Enterprise Distributed Application

Trabalho #2

Mestrado Integrado em Engenharia Informática e Computação
Tecnologias de Distribuição e Integração
EIC0077-2S

João Carlos Teixeira de Sá - 201107925 (ei11142@fe.up.pt)
João Pedro Matos Teixeira Dias - 201106781 (ei11137@fe.up.pt)

Faculdade de Engenharia da Universidade do Porto
Rua Roberto Frias, sn, 4200-465 Porto, Portugal

23 de Maio de 2015

Conteúdo

1 Introdução

O presente documento apresenta o desenvolvimento do projeto *Enterprise Distributed Application* para a unidade curricular de Tecnologias de Distribuição e Integração.

O projeto tem como objetivo o desenvolvimento de um sistema, *Enterprise Distributed Application*, capaz de gerir compras e vendas de livros de uma editora (*diginotes*).

O sistema é composto por um servidor presente numa loja que persiste a informação relativa aos livros existentes para venda tais como o título, preço e o stock disponível tanto na loja como em armazém. Este servidor encontra-se ligado à *internet* e está sempre disponível.

As compras de um livro por parte de um consumidor podem ser feitas diretamente na loja ou através de uma aplicação *web*. Caso exista stock do livro pretendido o mesmo é imediatamente entregue ao consumidor no caso da compra ser feita na loja ou enviado pelo correio caso seja efetuado o pedido através da aplicação *web*; no caso de não existir stock a loja deve enviar uma mensagem para o armazém a pedir o envio do livro em questão numa quantidade 10 vezes superior ao pretendido pelo cliente.

2 Tecnologias

O projeto foi desenvolvido usando a *framework Windows Communication Foundation* para o desenvolvimento da API disponibilizada da loja bem como a tecnologia *Microsoft Message Queuing* para o envio e receção das mensagens de pedido de stock de livros ao armazém utilizando em ambos os casos a linguagem de programação C#. Além disso foram também utilizadas as linguagens de programação PHP, HTML e Javascript para o desenvolvimento da aplicação *web*.

Adicionalmente foi utilizado para persistência de dados a tecnologia de base de dados *MongoDB* e para o desenvolvimento da interfaces de utilizador da loja e do armazém a tecnologia *Windows Presentation Foundation* usando a linguagem *XAML*.

Por último, para gestão de dependências do projeto foi usado o *NuGet* e para gestão de versões foi utilizada a ferramenta *Git* usando a plataforma *GitHub*.

3 Arquitetura

O sistema apresenta a estrutura típica de um projeto baseado em *.NET Remoting*, como apresentado na fig. 1,contando assim com uma aplicação cliente e uma aplicação servidor. Acrescenta-se ainda uma persistência de

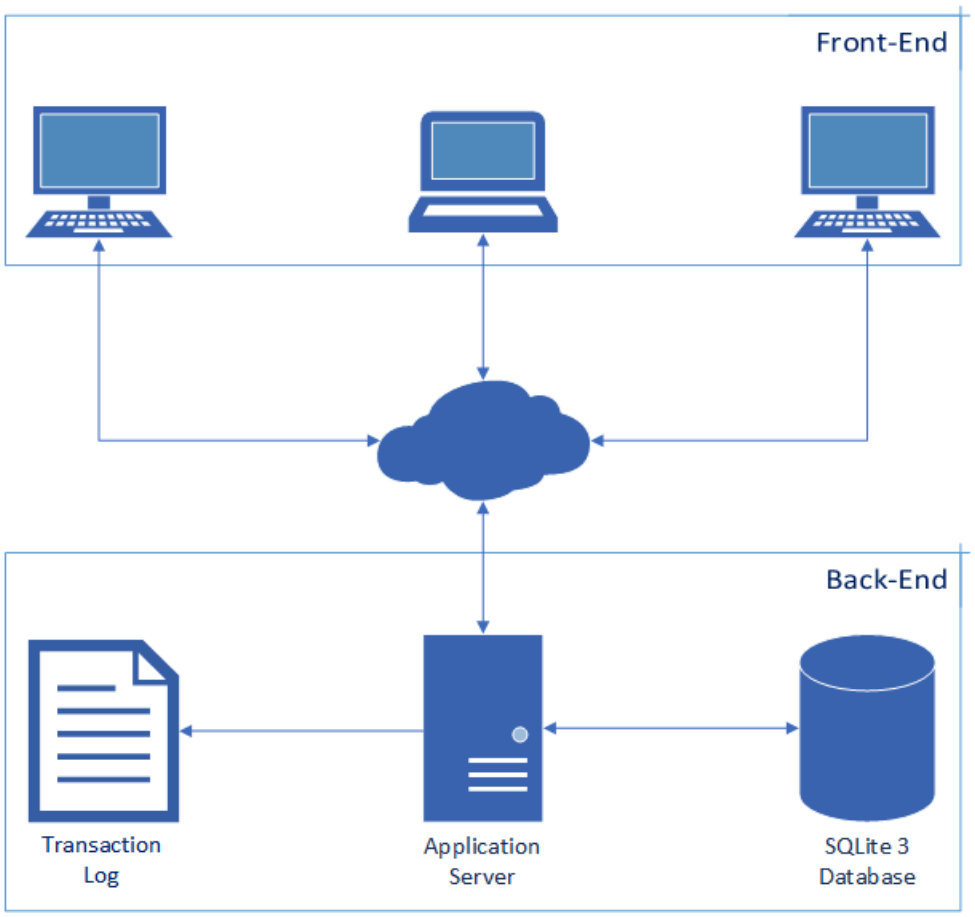

Figura 1: Arquitetura do Sistema.

dados garantida usando uma base de dados SQLite associada a aplicação servidor.

Apresenta-se também com uma divisão modular apresentando:

- Módulo comum
  - Definição das estruturas de dados partilhadas entre cliente e servidor.
- Módulo cliente
  - Aplicação cliente com interface gráfica (*GUI*).
- Módulo servidor
  - Aplicação servidor com lógica do sistema, transações e persistência de dados.

O sistema ainda utiliza objetos para representação das transações (*order*), dos valores digitais (*digicoin*) e para a informação dos utilizadores (*user*).

4 Front-end

4.1 Interface gráfica

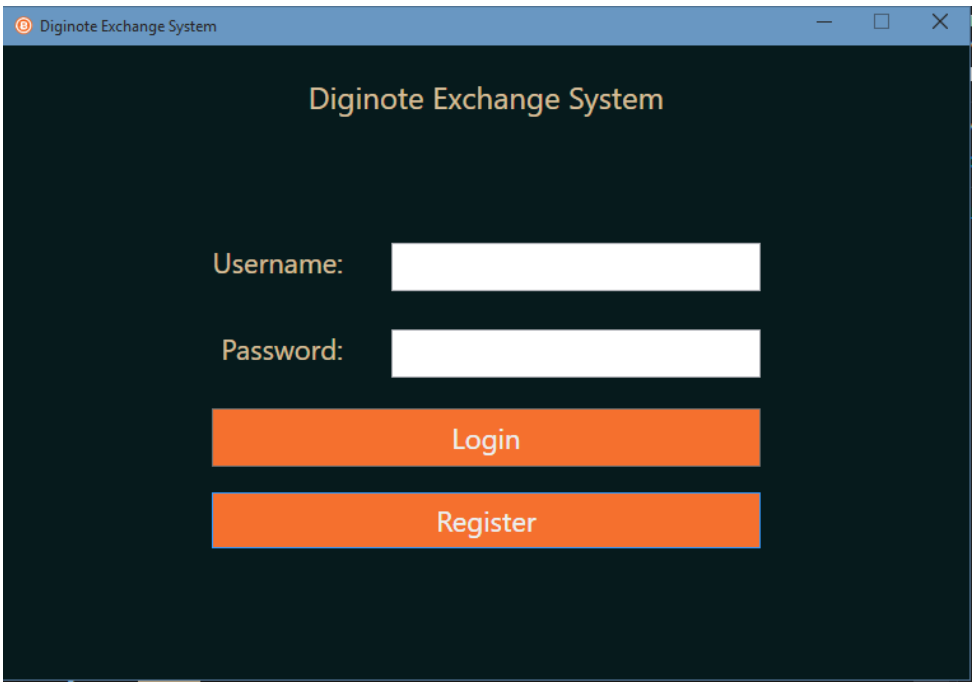

Figura 2: *Login.*

Ao nível da interface gráfica desenvolvida é disponibilizado numa primeira fase um ecrã de *login*, fig. 2, onde o utilizador pode entrar no sistema, ou, se registar no mesmo ao clicar em *regiter*.

Após o utilizador fazer *login* no sistema são disponibilizadas as opções de criar ações de venda e de compra, assim como editar as ordens existentes, fig. 3 e 4.

Por último, é mostrado alertas, fig. 5, quando existe mudança no valor do mercado, dando assim tempo ao utilizador para alterar as suas ações.

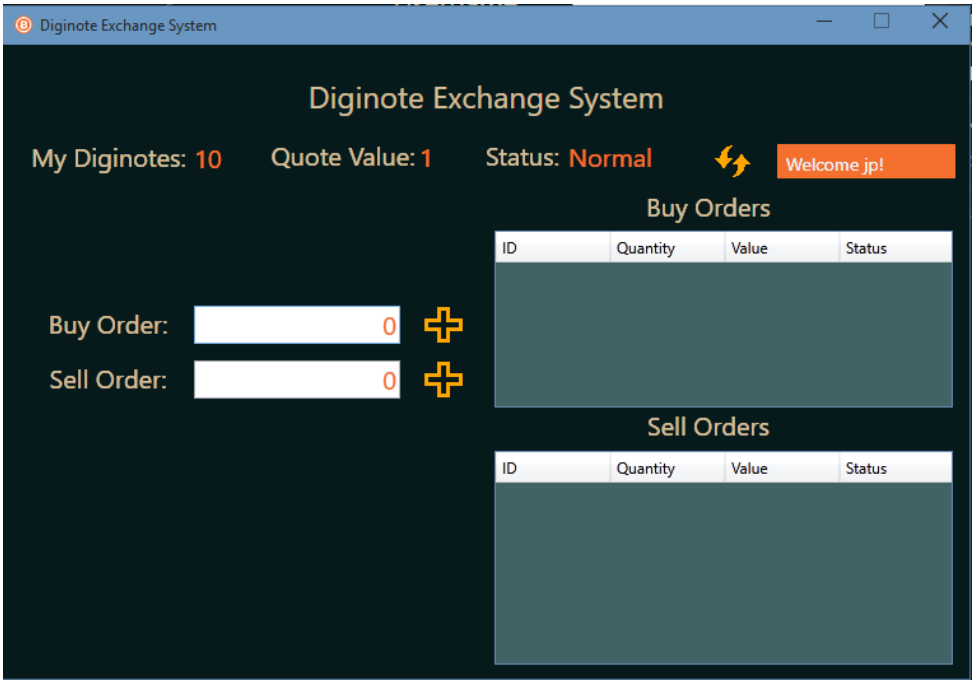

Figura 3: Ecrã inicial.

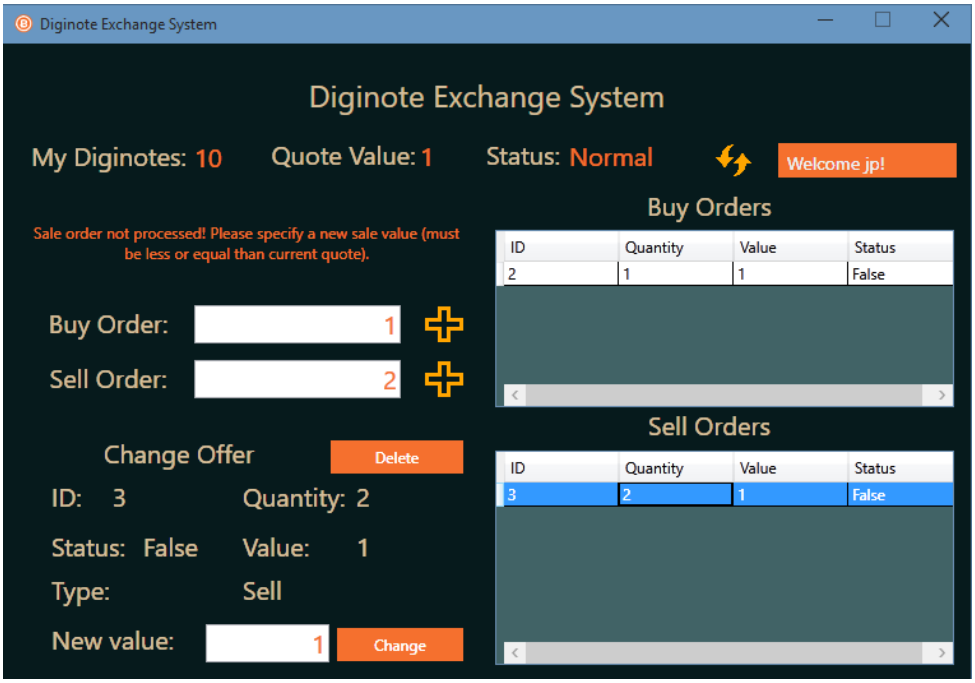

Figura 4: Edição.

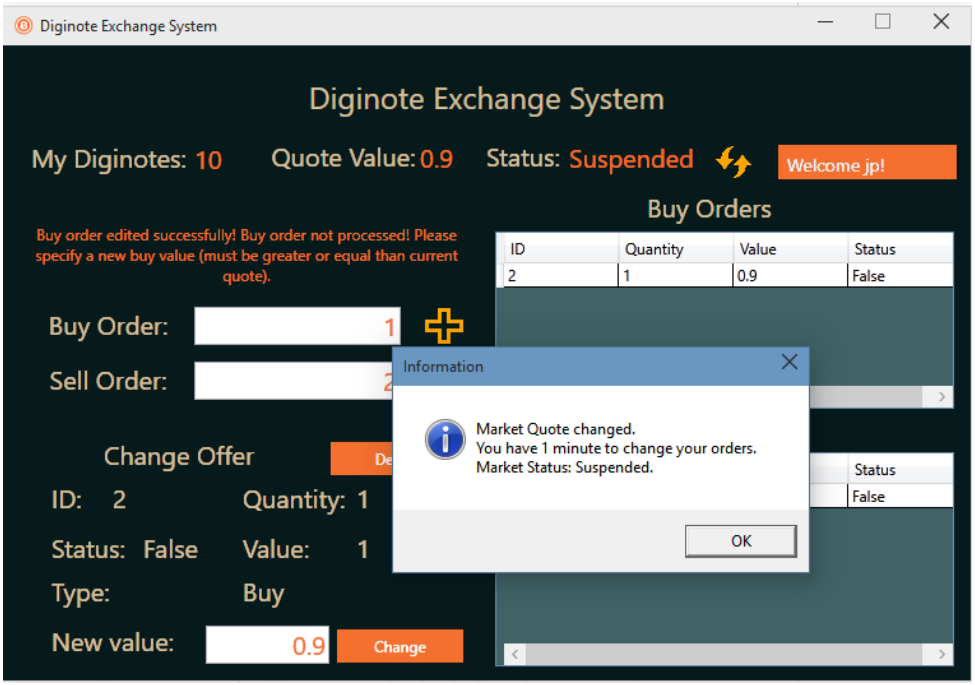

Figura 5: Alerta de mudança de valor de mercado.

4.2 Casos de Uso

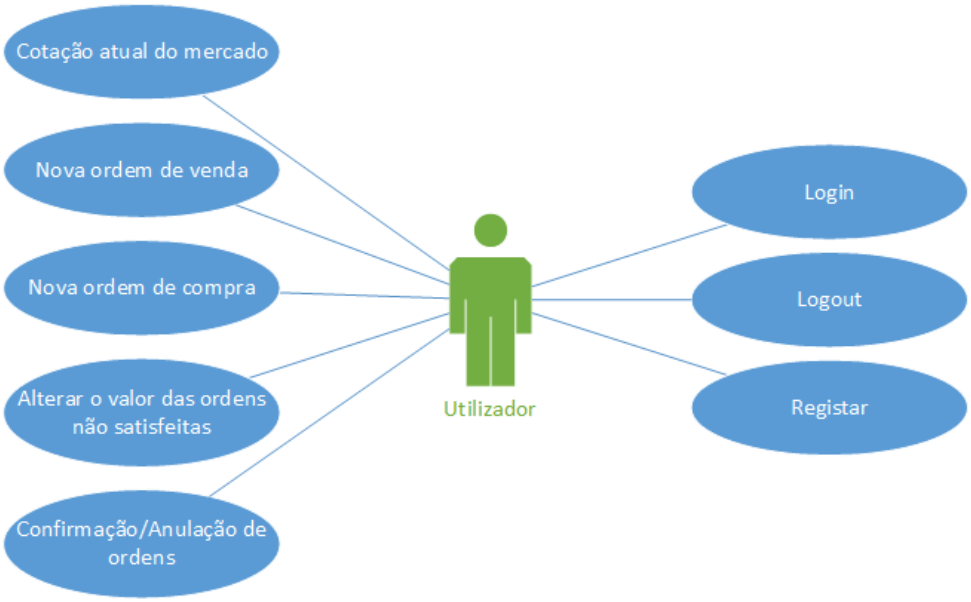

Figura 6: Diagrama de casos de uso.

5 Conclusão

O sistema encontra-se desenvolvido com todas as funcionalidades solicitadas no enunciado do projeto, possibilitando ao utilizador uma utilização total do sistema.

Testes

Para testar o correto funcionamento do sistema foram efetuadas várias experiências com diferentes clientes ligados a aplicação servidor, assim como, foram testados vários casos de falha ou do cliente e/ou servidor e garantida a persistência dos dados.

Deploy

O sistema pode ser utilizado abrindo a aplicação de servidor (*Server.exe*) e uma ou várias aplicações de cliente (*DESClient.exe*) usando o executáveis entregues. Pode ser também aberto o projeto de *Visual Studio* (*DiginoteExchangeSystem.sln*) e fazer *Start* ao mesmo na interface do *IDE*.

6 Recursos

6.1 Bibliografia

.NET Remoting Overview, `https://msdn.microsoft.com/library/kwdt6w2k\%28v=VS.71\%29.aspx`.

Distribution and Integration Technologies, Miguel Monteiro, Faculdade de Engenharia da Universidade do Porto, `paginas.fe.up.pt/~apm/TDIN/`.

6.2 *Software*

Visual Studio 2013 Ultimate, Microsoft, `http://www.visualstudio.com/`.

SQLite, `http://www.sqlite.org/`.

SQLiteClient, Community, `http://www.nuget.org/packages/Community.CsharpSqlite.SQLiteClient/`.

NuGet, `http://www.nuget.org/`.

GitHub, `http://github.com/`.