

Exploring the Inside-Out and Outside-In Approaches

Achieving Optimal Balance

Indumathi Chinnaswamy
Flying module - Presentation day
Date: 01.06.2023



TOM
FISH
BURNE

Introduction

- Importance of software development approaches
- Maintainable and customizable
- Choose the right approach

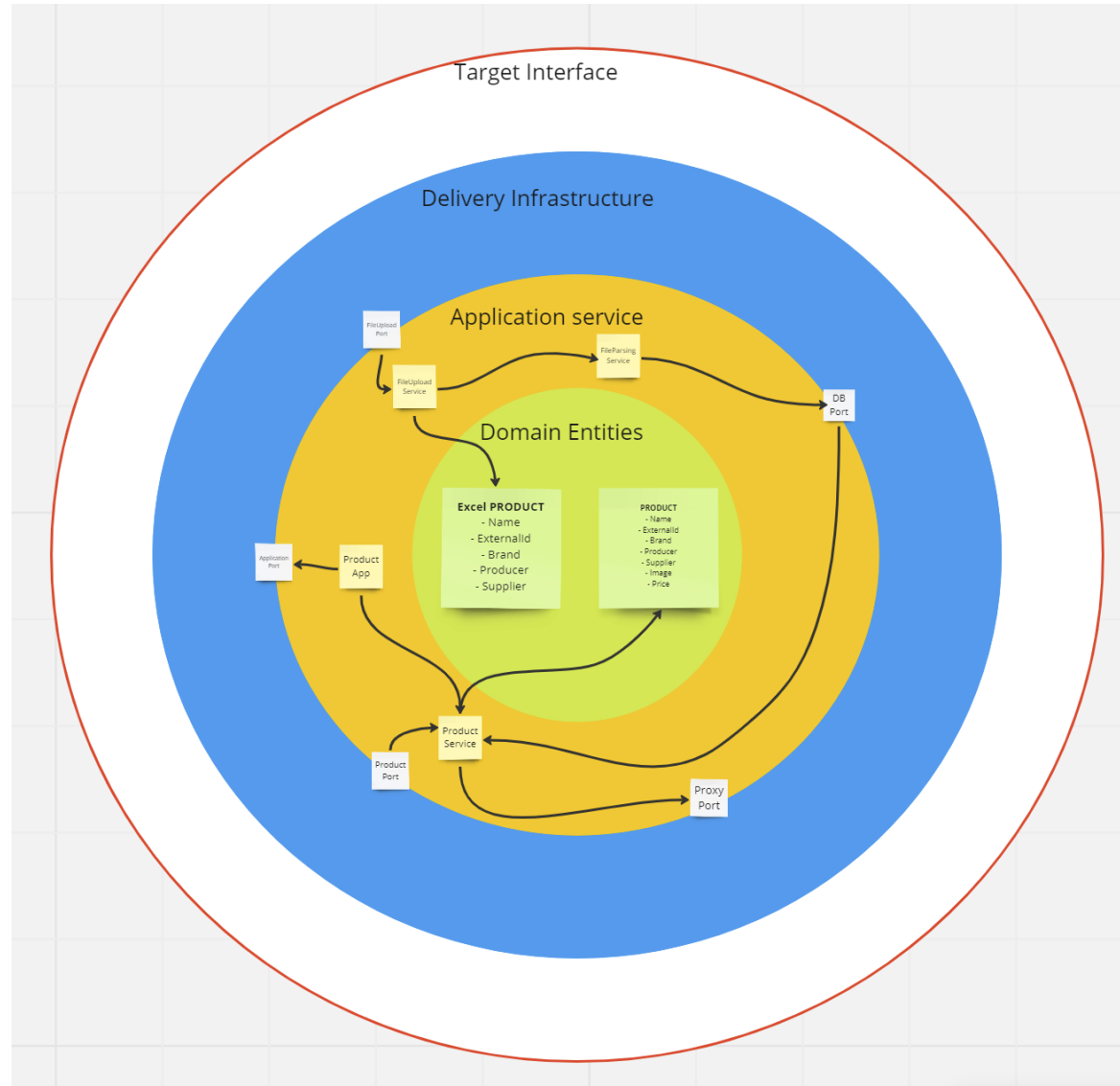
Inside-Out Approach

- Starts with the domain entity level (bottom-up).
- **Clear structure** from components to the system.
- Comprehensive **test coverage** from the beginning.
- Facilitates **iterative** development and **refactoring**.
- **Reusability** across different clients
- Require significant **upfront planning** and design.

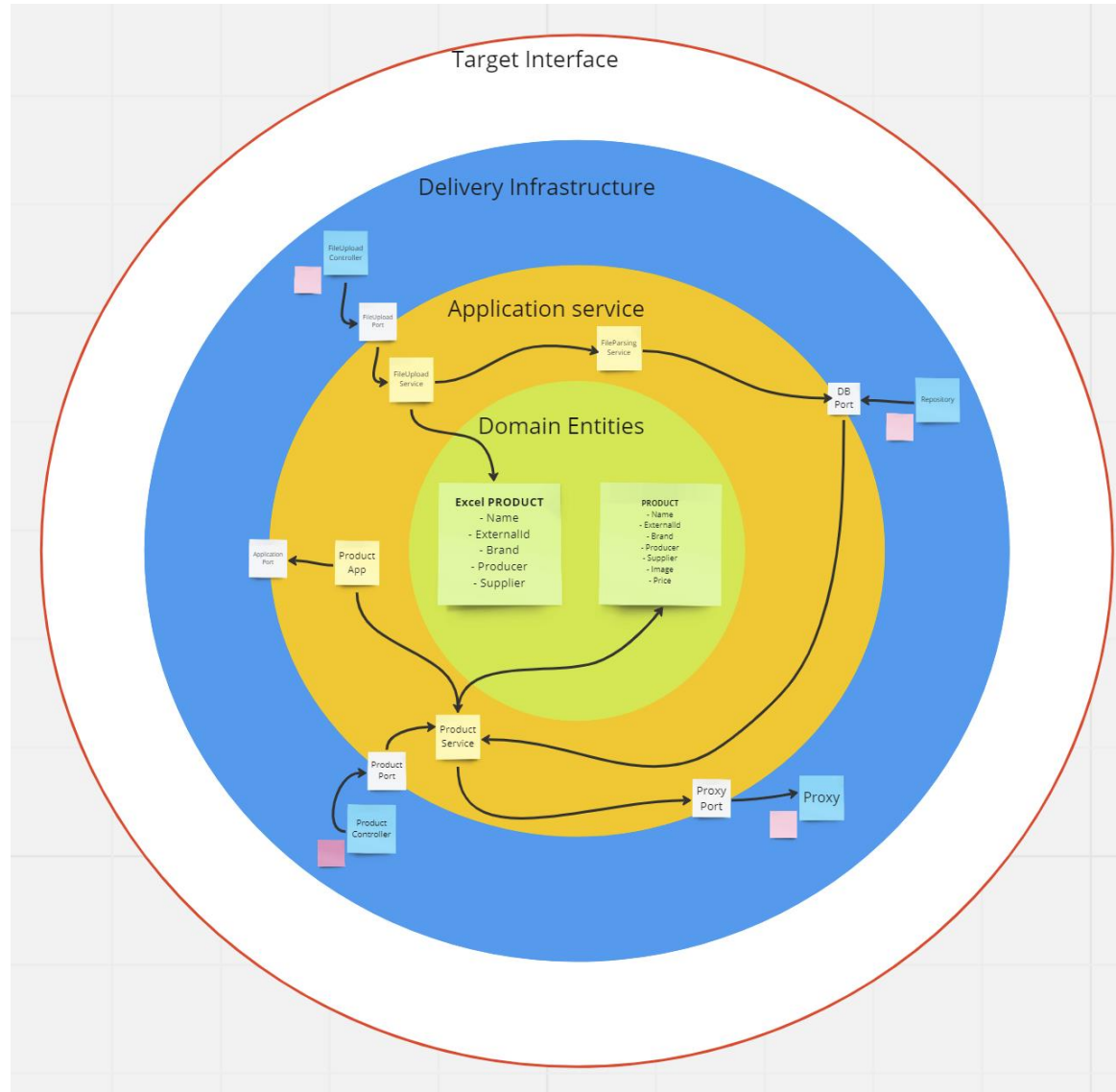
Clean/Onion/Hexagonal Architecture



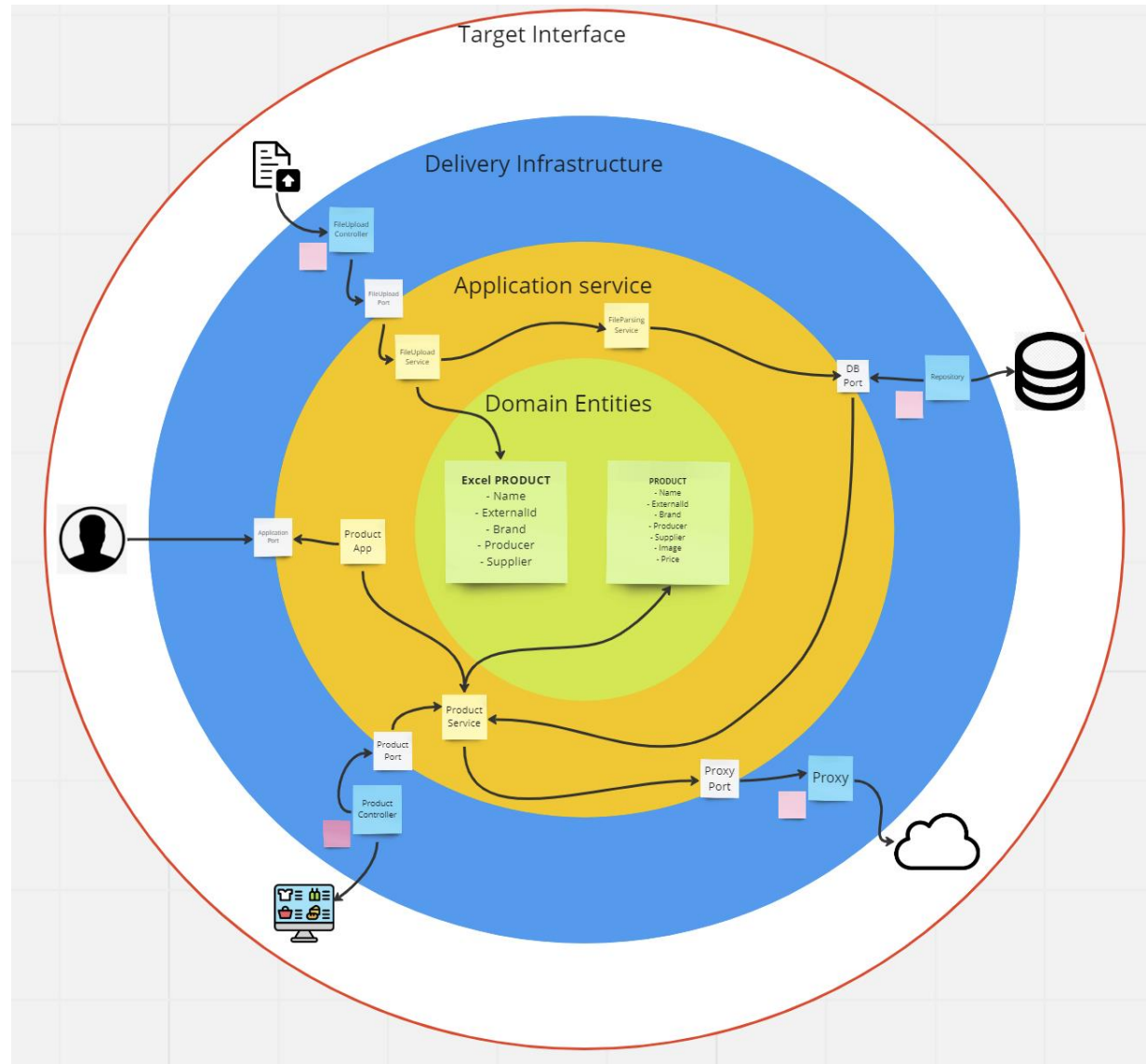
Clean/Onion/Hexagonal Architecture



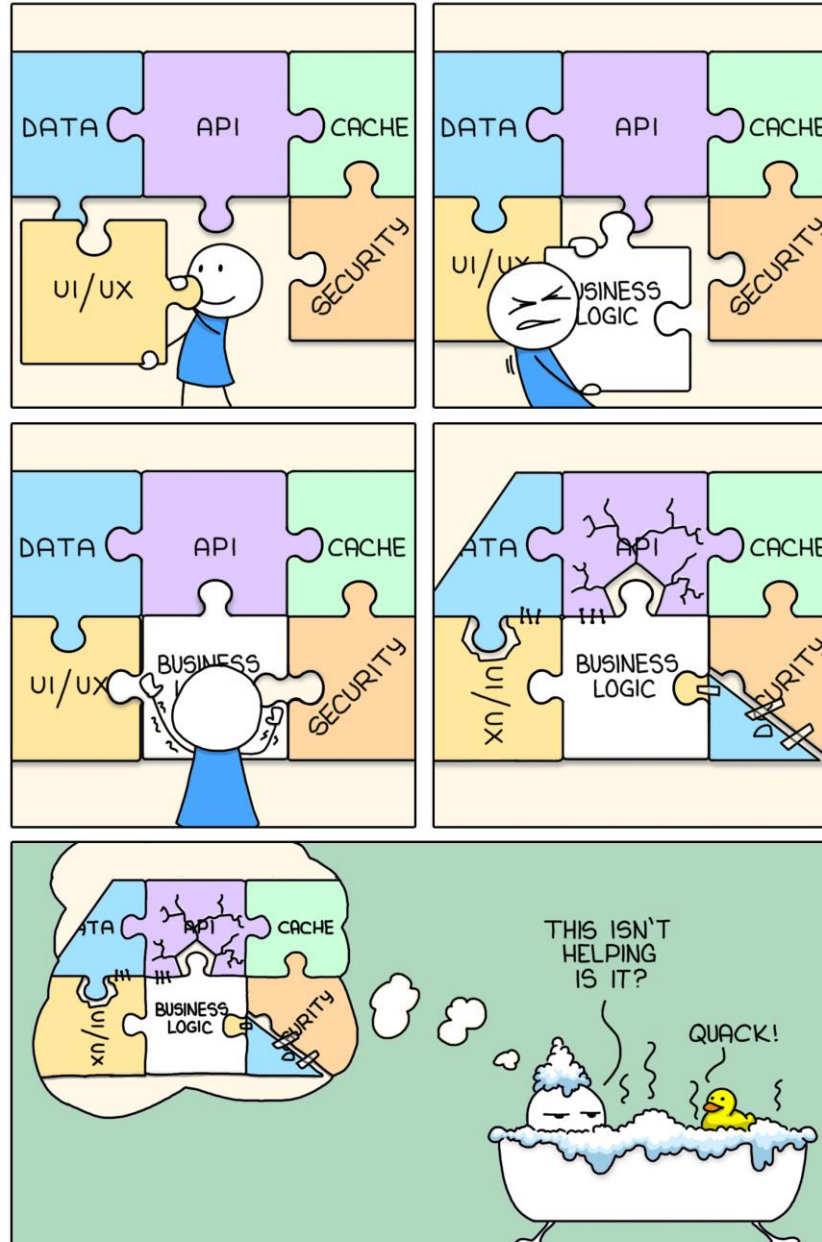
Clean/Onion/Hexagonal Architecture



Clean/Onion/Hexagonal Architecture



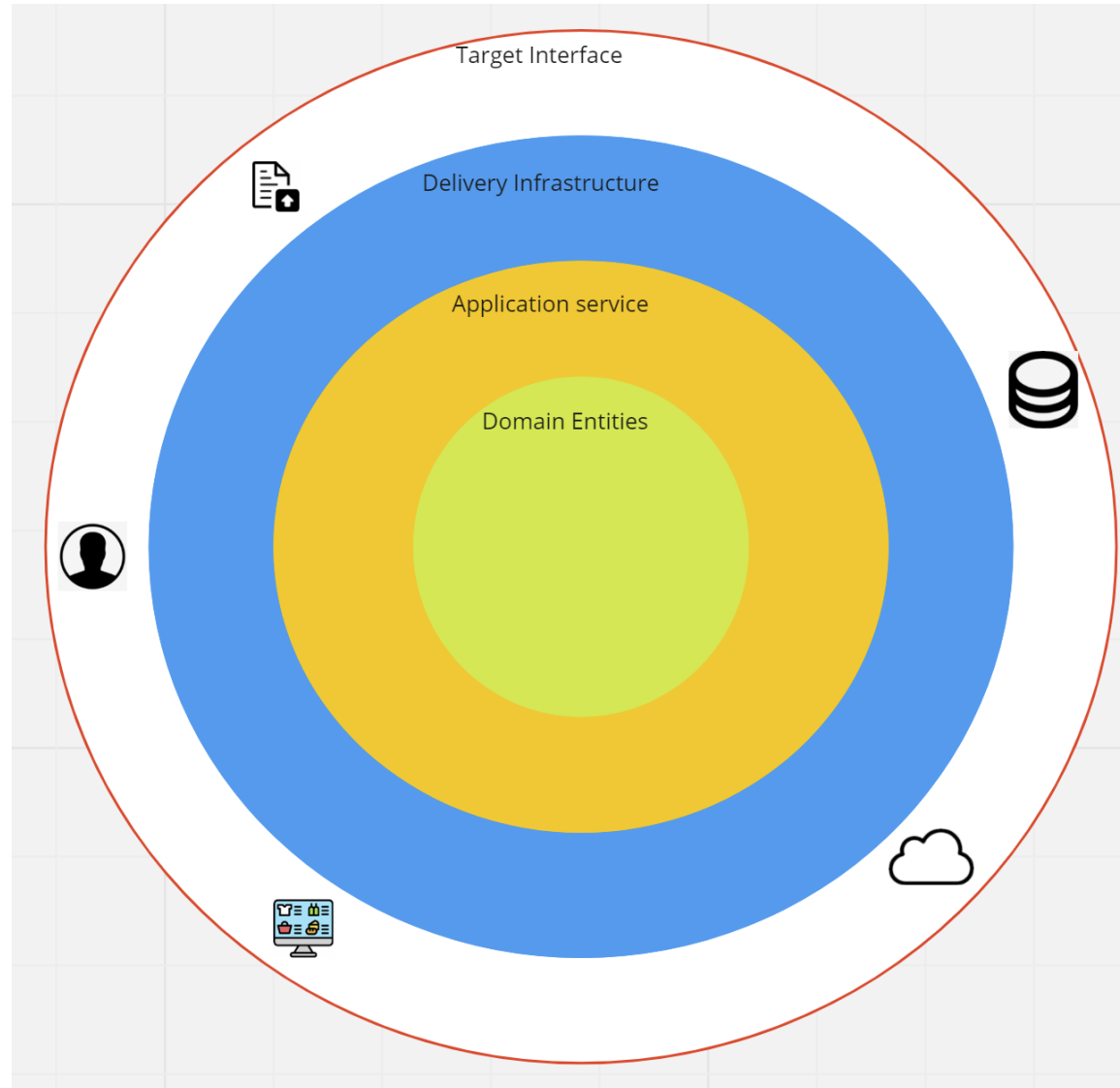
ARCHITECTURE



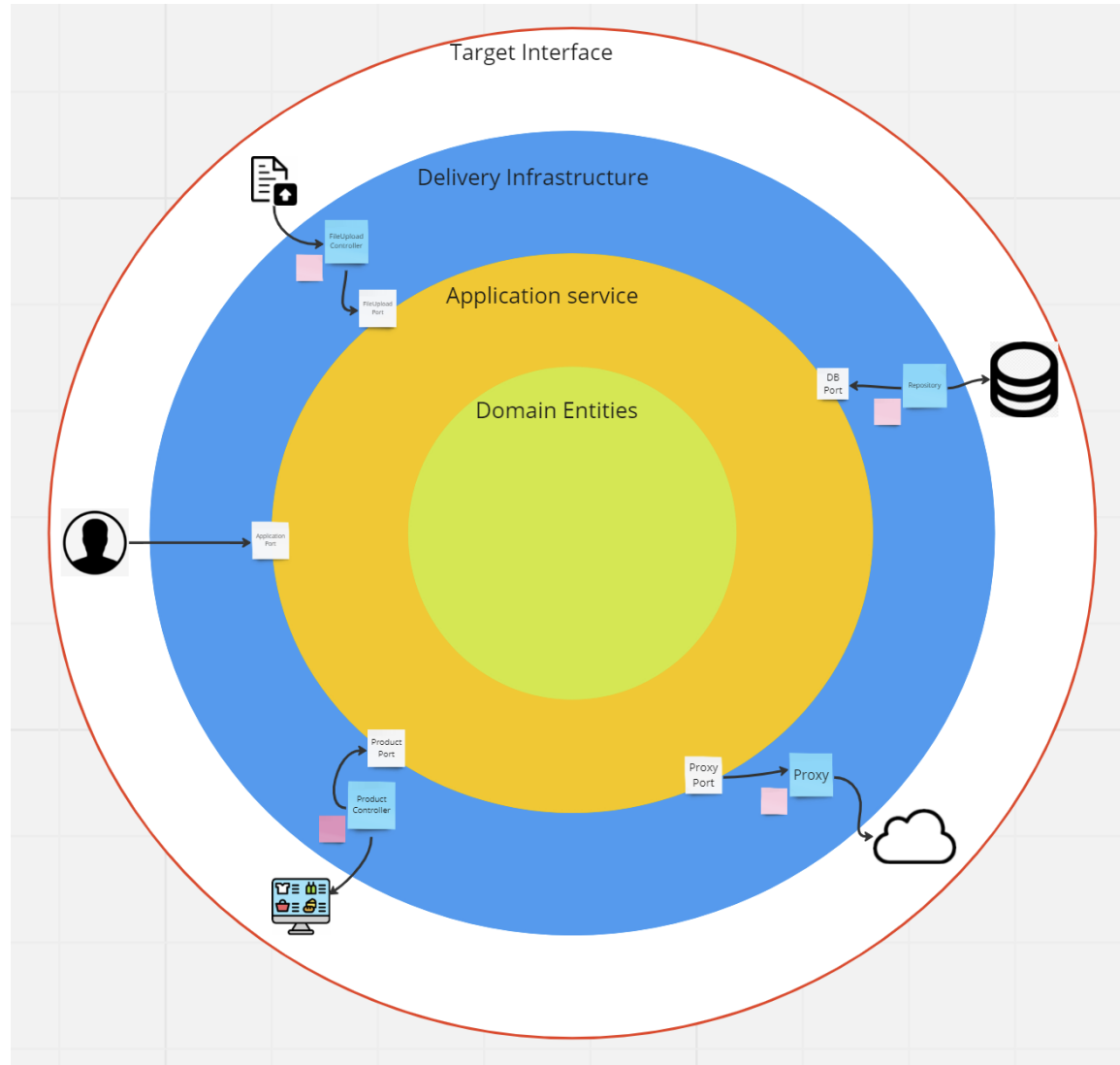
Outside-In Approach

- Top-down or "mockist TDD"
- **User experience** and external interfaces
- Rapid prototyping and iterations
- **Mock dependencies** and implement actual functionality.
- **Flexibility** for customization
- Early **validation** of high-level interactions and integration
- Clear **understanding** of system dependencies.
- Alignment with the "You Ain't Gonna Need It" (**YAGNI**) principle.

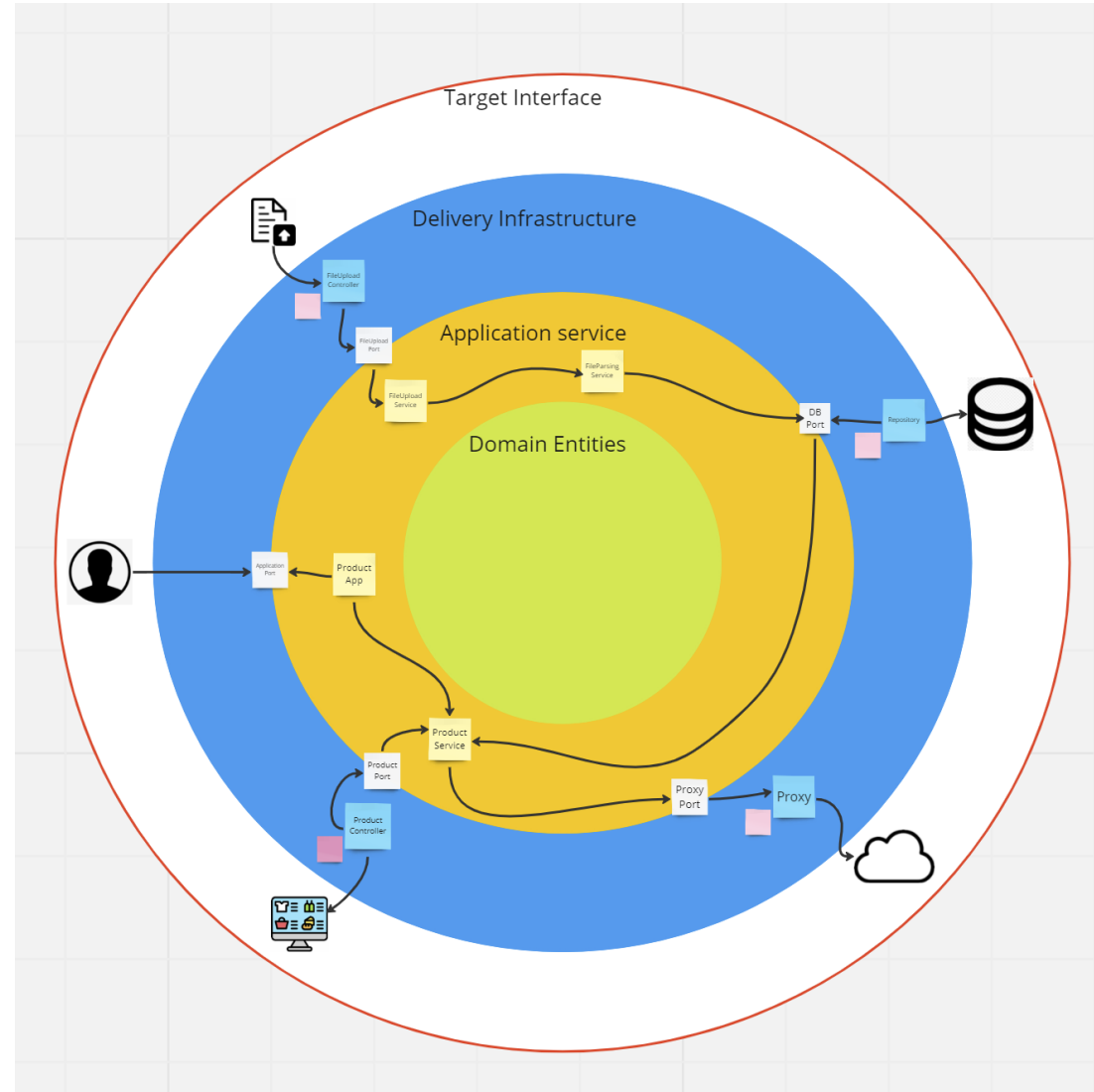
Clean/Onion/Hexagonal Architecture



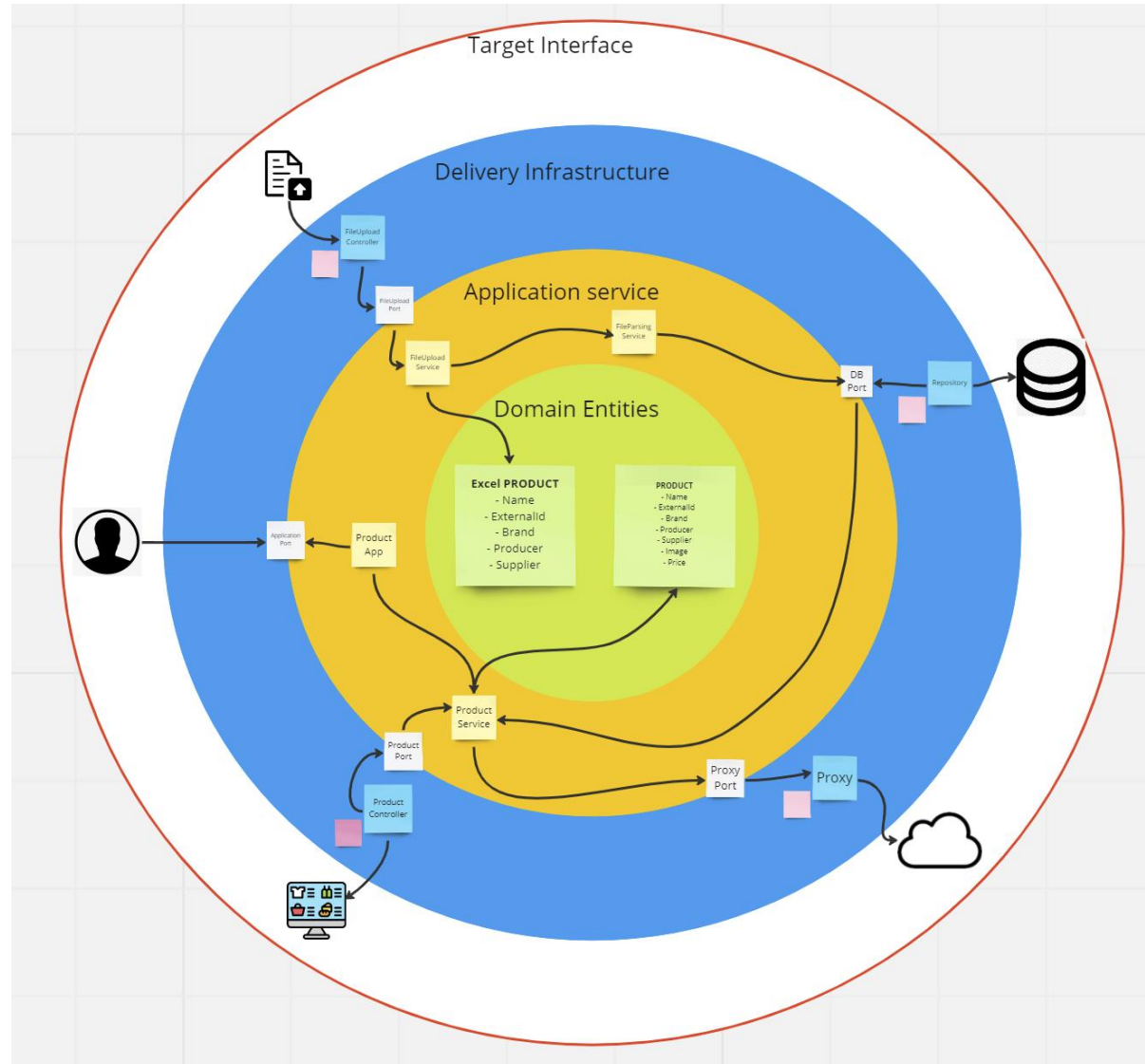
Clean/Onion/Hexagonal Architecture



Clean/Onion/Hexagonal Architecture



Clean/Onion/Hexagonal Architecture



THE LIFE OF A SOFTWARE
ENGINEER.

CLEAN SLATE. SOLID
FOUNDATIONS. THIS TIME
I WILL BUILD THINGS THE
RIGHT WAY.



Can you make it do
A, B, C, and D?

And have it done next
week?

MUCH LATER...

OH MY. I'VE
DONE IT AGAIN,
HAVEN'T I ?





Customer-centric
outside-in

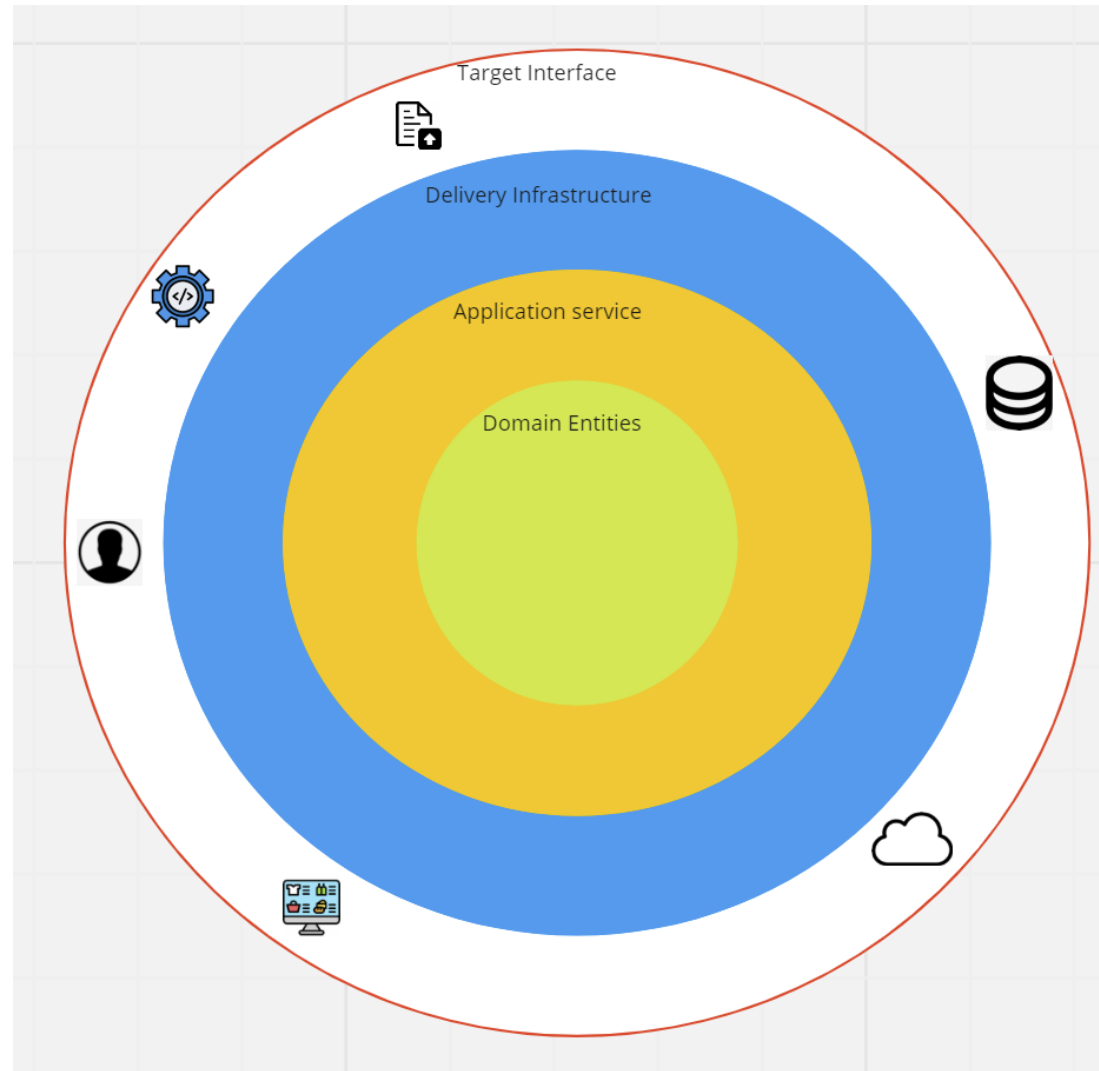


Product-centric
inside-out

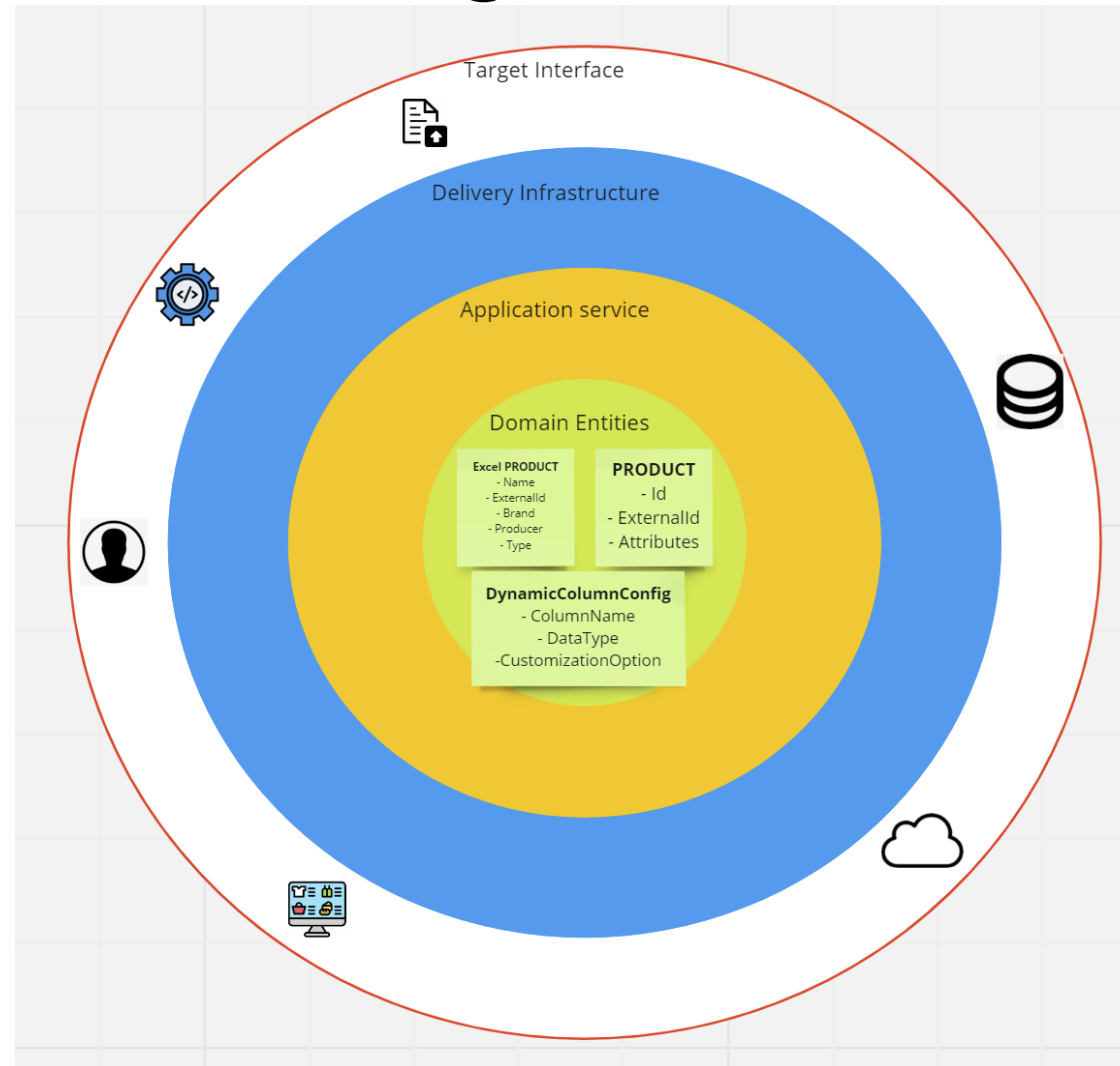
Combining the Approaches

- Integrates Inside-Out and Outside-In.
- Situations where the combined approach is beneficial:
 - **Large** enterprise solutions with **existing** architectural designs.
 - Uncertain **requirements** or evolving codebase.
 - Projects requiring a balance between high-level interactions and component-level details.

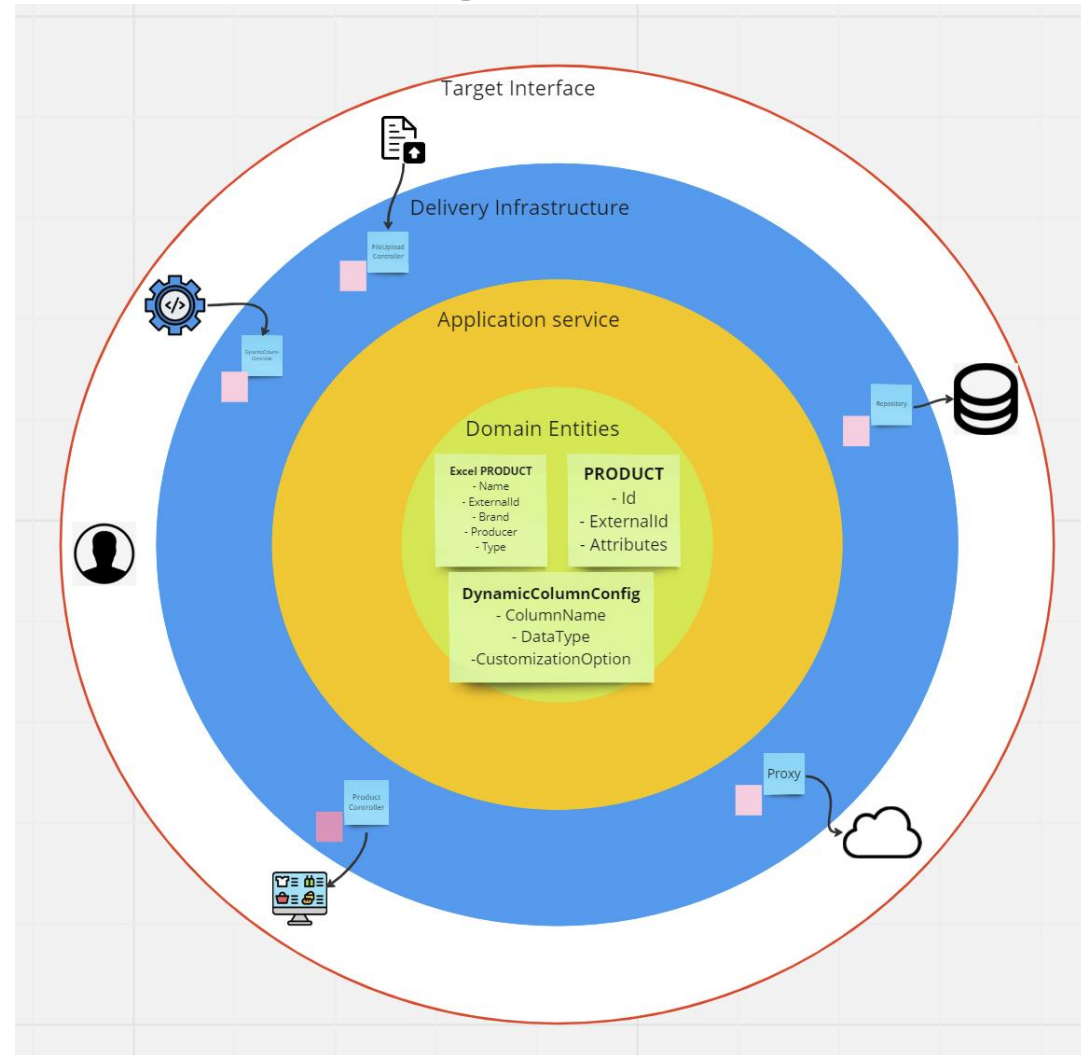
Clean/Onion/Hexagonal Architecture



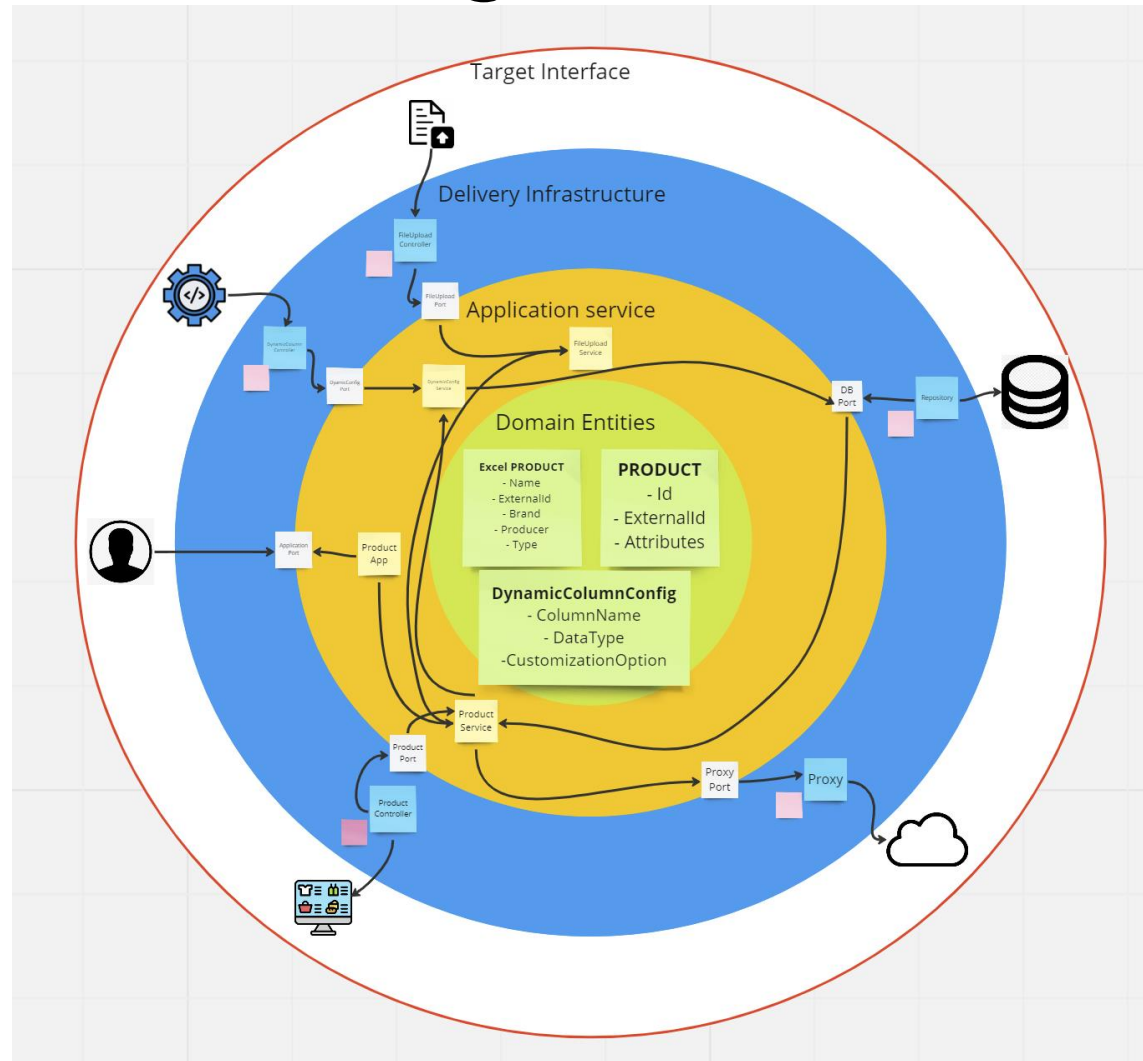
Clean/Onion/Hexagonal Architecture



Clean/Onion/Hexagonal Architecture



Clean/Onion/Hexagonal Architecture



Choosing the Right Approach

- Select the appropriate approach based on project context
 - **Nature** of the project, team **expertise**, and available **resources**.
 - Level of certainty in **requirements** and system **architecture**.
- Adaptable and flexible.



Conclusion

- Inside-Out approach for [clean core](#) domain and reusability
- Outside-In approach for [user-centered](#) design and customization
- Combine both approaches for a [comprehensive architecture](#)
- Understand project requirements when choosing an approach.

Reference

- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). "Design Patterns: Elements of Reusable Object-Oriented Software." Addison-Wesley Professional.
 - Offers design patterns for building modular, maintainable, and customizable software solutions.
- Patel, S. (2020). "Maintainability vs. Customizability: Finding the Right Balance in Software Development." [Online Article].
 - Explores the trade-off between maintainability and customizability in software development projects.
- Johnson, L. (2019). "Inside-Out vs. Outside-In Development: Choosing the Right Approach." [Online Article].
 - Provides insights into the characteristics and benefits of both Inside-Out and Outside-In approaches.
- Myers, J. (2022). "Combining Inside-Out and Outside-In Approaches for Robust Software Solutions."
 - Presents a case study on how a combined approach can create adaptable and user-centric software solutions.
- Smith, J. (2018). "Choosing the Right Software Development Approach." [Online Article].
 - Discusses factors to consider when selecting a software development approach.



THANK YOU



Indumathi Chinnaswamy



92541033



indumathi.chinnaswamy@bouvet.no