



Universidade do Minho

Conselho de Cursos de Engenharia

Licenciatura em Engenharia Informática

3ºAno

Disciplina de Laboratórios de Informática IV

Ano Lectivo de 2007/2008

ANUNK – Anúncios Online Georreferenciados

Agostinho Silva 38176

Miguel Regedor 43117

Junho, 2008

Data de Recepção	
Responsável	
Avaliação	
Observações	

ANUNK

Agostinho Silva 38176

Miguel Regedor 43117

Junho, 2008

Dedicamos este trabalho a todos os intervenientes no projecto ANUNK.

Resumo

Com o desenvolvimento da Web e crescente acesso à mesma, esta tornou-se num importante meio de divulgação de diversos conteúdos que de outra forma seriam inacessíveis.

O objectivo principal deste projecto é a criação de um portal de anúncios que aglomerasse a informação relativa a ofertas de venda e aluguer de imóveis, empregos, serviços, etc. Para benefício dos utilizadores a informação é georreferenciada facilitando localização das ofertas, mesmo em locais de acesso difícil ou desorganizados em termos de ruas tornando-se mais fácil a sua localização.

Neste contexto aclarou-se a ideia de numa fase inicial este projecto ser aplicado na região de Luanda devido ao seu estado actual de desorganização, que contrastam com um número crescente de pessoas a acederem à internet à procura de casa, emprego, etc., mas sem nunca esquecer que é um projecto com um âmbito global.

Área de Aplicação: Aplicação Web, Site de Anúncios Georreferenciados.

Palavras-Chave: Ruby, Ruby on Rails, JavaScript, Bases de Dados Relacionais, Google API's, AJAX, MySQL, Apache.

Índice

Resumo	i
Índice	ii
Índice de Figuras	iii
Índice de Tabelas	iv
1 Introdução	1
1.1 Contextualização	1
1.2 Apresentação do Caso de Estudo	1
1.3 Motivação e Objectivos	2
1.4 Estrutura do Relatório	2
2 Requisitos e planeamento do projecto	3
2.1 Análise de requisitos	3
2.2 Tecnologias e Ferramentas Utilizadas	4
2.2.1 Ruby on Rails	4
2.2.2 Padrão MVC	4
2.2.3 JavaScript	5
2.2.4 Apache	5
2.2.5 Outras Tecnologias e Ferramentas Utilizadas	6
2.3 Planeamento do projecto	6
2.3.1 Intervenientes no Sistema	7
Anexos	Erro!
Marcador não definido.	
Anexos	
I. Anexo 1	Erro!
Marcador não definido.	

Índice de Figuras

Figura 1 - Ilustração de inserção de uma figura e legenda.

Erro!

Marcador não definido.

Índice de Tabelas

Tabela 1 - Ilustração de inserção de uma tabela e sua legenda.

Erro!

Marcador não definido.

1 Introdução

1.1 Contextualização

A internet é um meio de comunicação ao dispor de um número crescente de pessoas, portanto é um meio apetecível para a difusão pública de diversos conteúdos, desde propaganda comercial até notícias ou informações de vários âmbitos. Devido ao alcance global, a Web, disponibiliza informações sobre qualquer parte do mundo à distância de um clique, sem necessidades de infra-estruturas muito complexas e em tempo útil.

Aproveitando estas vantagens da Web surge a ideia de criar um portal de anúncios que agrupasse várias categorias como imóveis, ofertas de emprego e serviços, venda de produtos vários, etc. Como o acesso aos *média* locais por vezes é difícil, sobretudo para quem se encontra distante, é desta forma que surge a concepção de uma plataforma que aglutine estes conteúdos num só portal facilitando a divulgação e a pesquisa deste tipo de informações.

O contexto de aplicação inicial seria a cidade de Luanda, pois é uma cidade bastante destruída e desorganizada, que conta com 4.5 milhões de habitantes, mas apesar destas dificuldades a cidade encontra-se em grande crescimento económico que leva a uma grande procura de informação sobre empregos, imóveis para aluguer e venda, entre outros produtos comercializáveis e serviços. Ou seja, esta é uma boa base para a expansão do projecto ANUNK dado que se pretende que esta plataforma seja uma montra global de anúncios.

1.2 Apresentação do Caso de Estudo

O projecto ANUNK nasce da ideia de agregar em uma só plataforma anúncios de vários tipos, tornando-os globalmente visíveis. Assim, esta plataforma tenta chegar ao maior número possível de anunciantes e compradores dado que não utilizar os normais meios de comunicação, como jornais, flyers e outros, mas usa a Web como suporte.

No ANUNK poderão ser inseridos diversas categorias de anúncios. Como exemplo teríamos os anúncios de imóveis, empregos, serviços e bens, que por norma são mais úteis quando estão associados a uma dada localização. Como tal, surge a necessidade de implementar uma funcionalidade que georreferencie os anúncios de forma a melhorar a plataforma ANUNK.

Outra necessidade numa plataforma deste tipo é a criação de ferramentas de pesquisa que auxiliem o utilizador na navegação e busca de anúncios do seu agrado.

A partir destas premissas iniciais estamos prontos para criar o nosso projecto, ao qual poderão ser adicionadas funcionalidades mediante necessidades que surjam.

1.3 Motivação e Objectivos

A principal motivação deste projecto foi criar uma plataforma com mais-valias como diversidade de produtos e bens anunciados e georreferenciados, acessível globalmente em tempo útil. No fundo, criar algo útil para a sociedade em que vivemos.

O objectivo do ANUNK é criar um portal com vários tipos de utilizadores: os anunciantes que após a criação de um registo podem colocar anúncios classificados online; os utilizadores comuns que vão em busca de oportunidades de negócio ou emprego; o administrador que faz uma gestão dos conteúdos divulgados.

Os anunciantes ao colocarem os anúncios terão que lhe atribuir um título, inseri-lo numa das categorias pelas quais o site distribui os vários anúncios e por fim associar o produto anunciado a uma localização.

A localização é feita recorrendo aos mapas do maps.google.com, que com bastante detalhe mostram a vista de satélite do local apontado pelo anúncio e, mesmo em locais desordenados como Luanda, não depende do nome de ruas para indicar o local.

1.4 Estrutura do Relatório

No capítulo 2.1 analisamos de uma forma global o temos de fazer que tecnologias usar.

No capítulo 2.2 explicamos melhor essas tecnologias.

Nos capítulos 2.3.... apresentamos uma solução para o problema.

No capítulo 3 damos um exemplo de utilização.

No capítulo 4 problemas ao longo da implementação.

E finalmente, no capítulo 5 uma conclusão.

2 Requisitos e planeamento do projecto

2.1 Análise de requisitos

Para a criação de um portal contendo anúncios georreferenciados através do Google Maps são necessárias ferramentas capazes de suportar estas funcionalidades e conhecimentos de como implementar as funcionalidades correctamente.

O site tem vários requisitos, sendo os mais importantes:

- Registo de novos utilizadores;
- Acessos seguros para os utilizadores, com login e palavra-chave;
- Permitir a inserção, remoção e edição de anúncios pelos utilizadores correspondentes;
- Filtrar anúncios por categoria;

Quanto aos dados, tem de estar guardados numa base de dados que esteja modelada de forma a receber os conteúdos inseridos pelos utilizadores do site, tanto no que respeita aos anúncios inseridos como nos dados dos vários utilizadores.

A georreferenciação recorrendo ao Google Maps é possível dada a existência de uma API da Google para utilização dos mapas e mais tarde decidir-se ia qual a tecnologia a usar para interligar os mapas com o nosso site.

Por fim, após a aplicação estar pronta a funcionar será necessário configurar o servidor no qual todas as tecnologias vão funcionar de forma aos clientes diversos clientes do sistema lhe poderem aceder.

2.2 Tecnologias e Ferramentas Utilizadas

2.2.1 Ruby on Rails

Ruby on Rails é um meta-framework gratuito que facilita o desenvolvimento de sites orientados a bases de dados, visto que é possível criar aplicações com base em estruturas pré-definidas. As aplicações criadas com o Framework Rails são desenvolvidas segundo o padrão de projecto **Model-View-Controller** (MVC).

O RoR é um meta-framework visto que é uma junção de cinco frameworks:

- **Active Record:** é uma camada de mapeamento objecto-relacional, que permite a interoperabilidade entre a aplicação e a base de dados e ao mesmo tempo permite a abstracção dos dados.
- **Action Pack:** agrupa o Action View (gerador da visualização do utilizador como, por exemplo, HTML, XML, JavaScript,...) e o Action Controller, responsável pelo controlo do fluxo de negócio.
- **Action Mailer:** responsável pelo serviço de envio e recepção de e-mails.
- **Active Support:** conjunto de classes e extensões de bibliotecas tidas como úteis para o desenvolvimento de aplicações em Ruby on Rails.
- **Action WebServices:** permite publicar APIs operáveis com Rails.

As principais vantagens no uso desta tecnologia prendem-se com a rapidez com que se desenvolve o site, a performance que o site atinge e o facto de a ferramenta ser gratuita.

2.2.2 Padrão MVC

Em aplicações complexas torna-se essencial a separação entre os dados (Model) e o layout (View). Assim as alterações feitas no layout não afectam a manipulação de dados, e também permite que os dados sejam alterados sem necessidade de mudar o layout.

O Model-View-Controller é então um padrão que resolve o problema da dependência dos dados em relação ao layout e vice-versa. O problema é resolvido com a separação das tarefas de acesso aos dados e lógica de negócio, lógica de apresentação e de interacção com o utilizador. A ligação entre estas duas camadas é feita pelo Controller que processa e responde a eventos, geralmente acções do utilizador, e pode invocar alterações no Model.

Assim no nosso trabalho este padrão enquadrava-se perfeitamente, dada a existência de uma camada de negócio na qual se efectuariam a gestão dos anúncios e a existência de uma camada

de apresentação na qual os diversos utilizadores interagem com o interface através do seu browser.

2.2.3 JavaScript

Javascript foi utilizado na interligação entre o Rails e o Google Maps, pois a API do Google Maps assim o exige e o JavaScript permite executar funções do lado do Cliente, sem intervenção do Servidor. Com JavaScript as funções auxiliares podem ser tratadas do lado do Cliente o que diminui a carga do Servidor. Um caso bastante evidente é na geração de statements SQL não necessitar da intervenção do Servidor, porque só é necessário a validação dos campos introduzidos e respectiva criação da string SQL. De seguida, a string SQL é enviada para o Servidor para que seja processada, devolvendo os registos.

A linguagem Javascript serve também para manipular elementos através da Document Object Model, como é exemplo, a recolha dos dados das caixas de texto. É, também, muito usada para atribuir eventos ao “rato” ex. (na passagem do rato por cima de algo, é executado uma dada função javascript).

(Retirado de [2])

O Javascript é uma linguagem de programação criada pela Netscape em 1995, que a princípio se chamava LiveScript, para atender, principalmente, as seguintes necessidades:

- Validação de formulários no lado Cliente;
- Interação com a página. O Javascript tem sintaxe semelhante à do Java, mas é totalmente diferente no conceito e no uso.

Características do Javascript:

1. Oferece tipos dinâmicos - tipos de variáveis não são definidos;
2. É interpretada, em vez de ser compilada;
3. Possui ótimas ferramentas padrão para listagens (como as linguagens de script, de modo geral);
4. Oferece bom suporte a expressões regulares (característica também comum a linguagens de script).

Sua união com o CSS é conhecida como DHTML. Usando o Javascript, é possível modificar dinamicamente os estilos dos elementos da página em HTML.

2.2.4 Apache

(Retirado de [2])

O Servidor Apache (ou Servidor HTTP Apache, em inglês: Apache HTTP Server, ou simples:Apache) é o mais bem sucedido Servidor WEB livre. Foi criado em 1995 por Rob

McCool, então funcionário do NCSA (National Center for Supercomputing Applications). Numa pesquisa realizada em Dezembro de 2005, foi constatado que a utilização do Apache supera 60% nos Servidores activos no mundo.

É a principal tecnologia da Apache Software Foundation, responsável por mais de uma dezena de projectos envolvendo tecnologias de transmissão via WEB, processamento de dados e execução de aplicativos distribuídos.

O Servidor é compatível com o protocolo HTTP versão 1.1. As funcionalidades são mantidas através de uma estrutura de módulos, o utilizador poderá escrever os seus próprios módulos, utilizando a API do software.

É disponibilizado em versões para os sistemas Windows, Novell Netware, OS/2 e diversos outros do padrão POSIX (Unix, Linux, FreeBSD, etc).

Configuração do Servidor Apache:

O Servidor é configurado por um ficheiro designado por httpd.conf, e opcionalmente pode haver configurações para cada directoria utilizando arquivos com o nome .htaccess, onde é possível utilizar a autenticação do utilizador pelo próprio protocolo HTTP utilizando uma combinação do ficheiro .htaccess com o ficheiro .htpasswd, que guardará os utilizadores e as passwords.

2.2.5 Outras Tecnologias e Ferramentas Utilizadas

Até agora descrevemos algumas das principais tecnologias usadas, mas houve necessidade de recorrer a mais tecnologias e ferramentas, em certos casos para facilitar o desenvolvimento do projecto, o caso do SVN, noutros casos porque era uma necessidade para a implementação do projecto.

O SVN serviu para controlo de subversões do projecto ao longo do seu progresso, facilitando o trabalho em grupo simultaneamente.

Outras ferramentas usadas foram o Dreamweaver para desenvolver o site, mais no que respeita ao interface com o utilizador.

2.3 Planeamento do projecto

Dado o nosso trabalho ser relativamente extenso, tem sentido planear primeiramente aquilo que deve ser feito para atingir o objectivo pretendido. Assim sendo começamos por entender quais os intervenientes na aplicação e as funcionalidades, que devemos oferecer aos mesmos, para posteriormente esquematizarmos uma boa solução.

Tendo como base para este primeiro planeamento a estrutura MVC que o Rails (Framework por nós utilizado) adopta, definimos então que o primeiro a fazer seria encontrar os modelos,

assim como as suas relações, e também os controladores necessários para corresponder às necessidades da aplicação, e facilitar futuras evoluções da mesma.

2.3.1 Intervenientes no Sistema

Neste sistema estão presentes vários intervenientes que diferem entre si nas permissões concedidas dentro do site. As entidades presentes são:

1. Root
2. Administrador
3. Utilizador
4. Visitante

Quando criamos este site deixamos em aberto a possibilidade de ainda introduzir mais intervenientes num estatuto hierárquico entre o Administrador e o Utilizador (utilizadores com acesso a funcionalidades extra). Falaremos seguidamente das permissões que cada tipo de utilizador tem à sua disposição, recorrendo a diagramas UML de Use-Case.

2.3.1.1 Root

O Root é a entidade suprema neste sistema, pois tem permissões para efectuar qualquer acção dentro do sistema.

2.3.1.2 Administrador

O Administrador tem como principal função a moderação dos anúncios inseridos fazendo o seu controlo mediante a aprovação, rejeição ou edição dos mesmos.

Neste caso apenas se coloca como pré-requisito para todas as acções que o login já esteja efectuado, após o login o Administrador pode efectuar qualquer uma das acções descritas tanto nos seus anúncios como nos anúncios dos vários utilizadores.

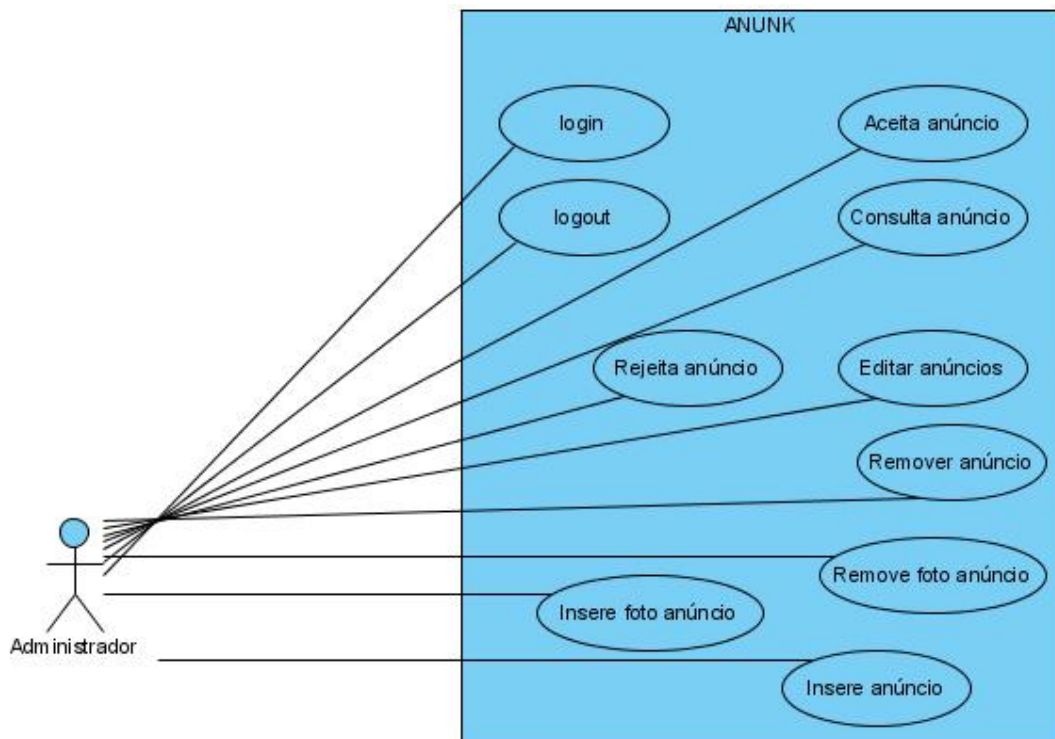


Ilustração 1 - Diagrama Use-Case para Administrador

2.3.1.3 Utilizador

O Utilizador está permitido de inserir anúncios e editar os próprios anúncios. Além disto o Utilizador pode pesquisar por anúncios de outros Utilizadores.

Os pré-requisitos representados por (*1) são: estar com login efectuado; o anúncio tem que ser do utilizador em questão.

O pré-requisito (*2) é que o Utilizador se encontre com login efectuado.

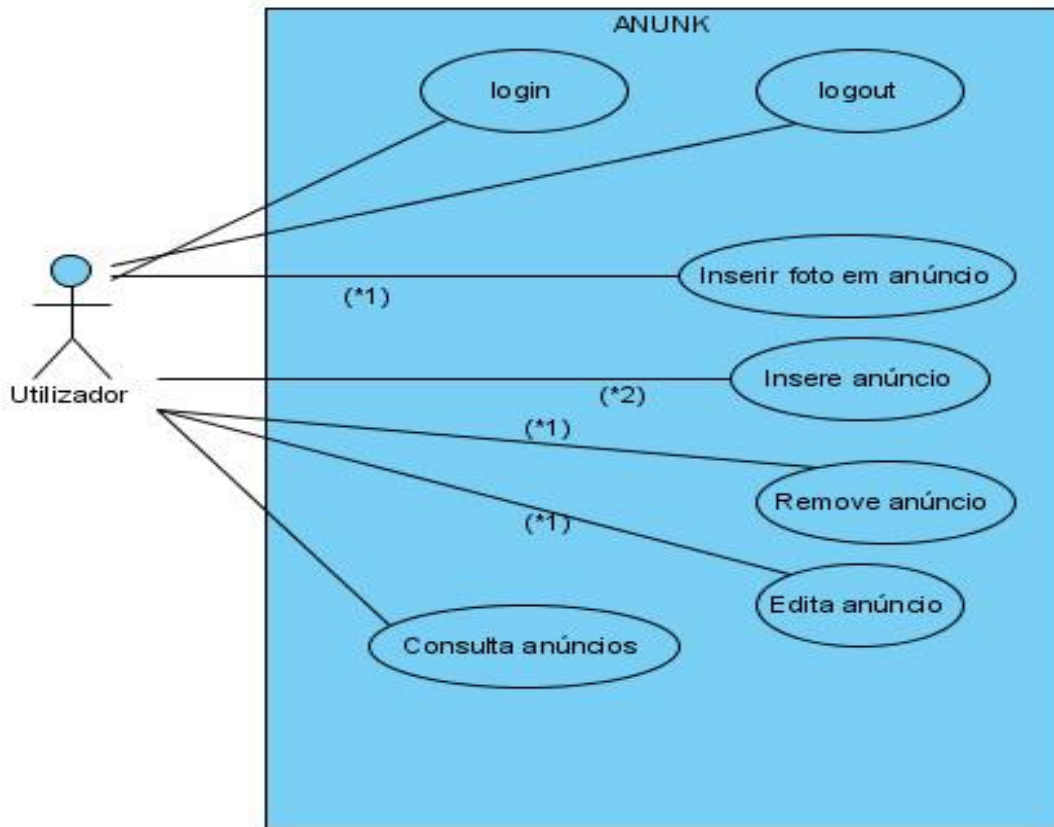


Ilustração 2 - Diagrama Use-Case para o Utilizador

2.3.1.4 Visitante

Ao Visitante não são concedidas mais permissões do que a consulta dos anúncios existentes e é permitido que o visitante crie um novo registo para que posteriormente possa efectuar o login como utilizador.

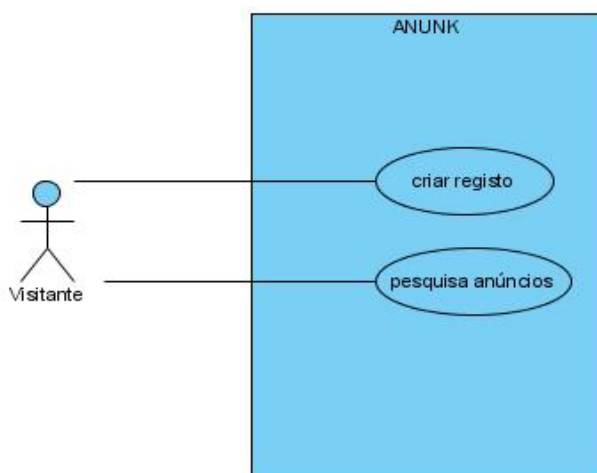


Ilustração 3 - Diagrama de Use-Case para o Visitante

2.4 Models

Na imagem abaixo podemos visualizar os modelos por nós definidos, assim como as suas relações. Estes irão dar origem a tabelas na base de dados e às primeiras classes da nossa aplicação.

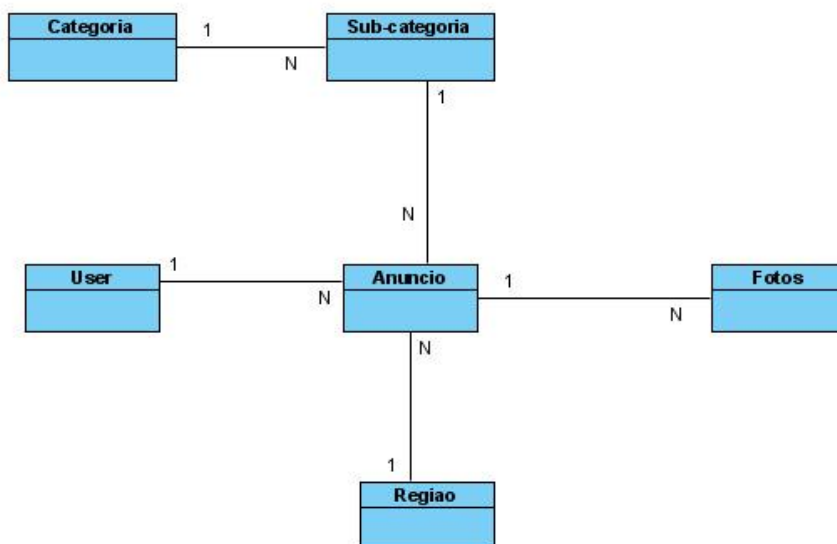
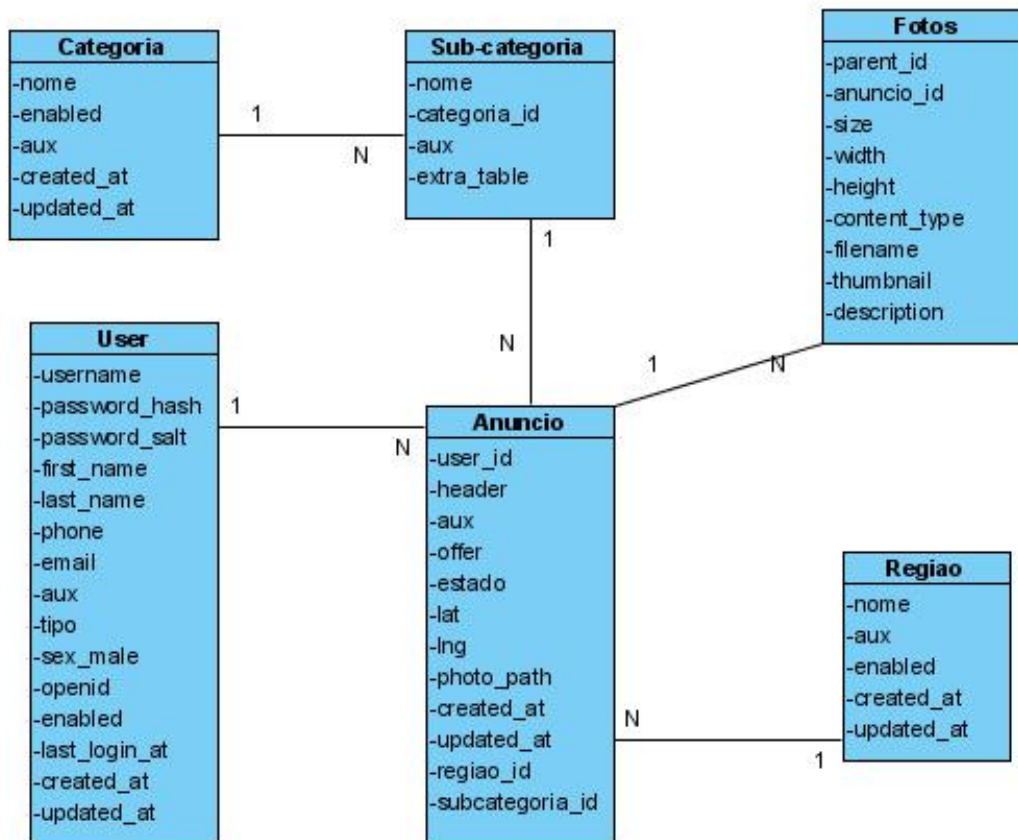


Ilustração 4 - Esquema dos Modelos

2.5 Base Dados

Aqui podemos ver as tabelas da nossa base de dados, assim como as suas relações e atributos.



2.6 Controller

Um controlador tem como função receber pedidos do cliente e processar os dados recorrendo aos modelos para criar a informação necessária, e relevante para o pedido. Depois apenas deve passar estas variáveis para o layout (camada de views), esta camada apenas deve definir a parte estética ou seja mostrar a informação passado de forma agradável!

Sendo assim criamos os seguintes controladores!

- anuncios_controller
- users_controller
- categorias_controller
- photos_controller
- regioes_controller
- categorias_controller
- subcategorias_controller

Como se pode verificar cada um acaba por representar um dos modelos já anteriormente definidos, além disto tivemos em conta convenções e boas praticas do rails, e todos estes

controladores respondem as mesmas ações: show(deve tb receber um id e mostra esse anuncio, user, etc, consoante o controlador), new e create para criar uma nova entidade(logicamente associada ao controlador), edit e create para alterar os dados de uma dessas entidades, list para as listar, e destroy para eliminar, evidente mente que estes controladores tem em conta quem esta a fazer o pedido e se tem ou não privilegios para o fazer. Embora algumas destas ações possam acabar por não ser usadas é boa pratica emplementar assim os modelos! E evitar criar ações diferentes das referiadas, mas evidentemente elas são necessárias e entao criamos essas ações mas em controladores diferentes que não representa nenhuma entidade presente no modelos, estes controladores são os seguintes:

- application_controller
ações referentes a lógica do controlador que possam estar acessíveis por todos os controladores como exemplo ações para saber se o pedido é de um utilizador com login efectuado e quais os privilégios deste!
- account_controller
este controlador apenas trata das ações de login e logout ou seja gera a sessão dos utilizadores que visitam o site!
- javascripts_controller
Este controlador ao contrario dos outros não prepara os dados para serem apresentados nas views mas sim, prepara os dados que vão permitir gerar javascript usando ruby (ou seja usamos ruby como meta linguagem), para posteriormente na view esse javascript gerar HTML temos esta necessidade devido à API do GMaps ser em JavaScript.

3 Exemplo de Utilização (Views)

Dado ter-se seguido o padrão MVC e já se ter falado anteriormente dos Models e do Controller chega a parte em que falamos das views, mais concretamente fazendo uma demonstração de utilização.

Quando um visitante vai à página inicial do ANUNK são lhe listados os anúncios mais recentes que foram introduzidos por utilizadores registados. É este o aspecto da página:

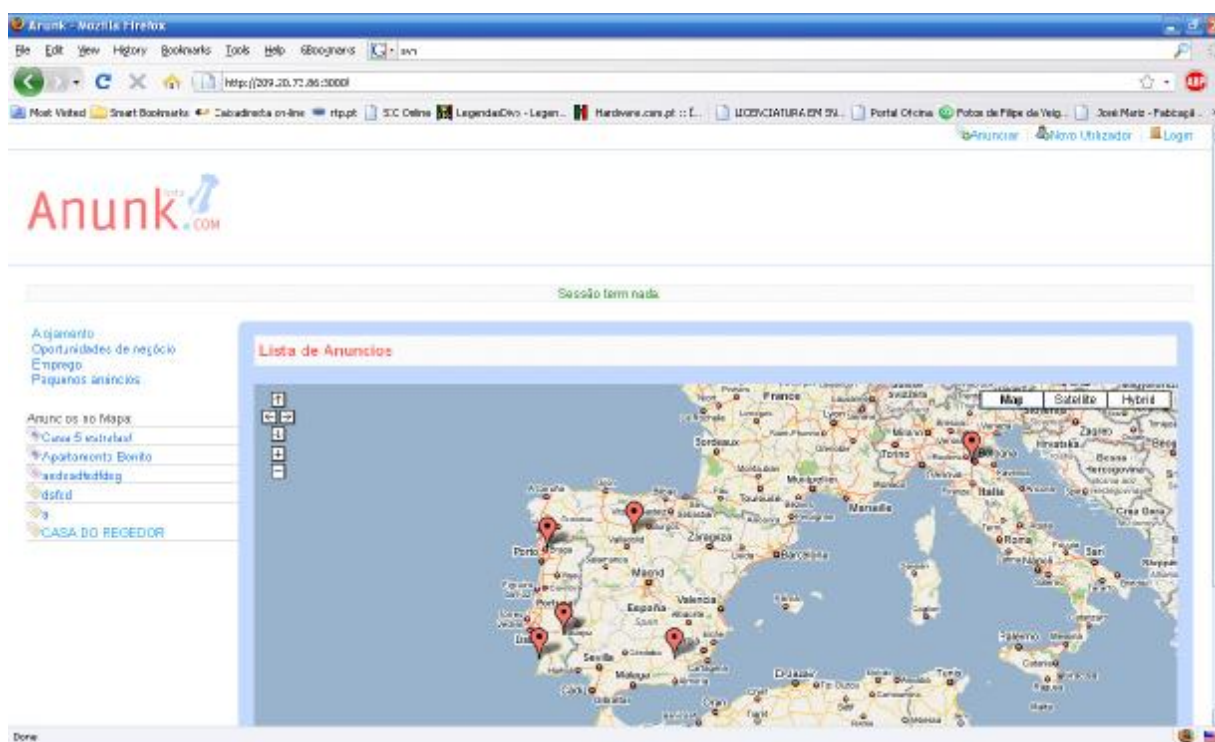


Ilustração 5 - Visualização do site pelo visitante

Caso o visitante se pretenda registar é só dirigir-se ao link para criar um novo utilizador e preencher os seguintes campos:

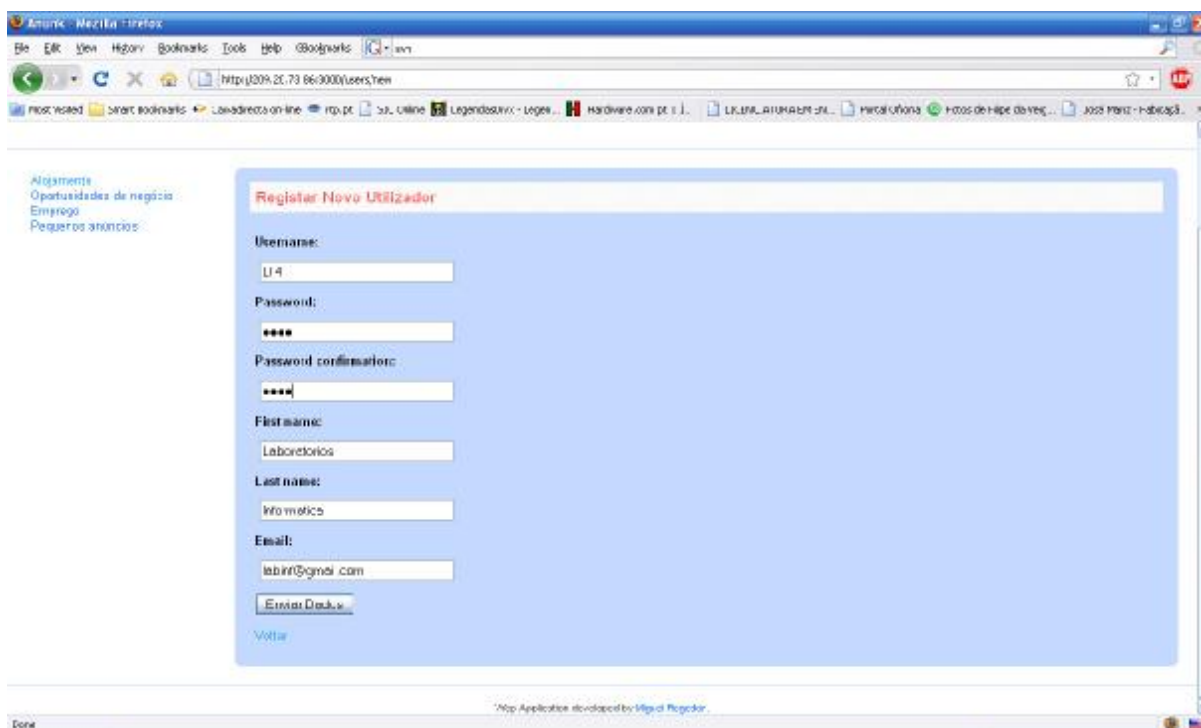


Ilustração 6 - Registo de Utilizador

Após o registo efectuado com sucesso o utilizador pode efectuar login e entre outras funcionalidades pode inserir anúncios, como se pode ver no exemplo a baixo:

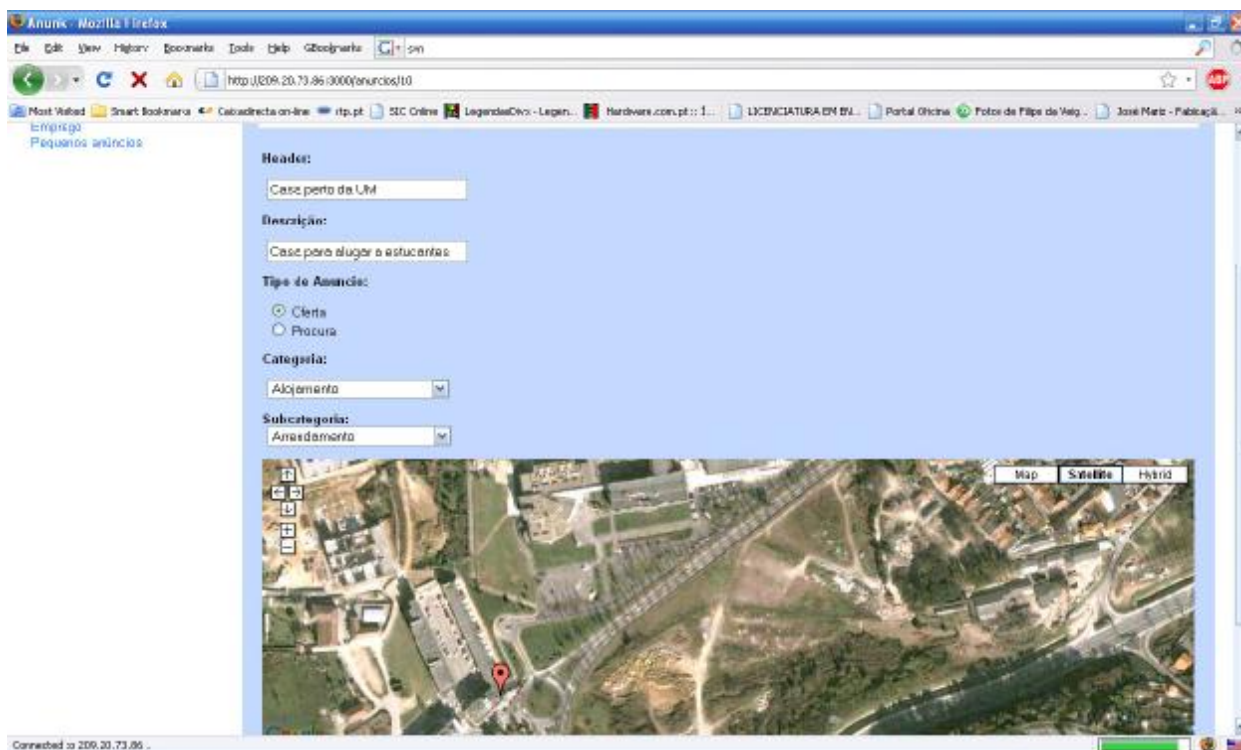


Ilustração 7 - Inserção de anúncio por utilizador

Caso o utilizador pretenda pode adicionar fotos ao anúncio para o tornar mais apelativo.

Posteriormente quando algum visitante do site procura informação sobre determinado ponto, com ou sem ajuda dos filtros, basta que passe o “mouse” por cima do ponto com um anúncio para que lhe sejam mostrados todos os detalhes do anúncio, assim como os contactos do utilizador que registou este ponto. A perspectiva é a seguinte:

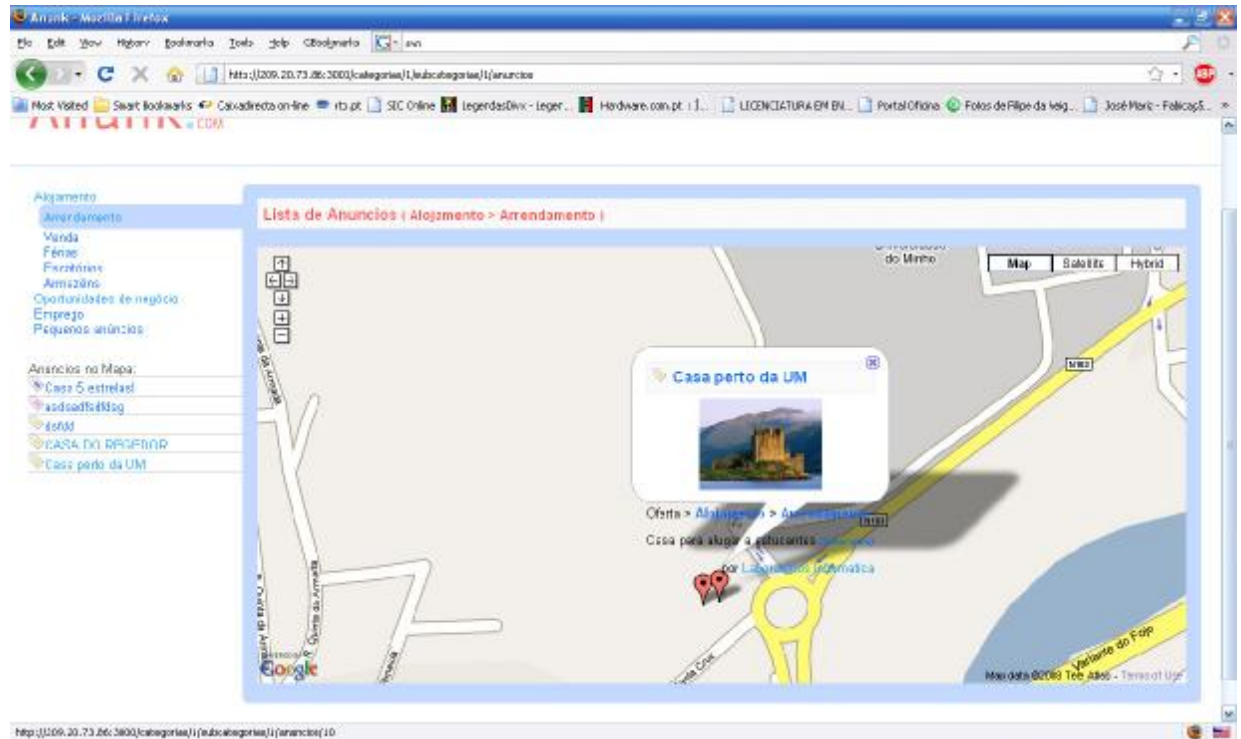


Ilustração 8 - Detalhe de anúncio

4 Problemas de implementação

Ao longo da implementação tivemos vários problemas, principalmente no que diz respeito à eficiência na altura de listar grandes quantidades de anúncios nos mapas, pois toda a informação a apresentar no mapa tem de ser enviada primeiramente para o server do googlemaps e dado que o javascript é criado usando Ruby como meta linguagem, tudo isto podia ficar um pouco moroso para certos pedidos sendo assim foi necessário criar algumas restrições na informação mostrada relativamente aos anúncios dentro dos balões que podemos encontrar nos mapas. Algo que também ainda não foi muito salientado ao longo deste relatório foi o facto de termos de configurar remotamente o servidor, instalar todas estas tecnologias Ruby, Rails, Apache e também um servidor svn que permite uma melhor organização neste tipo de projectos.

5 Conclusão e Trabalho Futuro

Este projecto ainda não se encontra terminado, neste momento ainda não consideramos características individuais das diferentes categorias, e isso será necessário assim como modos de visualização mais adaptados às mesmas, mas todo o projecto está preparado para essa evolução e mesmo após a entrega deste relatório o projecto deve continuar a evoluir.

Durante a execução deste trabalho confirmamos que as linguagens por nós usadas foram uma escolha acertada.

Bibliografia

Michael Fitzgerald, *Ruby pocket Reference*, O'Reilly.

Andre Lewis, *Google Maps Aplications with Rails and AJAX*, Apress.

Referências WWW

- [01] www.di.uminho.pt
Página principal do Departamento de Informática da Universidade do Minho. Aqui podemos encontrar informação referente aos membros do departamento, actividades de ensino, projectos de desenvolvimento e de investigação, etc.
- [02] <http://pt.wikipedia.org>
Página com diversos tipos de informações sobre várias das ferramentas e tecnologias usadas;
- [03] <http://www.rubyonrails.org/>
Página principal do RoR, a partir da qual podemos descarregar o RoR 2.0 e aceder a informações sobre esta tecnologia;

Lista de Siglas e Acrónimos

LI4	Laboratórios de Informática IV
BD	Base de Dados
RoR	Ruby on Rails
SVN	Subversion
API	Application Programming Interface
DHTML	Dynamic Hiper Text Markup Language
CSS	Cascading Style Sheets
HTTP	HyperText Transfer Protocol
SQL	Structured Query Language