

پروژه درس تحلیل و طراحی نرم افزار

موضوع: پلتفرم مدیریت هوشمند دانشگاه

دکتر فیضی

مدت زمان: ۸ هفته تیم: ۸ نفر امتیاز: ۸ نمره

۱. فلسفه و چشم انداز پروژه

هدف این پروژه، صرفاً ساخت یک نرم افزار نیست. بلکه، این پروژه یک سفر یادگیری هوشمندانه است که شما را از تئوری درس "تحلیل و طراحی نرم افزار" به دنیای واقعی مهندسی سیستم های توزیع شده و مقیاس پذیر می برد. ما به دنبال شبیه سازی یک محیط کاری واقعی هستیم که در آن شما به عنوان یک تیم فنی، با چالش های پیچیده معماری روبرو می شوید و برای حل آن ها از ابزارهای مدرن، از جمله دستیارها و ابزارهای هوش مصنوعی، به عنوان یک مربی و مشاور فنی استفاده می کنید.

موفقیت شما در این پروژه نه تنها در کارکرد نهایی کد، بلکه در میزان یادگیری، کیفیت مستندات، توانایی همکاری تیمی و دفاع از تصمیمات فنی تان سنجیده می شود.

۲. خواسته های کلیدی و غیرقابل مذاکره

برای اطمینان از دستیابی به اهداف آموزشی، تمام تیم ها موظف به پیاده سازی موارد زیر در پروتوتایپ خود هستند:

۱. معماری میکروسرویس: سیستم باید به صورت مجموعه ای از سرویس های مستقل و Loosely Coupled پیاده سازی شود.

۲. الگوی Saga: برای مدیریت حداقل یک فرآیند کسب و کار چندمرحله ای (مثلاً خرید بلیط)، باید از این الگو برای تضمین ثبات داده ها در سیستم توزیع شده استفاده شود.

۳. الگوی Circuit Breaker: برای افزایش تاب آوری سیستم، باید از این الگو برای جلوگیری از اثر آبشاری (Cascading Failure) در حداقل یک ارتباط بین سرویسی استفاده شود.

۴. ارتباط رویدادمحور: سرویس ها باید علاوه بر ارتباطات مستقیم، از یک Message Broker مانند RabbitMQ برای ارتباطات غیرهمزمان استفاده کنند.

۵. API Gateway: تمام درخواست های کلاینت باید از یک دروازه واحد عبور کنند.

۳. نیازمندی‌های پروژه

پروتوتایپ شما باید قابلیت‌های زیر را پوشش دهد. این نیازمندی‌ها، بوم شما برای پیاده‌سازی الگوهای معماری هستند.

۳.۱. نیازمندی‌های عملکردی (Functional Requirements)

این‌ها قابلیت‌های اصلی سیستم هستند که باید در پروتوتایپ شما کار کنند.

ماژول کاربر و احراز هویت (User & Auth Service):

- FR-۰۱: کاربران (دانشجو، استاد) باید بتوانند ثبت‌نام و وارد سیستم شوند.
- FR-۰۲: سیستم باید از توکن‌های JWT برای احراز هویت در درخواست‌های API استفاده کند.

ماژول منابع و رزرواسیون: (Resource & Booking Service)

- FR-۰۳: کاربران باید بتوانند لیست منابع قابل رزرو (مثلاً اتاق گروهی) را مشاهده کنند.
- FR-۰۴: کاربران باید بتوانند یک بازه زمانی برای یک منبع رزرو کنند. سیستم باید از رزرو همزمان (Overbooking) جلوگیری کند.

ماژول بازارچه خدمات و رویدادها: (Marketplace Service)

- FR-۰۵: یک "فروشنده" (مثلاً دانشکده) باید بتواند یک "محصول" (مثلاً بلیط کارگاه) تعریف کند.
- FR-۰۶: کاربران باید بتوانند محصول را به سبد خرید اضافه کرده و فرآیند خرید را تکمیل کنند.
- پیاده‌سازی Saga: فرآیند نهایی خرید (شامل کاهش موجودی و تأیید سفارش) باید با الگوی Saga پیاده‌سازی شود.

ماژول آموزش و آزمون آنلاین: (E-Learning & Exam Service)

- FR-۰۷: یک استاد باید بتواند یک آزمون ساده ایجاد کند.
- FR-۰۸: یک دانشجو باید بتواند در آزمون شرکت کند.
- پیاده‌سازی Circuit Breaker: این سرویس هنگام ارسال اعلان شروع آزمون به سرویس اعلان، باید از الگوی Circuit Breaker استفاده کند.

ماژول‌های تکمیلی (برای تکمیل پروتوتایپ):

- FR-09: یک داشبورد باید وضعیت زنده یک سنسور مجازی (مثلاً دمای یک کلاس) را نمایش دهد.
- FR-10: موقعیت یک شاتل مجازی باید روی یک نقشه نمایش داده شود.

۳.۲. نیازمندی‌های غیرعملکردی (Non-Functional Requirements)

این‌ها ویژگی‌های کیفی سیستم هستند که تصمیمات معماری شما را شکل می‌دهند.

این بخش، هسته اصلی چالش معماری پروژه است.

مقیاس‌پذیری: (Scalability)

- NFR-S01: معماری سیستم باید بتواند با افزایش ۱۰ برابری تعداد کاربران همزمان (مثلاً در زمان انتخاب واحد یا رزرو کلاس) با افزودن منابع (مقیاس‌پذیری افقی) کارایی خود را حفظ کند.
- NFR-S02: سیستم باید بتواند تعداد نامحدودی "فروشنده" و "محصول" در بازارچه پشتیبانی کند.

- پیامد معماری: این نیازمندی، معماری میکروسرویس را الزامی می‌کند. تیم باید سرویس‌ها را Stateless طراحی کند و از Load Balancer و Message Queue برای پردازش‌های غیرهمزمان (مانند تأیید سفارش) استفاده نماید.

چندمستأجاری: (Multi-Tenancy)

- NFR-MT01: داده‌های هر دانشکده یا "فروشنده" باید به طور کامل از سایرین ایزوله باشد. یک مدیر دانشکده نباید به داده‌های دانشکده دیگر دسترسی داشته باشد.
- پیامد معماری: این یک چالش بزرگ در طراحی دیتابیس و لایه اپلیکیشن است. تیم باید بین استراتژی‌های مختلف (مانند Database-per-Tenant یا Schema-per-Tenant) تصمیم‌گیری کرده و منطق ایزوله‌سازی را در تمام لایه‌ها پیاده‌سازی کند.

کارایی: (Performance)

- NFR-P01: زمان پاسخ‌دهی ۹۵٪ از درخواست‌های API باید کمتر از ۴۰۰ میلی‌ثانیه باشد.
- پیامد معماری: تیم باید از کشینگ توزیع‌شده (Redis)، بهینه‌سازی کوئری‌های دیتابیس و پردازشگرهای رویداد (Event Processors) برای داده‌های IoT استفاده کند.

امنیت: (Security)

- NFR-SE۰۱: سیستم باید از احراز هویت مبتنی بر JWT استفاده کند و دسترسی‌ها بر اساس نقش (RBAC) کنترل شود.
 - NFR-SE۰۲: در زمان برگزاری آزمون آنلاین، سیستم باید از دسترسی‌های غیرمجاز به سؤالات یا پاسخ‌های دیگران جلوگیری کند.
 - پیامد معماری: طراحی یک سرویس احراز هویت/مجوزدهی (Auth Service) و پیاده‌سازی مکانیزم‌های امنیتی در لایه API و پایگاه داده ضروری است.
- قابلیت اطمینان: (Reliability)
- NFR-R۰۱: در صورت از کار افتادن سرویس مانیتورینگ IoT، هسته اصلی سیستم (مانند رزرو منابع) باید به کار خود ادامه دهد.
 - NFR-R۰۲: سیستم نباید در زمان رزرو منابع، دچار "تخصیص بیش از حد (Overbooking)" شود، حتی زیر بار سنگین.
 - پیامد معماری: این نیازمندی به طراحی مقاوم در برابر خطا (Fault Tolerance) منجر می‌شود. استفاده از الگوی Circuit Breaker در ارتباط بین سرویس‌ها و مدیریت صحیح تراکنش‌ها برای جلوگیری از Overbooking الزامی است.
- قابلیت نگهداری: (Maintainability)
- NFR-MN۰۱: کد منبع باید از اصول SOLID پیروی کند و حداقل ۵ الگوی طراحی (مانند Observer برای اعلانات، Strategy برای انواع پرداخت، State برای وضعیت آزمون) به کار رود.
 - NFR-MN۰۲: تمام تصمیمات مهم معماری باید در اسناد Architecture Decision Records (ADRs) مستند شوند.
 - پیامد معماری: این نیازمندی بر طراحی ماژولار با کوپلینگ کم و مستندسازی کامل تأکید دارد.

۴. ساختار تیم شما: یک تیم عمومی و یک مربی هوش مصنوعی

برای تمرکز بر یادگیری جمعی و کاهش فشار نقش‌های تخصصی، تیم شما به صورت زیر عمل می‌کند:

- تیم عمومی ۸ نفره: شما یک تیم یکپارچه هستید. در هر فاز، مسئولیت‌ها را بر اساس علاقه و مهارت‌های فعلی تقسیم می‌کنید. در هر فاز، یک یا دو نفر به عنوان "رهبر فاز (Phase Lead)" انتخاب می‌شوند تا وظایف را هماهنگ کنند.
- عضو نهم تیم: مربی هوش مصنوعی (AI Mentor): شما یک دستیار هوش و بی‌خستگی دارید. از آن برای موارد زیر استفاده کنید:

- ترجمه مفاهیم پیچیده: الگوی Saga را به زبان ساده توضیح بده."
- تولید کد الگو(Boilerplate): "یک پروژه Spring Boot برای سرویس کاربر با JWT ایجاد کن."
- کمک به دیباگ: "این خطا در اتصال به دیتابیس به چه معناست و چگونه رفع می‌شود؟"
- مستندسازی: "یک پیش‌نویس از مستندات API برای این سرویس بنویس."

۵. فازبندی پروژه و میلستون‌ها (۸ هفته)

پروژه به ۴ فاز ۲ هفته‌ای تقسیم شده است. هر فاز یک هدف یادگیری مشخص دارد.

فاز ۱: کشف و بنیان‌گذاری (هفته‌های ۱ و ۲)

- هدف یادگیری: فهمیدن "چه چیزی" می‌سازیم و "چرا" معماری میکروسرویس برای آن مناسب است.
- خروجی (تحویل پایان هفته ۲ - ۱ نمره): یک مخزن GitHub با یک README.md شامل ویکی معماری تیم (دیگرام‌های C4 و تصمیمات اولیه).

فاز ۲: ساخت هسته اولیه (هفته‌های ۳ و ۴)

- هدف یادگیری: یادگیری "چگونه" دو میکروسرویس اولیه را بسازیم و آن‌ها را متصل کنیم.
- خروجی (تحویل پایان هفته ۴ - غیرنمره‌ای): یک ویدیوی ۳ دقیقه‌ای از اجرای سیستم هسته (ورود و مشاهده منابع).

فاز ۳: مواجهه با الگوهای پیشرفته (هفته‌های ۵ و ۶)

- هدف یادگیری: فهمیدن "چرا" الگوهای Saga و Circuit Breaker لازم هستند و "چگونه" آن‌ها را پیاده‌سازی کنیم.
- خروجی (تحویل پایان هفته ۶ - ۱ نمره): یک فایل Learning_Report.md که توضیح می‌دهد چگونه این الگوها را فهمیده و پیاده‌سازی کرده‌اید.

فاز ۴: یکپارچه‌سازی و روایتگری (هفته‌های ۷ و ۸)

- هدف یادگیری: توانایی "داستان معماری" پروژه را به صورت شفاف تعریف کنیم.
- خروجی (تحویل نهایی - ۶ نمره): کد کامل، ارائه نهایی، گزارش تعامل با AI و مستندات نهایی.

۶. فرایند ارزیابی و بارمبندی (مجموع ۸ نمره)

بخش	معیار ارزیابی	امتیاز
۱. فرآیند و مستندات (۳ نمره)		
تحويل ۱ (ویکی معماری)	کیفیت و کامل بودن دیاگرام‌ها و وضوح تصمیمات اولیه.	۱
گزارش تعامل با AI	نشان‌دهنده استفاده هوشمندانه و عمیق از AI برای حل مشکلات.	۱
گزارش یادگیری (Learning Report)	درک عمیق از چالش‌های Saga و Circuit Breaker	۱
۲. کیفیت فنی و معماری (۳ نمره)		
کیفیت معماری	انطباق پیاده‌سازی با دیاگرام‌ها، طراحی شل بین سرویس‌ها.	۱.۵
صحت عملکرد پروتوتایپ	اجرای موفقیت‌آمیز جریان‌های اصلی با docker-compose	۱.۵
۳. ارائه و دفاع از کار (۲ نمره)		
کیفیت ارائه	روایتگری قانع‌کننده، وضوح اسلایدها و مدیریت زمان.	۱
دفاع فنی	توانایی پاسخگویی به سوالات فنی و دفاع از تصمیمات.	۱
مجموع		

۷. چک‌لیست خروجی نهایی (در مخزن GitHub)

در پایان هفته هشتم، مخزن GitHub شما باید شامل موارد زیر باشد:

- کد منبع کامل تمام سرویس‌ها و فرانت‌اند.
- فایل docker-compose.yml برای اجرای کل سیستم.
- فایل ارائه نهایی (PDF یا PPT).
- فایل AI_Log.md (گزارش تعامل با AI).
- فایل Learning_Report.md.
- ویکی معماری تیم (به‌روز شده).

۸. سخنان پایانی

این پروژه یک فرصت منحصر به فرد برای رشد فنی و حرفه‌ای شماست. از چالش‌ها نترسید، از هوش مصنوعی به عنوان یک ابزار قدرتمند استفاده کنید و مهم‌تر از همه، از همکاری با یکدیگر لذت ببرید.

موفقیت شما در میزان یادگیری، همکاری و توانایی شما در روایتگری داستان فنی‌تان تعریف می‌شود. به این سفر هیجان‌انگیز خوش آمدید!