

**UNIVERSIDADE PAULISTA – UNIP**

Gabrielle de Lima Gabaldi **N523BJ1**

Kaique Marcelino Gomes **F1112H3**

Paola Regina Garbato **N528DG3**

Rafael Carneiro da Silva **N5065D6**

**PROJETO INTEGRADO MULTIDISCIPLINAR IV:**

Sistema de Controle Operacional para a Empresa BlockTech

**SÃO PAULO**

**2020**

Gabrielle de Lima Gabaldi **N523BJ1**

Kaique Marcelino Gomes **F1112H3**

Paola Regina Garbato **N528DG3**

Rafael Carneiro da Silva **N5065D6**

**PROJETO INTEGRADO MULTIDISCIPLINAR IV:**

Sistema de Controle Operacional Para a Empresa BlockTech

Projeto Integrado Multidisciplinar para a avaliação semestral do curso de Gestão Tecnológica em Análise e Desenvolvimento de Sistemas apresentado à Universidade Paulista - UNIP

Orientador: Prof. Emerson Abraham

**SÃO PAULO**

**2020**

Gabrielle de Lima Gabaldi **N523BJ1**

Kaique Marcelino Gomes **F1112H3**

Paola Regina Garbato **N528DG3**

Rafael Carneiro da Silva **N5065D6**

**PROJETO INTEGRADO MULTIDISCIPLINAR IV:**

Sistema de Controle Operacional Para a Empresa BlockTech

Projeto Integrado Multidisciplinar para a avaliação semestral no curso de Gestão Tecnológica em Análise e Desenvolvimento de Sistemas apresentado à Universidade Paulista - UNIP

Aprovado em:

BANCA EXAMINADORA

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_/\_\_\_\_**Prof. Emerson Abraham  
Universidade Paulista - UNIP

**DEDICATÓRIA**

Dedicamos esse trabalho aos alunos e professores, que se mantiveram determinados, apesar de todos os obstáculos e as dificuldades desse semestre, devido a situação atual.

*"Encontra ânimo na dor e no desafio. Nesta vida só nos são colocados à frente os obstáculos que somos capazes de ultrapassar."*

*Augusto Branco*

**RESUMO**

A empresa Innovate and Renew Technology oferece serviços baseados na tecnologia blockchain, oferecendo serviços para o investimento em criptomoedas. Para uma melhor experiência de seus usuários, contratou a BlockTech para que fosse desenvolvido um sistema de gerenciamento desses investimentos. Tal sistema deve contemplar um sistema Desktop, e um sistema Web, sendo que, o Desktop deve ser acessível apenas pela empresa, e o Web, tanto pela empresa, quanto para os usuários. O sistema dará uma melhor visão de seus investimentos, tendo em vista que terão uma maior visibilidade através de gráficos semanais, poderão acompanhar as criptomoedas em gráficos diários, semanais e mensais, além de acompanhar seu lucro ou perda, diariamente.

Palavras-chave: Blockchain. Criptomoedas.

**ABSTRACT**

Keywords:

**LISTAS DE ILUSTRAÇÕES**

**LISTA DE TABELAS**

**LISTA DE ABREVIATURAS E SIGLAS**

**SUMÁRIO**

# INTRODUÇÃO

# SITUAÇÃO PROBLEMA

A empresa Innovate and Renew Technology possui um sistema para investimentos em Criptomoedas, porém, não possuem um para a análise dos mesmos. Os usuários precisam ter seu próprio controle, tendo em vista que o sistema atual que utilizam só informa o valor em carteira que possuem, não sendo possível visualizar o valor das criptomoedas dos últimos dias, ou sequer o lucro que tiveram (ou perda). Com o novo sistema, tanto os clientes, quanto a empresa podem acompanhar seus investimentos, ou até mesmo, ver seu histórico de transações.

# ENGENHARIA DE SOFTWARE

## Requisitos não funcionais

Tabela 1 - Requisitos Não Funcionais

| **RNF** | **CÓD.** | **PRIORIDADE** | **GRAU DE DIFICULDADE** |
| --- | --- | --- | --- |
| O aplicativo web deve ser responsivo | 001 | Alta | Baixa - 2 |
| O sistema deve criptografar a senha dos usuários | 002 | Alta | Alta - 14 |
| O sistema deve ser entregue em 3 meses. | 003 | Alta | Alta - 15 |
| O aplicativo web deve utilizar a plataforma ASP.NET Core | 004 | Alta | Alta - 14 |
| O aplicativo desktop deve utilizar a plataforma Windows Forms | 005 | Alta | Média - 8 |

Fonte: Paola Regina Garbato (2020)

## Requisitos Funcionais – Aplicativo Web

**Tabela 2 - Requisitos Funcionais - Aplicativo Web**

| **RF** | **CÓD.** | **PRIORIDADE** | **GRAU DE DIFICULDADE** | **REGRA DE NEGÓCIO** |
| --- | --- | --- | --- | --- |
| O sistema deve ser acessível pelos usuários e o Administrador. | 001 | Alta | Baixa - 2 | O administrador possui uma única contra pré-definida |
| O sistema deve permitir cadastro de clientes. | 002 | Alta | Baixa - 3 | O cliente só poderá se cadastrar ao inserir todos seus dados |
| O sistema deve permitir a realização do login com a inserção do nome de usuário e senha definidos no cadastro. | 003 | Alta | Baixa - 3 |  |
| O sistema deve permitir o cadastro de novas criptomoedas. | 004 | Alta | Baixa - 3 | Acessível apenas pelo Administrador |
| O sistema deve permitir alteração e exclusão das criptomoedas do sistema. | 005 | Alta | Baixa - 4 | Acessível apenas pelo Administrador |
| O sistema deve permitir a busca das criptomoedas por seu nome. | 006 | Baixa | Média - 7 |  |
| O sistema deve permitir adicionar o valor das criptomoedas todos os dias, de acordo com as criptomoedas cadastradas. | 007 | Alta | Média - 8 | O valor das criptomoedas deve ser alterados uma vez no dia. Acessível apenas pelo Administrador. |
| O sistema deve permitir busca das criptomoedas por data. | 008 | Alta | Média – 8 | Deve ser exibido o nome, símbolo e data das criptomoedas. |
| O sistema deve permitir a alteração do valor da criptomoeda, a alteração e deleção da mesma. | 009 | Alta | Média - 7 | Acessível apenas pelo Administrador. |
| O sistema deve permitir visualizar todo histórico de transações realizadas. | 010 | Alta | Alta - 13 | O usuário pode visualizar suas transações. Já o Administrador, visualiza as transações de todos os usuários. |
| O sistema deve exibir o tipo de transação – Compra/Venda. Além de ordená-las por data. O administrador poderá filtrar a lista pelo nome do usuário, ou pelo nome da criptomoeda. O usuário poderá filtrar a busca pelo nome da criptomoeda. | 011 | Alta | Alta - 12 | Devem ser exibidas todas as informações da transação: data, nome do usuário, tipo de transação, valor, nome da criptomoeda e sua quantidade. |
| O sistema deve exibir informações sobre a conta do cliente, além de sua conta vinculada. | 012 | Alta | Baixa - 6 | O saldo total do cliente deve ser exibido. O valor deve ser baseado no valor atual das criptomoedas cujo usuário tenha investimento. |
| O sistema deve exibir uma lista de todas as aplicações do usuário. | 013 | Alta | Média - 9 | O Administrador deve visualizar uma lista das aplicações de todos os usuários. |
| O sistema deve conter um gráfico com o valor atual de todas as criptomoedas. | 014 | Alta | Alta – 14 |  |
| O sistema deve conter um gráfico para cada criptomoeda com o valor dos seus últimos 7 dias + o dia atual. | 015 | Alta | Alta – 14 |  |
| O sistema deve conter um gráfico para cada criptomoeda com o valor dos seus últimos 30 dias + o dia atual. | 016 | Alta | Alta - 14 |  |
| O sistema deve conter um gráfico com o valor de todas as criptomoedas nos últimos 7 dias + o dia atual. | 017 | Alta | Alta - 14 |  |
| O sistema deve conter um gráfico com o valor de todas as criptomoedas nos últimos 30 dias + o dia atual. | 018 | Alta | Alta - 14 |  |
| O sistema deve conter um gráfico dos últimos 7 dias + dia atual, indicando a porcentagem do valor da criptomoeda em relação ao dia anterior. | 019 | Alta | Alta - 16 |  |
| O sistema deve conter um gráfico dos últimos 30 dias + dia atual, indicando a porcentagem do valor da criptomoeda em relação ao dia anterior. | 020 | Alta | Alta - 16 |  |
| O sistema deve exibir todos os investimentos do usuário, indicando o valor investido e o valor atual. Além de gráficos semanais e mensais indicando seu investimento. | 021 | Alta | Alta – 16 | Para o administrador, deve ser exibido o valor total investido nas criptomoedas pelos seus clientes. |
| O sistema deve permitir a alteração do telefone, e-mail e senha do usuário. | 022 | Alta | Média – 8 | Para alterar a senha, é necessário inserir a senha atual. |

Fonte: Paola Regina Garbato (2020)

# ANÁLISE DE SISTEMAS ORIENTADA A OBJETOS

## Diagrama de Casos de uso

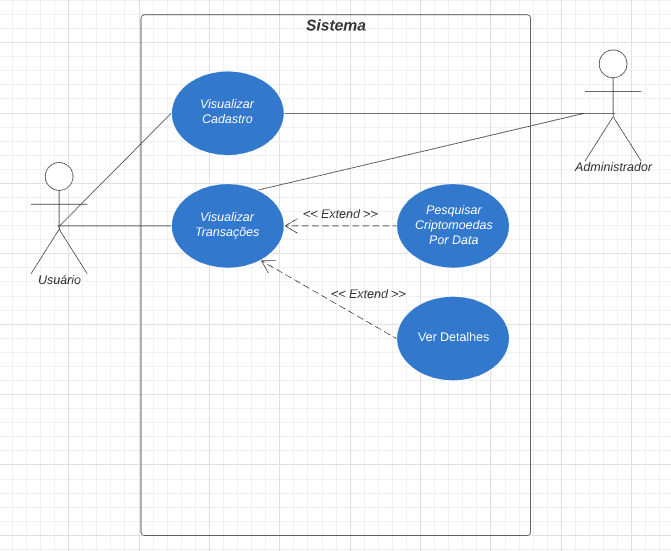
O Diagrama de Caso de Uso descreve as funcionalidades do sistema, como o sistema age conforme os requisitos solicitados pelo usuário. Para a montagem do Diagrama de caso de uso, definimos:

* O Cenário: Como o sistema deve interagir com o usuário
* O Ator: Usuário e Administrador
* Use Case: Conforme solicitado pelo Usuário
* A Comunicação: Tudo que liga o ator com o caso de uso

E em seguida, as funcionalidades do sistema:

* Visualizar cadastro
* Visualizar criptomoedas
* Visualizar transações
* Alterar criptomoeda
* Deletar criptomoeda
* Adicionar criptomoeda
* Pesquisar criptomoedas por data
* Ver detalhes

Figura 1 - Diagrama de Casos de Uso - Visualizar Cadastro e Visualizar Transações

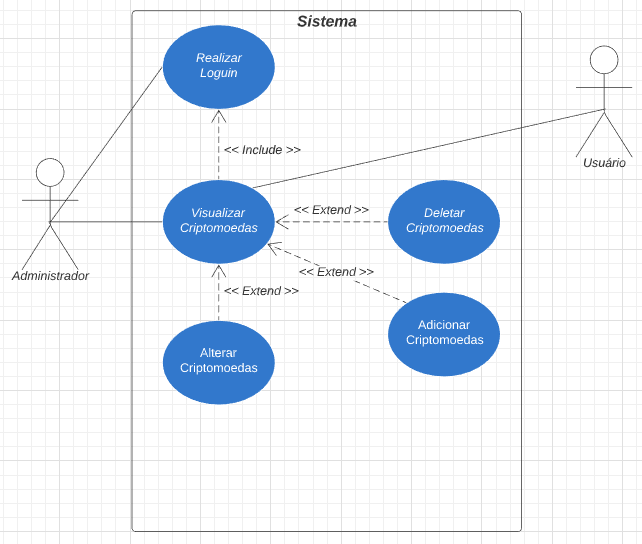


Fonte: Gabrielle de Lima Gabaldi (2020)

Como podemos ver na figura 1, o usuário, e o administrador podem visualizar os cadastros, e as transações. O <<extend>> indica que os itens “Pesquisar Criptomoedas por Data”, e “Ver Detalhes”, são opcionais.

Podemos observar na figura 2, que o Administrador deve realizar o login para ter acesso a visualização das Criptomoedas, indicado pelo <<include>>. E pode (opcionalmente) alterar, deletar, e adicionar criptomoedas. Para isso, o administrador deve selecionar a criptomoeda, para assim realizar as devidas alterações na mesma.

Figura 2 - Diagrama de Casos de Uso - Criptomoedas



Fonte: Gabrielle de Lima Gabaldi (2020)

## Diagrama de Sequência

O Diagrama de Sequência determina a Linha de vida, e as mensagens trocadas entre usuário e administrador, determinando em que ordem acontecerá os eventos, para demonstrar seu desempenho até o fim da linha de vida. Os atores são os mesmos do Diagrama de Caso de uso. Como mostra a figura 3, o diagrama de sequência indica:

1. O usuário realiza o login no sistema, e o sistema retorna uma mensagem de validação – liberando o acesso.

2. O usuário solicita a visualização de criptomoedas, e o sistema retorna a listagem das Criptomoedas.

3. O usuário seleciona a criptomoeda desejada, e o sistema retorna a criptomoeda.

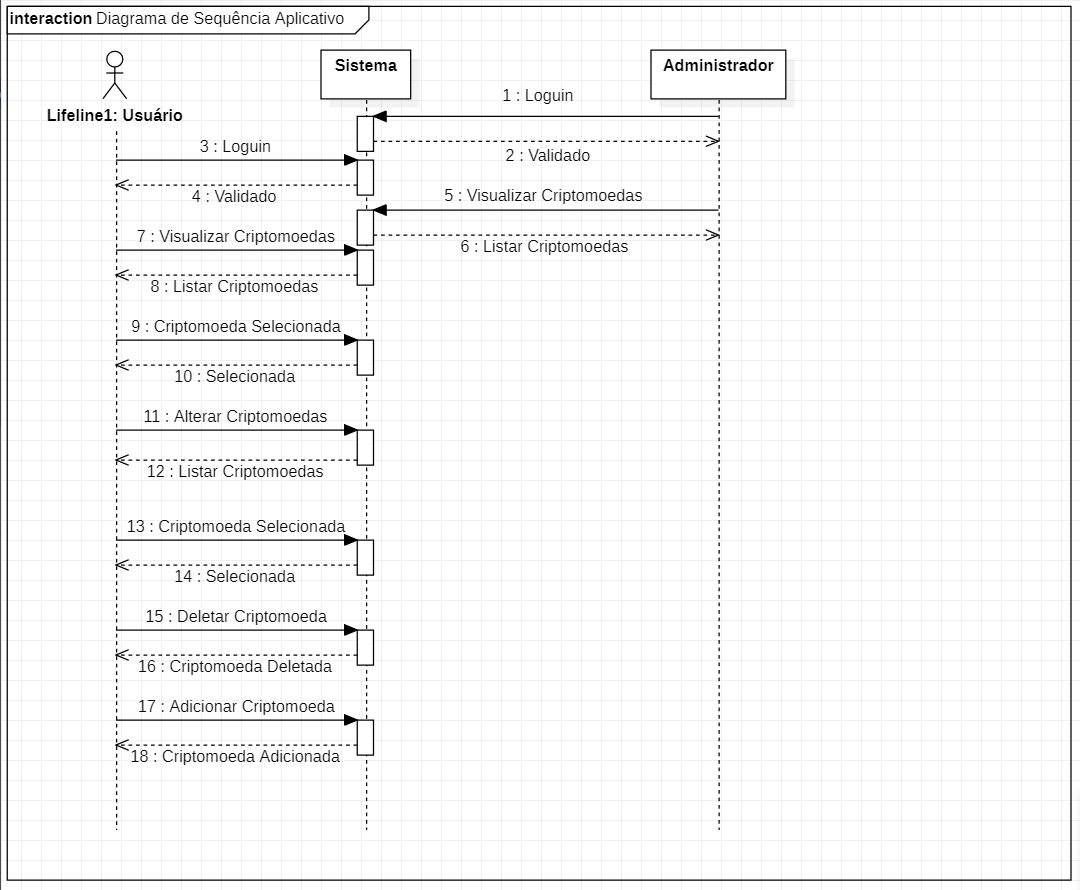
4. O usuário pede para deletar a criptomoeda, e o sistema retorna a criptomoeda solicitada pelo usuário.

5. O usuário pede para adicionar uma criptomoeda, e o sistema responde com a adição da criptomoeda.

6. O administrador realiza o login, e o sistema retorna validação - liberado o acesso ao sistema.

7. O administrador pede para visualizar as criptomoedas, o sistema lista as criptomoedas.

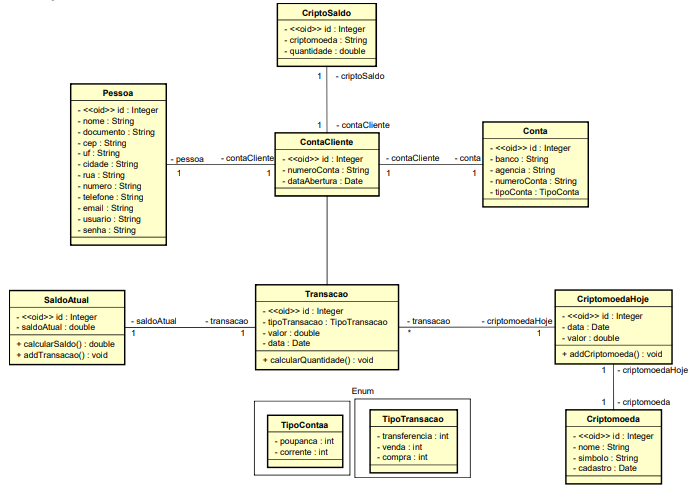
Figura 3 - Diagrama de Sequência

  
 Fonte: Gabrielle Gabaldi e Kaique Marcelino (2020)

## Diagrama de Classes

Em comparação a primeira parte do projeto, o diagrama de classes sofreu pequenas alterações, como pode ser visto na figura 1. Essas alterações se devem ao fato de, na primeira parte do projeto, o conhecimento em UML e Programação Orientada a Objetos serem extremamente rasos. Com a continuidade dos estudos, com um conhecimento maior em ambos, foi possível desenvolver um diagrama mais correto (assim como o programa).

Figura 4 - Diagrama de Classe



Fonte: Paola Regina Garbato (2020)

As alterações, foram: a classe Pessoa, antes era dividida em 4: Pessoa, e, Pessoa Física e Pessoa Jurídica herdando da primeira, e Cliente. Além disso, essa classe recebeu usuário e senha.

A classe ContaCliente, passou a armazenar somente o número da conta, e a data de abertura. Foi criada uma classe chamada Conta, pensando que, para que sejam feitos depósitos e transferências com a conta BlockTech, é necessário que haja uma conta vinculada. Para definir o tipo de conta, foi criado um Enum.

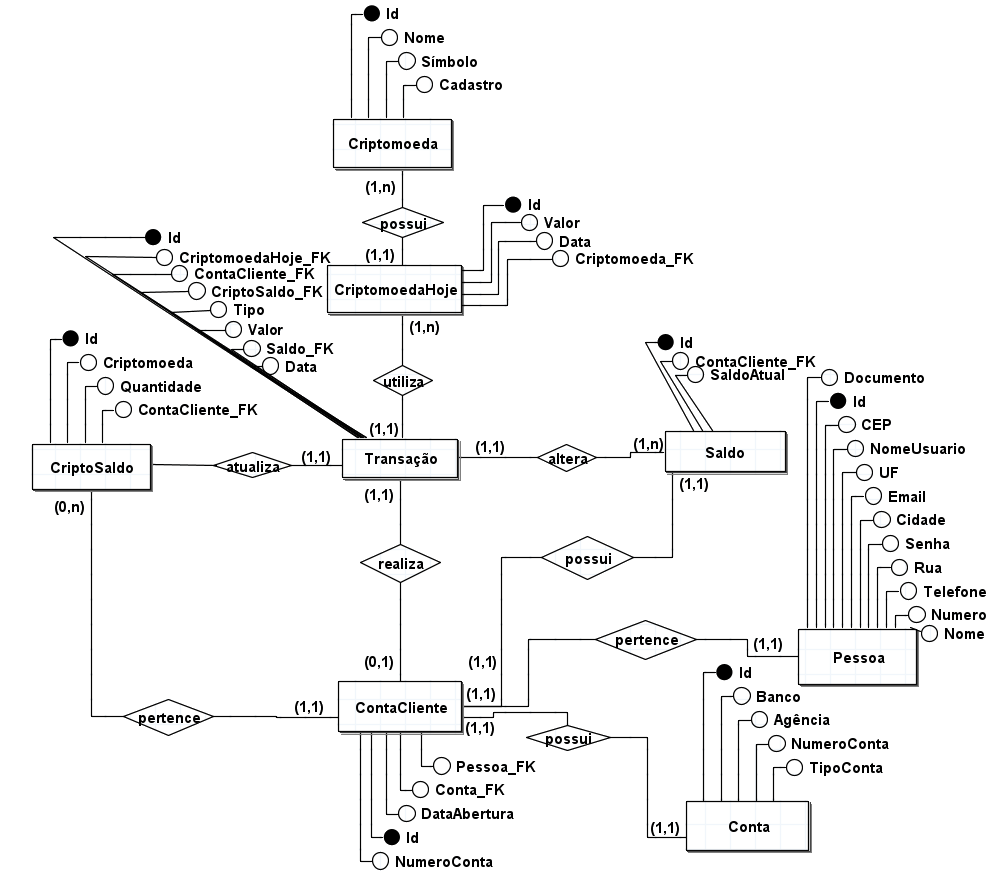
Agora, a classe Criptomoeda, armazena apenas o nome, o símbolo e sua data de cadastro, e, para adicionar seu valor, que é atualizado diariamente, foi criada uma nova classe chamada CriptomoedaHoje.

A classe Transação, substitui a antiga classe Operação. A classe Operação, antes, estava associada a três classes: Compra, Venda e Transferência. Essas classes, foram definidas na classe Transação pelo atributo “TipoTransacao”, e foi criado um Enum para adicionar essas três opções. Para armazenar o saldo em real e em criptomoedas dos clientes, foram criadas respectivamente as classes SaldoAtual, e CriptoSaldo.

# BANCO DE DADOS

Apesar do banco de dados ter sido elaborado na primeira parte do projeto, ao longo do desenvolvimento do projeto, foram necessárias pequenas alterações no mesmo. Todas as informações necessárias dos clientes, foram armazenadas, - porém, diferente do entregue na primeira parte do projeto, - apenas em uma tabela. Outra mudança, foi a criação de um usuário e senha, e também a tabela Operação que existia, passou a ser um ENUM no programa, e indicado como “tipo” na tabela transação, como pode ser visto na figura 1 – notação Peter Chen, e na figura 2 – Diagrama Entidade Relacionamento, e também na figura 3 - Diagrama feito no SGBD, que continua sendo o SQL Server Management Studio.

Figura 5 - Notação Peter Chen

****

Fonte: Paola Regina Garbato (2020)

Figura 6 - DER

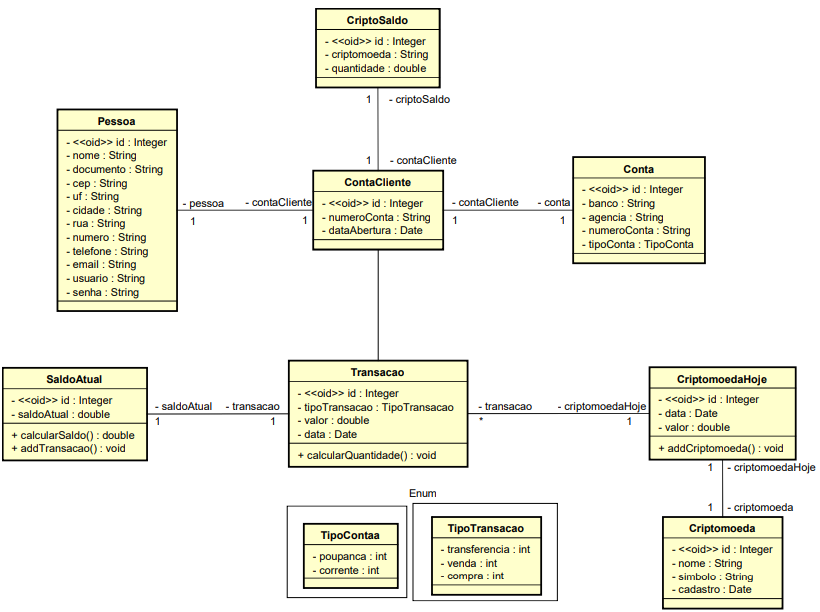
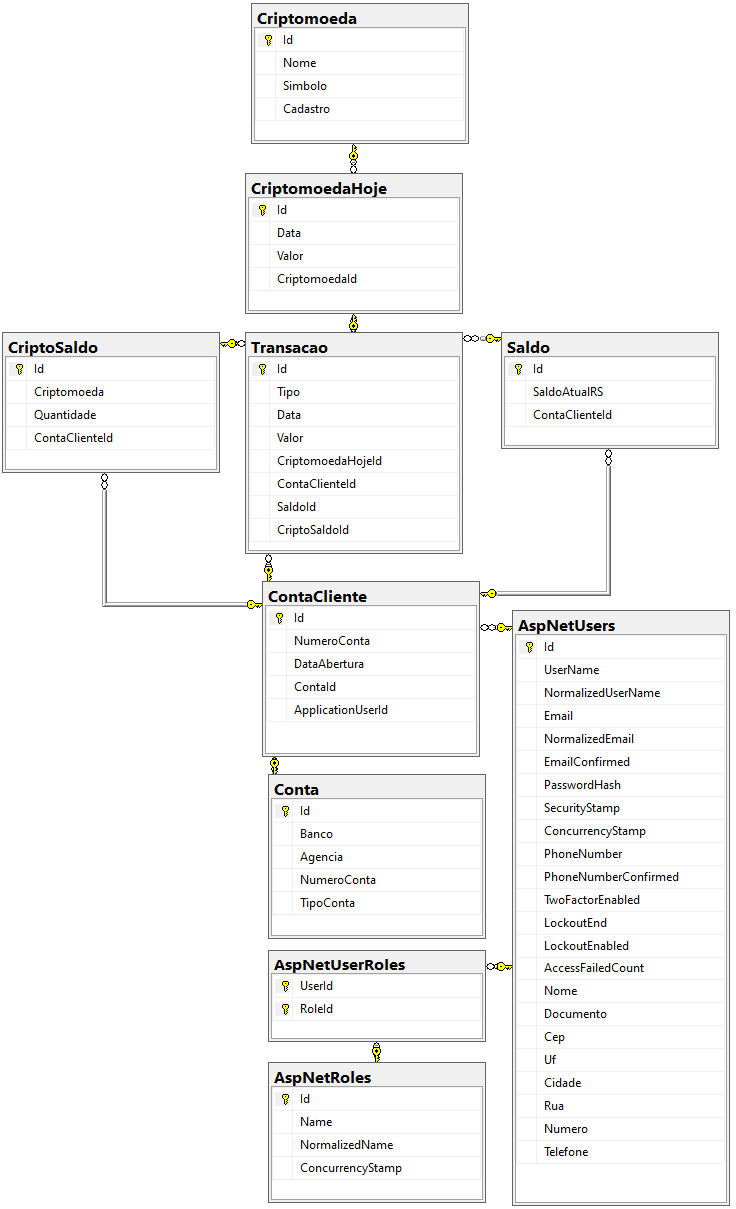


Figura 7 – DER - SQL Server Management Studio



Fonte: Paola Regina Garbato (2020)

## Dicionário de Dados

Devido a essas alterações, o dicionário de dados também foi atualizado.

Tabela 3 - Dicionário de Dados - Usuário

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Atributo** | | **Tipo de Dados** | **Tamanho** | | **Restrições** | | **Descrição** |
| Id | Varchar | | | 60 bytes | | PK, NOT NULL | Código criptografado de identificação do usuário |
| NomeUsuario | Varchar | | | 70 bytes | | NOT NULL | Nome de usuário |
| Nome | Varchar | | | 60 bytes | | NOT NULL | Nome ou Razão Social do usuário |
| Senha | Varchar | | | 20 bytes | | NOT NULL | Senha de acesso |
| Endereco | Varchar | | | 70 bytes | | NOT NULL | Endereço do cliente |
| CEP | Varchar | | | 8 bytes | | NOT NULL | CEP do cliente |
| UF | Char | | | 2 bytes | | NOT NULL | Estado do cliente |
| Cidade | Varchar | | | 60 bytes | | NOT NULL | Cidade do cliente |
| NumeroCasa | Char | | | 6 bytes | | NOT NULL | Número da casa do cliente |
| Telefone | Varchar | | | 12 bytes | | NOT NULL | Telefone do cliente |
| Email | Varchar | | | 70 bytes | | NOT NULL | E-mail do cliente |
| Documento | Varchar | | | 20 bytes | | NOT NULL | CPF/CNPJ do usuário |

Fonte: Paola Regina Garbato (2020)

Tabela 4 - Dicionário de Dados – Conta

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atributo** | **Tipo de Dados** | **Tamanho** | **Restrições** | **Descrição** |
| Id | Inteiro | 4 bytes | PK, NOT NULL | Código de identificação da conta vinculada |
| Banco | Varchar | 40 bytes | NOT NULL | Nome do banco |
| Agência | Varchar | 8 bytes | NOT NULL | Número da agência |
| NumeroConta | Varchar | 20 bytes | NOT NULL | Número da conta |
| TipoConta | Varchar | 16 bytes | NOT NULL | Tipo de conta |

Fonte: Paola Regina Garbato (2020)

Tabela 5 - Dicionário de Dados - ContaCliente

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atributo** | **Tipo de Dados** | **Tamanho** | **Restrições** | **Descrição** |
| Id | Inteiro | 4 bytes | PK, NOT NULL | Código de identificação da conta do usuário |
| NumeroConta | Varchar | 8 bytes | NOT NULL | Número da conta |
| DataAbertura | Date | 8 bytes | NOT NULL | Data de abertura da conta |
| IDConta | Inteiro | 4 bytes | FK, NOT NULL | Código de identificação da conta vinculada |
| IDUsuario | Varchar | 60 bytes | FK, NOT NULL | Código de identificação do usuário |

Fonte: Paola Regina Garbato (2020)

Tabela 6 - Dicionário de Dados - Criptomoeda

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atributo** | **Tipo de Dados** | **Tamanho** | **Restrições** | **Descrição** |
| Id | Inteiro | 4 bytes | PK, NOT NULL | Código de identificação da criptomoeda |
| Nome | Varchar | 70 bytes | NOT NULL | Nome da criptomoeda |
| Simbolo | Varchar | 4 bytes | NOT NULL | Símbolo da criptomoeda |
| Cadastro | Date | 8 bytes | NOT NULL | Data de cadastro da criptomoeda |

Fonte: Paola Regina Garbato (2020)

Tabela 7 - Dicionário de Dados - CriptomoedaHoje

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atributo** | **Tipo de Dados** | **Tamanho** | **Restrições** | **Descrição** |
| Id | Inteiro | 4 bytes | PK, NOT NULL | Código de identificação da criptomoeda hoje |
| Data | Date | 8 bytes | NOT NULL | Data de atualização da criptomoeda |
| Valor | Decimal | 8 bytes | NOT NULL | Valor da criptomoeda |
| IDCriptomoeda | Inteiro | 8 bytes | FK, NOT NULL | Código de identificação da criptomoeda associada |

Fonte: Paola Regina Garbato (2020)

Tabela 8 - Dicionário de Dados - CriptoSaldo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atributo** | **Tipo de Dados** | **Tamanho** | **Restrições** | **Descrição** |
| Id | Inteiro | 4 bytes | PK, NOT NULL | Código de identificação do saldo de criptomoedas |
| Criptomoeda | Varchar | 70 bytes | NOT NULL | Nome da criptomoeda |
| Quantidade | Decimal | 8 bytes | NOT NULL | Quantidade de criptomoeda |
| IDContaCliente | Inteiro | 4 bytes | FK, NOT NULL | Código de identificação da conta do cliente associado |

Fonte: Paola Regina Garbato (2020)

Tabela 9 - Dicionário de Dados - Saldo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atributo** | **Tipo de Dados** | **Tamanho** | **Restrições** | **Descrição** |
| Id | Inteiro | 4 bytes | PK, NOT NULL | Código de identificação do saldo |
| SaldoAtualRS | Decimal | 8 bytes | NOT NULL | Saldo atual em real |
| IDContaCliente | Inteiro | 4 bytes | FK, NOT NULL | Código de identificação da conta do cliente associado |

Fonte: Paola Regina Garbato (2020)

Tabela 10 - Dicionário de Dados - Transacao

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atributo** | **Tipo de Dados** | **Tamanho** | **Restrições** | **Descrição** |
| Id | Inteiro | 4 bytes | PK, NOT NULL | Código criptografado de identificação do usuário |
| Tipo | Varchar | 20 bytes | NOT NULL | Tipo de transação |
| Data | Date | 8 bytes | NOT NULL | Data da transação |
| Valor | Decimal | 20 bytes | NOT NULL | Valor da transação |
| IDCriptomoedaHoje | Inteiro | 4 bytes | FK, NOT NULL | Código de identificação da criptomoeda hoje |
| IDContaCliente | Inteiro | 4 bytes | FK, NOT NULL | Código de identificação da conta cliente |
| IDSaldo | Inteiro | 4 bytes | FK, NOT NULL | Código de identificação do saldo |
| IDCriptoSaldo | Inteiro | 4 bytes | FK, NOT NULL | Código de identificação do cripto saldo |

Fonte: Paola Regina Garbato (2020)

## Matriz de Rastreabilidade

# PROGRAMAÇÃO ORIENTADA A OBJETOS

## Web - ASP.NET

A versão web do software foi desenvolvida em C#, utilizando a IDE Visual Studio, que traz um grande suporte ao desenvolvedor. Foi utilizada a estrutura ASP.NET CORE MVC. O ASP.NET Core é uma extensão do .NET Framework.

### **Padrão MVC**

O MVC é um padrão de desenvolvimento de aplicações, que a divide em três partes: Model (modelo), Control (controlador) e View (visão).

**Modelo**:

* Mantém a estrutura dos dados, conhecido como "o sistema",
* Possui as entidades do negócio e os serviços,
* É a representação das regras do negócio,
* Quem possui a inteligência do sistema.

**Controlador**:

* Recebe e trata a interação do usuário com o sistema, como um botão que o usuário clica, - quem recebe a ação é o controlador,
* É o intermediário entre o sistema e as telas do sistema (View)

**Visão**:

* Apresentação para o usuário
* Define as estruturas e comportamento das telas

Essas camadas se relacionam da seguinte forma: por meio das telas (view), os usuários interagem com o sistema por meio de requisições. As requisições vão para o controlador (controller) que faz chamadas ao sistema (model) e retorna as respostas para as telas.

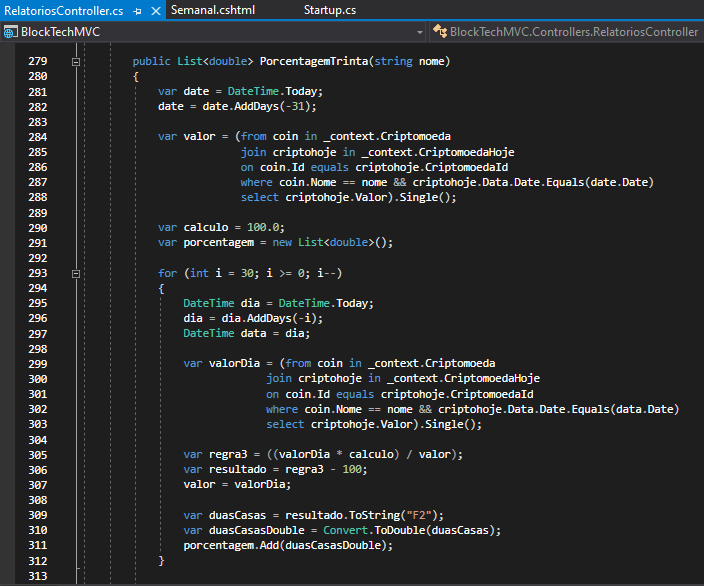
### **Entity Framework**

Pela dificuldade em criar um sistema orientado a objetos se comunicando com um banco de dados relacional, se utiliza muito ORM – Object-Relational Mappers, que significa Mapeamento objeto-relacional. A ORM é uma ferramenta que permite realizar a programação orientada a objetos, e automaticamente, ela é mapeada para o banco de dados relacional. A ORM da Microsoft se chama Entity Framework.

Para que essa ferramenta funcione, ela precisa de Providers. Os providers, são implementações da comunicação com o banco de dados utilizado, como por exemplo: Provider do SQLServer, Provider do MYSQL...

O Entity Framework possui duas classes principais que utilizamos, sendo elas, a **DBContext** e a **DbSET<TEntity>.** No DbSet, utilizamos operações do LINQ que, em tempo de execução, são convertidas para consultas SQL, como mostra o exemplo da figura 4.

Figura 8 - LINQ



Fonte: Paola Regina Garbato (2020)

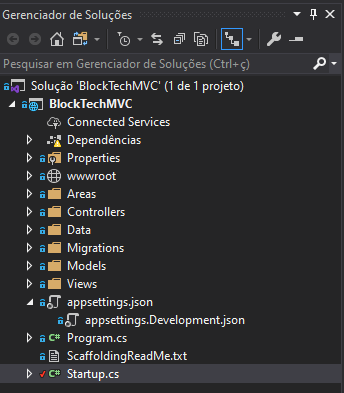
Esse exemplo mostra um método utilizado para obter a porcentagem do valor da criptomoeda, em relação ao dia anterior. Primeiramente, a data atual é armazenada na variável “date”, como mostra a linha 281. Foi utilizado DateTime.Today ao invés de DateTime.Now por não ser necessário obter o horário de cadastro das criptomoedas. Em seguida, foi utilizado o método AddDays para obter a data de 31 dias atrás. Esse método é necessário para que não haja erro de cálculos, se apenas subtraísse 31, como um número inteiro, o cálculo não daria certo – como ocorreu durante o desenvolvimento do software. Em seguida, foi realizada uma consulta com LINQ, guardando na variável “valorDia”, o valor da criptomoeda, 31 dias antes do dia atual, comparando o nome da criptomoeda com o nome passado por parâmetro no método, e a data com a data da variável “date”, declarada no início do método. Em seguida, foi criada uma lista chamada “porcentagem”, onde foram guardados todos os valores – utilizados nos gráficos. Então, foi feito um laço for, para percorrer um mês, ou seja, 30 dias. O laço for se inicia a variável com o valor 30, decrementa 1, e ocorre enquanto a variável for igual a 0. Dentro desse laço, foi declarada uma variável “dia”, que recebe o valor do dia atual, e subtrai o valor de dias de acordo com o valor da variável do laço for, então, primeiro recebe o valor de 30 dias atrás, depois 29, e assim, até retornar ao dia atual, onde subtrai 0 dias. Para cada um desses dias, foi feita uma consulta com LINQ, comparando novamente o nome da criptomoeda passada por parâmetro no método, e comparando a data com a armazenada na variável “data”, que recebe o valor da variável dia. Após obter o valor da criptomoeda no dia desejado, é feita uma regra de 3, sendo que, na primeira vez, é utilizado o valor obtido na linha 284, que seriam de 31 dias atrás. Após esse cálculo, para que seja obtido o valor de 30 dias, ao invés de 31, a variável “valor”, recebe o valor da criptomoeda – obtido na linha 299. Para que o resultado desse cálculo seja exibido apenas com duas casas decimais, ele foi convertido para string, com duas casas decimais. Em seguida, foi novamente convertido para o tipo double, e guardado dentro da Lista. E por fim, é retornada essa lista – como o método mostra, o retorno do mesmo é uma lista do tipo double.

Esse método foi chamado por todas as criptomoedas, para confecção de um gráfico para cada uma delas.

### **Estrutura do Projeto**

A figura 1 mostra toda a solução do projeto.

Figura 9 - Estrutura do Projeto



Algumas pastas, que serão descritas a seguir, sofreram alterações ao longo do projeto, outras, como “Properties”, não.

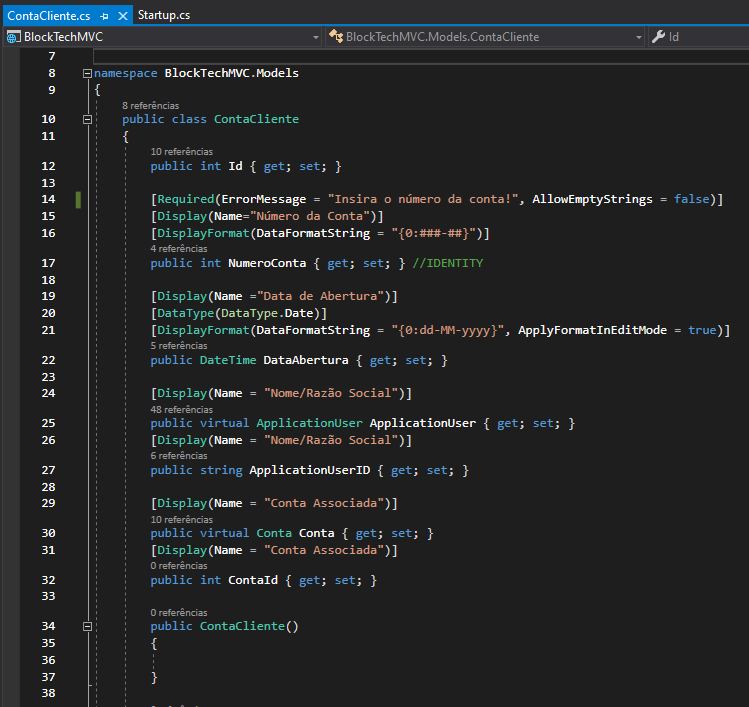
Em “Dependências” estão todos os pacotes do NuGet que foram utilizados no projeto, como o Microsoft.EntityFrameworkCore.SqlServer, - como foi citado, essa ORM facilitou o desenvolvimento de um projeto Orientado a Objetos com um banco de dados relacional. Com ele, o projeto foi feito seguindo o “Code-First”.

Em “wwwroot” estão todos os ícones, imagens, arquivos css, bootstrap, javascript utilizados no projeto – cada um em sua respectiva sub-pasta.

Em “Areas” estão os arquivos do Identity, como o Razor da página de login e cadastro. O Razor possibilita criarmos páginas HTML juntamente com o C#. Ele possui a extensão “cshtml”, sendo CS de CSharp.

Em “Controllers” estão todos os controladores citados anteriormente. Para que seja mais bem exemplificado, primeiro serão citados os Models. Na pasta “Model”, se encontram todas as classes – primeira coisa a ser criada no projeto. Como foi utilizado o Code-First, nessas classes já foram indicadas as relações que existiriam entre elas no banco de dados, como mostra o exemplo da figura 6.

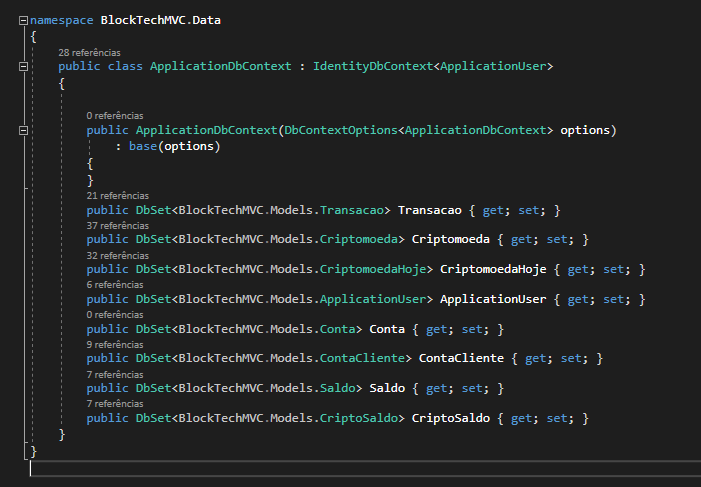
Figura 10 - Model - ContaCliente



Como mostra a figura 6, na própria classe já foi inserido um Id. Para indicar uma chave estrangeira do banco de dados, foi inserido o nome da entidade (classe) seguido do “ID”, exemplo na linha 32. O tipo da variável - que para Id geralmente é número inteiro, deve ser o mesmo tipo do ID da outra classe. Porém, como pode ser observado na linha 27, foi utilizado o tipo string, isso ocorreu, pois, na tabela de usuários, o ID é criptografado, assim como a senha, então, por haver números e letras, não é possível utilizar o tipo inteiro. Também foram utilizadas DataAnnotations como pode ser visto, para adicionar nomes diferentes aos das variáveis para serem exibidos ao usuário, definir tipos e máscaras, ou até mesmo mensagens de erro caso campos obrigatórios (Required) não sejam preenchidos.

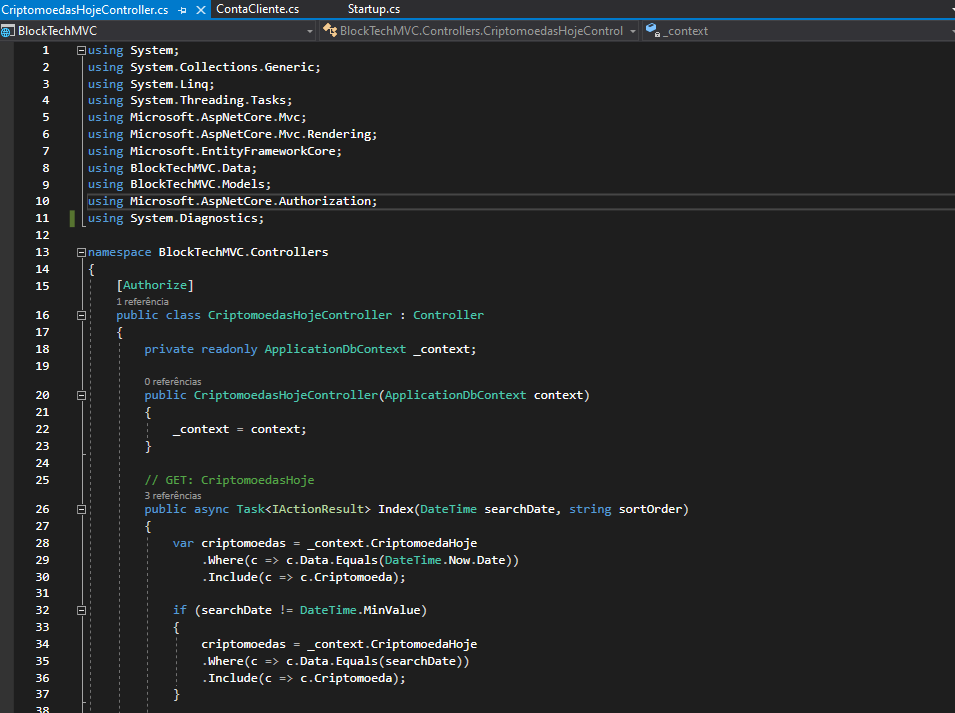
Na pasta “Data” existe o arquivo ApplicationDbContext - que herda do DbContext. Essa é a classe responsável para a comunicação com o banco de dados. Toda classe criada no Model deve ser inserida no ApplicationDbContext para que assim sejam criadas as tabelas correspondentes no banco de dados, como mostra a figura 7.

Figura 11 - ApplicationDbContext



Após a criação de todas as classes, foi criado o controlador das classes, que contém informações que serão exibidas para os clientes. Todas os controladores criados herdam da classe Controller, como pode ser visto na figura 8.

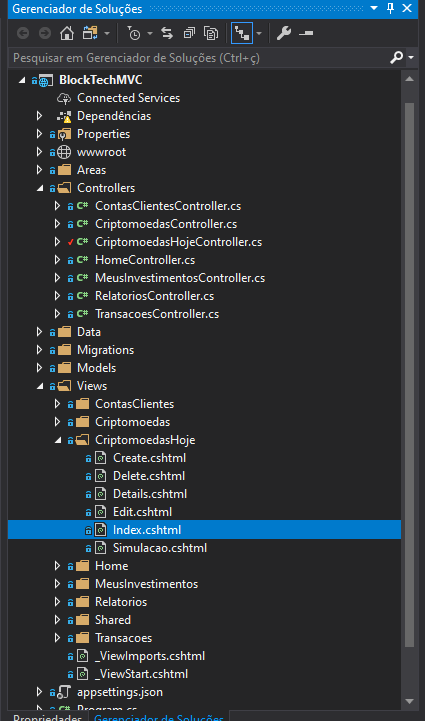
Figura 12 - Controlador - CriptomoedasHoje



No início de todos os controladores, possui uma instância do ApplicationDbContext para que possa consumir os dados do banco, e em seu construtor, é feita a injeção de dependência.

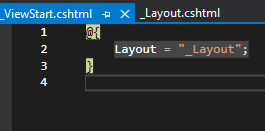
Para cada método do controlador, que retorna um “IActionResult”, é criado um Razor em uma View. No retorno do método do controlador, estão as informações que serão utilizadas na View. A View possui os arquivos Razor que serão exibidos para o usuário. Para essa exibição, é necessário, primeiro criar uma pasta com o mesmo nome do controlador. Exemplo: Na imagem 8 possui um controlador chamado “CriptomoedasHojeController”, então o nome da pasta na View deve ser “CriptomoedasHoje”. Então, nessa pasta, é criado um arquivo Razor com o mesmo nome do método do controller correspondente com o retorno IActionResult. A figura 9 mostra um exemplo do que foi citado. Todos os controllers possuem pastas correspondentes na View. Como pode ser visto como exemplo, existe o arquivo Razor chamado Index, assim como o método da figura 8.

Figura 13 - Views e Controllers



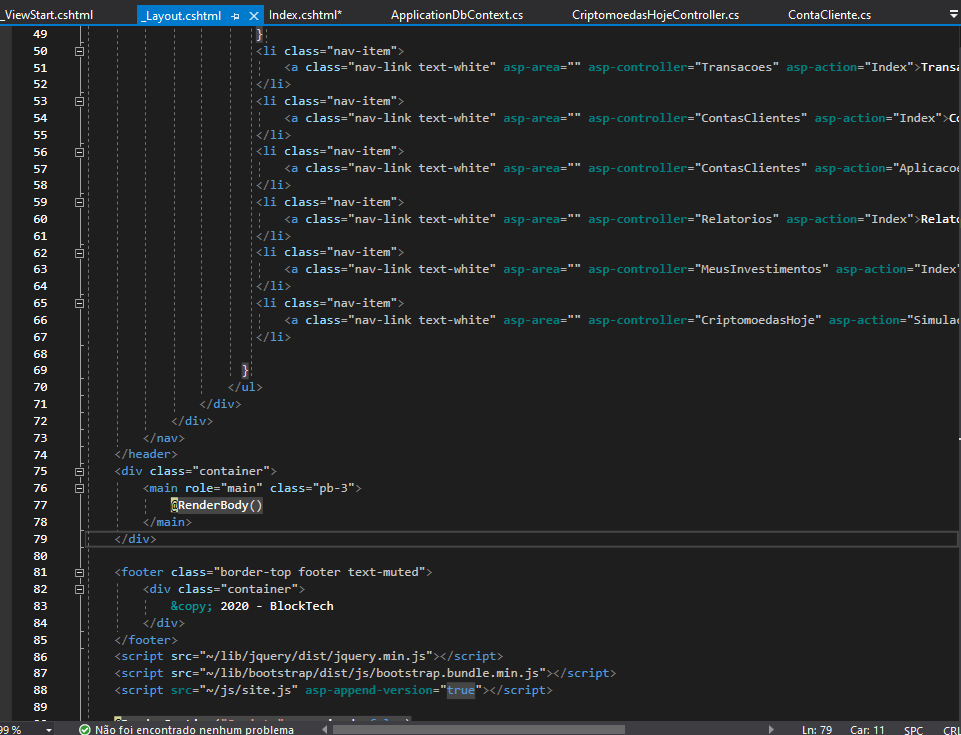
Dentro da pasta View, também há a pasta Shared. Nela existe um arquivo Razor chamado \_Layout.cshtml. Nesse arquivo estão: o cabeçalho, o rodapé, os arquivos CSS e JavaScript utilizados, todas as outras páginas herdam dessa. O arquivo \_ViewStart.cshtml, que também pode ser visto na figura 9, faz essa indicação, como pode ser visto na figura 10.

Figura 14 - ViewStart



Entre o cabeçalho e o rodapé, está o “@RenderBody()”. Como pode ser visto na linha 77, da figura 11. Ele indica que, todo o conteúdo das páginas que herdam dele, no caso, todas, serão renderizadas nesse local.

Figura 15 - Layout.cshtml

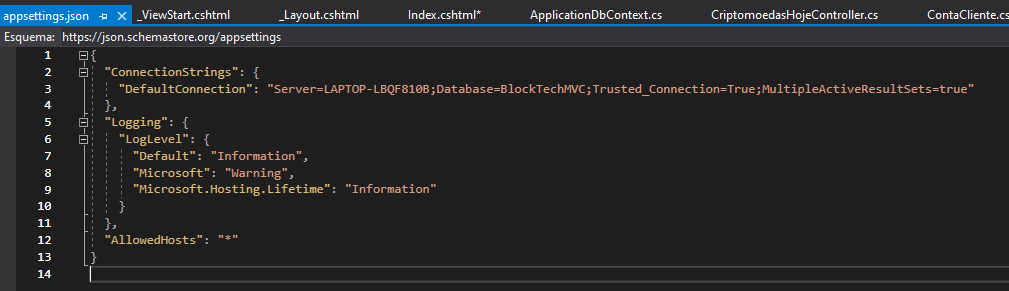


Esses arquivos citados começam com um underline, para que o usuário não possa acessá-los.

Na pasta “Migrations” estão os arquivos que indicam as criações de tabelas no banco de dados, e também as alterações. Toda vez que uma classe é alterada, é necessário adicionar uma nova “Migration” para que o banco de dados seja atualizado conforme a classe. Para adicionar essa migration, basta inserir no Console o Gerenciador de Pacotes, o comando: “Add-Migration NomeDaMigration”, e em seguida o comando para atualizar o banco: “Update-Database”.

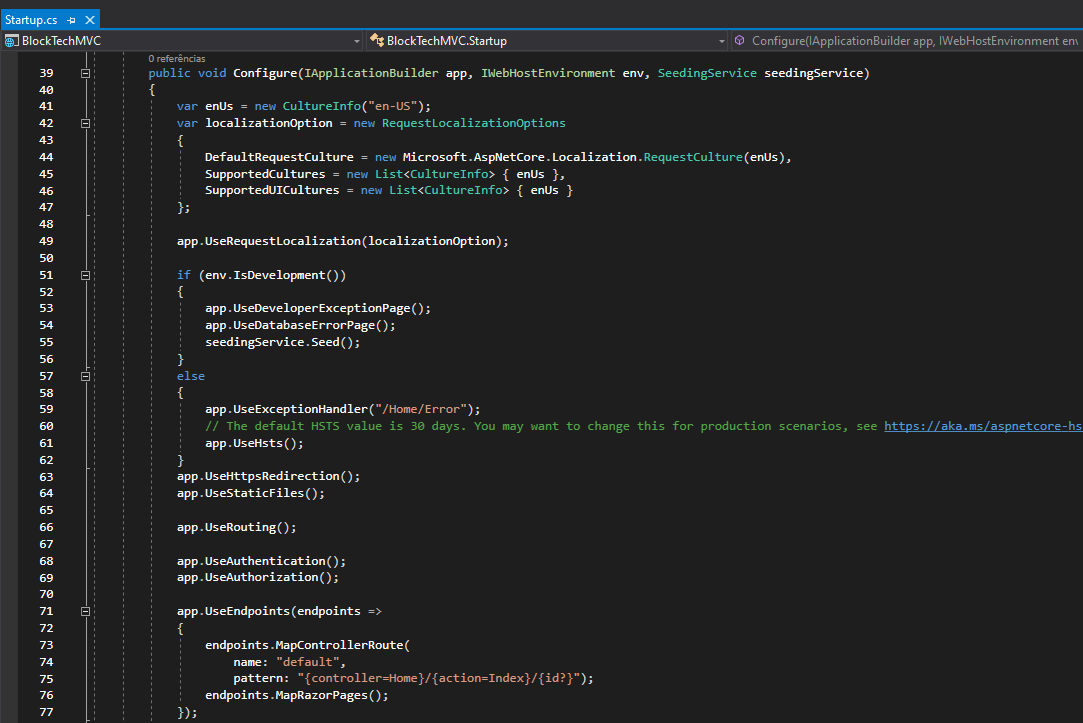
No arquivo “appsettings.json” está a rota do banco de dados utilizado, como pode ser visto na figura 12

Figura 16 - appsettings.json



No arquivo “Startup.cs” estão algumas configurações importantes, como a rota em que o site inicia, que no caso é a Index do Controller Home, como pode ser visto na linha 75 da figura 13.

Figura 17 - Startup.cs



Todo o programa, e suas telas podem ser visualizadas no Manual do Usuário – Apêndice B – página X

# CONCLUSÃO

**REFERÊNCIAS**

**GLOSSÁRIO**

**APÊNDICE A – MANUAL DESKTOP**

**APÊNDICE B – MANUAL WEB**

**APÊNDICE C - SCRIPT DE CRIAÇÃO DE TABELAS SQL SERVER**

USE [BlockTechMVC]

GO

/\*\*\*\*\*\* Object: Table [dbo].[AspNetRoles] Script Date: 04/11/2020 16:31:03 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetRoles](

[Id] [nvarchar](450) NOT NULL,

[Name] [nvarchar](256) NULL,

[NormalizedName] [nvarchar](256) NULL,

[ConcurrencyStamp] [nvarchar](max) NULL,

CONSTRAINT [PK\_AspNetRoles] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[AspNetUserRoles] Script Date: 04/11/2020 16:31:04 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUserRoles](

[UserId] [nvarchar](450) NOT NULL,

[RoleId] [nvarchar](450) NOT NULL,

CONSTRAINT [PK\_AspNetUserRoles] PRIMARY KEY CLUSTERED

(

[UserId] ASC,

[RoleId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[AspNetUsers] Script Date: 04/11/2020 16:31:04 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[AspNetUsers](

[Id] [nvarchar](450) NOT NULL,

[UserName] [nvarchar](256) NOT NULL,

[NormalizedUserName] [nvarchar](256) NULL,

[Email] [nvarchar](256) NOT NULL,

[NormalizedEmail] [nvarchar](256) NULL,

[EmailConfirmed] [bit] NOT NULL,

[PasswordHash] [nvarchar](max) NULL,

[SecurityStamp] [nvarchar](max) NULL,

[ConcurrencyStamp] [nvarchar](max) NULL,

[PhoneNumber] [nvarchar](max) NULL,

[PhoneNumberConfirmed] [bit] NOT NULL,

[TwoFactorEnabled] [bit] NOT NULL,

[LockoutEnd] [datetimeoffset](7) NULL,

[LockoutEnabled] [bit] NOT NULL,

[AccessFailedCount] [int] NOT NULL,

[Nome] [nvarchar](60) NULL,

[Documento] [nvarchar](14) NULL,

[Cep] [nvarchar](9) NULL,

[Uf] [nvarchar](2) NULL,

[Cidade] [nvarchar](58) NULL,

[Rua] [nvarchar](max) NOT NULL,

[Numero] [nvarchar](10) NULL,

[Telefone] [nvarchar](15) NULL,

CONSTRAINT [PK\_AspNetUsers] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY],

UNIQUE NONCLUSTERED

(

[Email] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY],

UNIQUE NONCLUSTERED

(

[UserName] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Conta] Script Date: 04/11/2020 16:31:04 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Conta](

[Id] [int] IDENTITY(1,1) NOT NULL,

[Banco] [nvarchar](max) NOT NULL,

[Agencia] [nvarchar](4) NULL,

[NumeroConta] [nvarchar](max) NOT NULL,

[TipoConta] [int] NOT NULL,

CONSTRAINT [PK\_Conta] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[ContaCliente] Script Date: 04/11/2020 16:31:04 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[ContaCliente](

[Id] [int] NOT NULL,

[NumeroConta] [int] IDENTITY(27,123412) NOT NULL,

[DataAbertura] [datetime2](7) NOT NULL,

[ContaId] [int] NULL,

[ApplicationUserId] [nvarchar](450) NULL,

CONSTRAINT [PK\_ContaCliente] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Criptomoeda] Script Date: 04/11/2020 16:31:04 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Criptomoeda](

[Id] [int] IDENTITY(1,1) NOT NULL,

[Nome] [nvarchar](max) NULL,

[Simbolo] [nvarchar](5) NULL,

[Cadastro] [datetime2](7) NOT NULL,

CONSTRAINT [PK\_Criptomoeda] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[CriptomoedaHoje] Script Date: 04/11/2020 16:31:04 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[CriptomoedaHoje](

[Id] [int] IDENTITY(1,1) NOT NULL,

[Data] [datetime2](7) NOT NULL,

[Valor] [decimal](20, 2) NOT NULL,

[CriptomoedaId] [int] NOT NULL,

CONSTRAINT [PK\_CriptomoedaHoje] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[CriptoSaldo] Script Date: 04/11/2020 16:31:04 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[CriptoSaldo](

[Id] [int] IDENTITY(1,1) NOT NULL,

[Criptomoeda] [nvarchar](max) NULL,

[Quantidade] [decimal](20, 6) NOT NULL,

[ContaClienteId] [int] NULL,

CONSTRAINT [PK\_CriptoSaldo] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Saldo] Script Date: 04/11/2020 16:31:04 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Saldo](

[Id] [int] IDENTITY(1,1) NOT NULL,

[SaldoAtualRS] [decimal](18, 2) NULL,

[ContaClienteId] [int] NULL,

CONSTRAINT [PK\_Saldo] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Transacao] Script Date: 04/11/2020 16:31:04 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Transacao](

[Id] [int] IDENTITY(1,1) NOT NULL,

[Tipo] [int] NOT NULL,

[Data] [datetime2](7) NOT NULL,

[Valor] [float] NOT NULL,

[CriptomoedaHojeId] [int] NOT NULL,

[ContaClienteId] [int] NOT NULL,

[SaldoId] [int] NULL,

[CriptoSaldoId] [int] NULL,

CONSTRAINT [PK\_Transacao] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[AspNetUserRoles] WITH CHECK ADD CONSTRAINT [FK\_AspNetUserRoles\_AspNetRoles\_RoleId] FOREIGN KEY([RoleId])

REFERENCES [dbo].[AspNetRoles] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserRoles] CHECK CONSTRAINT [FK\_AspNetUserRoles\_AspNetRoles\_RoleId]

GO

ALTER TABLE [dbo].[AspNetUserRoles] WITH CHECK ADD CONSTRAINT [FK\_AspNetUserRoles\_AspNetUsers\_UserId] FOREIGN KEY([UserId])

REFERENCES [dbo].[AspNetUsers] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[AspNetUserRoles] CHECK CONSTRAINT [FK\_AspNetUserRoles\_AspNetUsers\_UserId]

GO

ALTER TABLE [dbo].[ContaCliente] WITH CHECK ADD FOREIGN KEY([ApplicationUserId])

REFERENCES [dbo].[AspNetUsers] ([Id])

GO

ALTER TABLE [dbo].[ContaCliente] WITH CHECK ADD CONSTRAINT [FK\_ContaCliente\_Conta\_ContaId] FOREIGN KEY([ContaId])

REFERENCES [dbo].[Conta] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[ContaCliente] CHECK CONSTRAINT [FK\_ContaCliente\_Conta\_ContaId]

GO

ALTER TABLE [dbo].[CriptomoedaHoje] WITH CHECK ADD CONSTRAINT [FK\_CriptomoedaHoje\_Criptomoeda\_CriptomoedaId] FOREIGN KEY([CriptomoedaId])

REFERENCES [dbo].[Criptomoeda] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[CriptomoedaHoje] CHECK CONSTRAINT [FK\_CriptomoedaHoje\_Criptomoeda\_CriptomoedaId]

GO

ALTER TABLE [dbo].[CriptoSaldo] WITH CHECK ADD FOREIGN KEY([ContaClienteId])

REFERENCES [dbo].[ContaCliente] ([Id])

GO

ALTER TABLE [dbo].[Saldo] WITH CHECK ADD FOREIGN KEY([ContaClienteId])

REFERENCES [dbo].[ContaCliente] ([Id])

GO

ALTER TABLE [dbo].[Transacao] WITH CHECK ADD FOREIGN KEY([CriptoSaldoId])

REFERENCES [dbo].[CriptoSaldo] ([Id])

GO

ALTER TABLE [dbo].[Transacao] WITH CHECK ADD CONSTRAINT [FK\_Transacao\_ContaCliente\_ContaClienteId] FOREIGN KEY([ContaClienteId])

REFERENCES [dbo].[ContaCliente] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Transacao] CHECK CONSTRAINT [FK\_Transacao\_ContaCliente\_ContaClienteId]

GO

ALTER TABLE [dbo].[Transacao] WITH CHECK ADD CONSTRAINT [FK\_Transacao\_CriptomoedaHoje\_CriptomoedaHojeId] FOREIGN KEY([CriptomoedaHojeId])

REFERENCES [dbo].[CriptomoedaHoje] ([Id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Transacao] CHECK CONSTRAINT [FK\_Transacao\_CriptomoedaHoje\_CriptomoedaHojeId]

GO

ALTER TABLE [dbo].[Transacao] WITH CHECK ADD CONSTRAINT [FK\_Transacao\_Saldo\_SaldoId] FOREIGN KEY([SaldoId])

REFERENCES [dbo].[Saldo] ([Id])

GO

ALTER TABLE [dbo].[Transacao] CHECK CONSTRAINT [FK\_Transacao\_Saldo\_SaldoId]

GO