glossary

Glossary of Terms

This glossary defines key concepts and terminology used in the sample solution, focusing on Clean Architecture, Domain-Driven Design (DDD), and CQRS patterns.

Aggregate

A cluster of domain objects that can be treated as a single unit. The aggregate root enforces business invariants and manages the lifecycle of its entities.

Entity

An object with a unique identity that persists over time. Entities are managed by aggregate roots and are distinguished by their identity, not just their attributes.

Value Object

An immutable type that represents a descriptive aspect of the domain. Value objects implement value equality and are used for concepts like addresses, money, or strongly-typed IDs.

Domain Event

A message that represents a significant occurrence within the domain, such as an order being placed or a shipping address being changed.

Strongly-Typed ID

A value object used to represent identifiers (e.g., `OrderId`, `CustomerId`) instead of using primitive types, improving type safety and expressiveness.

Use Case

A business operation or workflow, implemented in the Application layer, that orchestrates domain logic and coordinates interactions between components.

Command

A request to perform an action that changes the state of the system (e.g., `PlaceOrderCommand`). Part of the CQRS pattern.

Query

A request to retrieve data without modifying state (e.g., `GetOrderByIdQuery`). Part of the CQRS pattern.

Port

An interface in the Application layer that defines a contract for infrastructure dependencies, such as repositories or external services.

Adapter

An implementation of a port, typically found in the Infrastructure layer, that connects the application to external systems or databases.

Minimal API

A concise way to define HTTP endpoints in ASP.NET, mapping requests directly to application commands/queries with minimal ceremony.

Dependency Injection (DI)

A design pattern for providing dependencies to classes via constructor or method parameters, promoting loose coupling and testability.

Conventional Commit

A standardized format for commit messages that improves traceability and automation in version control workflows.

Refer to this glossary for clarification of terms used throughout the architecture and design documents.