pytest برای همه‌ی ماژول‌ها

مایگریشن‌ها (Alembic):

0001\_students، 0002\_users، 0003\_courses، 0004\_enrollments، 0005\_grades

CI:

* نصب وابستگی‌ها و اجرای pytest روی main
* کش pip برای سرعت

### فهرست EndPointکلید های نمونه

Auth:

POST /api/v1/auth/register (JSON)

POST /api/v1/auth/login (OAuth2 Password Form)

Users/Students/Courses/Enrollments/Grades:

GET /api/v1/<entity>/?limit&offset (لیست با صفحه‌بندی)

POST /api/v1/<entity>/ (ایجاد)

GET /api/v1/<entity>/{id} (جزییات)

PATCH /api/v1/<entity>/{id} (ویرایش)

DELETE /api/v1/<entity>/{id} (حذف)

RBAC: ایجاد/ویرایش/حذف Enrollments/Grades فقط برای faculty/admin

Scheduling:

POST /api/v1/scheduling/generate (Greedy v0)

Analytics:

POST /api/v1/analytics/at-risk (نیازمند faculty/admin)

GET /api/v1/analytics/health

### تست و تضمین کیفیت

* تست‌های واحد و یکپارچه: health/auth/CRUD/RBAC/pagination/scheduling/analytics
* جداسازی دیتابیس تست با SQLite و تنظیم متغیر محیطی در هر تست
* اجرای خودکار در CI روی GitHub Actions
* پوشش سناریوهای مثبت/منفی (Unauthorized/Forbidden/Conflict/NotFound)

### امنیت و حریم خصوصی

* JWT با انقضای قابل پیکربندی
* RBAC مبتنی بر نقش
* هش گذرواژه با bcrypt
* پیشنهاد فاز بعد: Rate limiting, CORS policy, Security headers, Audit logging

### مستندسازی

* docs/ شامل RFP, SRS, HLA, Architecture, ERD, OpenAPI (اسکلت), TestPlan, SecurityPrivacy
* README شامل مقدمه و Quickstart و دستور اجرای محلی/Compose
* پیشنهاد: به‌روزرسانی OpenAPI کامل و تولید مستند خودکار با ReDoc/Swagger

### وضعیت مخزن و فرایند توسعه

* Conventional Commits (feat, chore, test, ci, authz)
* PR/CI روی شاخه‌ی main
* نمونه کامیت‌های مهم:
* feat(auth): JWT + endpoints + tests
* feat(db): Alembic setup + migrations
* feat(api): pagination helper + tests
* feat(scheduling): greedy endpoint + tests
* feat(analytics): at-risk prediction + tests

### ریسک‌ها و مدیریت آن‌ها

* داده‌ی واقعی دانشگاه: فعلاً از داده‌ی مصنوعی/نمونه استفاده می‌کنیم؛ آماده برای اتصال به SIS/LMS
* امنیت و تبعیت: نیاز به سخت‌گیری بیشتر پیش از ارائه عمومی (SSO، Audit، محدودیت نرخ)
* مقیاس‌پذیری: فعلاً MVP تک‌نمونه‌ای؛ برنامه‌ی ارتقا به K8s/ELK/Grafana در نقشه‌ی راه

### نقشه‌ی راه (Roadmap کوتاه)

* Sprint بعدی:
  + Frontend v1 (React + i18n) و اتصال به API
* تکمیل OpenAPI و مستندسازی ماشینی
* Reporting (CSV/PDF) و چند KPI پایه
* Observability اولیه (لاگ ساختاریافته، health/metrics)
* میان‌مدت:
* Scheduling v1 (الگوریتم بهتر/قیود بیشتر)
* Analytics v2 (ویژگی‌های غنی‌تر، مدل‌های پیشرفته)
* Recommender v1
* Integrations (LMS/SIS)
* بلندمدت:
* K8s، ELK/Grafana، SSO/Keycloak، CI/CD کامل، امنیت پررنگ

نتیجه گیری

پروژه بر اساس RFP/SRS با معماری مناسب و پیاده‌سازی تدریجی پیش می‌رود. هسته‌ی  Backend پایدار، تست‌پوش و قابل گسترش است. مسیر بعدی روی  UI/داشبورد و ارتقای تحلیل‌ها و کیفیت عملیاتی متمرکز خواهد بود. این نسخه برای ارائه‌ی میان‌دوره به دپارتمان قابل اتکا است و برای دمو آماده‌سازی ساده‌ای لازم دارد.

### پیوست‌ها

* اسکرین‌شات از موفقیت تست‌ها (CI green)
* نمونه پاسخ  API (JSON) برای چند endpoint
* دیاگرام ساده از جریان Scheduling/Analytics
* جدولی از نقش‌ها و دسترسی‌ها (RBAC Matrix)
* فهرست وابستگی‌ها (requirements) و نسخه‌ها