

CIAI-Phase 2

ArtBiz

2017/2018

Miguel Anciães nº 43367

Tiago Santo nº 41658

1. User Stories

**User Storie- Registar Utilizador**

- Um utilizador regista-se no sistema (a partir da landing page),

colocando nome,email, password e o tipo de utilizador

através do formulário fornecido , caso não coloque email,password e tipo de utilizador

vê mensagem de erro, caso contrário, vê a resposta de confirmação de registo na página de registo.

**User Storie- Login Utilizador**

- Um utilizador faz login (a partir da landing page) colocando email e password através do formulário fornecido,

caso não coloque email,password, ou coloque password ou email errados é mostrado uma mensagem de erro,

caso contrário vai para a landing page com indicação de login feito.

**User Storie - Editar Utilizador**

-Um utilizador seleciona o seu perfil(a partir da dashboard),

ve a sua página de perfil, modifica os campos associado do perfil e ve o

resultado na página de perfil.

**User Storie - Ver uma Peça**

**Sucesso**

-Um utilizador , seleciona uma peça (a partir da galeria publica) e ve a página peça.

-Um utilizador registado como autor, seleciona uma peça (a partir da sua galeria) e ve a página da peça.

**Erro: Peça nao existe**

-Um utilizador , seleciona uma peça (a partir da galeria publica) e ve erro de peça nao existente.

-Um utilizador registado como autor, seleciona uma peça (a partir da sua galeria) e ve erro de peça nao existente.

**User Storie - Adicionar Peça**

**Sucesso:**

- Um utilizador registado como autor, adiciona uma peça fornecendo o nome,

data,tecnicas usadas,descricao textual, keywords e conteúdo multimédia

e ve o resultado na sua galeria.

**Erro: Nome obrigatório**

- Um utilizador registado como autor, adiciona uma peça (a partir da dashboard) fornecendo ,

data,tecnicas usadas,descricao textual, keywords e conteúdo multimédia

e ve mensagem de erro de nome obrigatorio e mantêm-se no dashboard

**User Storie - Editar Peça**

**Sucesso:**

- Um utilizador registado como autor, seleciona uma peça para editar (a partir da sua galeria)

e ve a página da peça, modifica os valores associado da peça e ve o resultado das

alterações na página da peça.

- Um utilizador registado como autor, seleciona uma peça (a partir da sua galeria)

e ve a página da peça e seleciona para editar peça, modifica os valores associado da peça e ve o resultado das

alterações na página da peça.

**Erro: Nome obrigatorio**

- Um utilizador registado como autor, seleciona uma peça (a partir da sua galeria)

e ve a página da peça, modifica nome da peça para vazio e ve mensagem de erro de obrigatoriedade

de nome.

**User Storie - Remover Peça**

- Um utilizador registado como autor, seleciona uma peça (a partir da sua galeria)

e ve a página da peça e seleciona para remover peça, confirma a remoção e ve o resultado das

alterações na galeria.

- Um utilizador registado como autor, seleciona uma peça (a partir da sua galeria)

e ve a página da peça e seleciona para remover peça, cancela a remoção e ve o resultado das

alterações na página da peça.

**User Storie - Colocar Peça Venda**

**Sucesso:**

- Um utilizador registado como autor, seleciona uma peça (a partir da sua galeria)

ve a página da peça, seleciona vender peça (a partir da página da peça) ,

colocando um preço através de um formulario fornecido e ve o resultado na página da peça

**Erro: Peça já em venda**

- Um utilizador registado como autor, seleciona uma peça(já no estado de venda) (a partir da sua galeria)

ve a página da peça, seleciona vender peça (a partir da página da peça) e mostra mensagem de erro de peça já em venda.

**Erro: Preço Negativo**

- Um utilizador registado como autor, seleciona uma peça (a partir da sua galeria)

ve a página da peça, seleciona vender peça (a partir da página da peça) ,

colocando um preço negativo através de um formulario fornecido e ve o erro preço negativo.

**User Storie - Fazer bid de uma peça**

**Sucesso:**

- Um utilizador registado ,seleciona uma peça (a partir da galeria publica) (que esteja disponivel para venda),

vai para

a página da peça ,seleciona bid coloca valor e

se pretende ser identificado, através de formulário fornecido, e vê o resultado na página dos bids e informa o autor.

**Erro:Bid de uma peça que nao está em venda**

-Um utilizador registado ,seleciona uma peça (a partir da galeria publica) (que deixou de estar disponivel para venda),

vai para

a página da peça ,seleciona bid e vê mensagem de erro de peça já não está em venda.

**Erro: Bid menor que o preço base**

- Um utilizador registado ,seleciona uma peça (a partir da galeria publica) (que esteja disponivel para venda),

vai para a página da peça ,seleciona bid coloca valor menor que preço base e

se pretende ser identificado, através de formulário fornecido, e vê erro de preço de bid inferior a preço base.

**User Storie - Aceitar Bid**

**Sucesso:**

-Um utilizador registado como autor , seleciona inbox das notificações a (partir da landing page), seleciona notificação de bid,

vai para página da peça correspondente ao bid e aceita o bid , envia notificação ao utilizador que fez o bid e navega para página do bid.

**Erro: Aceitar Bid já aceite**

-Um utilizador registado como autor , seleciona inbox das notificações a (partir da landing page), seleciona notificação de bid,

vai para página da peça correspondente ao bid (já aceite) e aceita o bid , e vê mensagem de erro de bid já foi aceite.

**User Storie - Rejeitar Bid**

**Sucesso:**

-Um utilizador registado como autor , seleciona inbox das notificações a (partir da dashboard), seleciona notificação de bid,

vai para página da peça correspondente ao bid e rejeita bid , envia notificação ao utilizador que fez o bid e navega para página do bid.

**Erro: Rejeitar Bid já rejeitado**

-Um utilizador registado como autor , seleciona inbox das notificações a (partir da dashboard), seleciona notificação de bid,

vai para página da peça correspondente ao bid(já rejeitado) e rejeita bid , e vê mensagem de erro de bid já foi rejeitado

**User Storie - Anular Bid**

**Sucesso:**

-Um utilizador registado , seleciona meus bids a (partir da dashboard),

vai para página dos bids , seleciona o bid da lista de bids, vai para página da peça correspondente ao bid,

confirma a anulação do bid (na página do bid) , envia notificação ao autor e navega para os meus bids.

**Erro: Anular Bid já anulado**

-Um utilizador registado , seleciona meus bids a (partir da dashboard),

vai para página dos bids , seleciona o bid da lista de bids, vai para página da peça correspondente ao bid,

confirma a anulação do bid(já anulado) (na página da peça) , ve mensagem de erro de bid já anulado.

**User Storie - Finalizar Compra**

**Sucesso:**

- Um utilizador registado autor , seleciona meus bids a (partir da dashboard)

seleciona bids pesquisa e seleciona o bid , vai para página do bid e

finaliza compra e seleciona se torna bid publico

e envia notificação ao comprador e navega para pagina do bid.

**Erro: Bid Não aceite**

- Um utilizador registado autor , seleciona meus bids a (partir da dashboard)

seleciona bids pesquisa e seleciona o bid(que deixou de estar aceite) , vai para página do bid e

finaliza compra e seleciona se torna bid publico

e ve mensagem de erro de bid não está aceite.

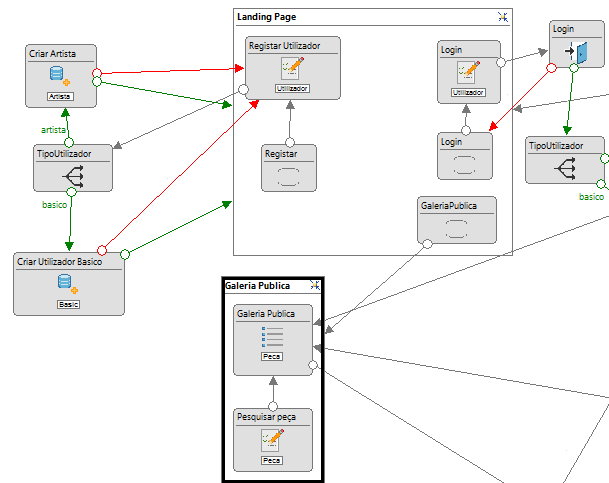
**User Storie - Pesquisar Peça Por Keyword**

-Um utilizador , escreve no input de pesquisa (na galeria publica) e clica no botao pesquisar,

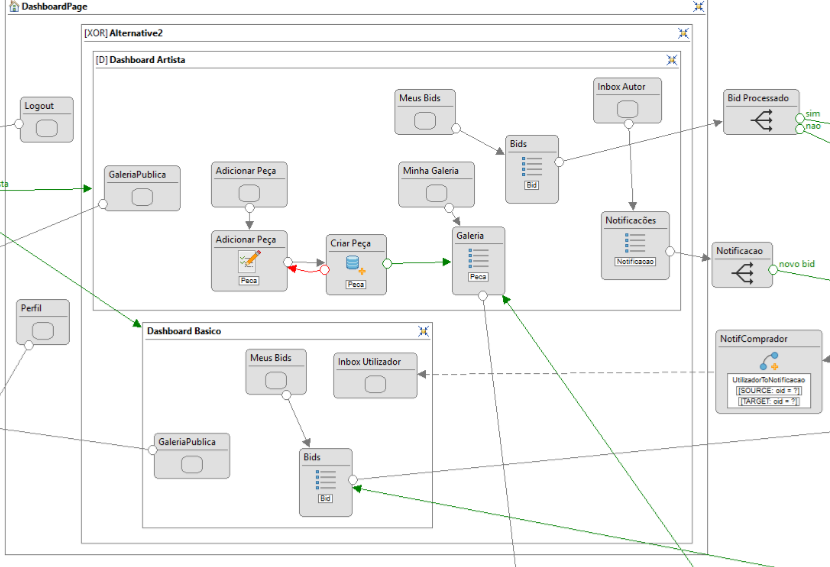
ve as peças relacionadas com as keywords que escritas na página de landing page.

2. IFML

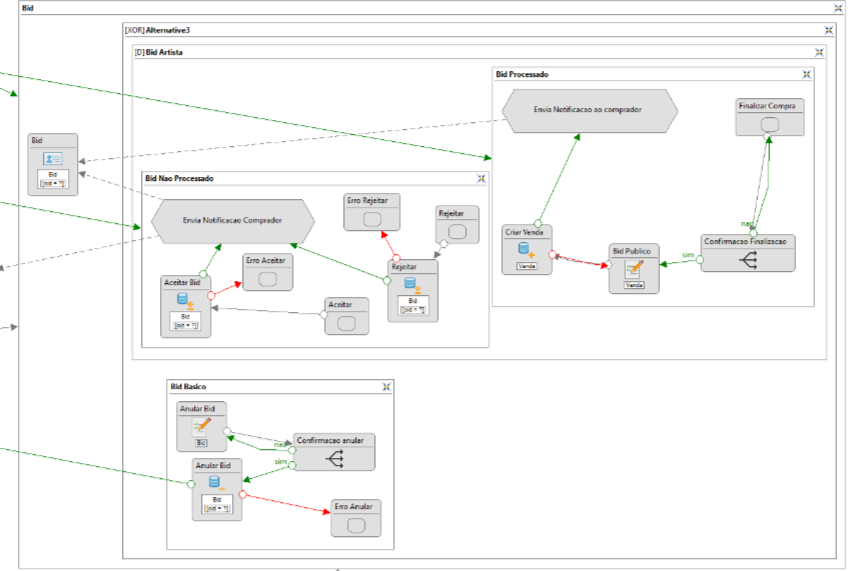
2.1- IFML que representa os user stories relativo ao registo de utilizador e galeria pública.



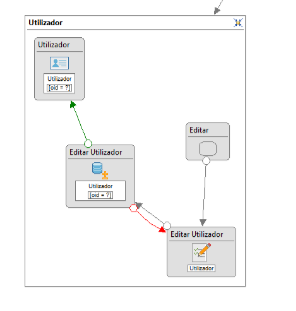
2.2- IFML que representa aos dashboards de cada utilizador.



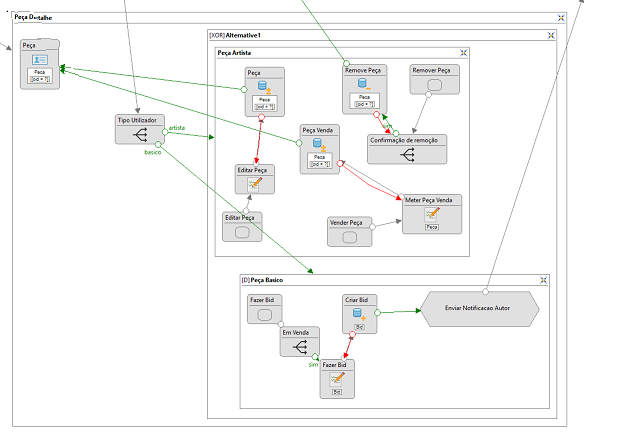
2.3 IFML relativo a user stories do bid



2.4-IFML relativo aos user stories do utilizador



2.5- IFML relative a user stories relative à peça



Na pasta definitions está o ficheiro ifml.png com o esquema IFML completo e na pasta ifml está presente o projeto do ifml.

3.REACT

A implementação em react do ifml está no bitbucket na pasta REACT.

Nesta segunda fase foram implementados todos os componentes relatavos a todas as user stories, utilizando pedidos AJAX para a conexão ao servidor, enviando sempre as credencias. Foi utlizada a biblioteca Sock JS para as notificações.

Para o layout foi utilizado a biblioteca Bootstrap.

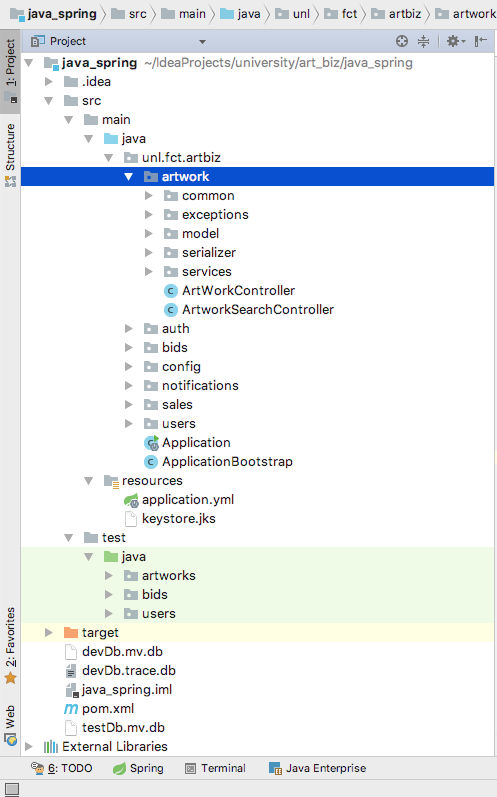
4. Servidor

# 4.1 Framework

A Framework utilizada na implementação na parte do servidor no projeto foi o Spring Framework, mais especificamente, uma componente desta chamada Spring-Boot. Esta Framework facilita a implementação da parte de servidor de uma aplicação e através de componentes consegue implementar todo o espectro de back-end de uma aplicação.

# 4.2 Estrutura do Projeto

Este projeto segue uma estrutura MVC – Model, View, Controller. Esta estrutura é a mais comum aquando do planeamento e construção de uma aplicação para a internet. Permite uma boa distinção entre os modelos, os controladores e as vistas sendo assim uma estrutura de boa compressão por parte de todos.

A imagem seguinte mostra a estrutura do projeto, omitindo algumas pastas para diminuição da imagem.

Na imagem vê-se claramente a distribuição de packages com model e controller, assim como a pasta main e a pasta de testes.

# 4.3 Ambientes

O nosso foi dividido em três ambientes diferentes:

- Ambiente de desenvolvimento,

- Ambiente de Produção,

- Ambiente de testes

Os três ambientes possuem propriedades diferentes e enumeradas num ficheiro global de configuração, tal como a porta onde a aplicação corre, ou mais importante, a diferenciação das tecnologias de armazenamento.

# 4.4 Implementação

Neste projeto foram implementadas todas as funcionalidades básicas pedidas no enunciado e ainda a parte de autenticação, autorização e segurança, assim como a conceção uma base de dados MySQL.

As funcionalidades básicas que implementamos foram:

* Criar Utilizador Básico/Artista
* Editar Utilizador
* Criar Peça
* Editar Peça
* Remover Peça
* Fazer um Bid a uma Peça
* Aceitar/Rejeitar Bid de uma peça
* Anular o Bid de uma Peça
* Finalizar Bid, criando uma venda
* Tornar publica uma venda
* Listagem de vendas
* Listagem de Bids por utilizador
* Pesquisa de Peças por keywords/artista/técnicas/nome/...
* Notificações aquando da alteração do estado de uma Bid (aceite/rejeite/...)
* Navegação para os detalhes de uma peça, artista, venda, bid

Foi feito o uso de todas as capacidades que a Framework Spring-Boot tem para oferecer.

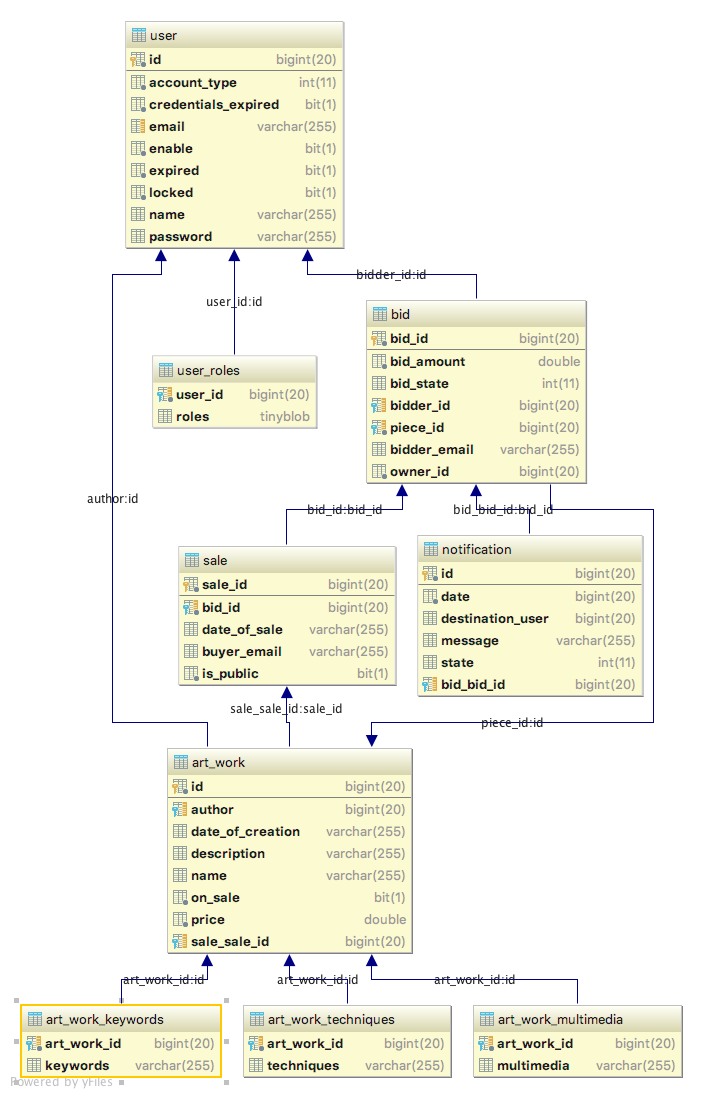
# 4.5 Armazenamento

Nos diferentes tipos de ambientes utilizamos diferentes tipos de tecnologia de armazenamento. No ambiente de desenvolvimento e testes utilizamos a base de dados em memória H2 com um bootstrap inicial. Esta base de dados ficava guardada num ficheiro e é apagada quando o servidor se desliga. Aquando a nova inicialização, o Spring fará a escrita de utilizadores e peças pré-definidos apenas para testes. Esta técnica permite testar a fundo as funcionalidades sem corromper a base de dados de forma irreversível.

No ambiente de produção, foi implementado uma conexão a uma base de dados MySQL para armazenamento persistente. Este base de dados corre localmente na porta 3306.

Fazendo uso ás capacidades do Spring Crud Repository, a migração de base de dados é trivial, mudando-se apenas o ficheiro de configuração apontando apenas ao Spring o driver a utilizar em cada caso.

A imagem seguinte é o UML, diagrama da base de dados:



# 4.6 Segurança

Foi implementado um mecanismo de segurança neste projeto. É um modelo de Role Based Access Control misturado com um modelo de Data Access Control. Os utilizadores têm diferentes papeis no sistema que o autorizam a fazer certas operações. Foi implementado também, através do uso de anotações customizadas que os utilizadores apenas podem aceder as suas propriedades sendo mostrado um erro 403 quando, por exemplo o utilizador 1 tenta fazer operações como utilizador 2.

O login é feito através de um sistema que acede a base de dados e compara as passwords. Estas passwords são guardadas de forma encriptada para o aumento da segurança.

# 4.7 Notificações

Para a funcionalidade de notificações, foi usada a tecnologia Web Sockets. Esta técnica omite a necessidade de pedidos AJAX contínuos para saber as notificações. Quando estas aparecem, é enviado por web socket, estando o utilizador subscrito a essa web socket.

# 4.8 SSL

A aplicação, em ambiente de produção, está a funcionar utilizando SSL/HTTPS com certificados assinados pelo próprio (self-signed certificates), na porta 8443.

# 4.8 Testes

Para cada funcionalidade foi implementado um teste em JUnit. Estes testes permitem confirmar que após algumas alterações de código este se conforma como esperado. No entanto esta funcionalidade deixou de funcionar aquando da implementação dos mecanismos de segurança, não sendo conseguido autenticar um utilizador para os testes. Não sendo esta uma funcionalidade requerida, o grupo não perdeu muito tempo na tentativa de so arranjar.