# ADR-003: انتخاب Clean Architecture به‌همراه DDD برای ساختار زیرساخت برنامه

## وضعیت (Status)

Accepted

## زمینه و مشکل (Context)

ما نیاز به یک معماری داخلی داشتیم که ویژگی‌های زیر را تأمین کند:  
  
- ساختار منسجم و لایه‌بندی مناسب برای پروژه‌ها.  
- امکان توسعه و نگهداری آسان توسط تیم‌های مختلف.  
- پشتیبانی از رویکرد Domain-Driven Design (DDD) برای مدیریت بهتر پیچیدگی دامنه.  
- ایجاد استانداردی مشترک و شناخته‌شده برای توسعه‌دهندگان.  
  
الگوهای معماری متداول مانند Hexagonal و Onion بررسی شدند، اما نیاز بود معماری انتخابی ما هم از نظر مفهومی ساده‌تر باشد و هم تکامل‌یافته‌تر برای نیازهای پروژه‌های ما.

## تصمیم (Decision)

ما تصمیم گرفتیم از Clean Architecture استفاده کنیم که در مقایسه با Hexagonal و Onion:  
  
- ساختار شفاف‌تری از لایه‌ها (Entities, Use Cases, Interfaces, Infrastructure) ارائه می‌دهد.  
- با اصول DDD هم‌راستا است و اجازه می‌دهد منطق دامنه به‌خوبی ایزوله بماند.  
- نسبت به Hexagonal/Onion شناخته‌شده‌تر است و توسعه‌دهندگان بیشتری با آن آشنایی دارند.  
- در عین حال انعطاف کافی برای اضافه کردن زیرساخت‌ها (Persistence, Messaging, External Services) به‌صورت جدا از هسته دامنه را فراهم می‌کند.

## پیامدها (Consequences)

مزایا:  
- انسجام بیشتر در ساختار پروژه‌ها.  
- ساده‌تر شدن یادگیری و پذیرش برای تیم توسعه.  
- قابلیت تکامل با اصول DDD.  
- تفکیک واضح مسئولیت‌ها بین لایه‌ها.  
- قابل مقایسه و در مواردی کامل‌تر از Hexagonal و Onion.  
  
معایب / چالش‌ها:  
- ممکن است در پروژه‌های کوچک کمی پیچیدگی و سربار غیرضروری ایجاد کند.  
- نیاز به نظم و رعایت دقیق لایه‌بندی توسط توسعه‌دهندگان دارد.  
- گاهی تداخل برداشت‌ها از اصطلاحات Clean، Hexagonal و Onion در تیم باعث ابهام می‌شود.

## گزینه‌های بررسی‌شده (Alternatives / Considered Options)

1. Hexagonal Architecture  
 - تمرکز روی Ports & Adapters.  
 - ساختاری خوب برای جداسازی وابستگی‌ها، اما نسبت به Clean معماری ساده‌تری دارد و همه توسعه‌دهندگان به آن مسلط نیستند.  
 - رد شد.  
  
2. Onion Architecture  
 - شبیه به Hexagonal اما با تأکید بیشتر بر هسته دامنه.  
 - ضعف در سادگی و شناخته‌شدن عمومی.  
 - رد شد.  
  
3. Clean Architecture  
 - ساختار شفاف‌تر، شناخته‌شده‌تر، و تکامل‌یافته‌تر.  
 - پذیرفته شد.

## ارجاعات (References)

- Robert C. Martin – Clean Architecture  
- Evans – Domain-Driven Design  
- تجربه تیم داخلی در پروژه‌های پیشین