Putting It All Together

Mel Grubb
Developer

@melgrubb | www.melgrubb.com

Overview

Four Pillars

- Encapsulation of data
- Abstraction of behavior
- Inheritance
- Polymorphism

SOLID

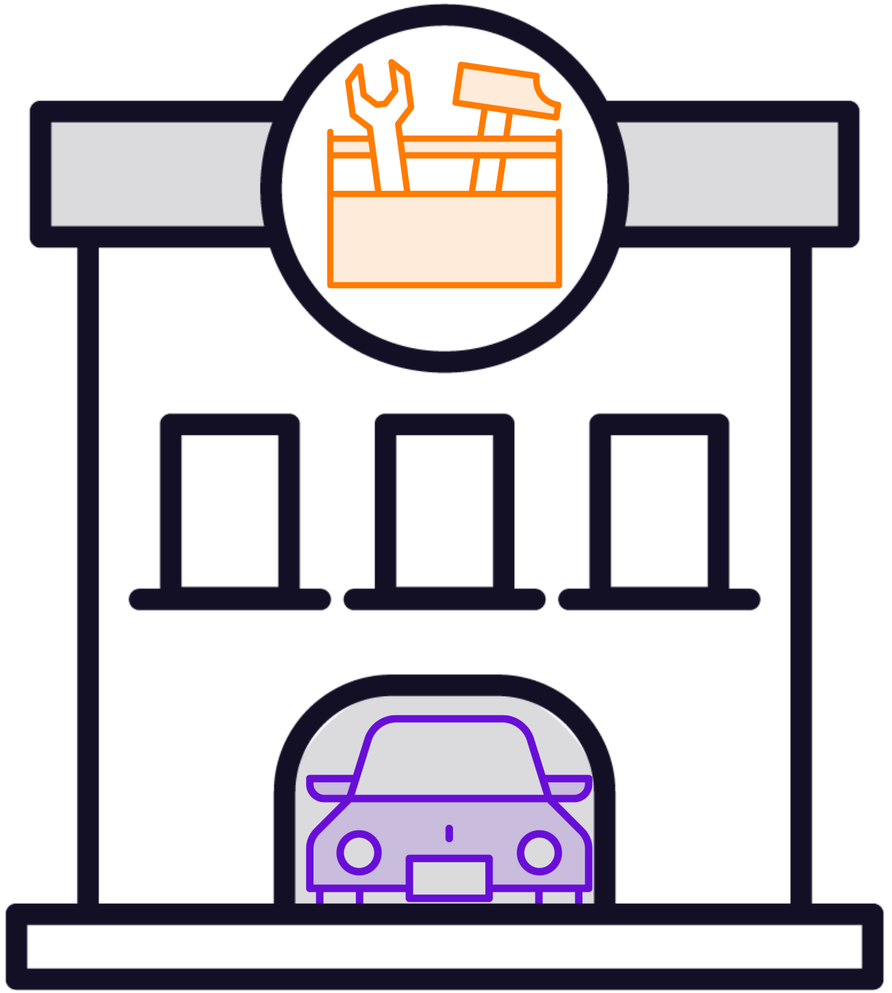
Design Patterns

See Also

Domain-driven Design Fundamentals

Julie Lerman & Steve Smith

Example: Auto Repair Quotation System



Use Case

“As a repair shop owner, I would like an automated system to generate quotes for vehicle repair orders so that I can provide accurate cost estimates for my customers.”

Out of Scope

Part inventory

Suppliers

Vehicle-specific parts

Labor categories

Cost snapshots

Design Tips

- Talk out the design
- Try to explain it from scratch
- Focus on areas of uncertainty
- Update the design to address those areas
- Repeat as needed

Implementing the Domain

Data Access

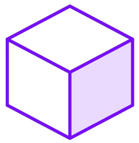
Business Logic

See Also

Entity Framework Core Path

POCO Objects

“Plain Old CLR Object”



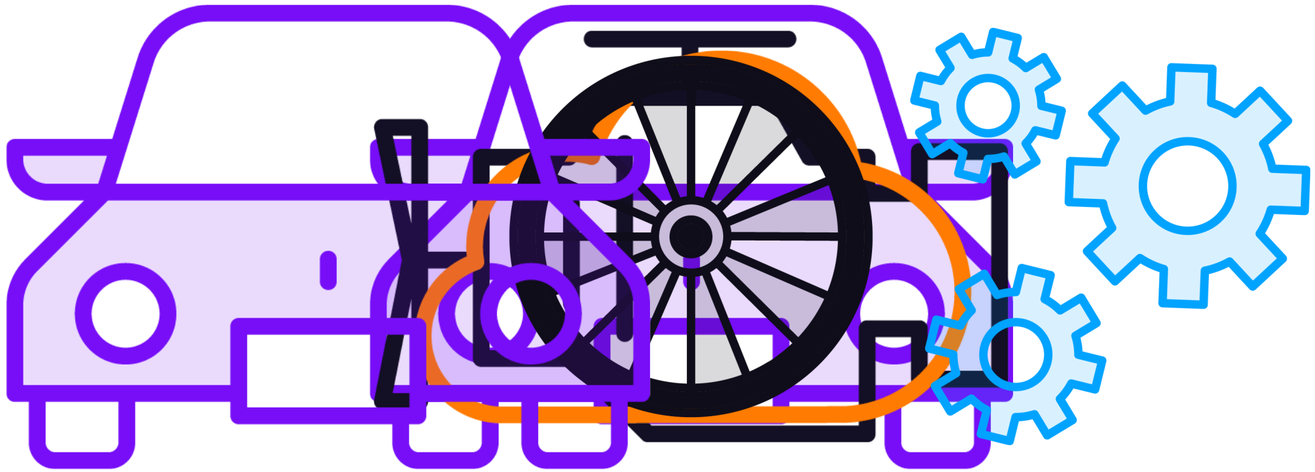
Minimal functionality

Aware of connections to other objects

Persistence agnostic

Works well with ORMs

Warranty Types



Summary

Summary

Four Pillars

- Encapsulation
- Abstraction
- Inheritance
- Polymorphism

SOLID

- Single-responsibility
- Open/Closed
- Liskov Substitution
- Interface Segregation
- Dependency Inversion

Summary

Nullability

Patterns

- Strategy
- Singleton

Equality

Immutability

Records

Options pattern

Thanks for Watching