1. انواع معماری نرم‌افزار

انواع معماری نرم‌افزار؛ هر ساختمانی نقشه خودش را می‌خواهد

یکی از اشتباهات رایج این است که فکر کنیم یک معماری برای همه پروژه‌ها مناسب است.

همان‌طور که نقشه یک خانه با نقشه یک فرودگاه فرق می‌کند، معماری نرم‌افزار هم به اندازه و نیاز پروژه بستگی دارد.

---

1. معماری Monolithic

ساده‌ترین و رایج‌ترین معماری

در این مدل تقریباً همه چیز داخل یک پروژه قرار می‌گیرد:

- Users
- Products
- Orders
- Payments
- Admin

همه با هم زندگی می‌کنند. 😄

مزایا

- یادگیری آسان
- توسعه سریع برای پروژه‌های کوچک
- دیپلوی ساده
- مناسب برای تیم‌های کوچک

معایب

- با بزرگ شدن پروژه مدیریت سخت می‌شود
- تغییر یک بخش ممکن است بخش‌های دیگر را خراب کند
- مقیاس‌پذیری محدودتر است

مثال واقعی

اکثر پروژه‌های Django که دانشجوها می‌سازند در ابتدا Monolithic هستند.

حتی خیلی از استارتاپ‌های موفق هم کارشان را با Monolith شروع کرده‌اند.

---

2. معماری Layered

معماری طبقه‌ای

در این مدل هر بخش مسئولیت مشخصی دارد.

مثلاً:

- Presentation Layer
- Business Layer
- Data Layer

مثل یک ساختمان چند طبقه.

مزایا

- سازماندهی بهتر
- تست آسان‌تر
- نگهداری راحت‌تر

معایب

- گاهی پیچیدگی اضافه ایجاد می‌کند
- برای پروژه‌های خیلی کوچک ممکن است بیش از حد باشد

مثال

بیشتر پروژه‌های سازمانی از این سبک استفاده می‌کنند.

---

3. معماری MVC

یکی از معروف‌ترین معماری‌های دنیا.

Model

مدیریت داده‌ها

مثال:

- کاربر
- محصول
- سفارش

View

نمایش اطلاعات به کاربر

Controller

مدیریت درخواست‌ها و منطق برنامه

مزایا

- ساختار واضح
- توسعه راحت‌تر
- یادگیری نسبتاً آسان

معایب

- در پروژه‌های بزرگ ممکن است Controllerها بیش از حد بزرگ شوند
- مدیریت وابستگی‌ها سخت‌تر می‌شود

مثال

- Laravel
- ASP.NET MVC
- بسیاری از فریم‌ورک‌های وب

---

4. معماری Microservices

محبوب شرکت‌های بزرگ

در این مدل پروژه به چند سرویس مستقل تقسیم می‌شود.

مثلاً:

- User Service
- Payment Service
- Notification Service
- Appointment Service

هر سرویس زندگی خودش را دارد.

مزایا

- مقیاس‌پذیری عالی
- توسعه مستقل هر بخش
- مناسب تیم‌های بزرگ

معایب

- بسیار پیچیده‌تر
- دیپلوی سخت‌تر
- مانیتورینگ دشوارتر
- ارتباط بین سرویس‌ها چالش‌برانگیز است

مثال

شرکت‌های بزرگی مثل:

- Netflix
- Amazon
- Uber

از معماری‌های نزدیک به Microservices استفاده می‌کنند.

---

5. معماری Modular

بهترین دوست پروژه‌های متوسط

در این مدل پروژه به ماژول‌های مستقل تقسیم می‌شود.

- Users Module
- Payments Module

- Appointments Module
- Reports Module

اما همه هنوز داخل یک پروژه هستند.

مزایا

- ساده‌تر از Microservices
- مرتب‌تر از Monolith
- توسعه آسان
- مناسب اکثر پروژه‌های واقعی

معایب

- استقلال کامل Microservices را ندارد
- در پروژه‌های خیلی بزرگ محدودیت پیدا می‌کند

پس کدام معماری بهتر است؟

این سؤال شبیه این است که بپرسیم:

کامیون بهتر است یا موتور؟

بستگی دارد چه کاری می‌خواهید انجام دهید.

برای اکثر دانشجوها و استارتاپ‌ها:

Monolithic Modular بهترین انتخاب است.

برای پروژه‌های کوچک:

Monolithic کاملاً کافی است.

برای پروژه‌های خیلی بزرگ:

Microservices می‌تواند انتخاب مناسبی باشد.

اشتباه خطرناک

بعضی افراد قبل از داشتن حتی ۱۰ کاربر سراغ Microservices می‌روند!

انگار برای رفتن به سوپرمارکت محله با هواپیمای شخصی سفر کنید. 😄

جمع‌بندی

معماری خوب آن معماری نیست که پیچیده‌تر باشد.

معماری خوب آن است که:

- نیاز فعلی پروژه را برطرف کند
- توسعه آینده را آسان کند
- پیچیدگی غیرضروری ایجاد نکند

یک معمار حرفه‌ای به دنبال «بهترین معماری دنیا» نیست؛

به دنبال «مناسب‌ترین معماری برای مسئله فعلی» است.