Fluxo de Caixa – Implementação

# Indice

Sumário

[Indice 1](#_Toc172411177)

[1 – Breve descrição 2](#_Toc172411178)

[2 – Serviços 2](#_Toc172411179)

[3 – Repositórios de Código 2](#_Toc172411180)

[4 - Pacotes Externos 3](#_Toc172411181)

[5 – Setup Banco 3](#_Toc172411182)

[6 – Instalação 3](#_Toc172411183)

[6.1 – Uso do build 3](#_Toc172411184)

[6.2 – Uso do build 4](#_Toc172411185)

[6.3 – Rodar o Projeto 4](#_Toc172411186)

[7 - Diagrama de Banco (básico) 4](#_Toc172411187)

# 1 – Breve descrição

O sistema foi desenvolvido usando .Net Core, com implementações de design paterns:]

* DDD
* Alguns conceitos Solid
* Repository
* ValidationResult
* 0(ZERO) Procedure
* Injeção de dependência
* Mensageria
* Hosted Workers

O serviços de apis/workers está on line disponibilizado e publicados nos endereços:

<https://fc-lancamento.dietcode.com.br/swagger/index.html>

<https://fc-consolidado.dietcode.com.br/swagger/index.html>

<https://fc-worker.dietcode.com.br/swagger/index.html>

Tem implementado, fake a camada de autenticação. Se fosse totalmente implementada seria usado certificado em banco para criptografar e decriptografar os dados em um campo binário na tabela e usando JWT.

O serviço de Ping é apenas um Ping, se houver resposta positiva o serviço está on, se der erro o site todo está fora.

# 2 – Serviços

Serviço de Lançamento => Service Rest onde é feito os lançamentos de Credito ou débito via end point. Após o lançamento é enviado uma mensagem em fila no rabbit mq.

Serviço Worker => Este fica em escuta, quando cai uma mensagem no sistema ele puxa e registra a mensagem.

Serviço Consolidado => Serviço de consulta de dados consolidados, que podem ser do dia, do mês, de hoje para sete dias atrás, um mês específico, ou faixa de datas, ou todos os lançamentos.

# 3 – Repositórios de Código

No Repositório Git. Subi o projeto principal, e lá tem também a minha biblioteca de apoio para uso em sistemas.

O repositório deste projeto está em:

<https://github.com/Humberfrench/FluxoDeCaixa>

O repositório dos pacotes nugget está em:

<https://github.com/Humberfrench/DietTools> (Dietcode é o nome fantasia da minha empresa e marca)

Existe outros repositórios alguns 100% implementados e outros foram estudos que demonstram a referência e o conhecimento de DDD. O meu conhecimento foi obtido através do Eduardo Pires, um colega de trabalho disponibilizou ao entrarmos na empresa os vídeos dele falando disso para entrarmos no conhecimento e aplicar o uso da metodologia para isso

**Jack** – Para controle de assistência social e distribuição de sacolinhas de Natal, usando DDD, NHibernate, Net Framework, Ajax com Javascript.

**Reddington** – Para gerenciamento de escolinha de moral cristã a alunos da assistência social, usando DDD, NHibernate, Net Framework, Ajax com Javascript.

**Jim Halpert** – Incompleto. Api em Net Core para munir o sistema front de estudo usando Angular (não finalizado, era estudo)

# 4 - Pacotes Externos

No repositório Git será encontrado a pasta Nuget Packages. Lá contém os pacotes publicados:

* Dietcode Database – É o pacote de repositório que usa Dapper Contrib para CRUD de dados
* Dietcode Lib – Lib de Helper.

Para instalar os pacotes basta referenciar a pasta dele no config de source do nugget no Visual Studio. Se não encontrar será preciso exportar o pacote com um comando nugget.

dotnet nuget push <caminho do arquivo >\nome.nupkg -s <caminho destino>

É preciso usar o command prompt como administrador e ir até a pasta que tem o nuget.exe que está dentro das pastas do visual Studio.

Outros pacotes são públicos pela Microsoft e parceiros.

# 5 – Setup Banco

No repositório existe uma pasta SQL. Dentro dela tem uma cópia de backup do banco. E em outra pasta os scripts de criação do banco, duas tabelas e duas procedures.

O projeto se rodar vai apontar para este banco: Server=dietcode.com.br\\Dev. Ele está em minha casa em um servidor meu interno. É um servidor genérico de dev. Não tem problemas o uso e abuso de inserir e mexer nos dados.

# 6 – Instalação

## 6.1 – Uso do build

Criar o banco no servidor ou local. Pode ser via restore do banco (.bak) que está disponibilizado junto do repositório Git. OU rodar os scripts na ordem:

1. Criar banco
2. Rodar scripts para criar tabela de tipo de lançamento
3. Rodar scripts para criar tabela de Lançamento
4. Rodar scripts para criar aos usuários

Após isso alterar a string de conexão para conectar-se ao banco. É preciso dar permissões ao usuário que vai ser usado para conectar a essa base. Poderia usar o worker, usuário criado junto do banco.

Antes de tudo criar uma pasta e criar um site no iis, com DNS, ou na pasta criada adicionara aplicação/diretório virtual para que possa colocar o app publicado.

A mais simples é no repositório GIT clonar o projeto e na pasta publish tem a última versão publicada já, é só implementar no iis. E chamar. POIS está configurado para um banco online e vai rodar.

## 6.2 – Rabbit

Caso não haja um server local, criar as filas em um server genérico ou instale um local. Ou usar a estrutura que eu tenho on line em meu server, e estes dados de apontamento estão no código do worker e do serviço de lançamento.

Será preciso criar uma queue: Lancamentos do tipo Durable

Uma Exchange tipo topic: Lancamentos-created. Vincular a queue Lancamentos nesta Exchange.

## 6.3 – Rodar o Projeto

Antes de tudo é preciso ajustar o projeto e o visual studio para ler os pacotes.

Se faz preciso criar um apontamento nas configurações do projeto para o caminho

https://nuget.dietcode.com.br/nuget

Assim com o caminho ajustado o pacote poderá ser instalado. Rodar o projeto.

# 7 - Diagrama de Banco (básico)

O banco contém duas tabelas:

Diagrama

Descrição gerada automaticamente

A de lançamento que contêm todos os lançamentos uma tabela auxiliar que tem os cadastros de tipos de lançamentos.