The **My Account Balance API** is based on my previous project of Event Sourcing on GitHub called ivanpaulovich/jambo (please checkout).

Architectural Styles used

- CQRS
- Domain-Driven Design
- Aggregates + Event Sourcing
- Optimistic Concurrency

Data

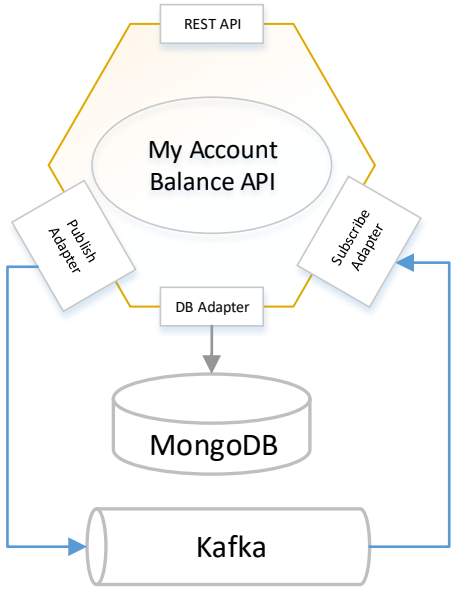
- Kafka stream as the source of true
- MongoDB as Event Stream projection
- Avoids Single Point of Failura
- Distribuited Data

Authentication

- Bearer Authentication

Principles

- SOLID
- Microservices
  - Independent
  - Autonomous
  - Modelled aroung business Domain



Application architecture based on the IDDD (Chapter 4 of Vaughn Vernon) that explains Hexagonal Application Architecture and based on Events-Driven Architecture.

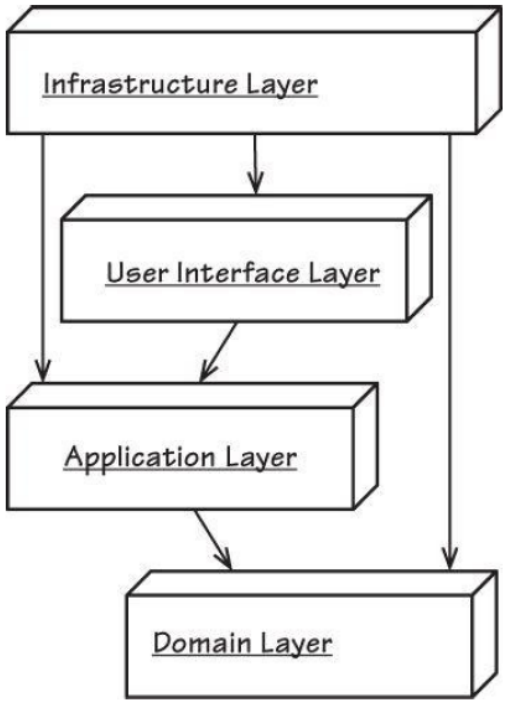


Figure 4.3. The possible Layers when the Dependency Inversion Principle is used. We move the Infrastructure Layer above all others, enabling it to implement interfaces for all Layers below.

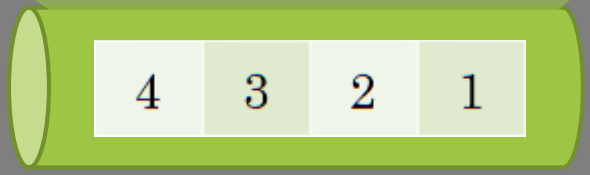
Frameworks used

- WebAPI
- Autofac
- MediatR
- MongoDB.Driver
- Dapper
- Confluent.Kafka
- SwashBuckle
- Bearer Authentication

Customer Accounts Transactions

1	Deposited	C/C: 4030-1 Data: 17/08/2017 Valor: 500,00€
2	Deposited	C/C: 2010-0 Data: 20/08/2017 Valor: 300,00€
3	Withdraw	C/C: 2010-0 Data: 23/08/2017 Valor: 100,00€
4	Withdraw	C/C: 4030-1 Data: 25/08/2017 Valor: 200,00€

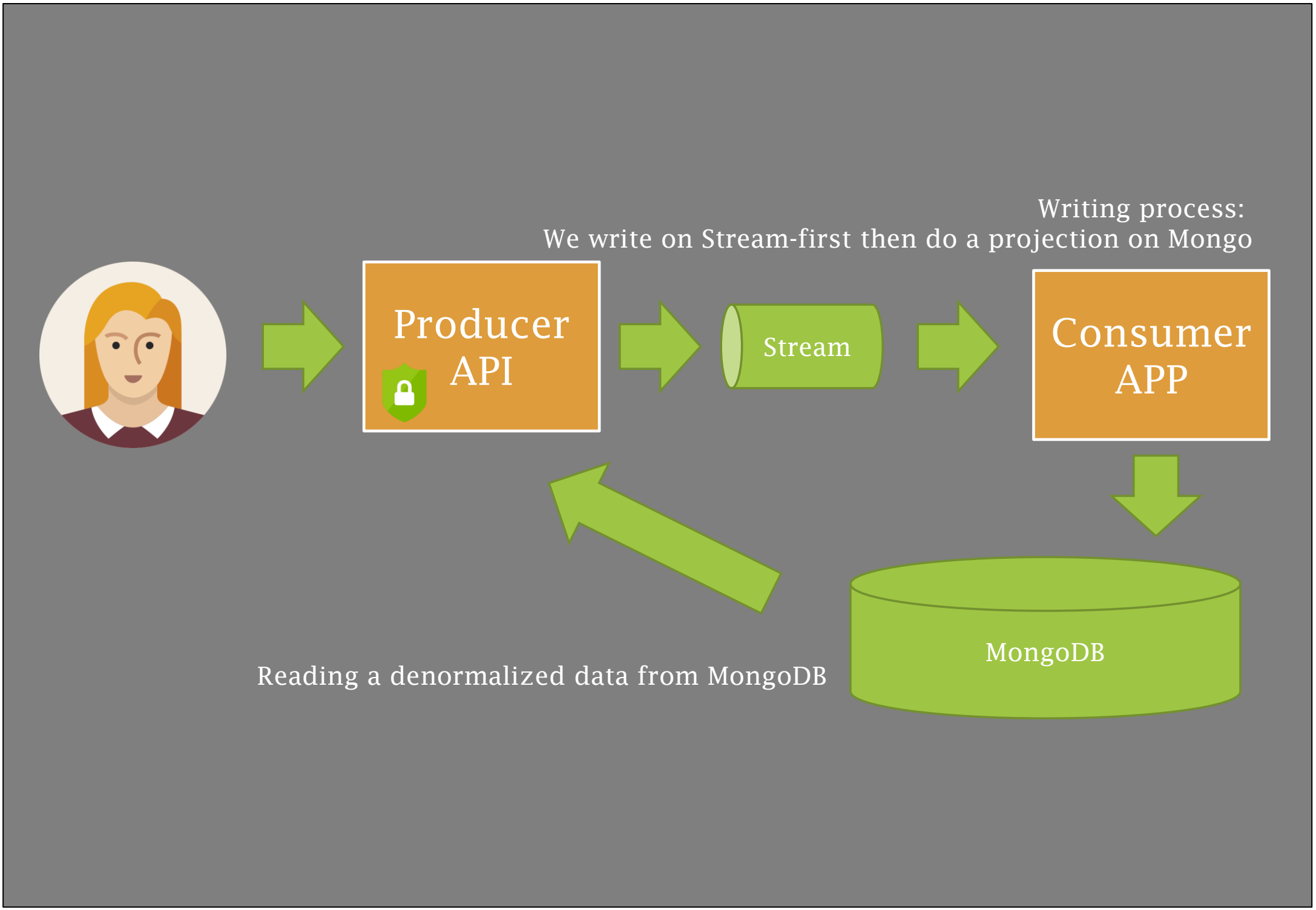
Publish



Projection

Customer Accounts

C/C	Balance
2010-0	200,00€
4030-1	300,00€


Writing process:
We write on Stream-first then do a projection on Mongo
Producer API
Stream
Consumer APP
MongoDB
Reading a denormalized data from MongoDB