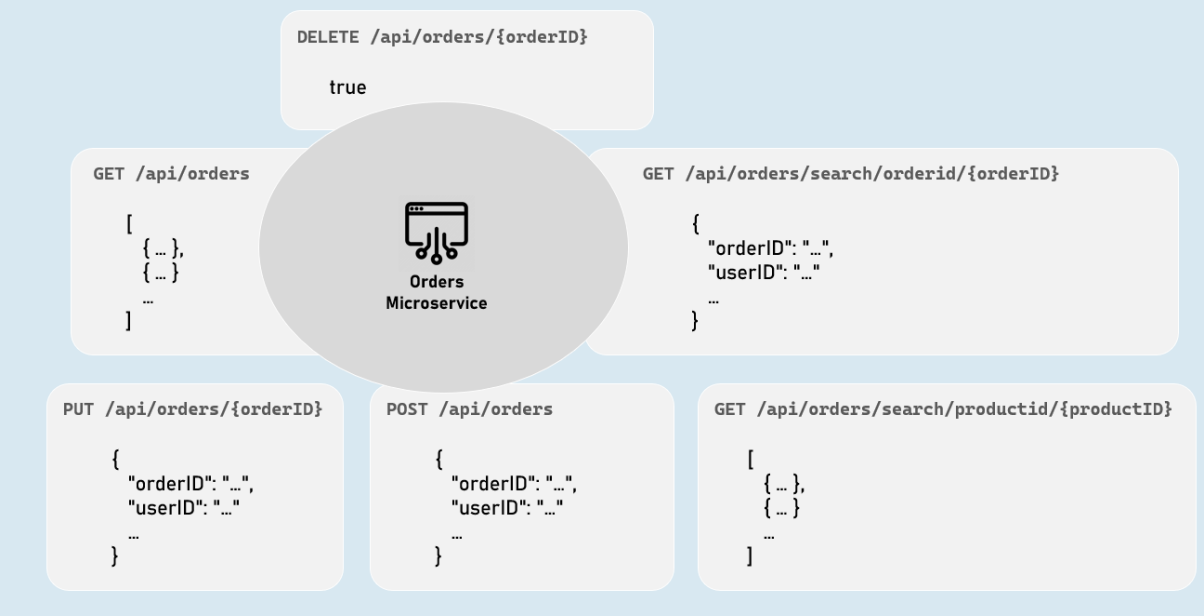
.NET Core Microservices – True Ultimate Guide

Section 6 – Orders Microservice – Cheat Sheet

Orders Microservice Endpoints



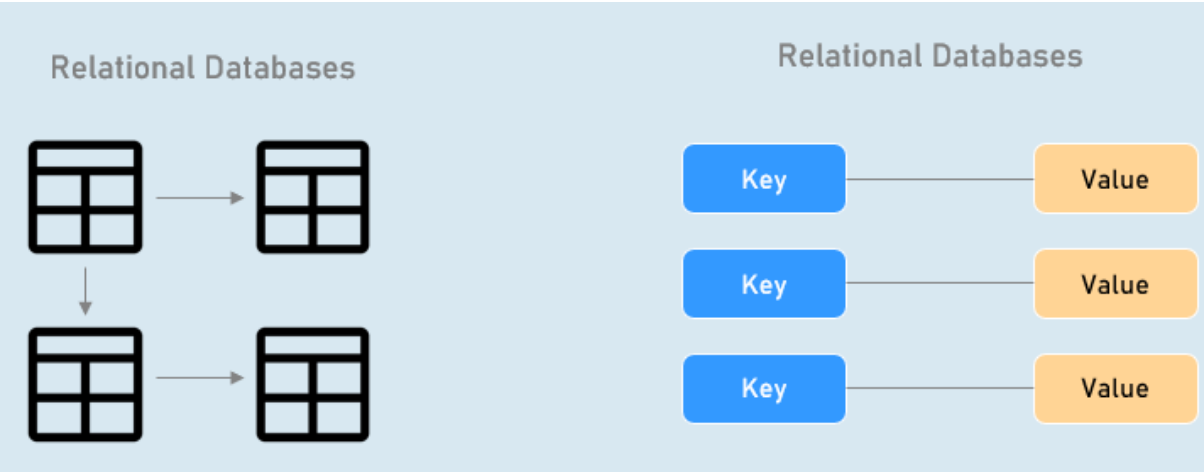
Orders Microservice

Layered Architecture

- Presentation (WebAPI Controller Endpoints)
- Business Logic Layer (BLL)
- Repository Layer (RL)
- Data Access Layer (DAL) / MongoDB Driver
- Database (MongoDB)

NoSQL Databases

NoSQL (Non-SQL or Not-Only-SQL) databases, are a category of databases that provide mechanisms for store data as JSON documents, rather than tabular relations used in relational databases.



When to use NoSQL Databases?

Flexible Data Models: For scenarios where data structures are dynamic or semi-structured and require diverse data types like JSON, XML, key-value pairs; or a fixed schema is not applicable.

High Performance Needs: In applications requiring low-latency responses.

Scalability: When the application demands horizontal scaling to handle large volumes of data and high traffic.

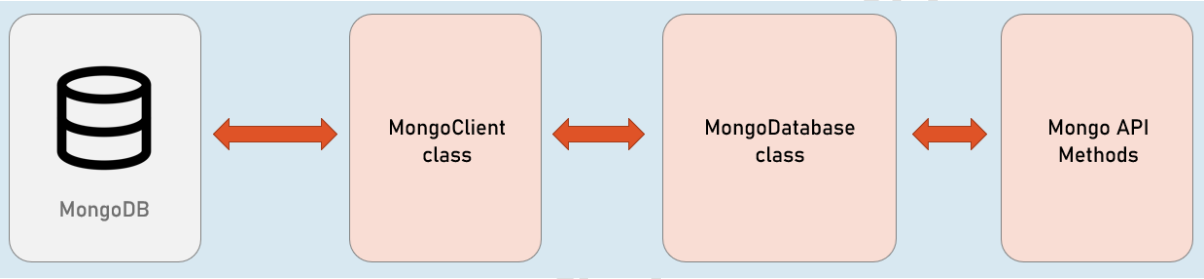
Cloud-Native and Distributed Environments: NoSQL databases are well-suited for cloud deployments and distributed systems due to their inherent scalability and fault tolerance.

Cost-Effectiveness: NoSQL databases can be more cost-effective for certain use cases, especially in cloud environments, due to their pay-as-you-go pricing models.

Examples of NoSQL Databases

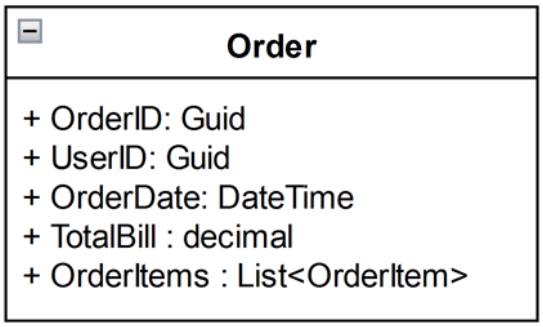
- MongoDB
- Redis
- Cassandra
- Amazon DynamoDB
- Cosmos DB
- Couchbase

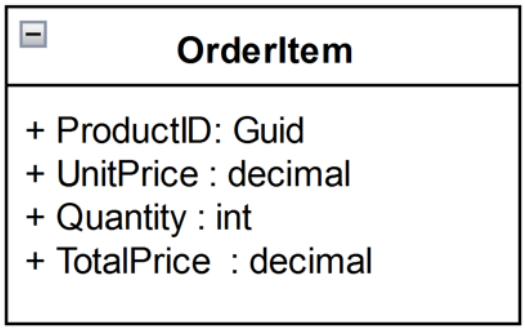
MongoDB



Order Models

Entity models of "Orders Microservice"





Orders Repository

```
IOrdersRepository
+ GetOrders() : Task<IEnumerable<Order>>
+ GetOrdersByCondition(FilterDefinition<Order> filter) : Task<IEnumerable<Order?>>
+ GetOrderByCondition(FilterDefinition<Order> filter) : Task<Order?>
+ AddOrder(Order order) : Task<Order?>
+ UpdateOrder(Order order) : Task<Order?>
+ DeleteOrder(Guid orderID) : Task<bool>
```

Orders DTO

```
OrderAddRequest
+ UserID: Guid
+ OrderDate: DateTime
+ OrderItems : List<OrderItemAddRequest>
```

```
OrderItemAddRequest
+ ProductID: Guid
+ UnitPrice : decimal
+ Quantity : int
```

```
OrderUpdateRequest
+ OrderID: Guid
+ UserID: Guid
+ OrderDate: DateTime
+ OrderItems : List<OrderItemUpdateRequest>
```

```
OrderItemUpdateRequest
+ ProductID: Guid
+ UnitPrice : decimal
+ Quantity : int
```

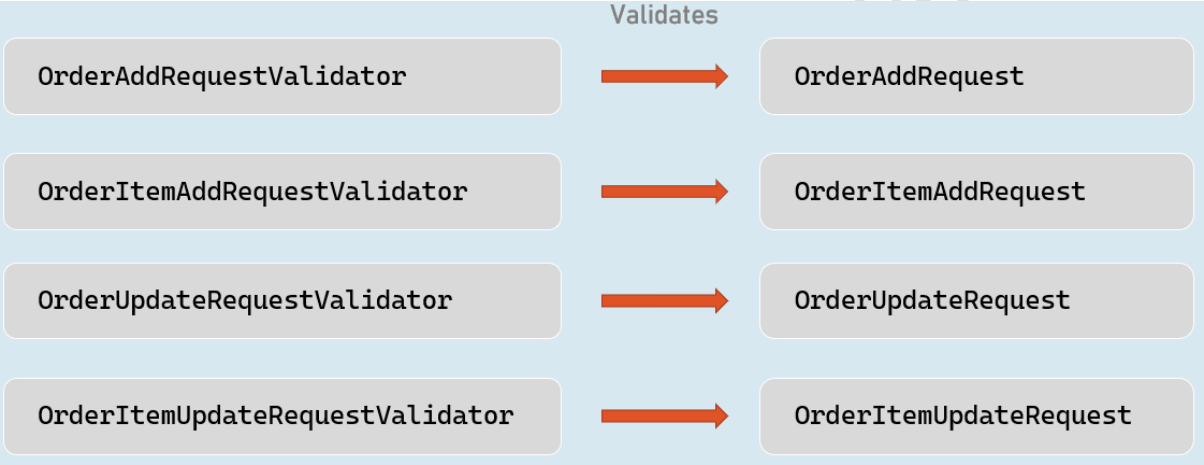
```
OrderResponse
+ OrderID : Guid
+ UserID : Guid
+ OrderDate : DateTime
+ TotalBill : decimal
+ OrderItems : List<OrderItemResponse>
```

```
OrderItemResponse
+ ProductID : Guid
+ UnitPrice : decimal
+ Quantity : int
+ TotalPrice : decimal
```

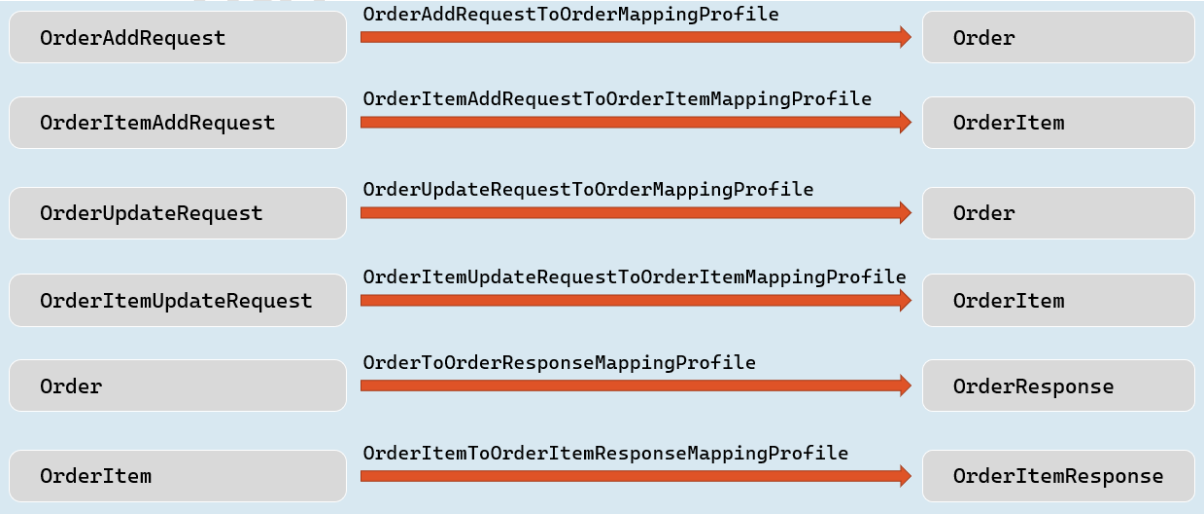
Orders Service

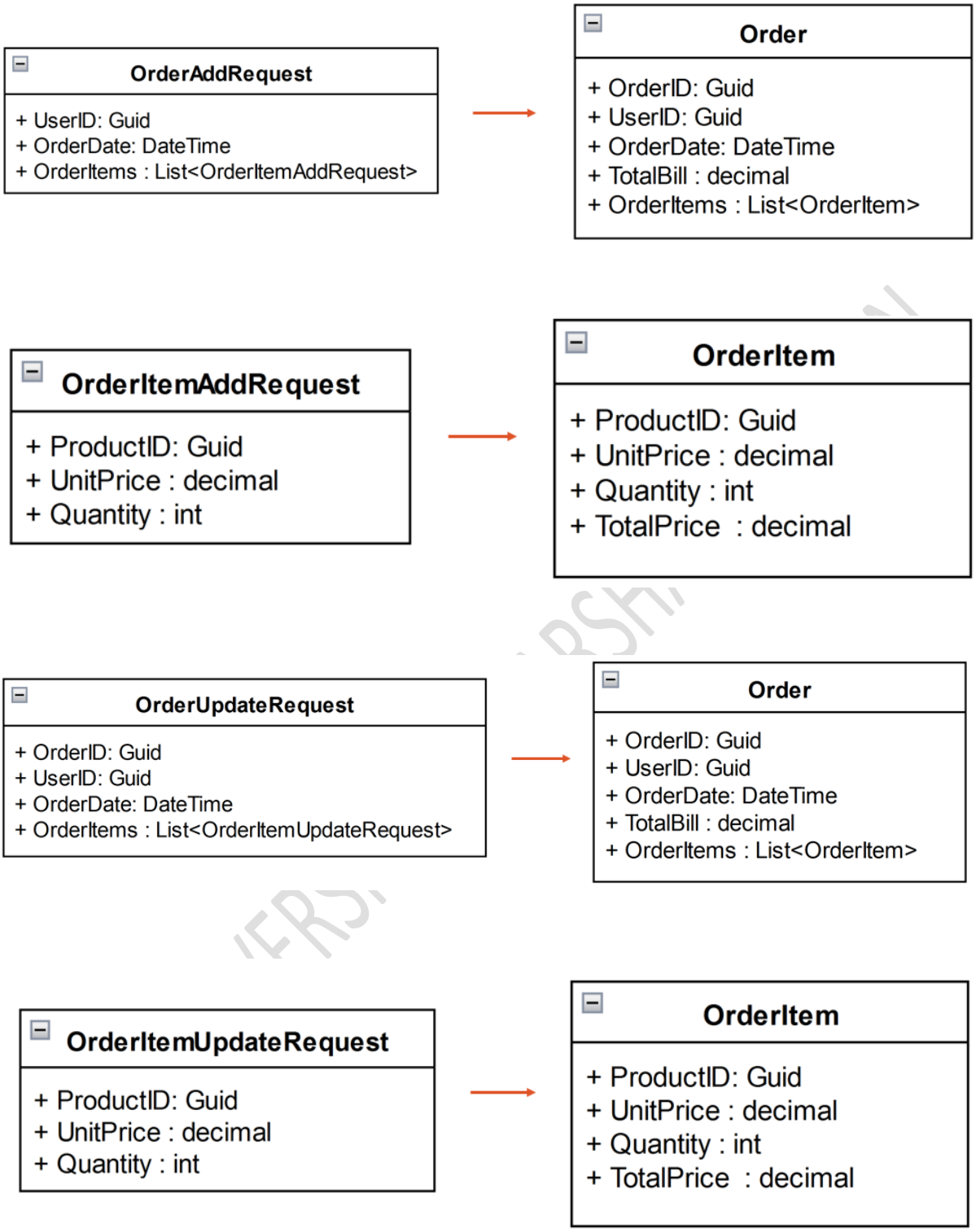
```
IOrdersService
+ GetOrders() : Task<List<OrderResponse?>>
+ GetOrdersByCondition(FilterDefinition<Order> filter) : Task<List<OrderResponse?>>
+ GetOrderByCondition(FilterDefinition<Order> filter) : Task<OrderResponse?>
+ AddOrder(OrderAddRequest orderAddRequest) : Task<OrderResponse?>
+ UpdateOrder(OrderUpdateRequest orderUpdateRequest) : Task<OrderResponse?>
+ DeleteOrder(Guid orderID) : Task<bool>
```

Order Validator



Order Mappers




OrderAddRequest
+ UserID: Guid
+ OrderDate: DateTime
+ OrderItems : List<OrderItemAddRequest>
Order
+ OrderID: Guid
+ UserID: Guid
+ OrderDate: DateTime
+ TotalBill : decimal
+ OrderItems : List<OrderItem>
OrderItemAddRequest
+ ProductID: Guid
+ UnitPrice : decimal
+ Quantity : int
OrderItem
+ ProductID: Guid
+ UnitPrice : decimal
+ Quantity : int
+ TotalPrice : decimal
OrderUpdateRequest
+ OrderID: Guid
+ UserID: Guid
+ OrderDate: DateTime
+ OrderItems : List<OrderItemUpdateRequest>
Order
+ OrderID: Guid
+ UserID: Guid
+ OrderDate: DateTime
+ TotalBill : decimal
+ OrderItems : List<OrderItem>
OrderItemUpdateRequest
+ ProductID: Guid
+ UnitPrice : decimal
+ Quantity : int
OrderItem
+ ProductID: Guid
+ UnitPrice : decimal
+ Quantity : int
+ TotalPrice : decimal