



# RELATÓRIO

# TRABALHO FINAL CURSO

## WEB-BASED PORTAL STORE

## ÍNDICE

LISTA DE IMAGENS .....	2
1. RESUMO .....	3
2. ABSTRACT .....	4
3. ENQUADRAMENTO .....	5
3.1. A ESCOLHA DO PROJECTO.....	5
3.2. O PROJECTO NO ÂMBITO ACADÉMICO .....	5
3.3. OBJECTIVO DO DOCUMENTO.....	5
4. ACRÓNIMOS .....	6
5. RUBY ON RAILS .....	7
5.1. RAILS É AGILE.....	7
5.2. DRY .....	7
5.3. MODELS, VIEWS E CONTROLLERS .....	8
6. AGILE.....	9
7. GESTÃO DE VERSÕES .....	10
8. CLOUD COMPUTING .....	12
9. CASO DE USO SIMPLIFICADO .....	13
10. MODELO DE DADOS .....	14
11. DECISÕES TOMADAS NO DESENVOLVIMENTO .....	15
11.1. DESENHO DA PÁGINA .....	15
11.2. REGISTO DE UTILIZADORES .....	15
11.3. MÉTODO DE PAGAMENTO (PAYPAL).....	15
12. IMAGEM DO PROGRAMA .....	16
12.1. PÁGINA DE LOGIN.....	16
12.2. PÁGINA DE REGISTO.....	16
12.3. PÁGINA PRINCIPAL.....	17
12.4. DETALHE DO PRODUTO .....	17
12.5. SACO DE COMPRAS.....	18
13. ESTATÍSTICAS DESENVOLVIMENTO.....	19
14. MELHORIAS FUTURAS.....	20
15. CONCLUSÃO .....	21

## Lista de Imagens

Imagem 5-1: Arquitectura modelo MVC.....	8
Imagem 7-1: Gestão de Versões: Histórico e detalhe de alterações (local) .....	10
Imagem 7-2: Gestão de Versões: Histórico e detalhe de alterações (remoto).....	11
Imagem 9-1: Caso de uso simplificado .....	13
Imagem 10-1: Modelo de dados .....	14

## 1. RESUMO

Sendo o projecto efectuado num contexto académico foi estabelecido como objectivo principal a aquisição de novos conhecimentos e competências.

Para a criação de uma página de comércio electrónico decidi utilizar uma framework MVC e escolhi o *Ruby on Rails* como linguagem de programação.

Foi utilizado um modelo de desenvolvimento *Agile* com suporte de um gestor de versões.

Foi ainda objectivo a utilização de técnicas de *Cloud Computing*, essencialmente a componente de *Storage* oferecida pela Amazon.

Palavras chave: Agile, Controle de Versões, Cloud Computing, MVC.

## 2. ABSTRACT

Being the project realized in an academic context, the main goal will be the acquisition of new knowledge and competences.

To create an e-commerce platform I decided to use an MVC framework and the choice had been for Ruby on Rails.

The method used to the development of the project was the Agile, and the development was supported with a Configuration Management tool named Git.

It was also objective the using of Cloud Computing servers, essentially the storage component offered by Amazon.

Keywords: Agile, Configuration Management, Cloud Computing, MVC

### 3. ENQUADRAMENTO

#### 3.1.A ESCOLHA DO PROJECTO

Antes de iniciar a minha actividade académica tive a ideia e vontade de criar um projecto na área do comércio electrónico. A ideia tinha como base a criação de um portal onde criadores poderiam vender produtos criados à mão (bijutarias, artigos de decoração, entre outros) essencialmente para um público-alvo do sexo feminino.

Durante o meu percurso académico tomei conhecimento de um projecto semelhante que nasceu nos Estados Unidos da América – o Etsy<sup>1</sup>. O Etsy tornou-se o maior portal de vendas de produtos feitos há mão e uma referência internacional. Entretanto têm aparecido alguns projectos semelhantes a nível Europeu, mais precisamente, em Inglaterra e Alemanha.

Tendo sido a entrada na universidade uma das causas para ter deixado o projecto na gaveta, optei por dar início ao mesmo no fim deste percurso.

#### 3.2.O PROJECTO NO ÂMBITO ACADÉMICO

Quando iniciei o meu percurso académico o objectivo principal foi a aquisição de novos conhecimentos, e a abordagem a este projecto foi com o mesmo objectivo em mente.

Decidi utilizar para o desenvolvimento do projecto a linguagem *Ruby on Rails*<sup>2</sup> que desconhecia até então.

Foi aplicada uma metodologia Agile. A fase de desenvolvimento foi feita com suporte de uma ferramenta de gestão de configurações- *git*<sup>3</sup>.

#### 3.3.OBJECTIVO DO DOCUMENTO

O objectivo do documento é descrever as fases, os processos e as decisões que foram tomadas no decorrer da implementação do projecto. Todas as explicações serão acompanhadas sempre que necessário com imagens, tabelas ou modelação.

---

<sup>1</sup> <http://www.etsy.com>

<sup>2</sup> <http://www.rubyonrails.com>

<sup>3</sup> <http://git-scm.com/>

## 4. ACRÓNIMOS

São utilizados os seguintes acrónimos ao longo do documento e foram identificados como necessários a sua descrição, para que o entendimento ao longo do documento seja claro.

Acronym	Definition
CRUD	Create, Read, Update and Delete
DRY	Don't Repeat Yourself
MVC	Model, View, Controller
SEO	Search Engine Optimization

## 5. RUBY ON RAILS

### 5.1. RAILS É AGILE

A agilidade é parte do Ruby on Rails.

Se olharmos para os valores expresso no *Agile Manifesto*<sup>4</sup> como um conjunto de 4 preferências, o desenvolvimento agile valoriza o seguinte:

- Indivíduos e interações – mais do que processos e ferramentas
- Software funcional – mais do que documentação abrangente
- Colaboração com o cliente – mais do que negociação contratual
- Responder à mudança – mais do que seguir um plano

O Ruby on Rails tem tudo a ver com Indivíduos e interações. Não existem ferramentas pesadas, configurações complexas ou processos elaborados. Existe apenas um grupo de programadores que através de um processo iterativo produzem um produto visível e utilizável pelo cliente.

O processo de desenvolvimento de um projecto em Ruby on Rails não é conduzido por documentação extensa mas sim por um grupo composto por clientes e programadores que trabalham juntos nas necessidades e no modo para responder às mesmas. Rails encoraja fortemente a colaboração com o cliente, porque permite ao cliente perceber o quão tão rapidamente o Rails pode responder a alterações e ao mesmo tempo elevar a sua confiança de que o que vai ser entregue no final é exactamente aquilo que foi pedido.

### 5.2. DRY

*DRY – Don't repeat yourself*, ou em português: Não se repita. É um conceito intrínseco no *Ruby on Rails*. Com este modelo, evitamos a repetição de código por toda a aplicação. Para reaproveitamento de código existem os *helpers* e os *partials* que ao serem utilizados reduzem a redundância.

Um *helper* contem funções para auxiliar a camada de visualização da aplicação retirando lógica complexa do código de apresentação (ex: formatação de datas, moedas, etc).

Um *partial* são páginas parciais da camada de apresentação e que normalmente são usadas em mais do que um local.

---

<sup>4</sup> <http://www.agilemanifesto.org>

### 5.3. MODELS, VIEWS E CONTROLLERS

Tem como função isolar as regras de negócio da interface exibida ao utilizador.

O **Model** além de tratar dados, tem também responsabilidade no que toca a restrições do negócio. Se pretendemos dar um desconto apenas a compras superiores a €20, o *model* irá forçar esta regra. Este tipo de restrição tem lógica existir no *Model*, porque garante que não existe nada na nossa aplicação que crie dados errados.

A **View** é responsável para gerar a interface com o utilizador e é em grande parte composta por dados que vêm do *Model*. Numa loja online temos uma lista de produtos que estão armazenados num modelo de dados e que é acedida pelo *model*, mas é a *view* que tem responsabilidade de formatar os dados e mostrá-lo para o utilizador.

O **Controller** orquestra a aplicação. Os *controllers* recebem eventos do exterior (normalmente *inputs*), interage com o *model*, e mostra a *view* apropriada ao utilizador.

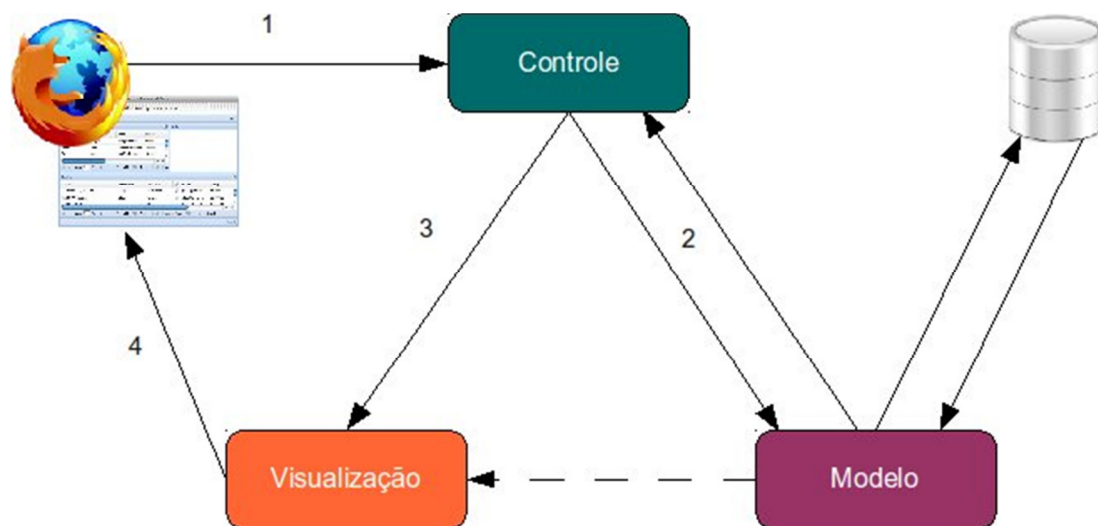


Imagem 5-1: Arquitectura modelo MVC



## 6. AGILE

A maioria dos métodos ágeis tenta minimizar o risco inerente num projecto de desenvolvimento de software através de iterações. Cada iteração é como um mini projecto de software onde se inclui todas as tarefas necessárias para implementar o incremento de funcionalidade: planeamento, requisitos, codificação, teste e documentação.

Enquanto num processo convencional de desenvolvimento, cada iteração não está necessariamente associada à implementação de uma nova funcionalidade, num projecto de software ágil, a cada iteração procura-se obter uma nova versão do software com a implementação de novas funcionalidades.

Os métodos ágeis enfatizam a comunicação com o cliente ao invés de documentação escrita e extensa. A documentação de um projecto num método ágil é mínima comparada com outros métodos.

Entre os objectivos da metodologia ágil reforço os seguintes:

- Satisfazer o cliente através de um processo de entregas rápidas e contínuas de software com valor;
- Acatar alterações de requisitos, mesmo em estados avançados de desenvolvimento;
- Efectuar entregas de software funcional com frequência, com intervalos de poucas semanas ou poucos meses, com preferência para a escala temporal mais curta;
- Comerciais e equipa de desenvolvimento têm de trabalhar em conjunto diariamente ao longo do projecto;
- Software funcional é a medida primária de progresso;

Apesar de não ter um cliente final com que pudesse avaliar a implementação do projecto, tentei criar um processo iterativo no que toca a funcionalidades.

## 7. GESTÃO DE VERSÕES

Num processo de desenvolvimento de software é essencial o versionamento de versões, tanto no ponto de vista da segurança mas também no que diz respeito à partilha de *baselines* com outros membros da equipa.

Apesar do projecto ter sido feito individualmente e não ter que partilhar código com outros membros da equipa, a sua utilização foi importante não só para isolar iterações em curso (*branching*) daqueles que estavam maduros (*trunk*), mas também para partilhar código entre diversos computadores.

O projecto que foi entregue em formato digital não é apenas a versão final. O projecto tem todo o histórico de desenvolvimento. Para sua visualização é necessário o download da ferramenta GIT.

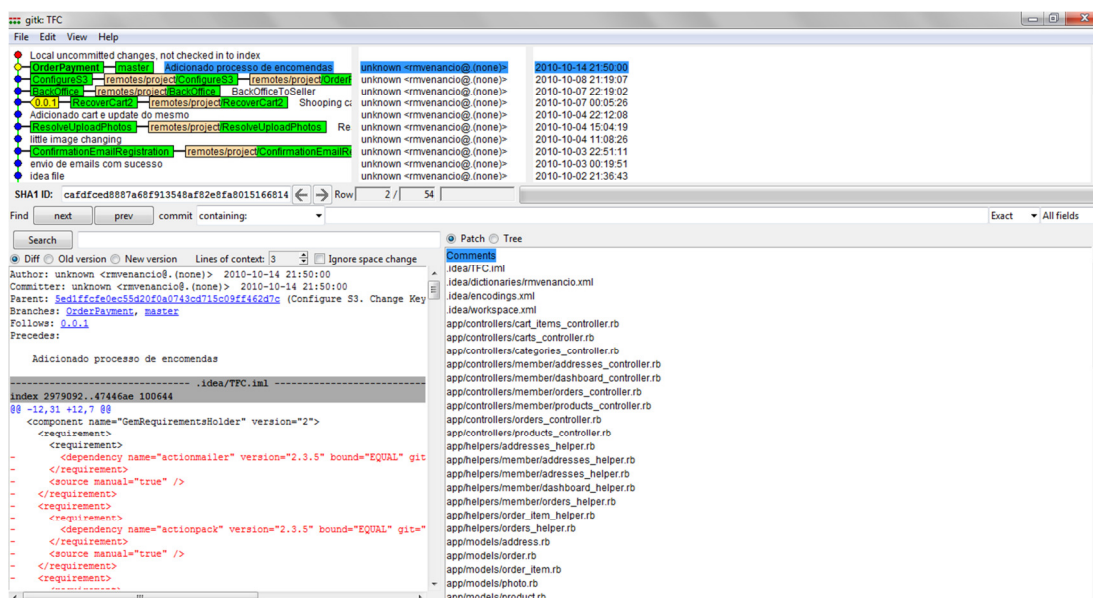


Imagem 7-1: Gestão de Versões: Histórico e detalhe de alterações (local)

Visto que existiu uma fase do desenvolvimento onde foram utilizados vários computadores, existiu a necessidade da criação de um repositório central para armazenamento do código. O objectivo era que fosse mantida a coerência entre as fontes e que pudesse sempre trabalhar com base nas últimas versões independente da máquina que utilizasse.

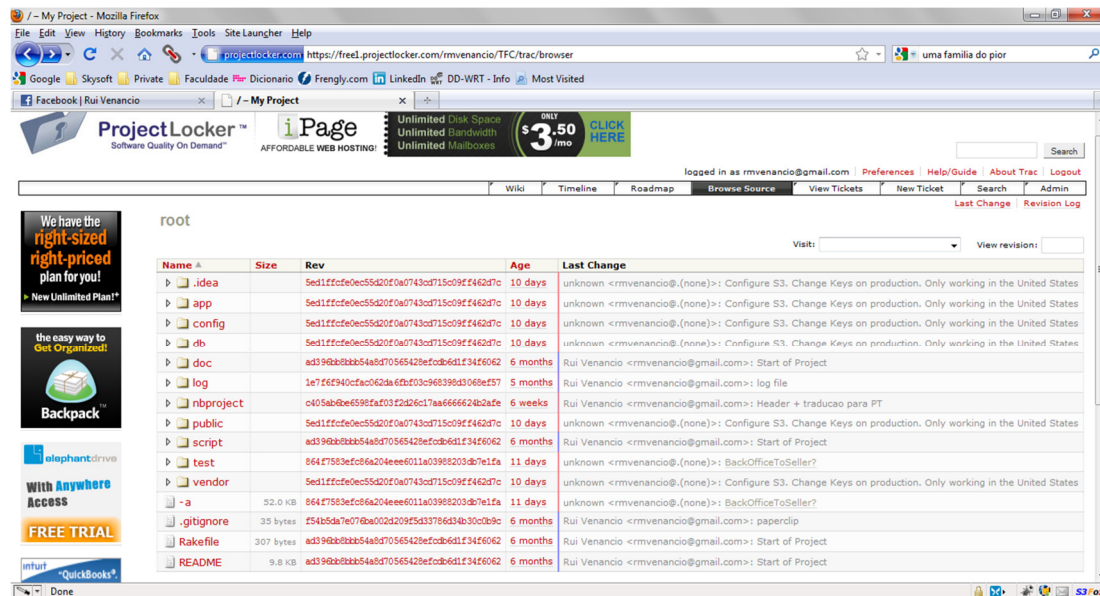


Imagem 7-2: Gestão de Versões: Histórico e detalhe de alterações (remoto)

Durante várias situações na fase de codificação foi necessário regredir a fases anteriores de desenvolvimento. A existência de um repositório de gestão de versões foi essencial.

## 8. CLOUD COMPUTING

Cada produto pode ter várias fotografias associadas. As fotos ao serem adicionadas ao produto são alvo de tratamento e são guardadas em filesystem em 4 tamanhos distintos:

75x75 – Tamanho pequeno

170x135 – Tamanho dos thumbnails

430x323 – Tamanho grande

\* x \* - Tamanho Original da foto

Estou a assumir que um utilizador não sabe como fazer o redimensionamento de uma fotografia no computador, por isso estou a aceitar que seja feito o upload de qualquer imagem até 5Mb e a mesma é guardada no seu formato original.

Com o crescimento de produtos e do volume de fotografias, é difícil prever qual o volume mensal de fotografias adicionadas assim como o volume de transferências (*bandwidth*).

Optei por alojar todas as imagens no serviço de Storage (S3<sup>5</sup>) da Amazon. Com esta opção, na altura que o sistema for disponibilizado ao público não preciso de me preocupar com o crescimento do volume de dados físicos.

O Ruby on Rails disponibiliza alguns plugins que facilita a utilização do storage da Amazon.

---

<sup>5</sup> <http://aws.amazon.com/s3/>

## 9. CASO DE USO SIMPLIFICADO

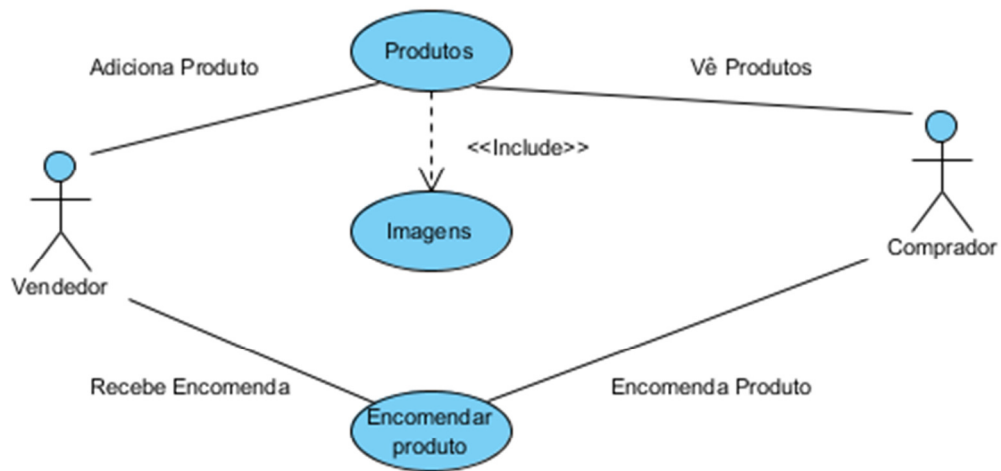


Imagem 9-1: Caso de uso simplificado

## 10. MODELO DE DADOS

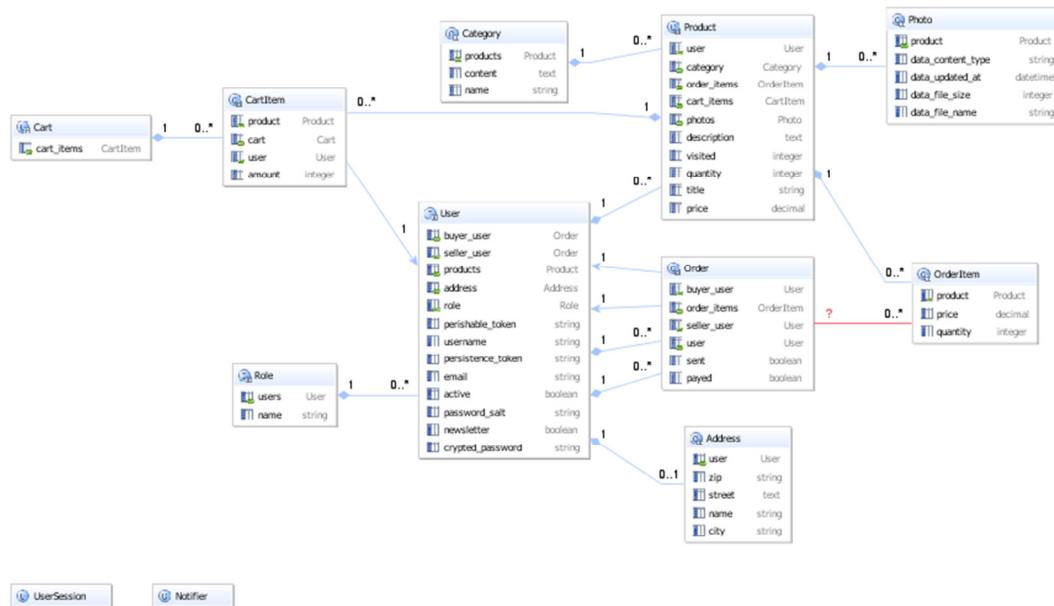


Imagem 10-1: Modelo de dados

## 11. DECISÕES TOMADAS NO DESENVOLVIMENTO

### 11.1. DESENHO DA PÁGINA

O desenho da página recorreu a algumas técnicas de SEO.

Uma tarefa dos *spiders* dos motores de pesquisa é percorrer páginas de internet para indexação de conteúdo.

Um dos erros comuns é o desenho de páginas recorrendo a tabelas. O problema é quando existe uma definição hierárquica de tabelas dentro de tabelas, os spiders acabam por desistir de percorrer as páginas por se tornarem demasiado pesadas e por consequência a página não é indexada/catalogada.

Sendo um site de comércio electrónico, é essencial a sua indexação. Para isso e conforme aconselhado por normas de SEO todo o design recorreu a *divs* e *css* de modo a organizar conteúdos.

### 11.2. REGISTO DE UTILIZADORES

Quando um utilizador se regista na aplicação, terá que escolher se é vendedor ou comprador. Dependendo do tipo de *role* o que será apresentado na componente de backoffice será diferente, nomeadamente a parte de gestão de produtos (só o vendedor poderá adicionar produtos).

De modo a evitar registos por robots, é enviado um email de confirmação ao utilizador.

As passwords estão encriptadas na base de dados.

### 11.3. MÉTODO DE PAGAMENTO (PAYPAL)

Para pagamento de compras recorri ao *paypal*. Contudo e devido a limitação de tempo, recorri a opção de integração mais fácil de implementar que é também a menos segura. Após pagamento da encomenda, o utilizador retorna à *root* da aplicação. O indicado seria marcar a compra como paga, todavia isto não ocorre. Terá de ser o vendedor a confirmar que recebeu o pagamento no backoffice.

## 12. IMAGEM DO PROGRAMA

### 12.1. PÁGINA DE LOGIN

The screenshot shows the login page of the Yuzbe website. The header features the 'yuzbe' logo on the left and 'Entrar' and 'Registrar' links on the right. A navigation menu includes 'Home', 'Acessórios', 'Arte', 'Bijuteria', 'Casa', 'Casamento', and 'Criança'. The main content area is titled 'Entrar no Yuzbe' and contains three sections: 'Já é membro ? Entrar' with fields for 'Nome do Utilizador' and 'Password' and an 'Entrar' button; 'Ainda não é membro?' with explanatory text and a 'Registrar' button; and 'Yuzbe no Social' with Facebook and Twitter icons.

### 12.2. PÁGINA DE REGISTO

The screenshot shows the registration page of the Yuzbe website. The header and navigation menu are identical to the login page. The main content area is titled 'Registrar no Yuzbe' and contains a registration form with the following fields and instructions:

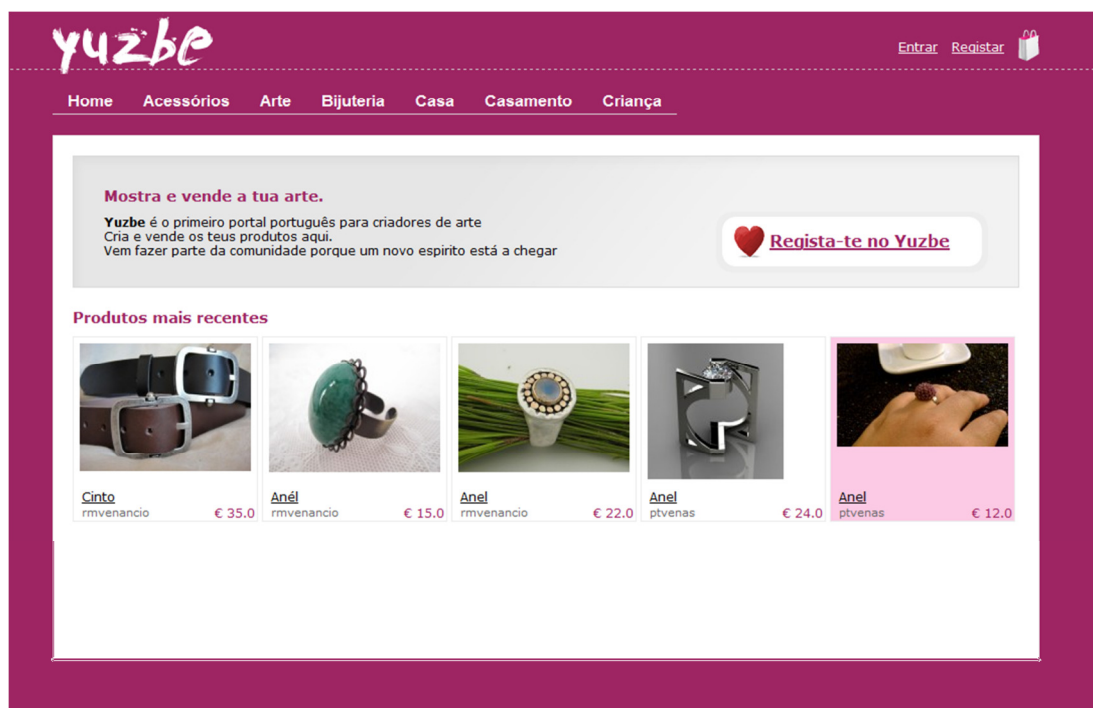
- Nome do Utilizador**: Text input field.
- Password**: Text input field.
- Confirmação Password**: Text input field.
- Email**: Text input field.
- Quero Comprar** (selected) / **Quero Vender**: Radio buttons.
- ☒ **Sim, quero receber a newsletter do yuzbe**: Check box.
- Registrar**: Submit button.

Instructions for the form:

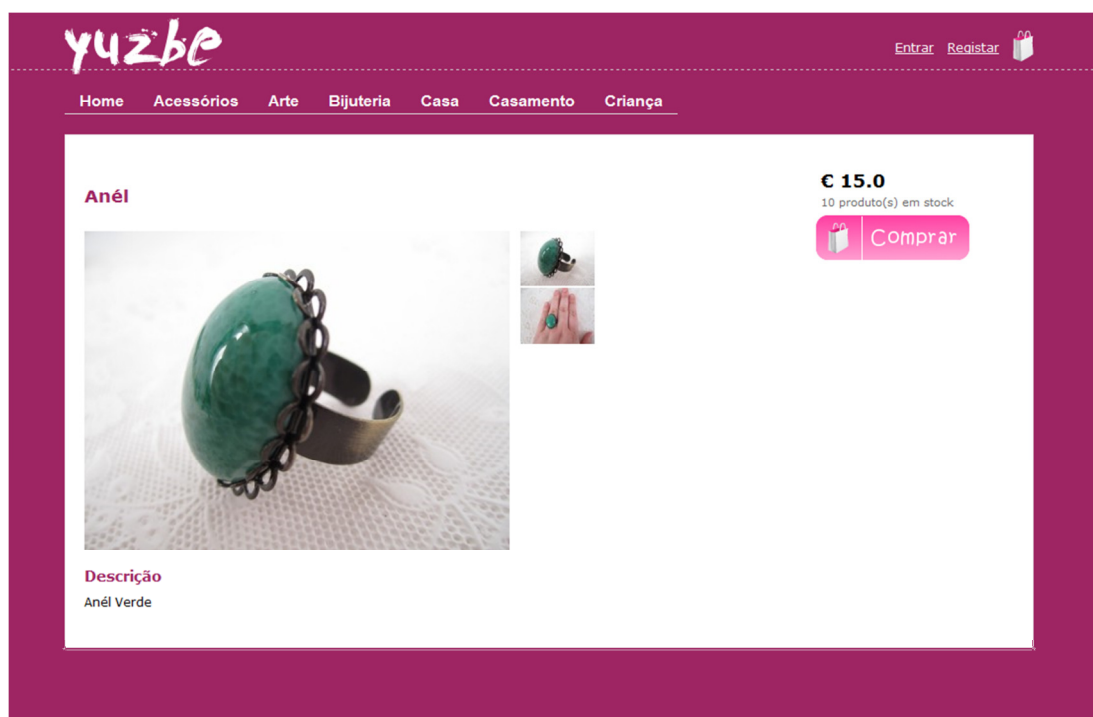
- O nome de utilizador deverá ser composto apenas com caracteres alfa-numéricos (e sem acentuação)- ex: ola123;
- O nome de utilizador é único no Yuzbe;
- O nome do utilizador não pode ser alterado;
- Será atribuído a todos os utilizadores um endereço próprio- [http://www.yuzbe.com/nome\\_de\\_utilizador](http://www.yuzbe.com/nome_de_utilizador).
- O tamanho mínimo da password é de 6 caracteres;
- Nunca pediremos a sua password por email ou outro meio;
- As password faz distinção entre maiúsculas e minúsculas- ex: praia é diferente de PRAIA e de Praia.
- Vai ser enviado um email para este endereço com um código de confirmação;
- Caso não receba o email, verifique a pasta de spam. Adicione o email [registration@yuzbe.com](mailto:registration@yuzbe.com) como seguro;
- O seu email será privado e não será fornecido a nenhuma entidade externa;
- Em qualquer altura poderá mudar o perfil para poder vender no yuzbe.





### 12.3. PÁGINA PRINCIPAL



### 12.4. DETALHE DO PRODUTO





## 12.5. SACO DE COMPRAS


[Entrar](#) [Registar](#) 

Home Acessórios Arte Bijuteria Casa Casamento Criança



O produto foi adicionado ao carrinho de compras.

**Saco de Compras**

 Finalizar Compra  Continuar a comprar

**ptvenas**  
 Anel  
Preço: 12.0  
Quantidade : 1  
Preço Total: 12.0  
X

Sub-Total: € 12.0

**rmvenancio**  
 Anel  
Preço: 15.0  
Quantidade : 1  
Preço Total: 15.0  
X  
 Anel  
Preço: 22.0  
Quantidade : 1  
Preço Total: 22.0  
X

Sub-Total: € 37.0

**Informação**

- No yuzbe as suas compras são feitas directamente ao vendedor.
- Os produtos estão ordenados por vendedor.
- O pagamento é feito para cada vendedor existente na sua encomenda.
- Para comprar, terá que usar obrigatoriamente o [paypal](#).
- Para confirmar a compra e seguir para o próximo passo, tem de ter o [login](#) feito.

### 13. ESTATÍSTICAS DESENVOLVIMENTO

Name	Lines	LOC	Classes	Methods	M/C	LOC/M
Controllers	620	476	22	66	3	5
Helpers	97	82	0	6	0	11
Models	152	110	12	8	0	11
Functional tests	458	366	25	36	1	8
Unit tests	173	132	33	3	0	42
Total	1500	1166	92	119	1	7
Code LOC: 668    Test LOC: 498    Code to Test Ratio: 1:0.7						

## 14. MELHORIAS FUTURAS

Antes de efectuar o *deploy* para um servidor definitivo, a parte de pagamentos do paypal tem de ser revisto.

Tudo o resto vai ficar inalterado. Futuras alterações e evoluções dependem do feedback de utilizadores reais.

Contudo é de prever que seja possível ver os produtos por cada utilizador e a criação de RSS por utilizador e global.

## 15. CONCLUSÃO

Tenho bastante experiência de desenvolvimento em plataformas Microsoft .NET. Inclusive poderia ter jogado pelo seguro e implementado o projecto com esta tecnologia, mas quis arriscar e fazer algo que me trouxesse mais valor acrescentado e escolhi uma linguagem algo que até ao momento era-me totalmente desconhecida.

Para quem vem do mundo .NET como eu, a framework mvc impõe alguma abstracção no que toca aos conhecimentos até então adquiridos. O mvc é mais puro no que toca a invocar POST, GET, PUT, enquanto no .NET tudo isso está encapsulado com os métodos e objectos criados. Esta foi a primeira dificuldade que tive que ultrapassar.

O Ruby tem uma linguagem que na minha opinião é mais fácil de entender e de ler e isso trás vantagens a nível de manutenção de código. Contudo, e pela linguagem ser diferente de todas as outras é moroso começar a produzir código.

O Rails foi uma surpresa no que toca ao processo de *scaffolding*, e à relação de entidades de dados. Ao construir a aplicação, as relações que invocamos nos objectos de dados é muito mais natural. A quantidade de código necessário para colocar algo a correr é muito reduzido.

O uso da componente de storage da amazon é uma boa solução para armazenamento de ficheiros com forte tendência de crescimento. Ao utilizar esta componente, apenas teremos que nos preocupar com a componente de alojamento e de base de dados.

No que toca a metodologia Agile, este trabalho levou-me a querer saber um pouco mais sobre a mesma. Após finalização deste trabalho pretendo alargar os meus conhecimentos nesta metodologia, mais concretamente em *Scrum* e *Extreme Programming (XP)*.

Em tom de conclusão, este foi um trabalho que me deu gosto fazer e não serviu apenas para cumprir currículo. É um produto que vai estar disponível para o público em geral e espero que tenha um bom crescimento e sucesso.

O trabalho final de curso não serviu apenas para cimentar alguns conhecimentos obtidos ao longo do curso, mas serviu essencialmente para a aquisição de novos conhecimentos.