



Universidade do Minho

Mestrado em Engenharia Informática – 1º Ano

Análise e Conceção de Software

Engenharia de Aplicações Web

Implementação da Aplicação Web

Elementos do Grupo:

Luís André Mota Araújo - PG19835

Rafael Eduardo de Carvalho Pereira - PG19753 (Gestor)

Rui Alexandre Afonso Pereira - PG20201

Braga, janeiro de 2012

Índice

Índice de Figuras	3
1. Contextualização	4
1.1. Introdução	4
1.2. Objetivos	4
2. Implementação	5
2.1. Componente Funcional	5
2.1.1. Vista de Visitantes	5
2.1.2. Vista de Utilizadores Registados	8
2.1.3. Vista de Administrador	13
2.1.4. Serviço de Eventos e Google Maps API	14
2.2. Componente Não Funcional	17
3. Notas Finais e Trabalho Futuro	18
4. Conclusão	19
5. Bibliografia	20
6. Anexos	21
6.1. Base de Dados e Estado Inicial	22
6.2. Código Javascript – Google Maps API	27
6.3. Código do Template da <i>Power Index Unit</i> alterado	30
6.4. Screenshots da Aplicação	35

Índice de Figuras

Figura 1 Logótipo da ferramenta de desenvolvimento de Aplicações Web orientadas aos dados usada para o desenvolvimento da aplicação Web Eventhis.	4
Figura 2 Página de Pesquisa de Eventos.	5
Figura 3 Página de Autenticação.	6
Figura 4 Área de Registo de Utilizadores.	7
Figura 5 Área de Visualização de Eventos.	7
Figura 6 Página de (Eventos) Favoritos.	8
Figura 7 Área de Edição de Registo.	9
Figura 8 Área de Visualização de Eventos.	11
Figura 9 Área de Gestão de Eventos.	12
Figura 10 Páginas da Vista de Administrador.	13
Figura 11 Serviço Web (REST) que permite retornar os eventos para uma área indicada.	15
Figura 12 <i>XML Out Unit</i> que extrai os eventos de uma determinada área usando uma <i>Seletor Unit</i> e converte para XML.	15
Figura 13 Exemplo de pedido e output XML do serviço Web.	16
Figura 14 Resultados que advém do serviço Web.	16
Figura 15 Alteração do Estilo para a introdução de imagens nos comentários.	17
Figura 16 Exemplo de seleção de coordenada geográfica para a realização de um evento.	18
Figura 17 Página de Autenticação.	35
Figura 18 Página que permite ver um evento detalhado para utilizadores autenticados.	36
Figura 19 Página de Criação de Evento.	37
Figura 20 Página de Edição de Evento.	37
Figura 21 Página Favoritos.	38
Figura 22 Página de Registo na Aplicação Web.	38
Figura 23 Página principal para utilizadores registados.	39
Figura 24 Página principal para Visitantes.	39

1. Contextualização

1.1.Introdução

Este documento é suporte à fase de implementação da aplicação Web Eventhis, que está relacionada com publicação e publicitação de eventos na Web, facilitando aos utilizadores encontrarem atividades para ocuparem o seu tempo livre.

Durante as todas as fases anteriores foi sendo pensada e desenhada uma aplicação Web seguindo uma abordagem de engenharia. O processo usado foi muito cerimonioso mas teve como output documentação muito útil para a implementação da aplicação Web, pelo que serão feitas variadas referências à mesma, para justificar decisões da implementação.

1.2.Objetivos

O objetivo principal desta fase foi a implementação da aplicação Web pensada e desenhada em todas as fases ulteriores usando a aplicação *Webratio*, tal implicou alguma disciplina de desenvolvimento.



Figura 1 Logótipo da ferramenta de desenvolvimento de Aplicações Web orientadas aos dados usada para o desenvolvimento da aplicação Web Eventhis.

Este documento serve de suporte a todo o desenvolvimento, sendo a documentação para futuras alterações/iteraões do projeto, uma vez que este é o primeiro protótipo da aplicação desenvolvida.

2. Implementação

Foram identificadas duas componentes distintas na implementação da aplicação *Eventhis*, uma claramente funcional, que diz respeito à implementação de todas as funcionalidades (use cases) previstas e outra, não funcional, que visou alterar a representação visual da aplicação, permitindo tornar a aplicação mais apelativa em duas páginas da aplicação, como será explicado.

2.1.Componente Funcional

2.1.1. Vista de Visitantes

Na vista de visitantes (Guests) são disponibilizadas todas as funcionalidades relacionadas com a pesquisa e visualização de eventos e com a autenticação e criação de mais registo na aplicação.

Como foi explicado em etapas anteriores a pesquisa de eventos disponível para os utilizadores da aplicação é multicritérios, sendo possível pesquisar no título ou descrição de um evento e ainda restringir os resultados a um dos tipos existentes.

Abaixo temos a representação da página de pesquisa em *Webratio*, podemos verificar que são usadas duas *Units*, a Tipos e a Resultados, e o formulário Pesquisar Eventos. A *Unit* Tipos é um seletor que permite extrair da base de dados todos os tipos de eventos possíveis, sendo os mesmos usados para construir o formulário Pesquisar Eventos, mais especificamente o um campo de seleção (TipoEvento) que permitirá ao utilizador especificar um tipo para pesquisa.

Quando um utilizador da aplicação Web submete o formulário, a *Unit* Resultados apresenta os resultados da pesquisa, disponibilizando para todos os eventos resultado da pesquisa um link para a página do respetivo evento, a *Unit* Resultados é um *PowerIndex*.

Pode-se verificar que a *Unit* Resultados pode receber o identificador de um tipo e texto que é procurado na descrição ou título dos eventos do tipo indicado, se indicado, pois caso não o seja a pesquisa não é para todos os eventos disponíveis.

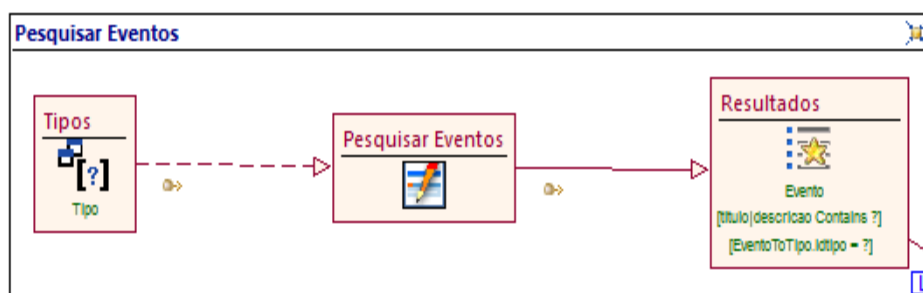


Figura 2 Página de Pesquisa de Eventos.

É na vista de visitantes que está disponível a página de autenticação, que está disponível para acesso a partir de qualquer página desta vista, uma vez que a página de autenticação é uma *Landmark*, como se pode verificar a partir pela figura 3.

Na página de autenticação é disponibilizado um formulário (Login) no qual o utilizador pode inserir o seu username e password, podendo realizar o seu login, para tal o utilizador tem que submeter o formulário que será encaminhado para uma *Login Unit*, onde serão validados os dados inseridos.

Caso o login seja bem-sucedido, o utilizador é encaminhado para a vista de utilizadores registados, no caso de este ser um utilizador registado, ou para vista de administrador, no caso de o mesmo ser um administrador. Tal foi especificado a partir dos *group_id* atribuído aos diferentes utilizadores, sendo que o módulo associado a um utilizador simples e a um administrador é diferente, tal pode ser verificado pela tabela 1, aos módulos foram por sua vez atribuídos os identificadores das vistas respetivas (tabela 2).

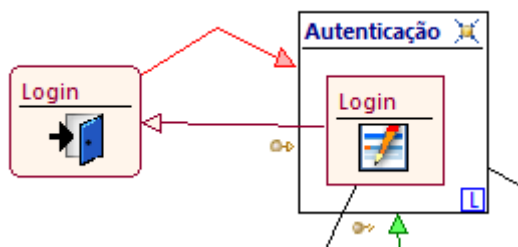


Figura 3 Página de Autenticação.

oid	groupname	module_oid
1	registados	1
2	administradores	2

Tabela 1 Tabela grupo.

oid	moduleid	modulename
1	sv4	registados
2	sv1	administradores

Tabela 2 Tabela module.

Um utilizador não registado pode seguir um link na página de autenticação que o levará para uma página de registo que está contida na área de registo, nesta página existe um formulário de registo que poderá ser submetido a uma *Operation Unit* que tratará de introduzir o novo utilizador no sistema, a referida *Operation Unit* foi denominada AddUtilizador e associada à entidade Utilizador, tal é verificável na imagem abaixo.

No caso de o registo ser bem-sucedido, o novo utilizador é encaminhado para a página de autenticação onde poderá realizar login.

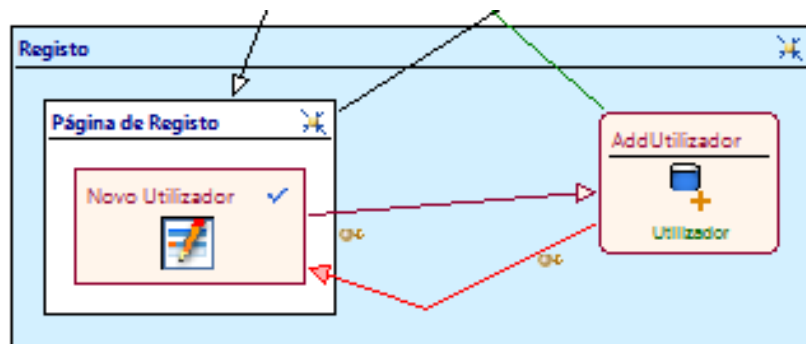


Figura 4 Área de Registo de Utilizadores.

A área de visualização de eventos tem duas páginas, a página principal para visitantes e a Página do Evento. A página principal para Visitantes tem um formulário e um *Power Index*, no entanto, tal não é usado na aplicação final, foi apenas para realizar testes, estas duas *Units* serviriam para o utilizador indicar uma zona geográfica, usando o formulário Zona, e seriam mostrados, usando o *Power Index* (MostrarEventos), os eventos da zona indicada. Na aplicação final acabamos por substituir estas duas *Units* por um mapa, provido pela API do *Google Maps* e por um *Web Service* que criámos para extrair da base os eventos da zona do Mapa em visualização, tal será explicado totalmente na secção 2.1.4.

Partindo da página principal é possível ir para a página de um evento escolhido, aí é disponibilizada toda a informação existente relativa ao evento, usando a *Data Unit* Evento, e aos seus comentários com a *Power Index Unit* Comentários. Como só os utilizadores registados é que podem comentar eventos não foi dada essa opção na Página do Evento da vista de visitantes.

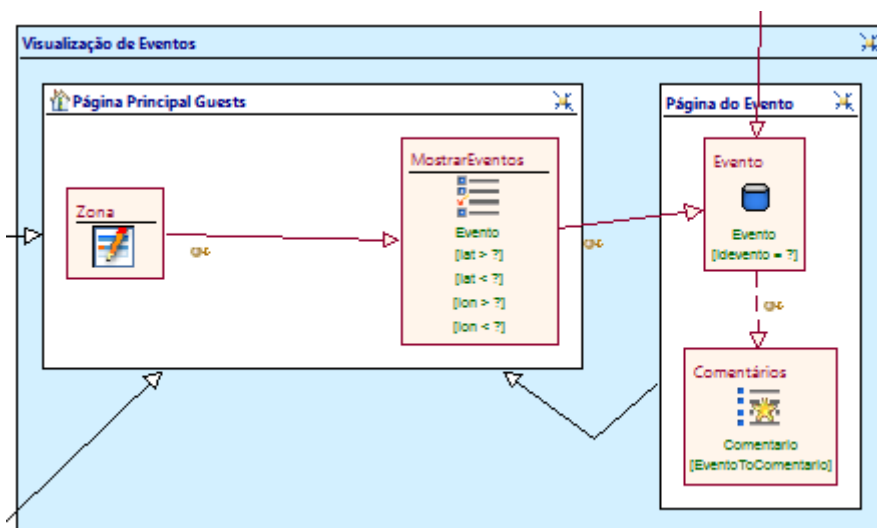


Figura 5 Área de Visualização de Eventos.

Foi criado um projeto de estilo para alterar *Power Index Unit* comentários, pois a representação gerada pelo *Webratio* não era muito agradável, essa componente da implementação é claramente não funcional, pelo que será explicada com maior detalhe na secção 2.2.

2.1.2. Vista de Utilizadores Registados

Todos os utilizadores que acedem à vista de utilizadores registados já estão devidamente autenticados, pelo que já é possível aceder aos seus registos. Os eventos marcados como favoritos pelos utilizadores são guardados na base de dados como uma associação entre o utilizador e o evento em causa, é para permitir ao utilizador a consulta dos eventos que marcou como favoritos que serve a página Favoritos, cuja implementação/modelação usando o *Webratio* pode ser apreciada na figura 6.

Para apresentar a um determinado utilizador os eventos que marcou como favoritos temos que identificar o utilizador autenticado, para tal foi usada uma *Get Unit* que foi denominada *GetLoggedUser* que devolve o identificador do utilizador autenticado na aplicação. Como é óbvio os eventos que já terminaram não têm interesse para os utilizadores, logo torna-se necessário só mostrar aos utilizadores os eventos marcados como favoritos que ainda não terminaram. Daí ter sido necessária a *Time Unit* *DataHoraActual* para filtrar na *Index Unit* *Favoritos* os eventos a mostrar.

A *Index Unit* *Favoritos* mostrará todos os eventos marcados pelo utilizador autenticado que ainda não terminaram, a partir dos eventos mostrados será possível seguir um link para a página do evento em que é possível aceder a toda a informação disponível relativa ao evento ou seguir um link que leva a que o evento seja desmarcado como favorito, sendo nesse caso usada a *Disconnect Unit* denominada *DesmarcarFavorito*, que necessita receber a identificação do evento a ser desmarcado (*idevento*) e do utilizador que o está a desmarcar (*oid*).

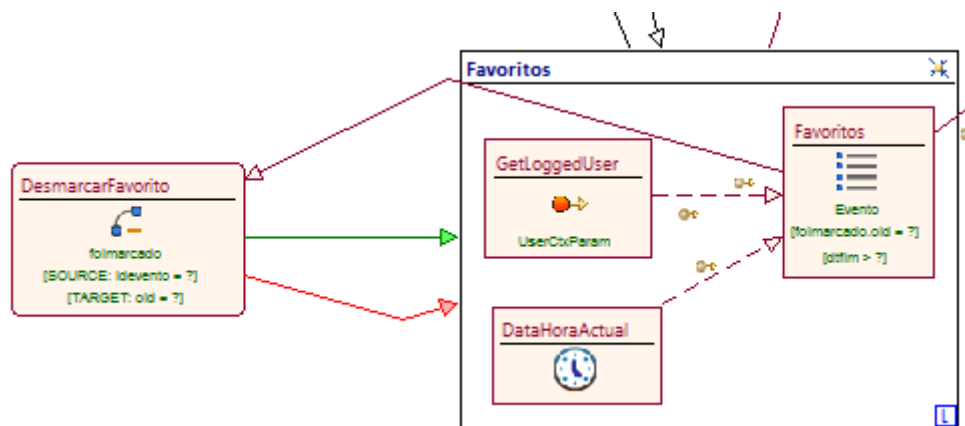


Figura 6 Página de (Eventos) Favoritos.

Na área de Edição de Registo está disponível a página que permite ao utilizador autenticado editar a sua informação de registo, que é justamente a página Página de Edição de Registo.

O *Seletor Unit* UtilizadorAutenticado é usado para preencher o formulário EditarUtilizador com a informação atual do utilizador, neste formulário o utilizador poderá com proceder a alterações ao seu registo. Para as alterações efetuadas serem refletidas nos dados da base de dados é necessário que os dados do formulário sejam enviados para a *Operation* AlterarUtilizador, sendo as alterações aplicadas com sucesso, o utilizador é redirecionado para a página principal da vista de utilizadores registados, caso contrário é enviado novamente para a página Edição de Registo.

Para que o *Seletor* UtilizadorAutenticado consiga extrair da base a informação relativa ao utilizador que está autenticado é necessário usar GetLoggedUser, que é uma *Get Unit*, para identificar o utilizador que está a utilizar a aplicação no momento.

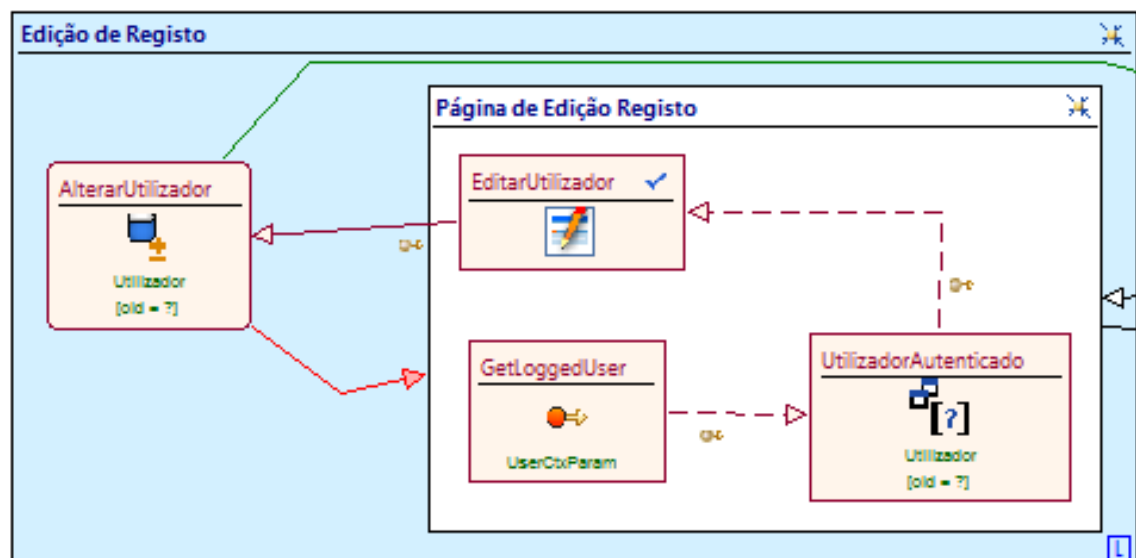


Figura 7 Área de Edição de Registo.

A área de Visualização de Eventos é a mais importante da aplicação, uma vez que é na mesma que estão concentradas a maior parte das funcionalidades que permitem a interação entre utilizadores e eventos registados no sistema.

Existem duas páginas nesta vista área, a página principal para utilizadores registados e a página do evento, na página principal para utilizadores registados é identificado o utilizador autenticado usando a *Get Unit* *GetLoggedUser* e são apresentados os dados iniciais, que são os tuplos latitude longitude que identificam a área geográfica que o utilizador definiu como preferencial no ato de registo, sendo essa informação usada para mostrar os eventos disponíveis na sua área preferencial.

Como já foi referido, nas páginas principais os eventos são mostrados num mapa, logo as *Units* *DadosIniciais* e *MostrarEventos* não vão ser usadas, pelo que são escondidas usando código do lado cliente (Javascript), no entanto a informação dada pela *Data Unit* *DadosIniciais* é usada para situar o mapa provido pela API da Google na zona preferencial do utilizador. A *Index Unit* *MostrarEventos* foi inserida exclusivamente para testes, uma vez que representa a aplicação final na plenitude, a partir dos eventos representados no mapa poderá navegar-se para a página do evento, tal como é representado na figura 8 mas usando a *Index Unit* *MostrarEventos*.

Na Página do Evento é apresenta toda a informação relativa ao evento identificado através do parâmetro que chega, via link, à *Data Unit* *Evento* e depois é distribuído por todas as *Units* que dele necessitam, como é o caso da *Power Index Unit* *Comentários* que apresenta todos os comentários efetuados relativos ao evento em causa.

Há operações que só podem estar disponíveis para determinados utilizadores, por exemplo o link que permite ir para a página de alteração/remoção do evento só deve ser apresentados ao utilizador criador do mesmo, tal foi fácil de fazer uma vez que a *DataUnit* *Evento* tem o identificador do utilizador que criou o evento, pelo que se criou uma variável de página que compara o identificador do utilizador autenticado e a ver a página no momento com o identificador do utilizador que criou o evento. Depois usou-se uma condição de expressão que mostra o link que permite aceder à página de edição do evento caso o utilizador seja o criador e esconde o link caso não o seja.

Contudo, existiu uma situação em que não estava disponível em nenhuma *Unit* os dados necessário para resolver o problema, essa situação diz respeito ao problema de identificar se utilizador que está a visualizar a página do evento já o marcou como favorito ou não. Devido ao explicado foi que introduzida a *Seletor Unit* *EstadoFavorito*, que a partir do identificador do evento e do identificador do utilizador a usar a aplicação, que advém da *Get Unit* *GetLoggedUser*. A partir do output do *EstadoFavorito* já foi possível criar uma variável e uma condição de expressão que mostrasse o link desmarcar favorito no caso de o evento em visualização ser favorito do utilizador ou mostrar o link marcar como favorito na situação inversa.

Na página do Evento foi introduzido um formulário (Novo Comentário) que permite ao utilizador que está a visualizar o evento criar comentários, a informação que está no formulário vai, quando este é submetido para uma *Operation Unit* que insere o novo

comentário (CriarComentário), no entanto é necessária mais informação do que a que está no formulário, pois é necessário o identificador do utilizador que está a criar o comentário, o identificador do evento ao qual o comentário vai ser associado e data-hora de criação do mesmo, os identificadores estão todos disponíveis em *Units* já referidas, no entanto a necessidade da data-hora da criação levou a introduzir-se um Time Unit TempoComentarios.

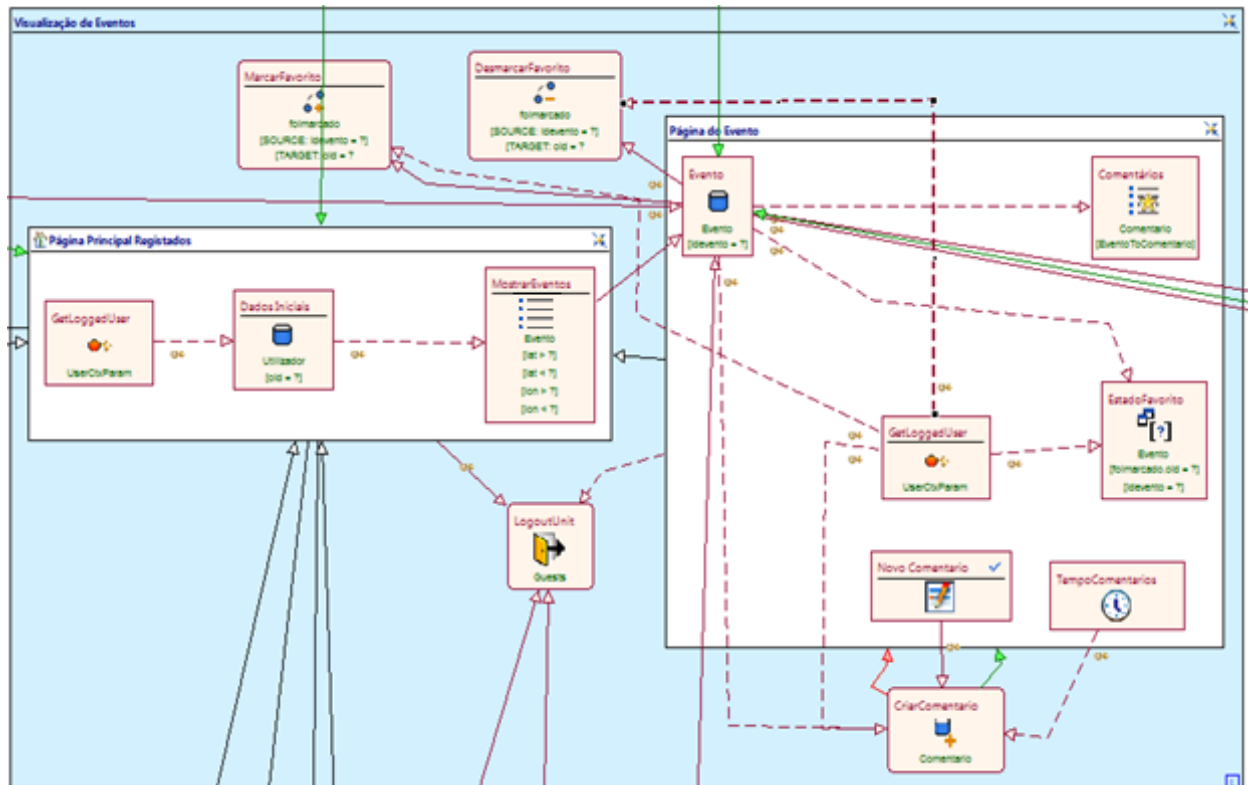


Figura 8 Área de Visualização de Eventos.

Na área de gestão de eventos estão disponíveis as páginas relacionadas com a criação, alteração e remoção de eventos. As Operation Units usadas são as RemoçãoEvento, ModificarEvento e CriarEvento. Tanto na página de Edição de Evento como na de Criar Evento foi necessário utilizar um *Seletor Unit* para extrair os tipos de eventos disponíveis da base.

Poderiam ter sido passadas as informações relativas ao evento a ser alterado para o formulário EditarEvento, no entanto, quando tal é feito por link a codificação dos caracteres acentuados fica alterada, pelo que foi utilizada a *Seletor Unit* DadosEventos.

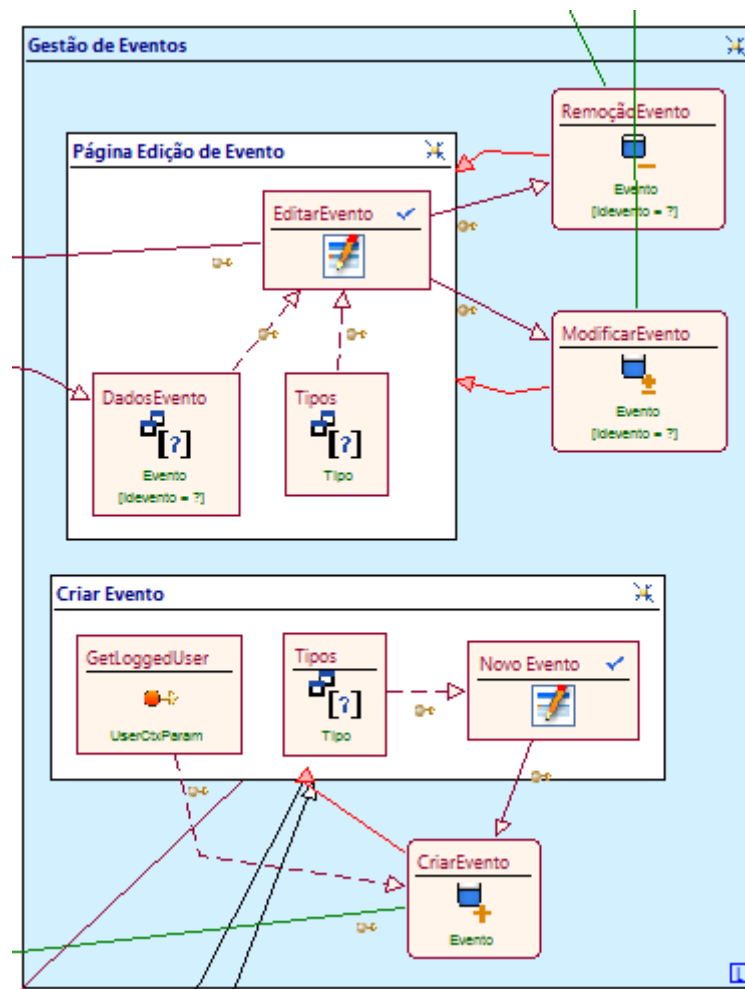


Figura 9 Área de Gestão de Eventos.

2.1.3. Vista de Administrador

Os administradores da aplicação Web têm a seu cardo o envio de newsletters para os utilizadores da aplicação informando acerca de novos eventos.

Na página principal da vista para administradores podem ser vistos os eventos para os quais foi solicitado o aparecimento na newsletter seguinte, tal é realizado usando a *Index Unit* Eventos da Newsletter, o formulário de enviar newsletter só deve estar visível no caso de existirem eventos para serem publicitados, pelo que foi necessário criar uma variável e uma condição de expressão. O valor da variável de página criada advém da *Seletor Unit* NumeroEventosNewsletter, que como o próprio nome indica conta o número de eventos que estão disponíveis para fazer uma newsletter, caso não esteja nenhum a condição de expressão será “false” e por isso o formulário que permite enviar a newsletter não será mostrado.

Depois do envio da newsletter é necessário desmarcar todos os eventos que estavam marcados para serem enviados na próxima newsletter, uma vez que esses já foram publicitados, para tal usa-se uma *Query Unit* que executa na base de dados o código SQL que desmarcará os eventos. O Email é enviado usando a *Mail Unit* Enviar Email.

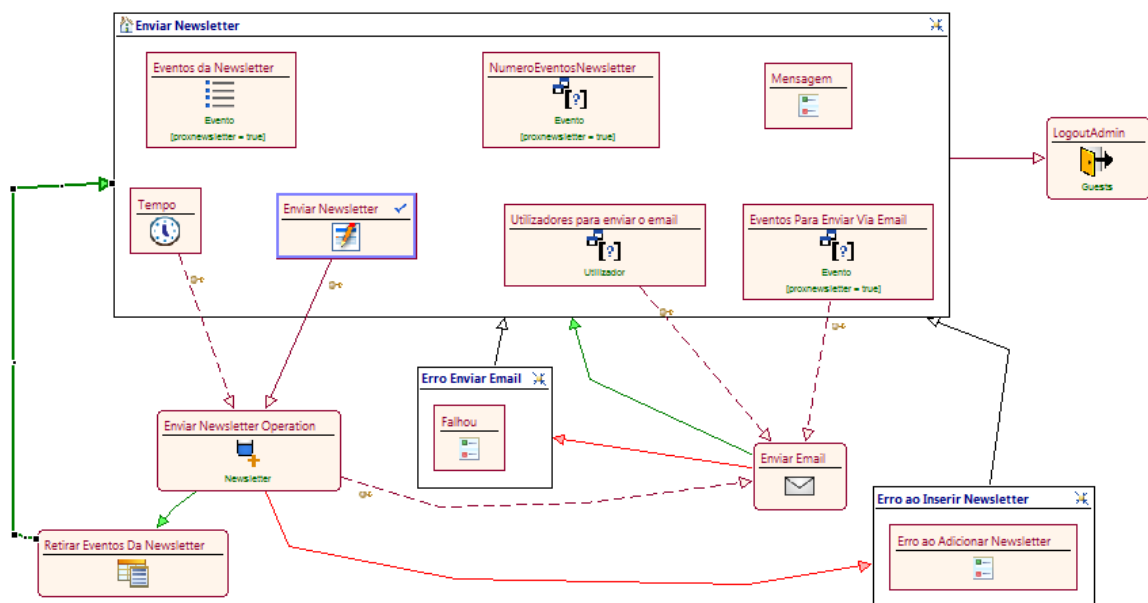


Figura 10 Páginas da Vista de Administrador.

2.1.4. Serviço de Eventos e Google Maps API

Como foi explicado durante as fases anteriores do projeto os eventos devem ser visualizados e criados contextualizados com a localização geográfica dos mesmos, pelo que é imperativo usar uma API de Mapas disponível gratuitamente, de entre as existentes foi escolhida a Google Maps API pela facilidade de uso e informação disponibilizada.

Todo o código relativo à introdução de mapas nas páginas principais, criação e edição de registo de utilizador e criação e edição de eventos só foi introduzido depois de aplicação estar toda construída, pois tal é feito usando código Javascript e seria mais complexo fazê-lo no Webratio.

Seria impossível carregar todos os eventos disponíveis para um mapa, pelo menos numa aplicação real em que o número de eventos fosse muito elevado, pelo que surgiu a necessidade de utilizar AJAX (Asynchronous JavaScript and XML) para pedir ao servidor os eventos existentes nas zonas em visualização no mapa. Assim, se um utilizador só está a visualizar a cidade de Braga só é feito download de informação relativa a eventos na área em causa. Como é óbvio a informação resposta do servidor tem que ser entendida pelo cliente, pelo que tem que ser possível tratar o resultado usando Javascript.

As necessidades identificadas nos parágrafos anteriores levaram à necessidade de criar um Serviço Web que conseguisse responder com informação relativa a eventos de uma determinada área indicada por parâmetro, sendo a resposta numa linguagem passível de processamento pelo browser cliente, no caso XML, mas poderia ser JSON ou outra que quiséssemos criar.

O Serviço Web criado foi implementado usando uma *Solicit Unit* que é de forma simplista uma *Unit* que recebe um pedido, ao qual foi dado o nome de Area, os pedidos tem como parâmetros dois tuplos GPS que especificam uma área geográfica. Os parâmetros são enviados para uma XML Out Unit que usa um Seletor para extrair os eventos que ainda não terminaram e se situam na área parâmetro e converte em XML sendo depois enviado como resposta para o browser que fez o pedido que o interpretará.

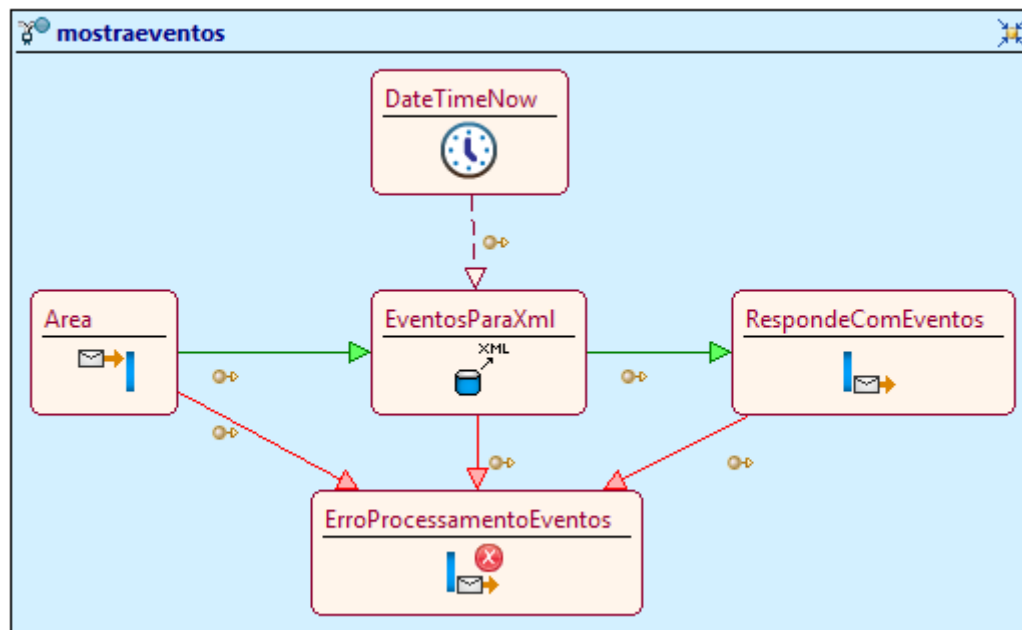


Figura 11 Serviço Web (REST) que permite retornar os eventos para uma área indicada.

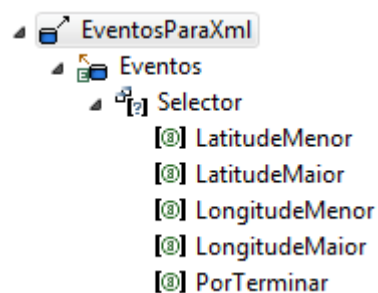


Figura 12 XML Out Unit que extrai os eventos de uma determinada área usando uma *Seleção Unit* e converte para XML.

Podemos observar na imagem abaixo um pedido REST feito ao servidor, neste caso usando o verbo HTTP GET e a resposta XML devolvida para a área indicada.

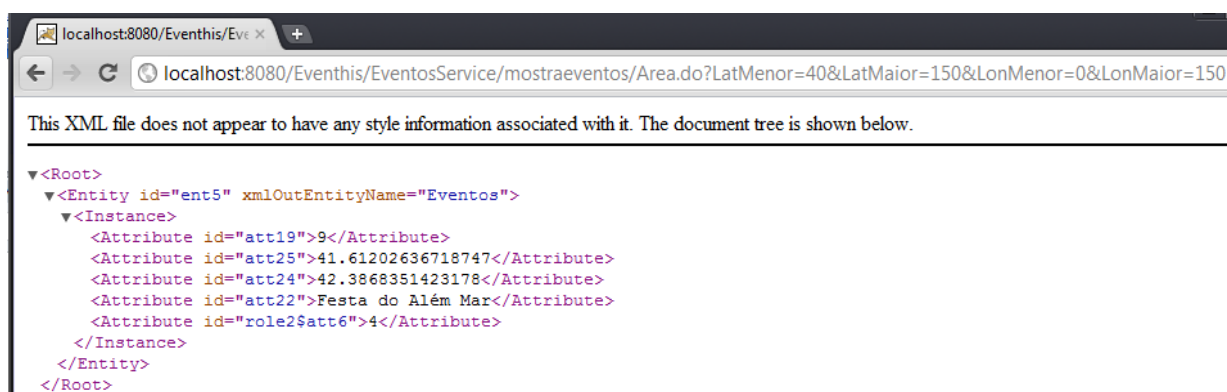


Figura 13 Exemplo de pedido e output XML do serviço Web.

Como o Javascript é melhor a tratar JSON (JavaScript Object Notation), pois é a notação que utiliza para representar os próprios objetos, foi decidido usar a API XMLObjectifier que permitiu converter o XML em JSON, sendo posteriormente tratado e introduzido no mapa.

No anexo 6.3 pode-se consultar o código *JavaScript* usado para dar suporte ao explicado nesta secção, estando o mesmo totalmente comentado.

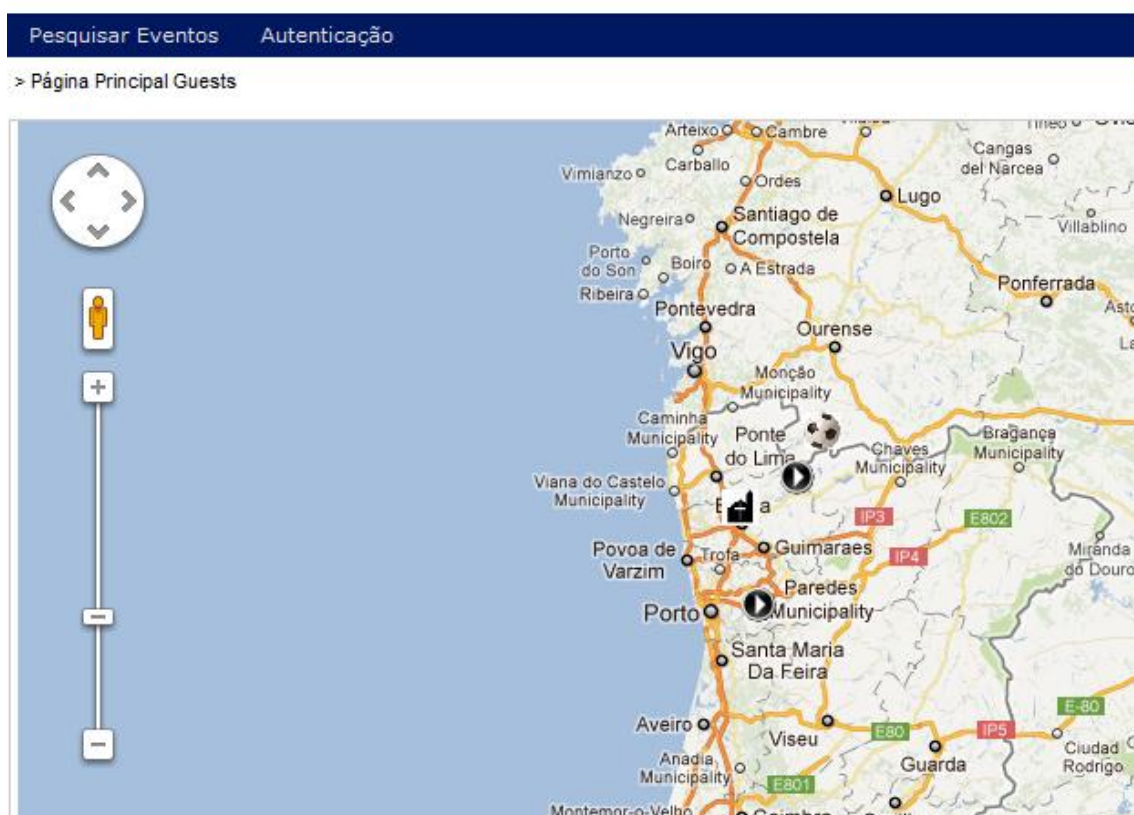
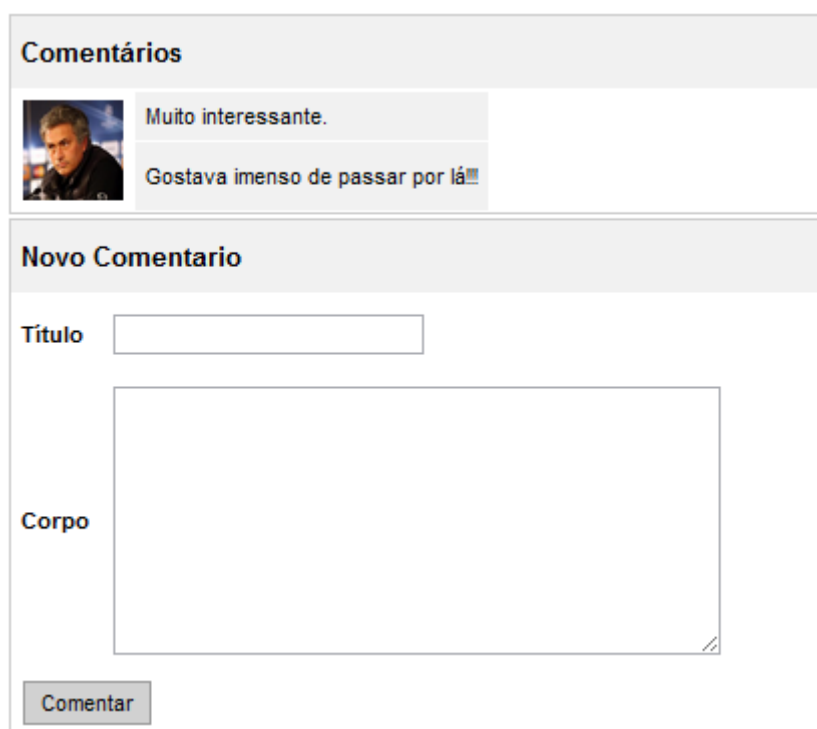


Figura 14 Resultados que advém do serviço Web.

2.2.Componente Não Funcional

Os comentários dos utilizadores ficavam por defeito na horizontal, como a imagem ficava grande e o título do comentário e corpo de seguida, tudo na mesma linha, tal não era muito estético, pelo que foi decidido proceder à criação de um Projeto de Estilo para criar uma nova apresentação que pudesse ser aplicada na *Power Index Unit* responsável por apresentar os comentários dos utilizadores.

O resultado gráfico da alteração pode ser visualizado na figura introduzida abaixo, o código do templates usado pode ser consultado no anexo 6.3.



The image shows a web interface for comments. At the top, there is a section titled "Comentários" in a light gray box. Below this, a user's profile picture is shown next to their comment text: "Muito interessante." and "Gostava imenso de passar por lá!!!". Below the comments, there is a section titled "Novo Comentario" in a light gray box. This section contains a form with two main input fields: "Título" (Title) and "Corpo" (Body). The "Título" field is a small text input box. The "Corpo" field is a larger text area. At the bottom of the form, there is a button labeled "Comentar" (Comment).

Figura 15 Alteração do Estilo para a introdução de imagens nos comentários.

3. Notas Finais e Trabalho Futuro

Para todos os formulários criados na implementação da aplicação foram definidas regras de validação, tendo sido todas testadas. Nos formulários que requeriam introdução de coordenadas GPS foram introduzidos mapas, sendo possível selecionar a área geográfica ou ponto geográfico a partir do mapa, ou introduzir diretamente a coordenada nas caixas de texto, tal pode ser verificado na imagem abaixo inserida.

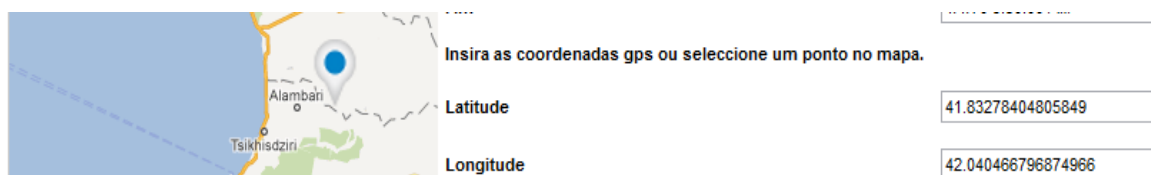


Figura 16 Exemplo de seleção de coordenada geográfica para a realização de um evento.

A possibilidade de criar uma Map Unit foi discutida, no entanto, seriam necessários conhecimentos que demorariam a adquirir, nomeadamente de *Java Server Pages*, *Groovy Script*, *Expression Language* e *JavaServer Pages Standard Tag Library*, algo que não poderia ser realizado em tempo útil, no entanto, seria muito mais interessante, pois *Unit* criada poderia ser utilizada em muitos projetos e facilitaria o *debug* e desenvolvimento da aplicação, pelo que tal tarefa é tida como primordial para evolução futura do trabalho desenvolvido.

Foram implementadas todas as funcionalidades previstas para o projeto, no entanto o modelo de negócio para a aplicação Web teria que ser completado, uma vez que não foram incluídos os pagamentos para quem quisesse que o evento fosse publicitado via newsletter.

A aplicação precisa de ser melhorada no que diz respeito ao *design* da mesma, pois devido a limites temporais, foram efetuadas poucas alterações à camada de visualização, tal é cada vez mais importante e merece atenção em futuras alterações/evoluções.

O administrador da aplicação Web deveria ter mais funcionalidades que lhe permitissem gerir a informação disponibilizada, podendo apagar informação ofensiva ou pejorativa, tal seria algo a evoluir na aplicação, mas implicaria também uma nova iteração sobre o processo de desenho, uma vez que tal não fez parte dos requisitos, pois a aplicação Web estava a ficar com demasiadas funcionalidades.

4. Conclusão

Durante a fase de implementação ficou claro que a abordagem seguida no estudo, análise e desenho da aplicação a implementar foi bem conseguido, uma vez que facilitou imenso a implementação.

A implementação da aplicação Web usando o *Webratio* limitou-se à representação em Web ML do que já tinha sido estudado e pensado na fase de *Coarse Design* da aplicação, o que foi de maior complexidade de implementação foram os mapas, que levaram à necessidade de código cliente e à criação de um serviço web, que não têm representação em *Coarse Design* daí que tal ainda não estivesse definido e a componente não funcional da implementação, que levou à criação de um projeto de estilo que melhorou a apresentação dos comentários dos eventos.

A aplicação Web é apenas um protótipo, todas as funcionalidades estão implementadas, no entanto é necessário aperfeiçoar muitas funcionalidades e ter um cuidado especial com a camada de apresentação da aplicação Web.

Desenvolver este projeto foi uma experiência muito enriquecedora, tendo ficado demonstrada a facilidade com que se implementam aplicações Web orientadas aos dados usando o *Webratio*, mas também ficou patente que quando é necessário utilizar código cliente e mecanismos como o AJAX, a experiência torna-se mais complexa, pois deve ser colocado no final do desenvolvimento em *Webratio*, sendo que uma nova geração de código apaga todo o código Javascript já incluído.

5. Bibliografia

- [1] Google Maps JavaScript API V3,
<http://code.google.com/apis/maps/documentation/javascript/>
Usado para contruir as páginas baseadas em mapas.
Ultimo acesso: 7/01/2012
- [2] Introdução ao JSON, <http://www.json.org/>,
Usado para obter conhecimentos relativos ao parsing de JSON no browser cliente, para interpretar o output do Serviço Web desenvolvido e inserir os eventos no mapa.
Ultimo acesso: 8/01/2012
- [3] Terracoder, <http://www.terracoder.com/index.php/xml-objectifier/xml-objectifier-documentation>,
Usado para converter o output XML do serviço Web criado para JSON, para facilitar o tratamento usando o Javascript,
Ultimo acesso: 7/01/2012
- [4] Web Modeling User Guide,
http://downloads.webratio.com/6.1/WebRatio_WebML_User_Guide.pdf,
Usado como auxiliar durante toda a implementação com o Webratio, foram particularmente úteis as informações relativas a Serviços Web e Web Style Projects.
Ultimo acesso: 7/01/2012

6. Anexos

6.1. Base de Dados e Estado Inicial

```
create database eventos;
use olaola;
-- Grupo [Group]
create table `grupo` (
  `oid` integer not null,
  `groupname` varchar(255),
  primary key (`oid`)
);

-- Module [Module]
create table `module` (
  `oid` integer not null,
  `moduleid` varchar(255),
  `modulename` varchar(255),
  primary key (`oid`)
);

-- Utilizador [User]
create table `utilizador` (
  `oid` integer not null,
  `email` varchar(255),
  `password` varchar(255),
  `teste` varchar(255),
  `nome` varchar(255),
  `foto` varchar(255),
  `lat` double precision,
  `lon` double precision,
  `lat2` double precision,
  `lon2` double precision,
  `username` varchar(255),
  primary key (`oid`)
);

-- Newsletter [ent1]
create table `newsletter` (
  `nr` integer not null,
  `titulo` varchar(255),
  `dtenvio` datetime,
  primary key (`nr`)
);

-- Tipo [ent2]
create table `tipo` (
  `idtipo` integer not null,
  `nome` varchar(255),
```

```

primary key (`idtipo`)
);

-- Comentario [ent4]
create table `comentario` (
  `idcomentario` integer not null,
  `titulo` varchar(255),
  `texto` longtext,
  `dtcom` datetime,
  primary key (`idcomentario`)
);

-- Evento [ent5]
create table `evento` (
  `idevento` integer not null,
  `dtfim` datetime,
  `dtinicio` datetime,
  `titulo` varchar(255),
  `descricao` varchar(255),
  `lat` double precision,
  `lon` double precision,
  `proxnewsletter` bit,
  primary key (`idevento`)
);

-- Group_DefaultModule [Group2DefaultModule_DefaultModule2Group]
alter table `grupo` add column `module_oid` integer;
alter table `grupo` add index fk_grupo_module (`module_oid`), add constraint
fk_grupo_module foreign key (`module_oid`) references `module` (`oid`);
create index `idx_grupo_module` on `grupo`(`module_oid`);

-- Group_Module [Group2Module_Module2Group]
create table `group_module` (
  `grupo_oid` integer not null,
  `module_oid` integer not null,
  primary key (`grupo_oid`, `module_oid`)
);
alter table `group_module` add index fk_group_module_grupo (`grupo_oid`), add constraint
fk_group_module_grupo foreign key (`grupo_oid`) references `grupo` (`oid`);
alter table `group_module` add index fk_group_module_module (`module_oid`), add
constraint fk_group_module_module foreign key (`module_oid`) references `module` (`oid`);
create index `idx_group_module_grupo` on `group_module`(`grupo_oid`);
create index `idx_group_module_module` on `group_module`(`module_oid`);

-- User_DefaultGroup [User2DefaultGroup_DefaultGroup2User]
alter table `utilizador` add column `grupo_oid` integer;
alter table `utilizador` add index fk_utilizador_grupo (`grupo_oid`), add constraint

```

```

fk_utilizador_grupo foreign key (`grupo_oid`) references `grupo` (`oid`);
create index `idx_utilizador_grupo` on `utilizador`(`grupo_oid`);

-- User_Group [User2Group_Group2User]
create table `user_group` (
  `utilizador_oid` integer not null,
  `grupo_oid` integer not null,
  primary key (`utilizador_oid`, `grupo_oid`)
);
alter table `user_group` add index fk_user_group_utilizador (`utilizador_oid`), add constraint
fk_user_group_utilizador foreign key (`utilizador_oid`) references `utilizador` (`oid`);
alter table `user_group` add index fk_user_group_grupo (`grupo_oid`), add constraint
fk_user_group_grupo foreign key (`grupo_oid`) references `grupo` (`oid`);
create index `idx_user_group_utilizador` on `user_group`(`utilizador_oid`);
create index `idx_user_group_grupo` on `user_group`(`grupo_oid`);

-- Tipo_Evento [rel1]
alter table `evento` add column `tipo_idtipo` integer;
alter table `evento` add index fk_evento_tipo (`tipo_idtipo`), add constraint fk_evento_tipo
foreign key (`tipo_idtipo`) references `tipo` (`idtipo`);
create index `idx_evento_tipo` on `evento`(`tipo_idtipo`);

-- Utilizador_Comentario [rel2]
alter table `comentario` add column `utilizador_oid` integer;
alter table `comentario` add index fk_comentario_utilizador (`utilizador_oid`), add constraint
fk_comentario_utilizador foreign key (`utilizador_oid`) references `utilizador` (`oid`);
create index `idx_comentario_utilizador` on `comentario`(`utilizador_oid`);

-- Evento_Comentario [rel3]
alter table `comentario` add column `evento_idevento` integer;
alter table `comentario` add index fk_comentario_evento (`evento_idevento`), add constraint
fk_comentario_evento foreign key (`evento_idevento`) references `evento` (`idevento`);
create index `idx_comentario_evento` on `comentario`(`evento_idevento`);

-- Utilizador_Evento [rel4]
alter table `evento` add column `utilizador_oid` integer;
alter table `evento` add index fk_evento_utilizador (`utilizador_oid`), add constraint
fk_evento_utilizador foreign key (`utilizador_oid`) references `utilizador` (`oid`);
create index `idx_evento_utilizador` on `evento`(`utilizador_oid`);

-- Favoritos [rel6]
create table `favoritos` (
  `utilizador_oid` integer not null,
  `evento_idevento` integer not null,
  primary key (`utilizador_oid`, `evento_idevento`)
);

```



```

alter table `favoritos` add index fk_favoritos_utilizador (`utilizador_oid`), add constraint
fk_favoritos_utilizador foreign key (`utilizador_oid`) references `utilizador` (`oid`);
alter table `favoritos` add index fk_favoritos_evento (`evento_idevento`), add constraint
fk_favoritos_evento foreign key (`evento_idevento`) references `evento` (`idevento`);
create index `idx_favoritos_utilizador` on `favoritos` (`utilizador_oid`);
create index `idx_favoritos_evento` on `favoritos` (`evento_idevento`);

-- Newsletter_Evento [rel7]
alter table `newsletter` add column `evento_idevento` integer;
alter table `newsletter` add index fk_newsletter_evento (`evento_idevento`), add constraint
fk_newsletter_evento foreign key (`evento_idevento`) references `evento` (`idevento`);
create index `idx_newsletter_evento` on `newsletter` (`evento_idevento`);

-- Script de configuração dos módulos para autenticação

INSERT INTO `module` (`oid`,`moduleid`,`modulename`) VALUES (1,'sv4','registados');
INSERT INTO `module` (`oid`,`moduleid`,`modulename`) VALUES (2,'sv1','administradores');

INSERT INTO `grupo` (`oid`,`groupname`,`module_oid`) VALUES (1,'registados',1);
INSERT INTO `grupo` (`oid`,`groupname`,`module_oid`) VALUES (2,'administradores',2);

-- Utilizadores e eventos de teste

INSERT INTO `utilizador`
(`oid`,`email`,`password`,`nome`,`foto`,`lat`,`lon`,`lat2`,`lon2`,`username`,`grupo_oid`,`teste`)
VALUES (1,'rafael@gmail.com','nova','Rafael Eduardo de Carvalho
Pereira','jm.jpg?field=page10.fld38',42.71384201504933,-
5.354429687499987,54.29017450293639,10.026429687500013,'rafael',1,'upload/images/1/j
m.jpg');

INSERT INTO `utilizador`
(`oid`,`email`,`password`,`nome`,`foto`,`lat`,`lon`,`lat2`,`lon2`,`username`,`grupo_oid`,`teste`)
VALUES (2,'admin@eventhis.com','1234567','Natal
Oliveira','http://www.google.pt/ola.jpg',0,0,100,100,'admin',2,'upload/images/1/IMG2011090
5_001.jpg');

INSERT INTO `utilizador`
(`oid`,`email`,`password`,`nome`,`foto`,`lat`,`lon`,`lat2`,`lon2`,`username`,`grupo_oid`,`teste`)
VALUES (3,'la@gmail.com','12345','Luis
Araujo','http://multimedia.fnac.pt/multimedia/PT/images_produits/PT/grandes150/4/5/8/404
4156023854.gif',0,0,100,100,'lala',1,'upload/images/1/IMG20110905_001.jpg');

INSERT INTO `utilizador`
(`oid`,`email`,`password`,`nome`,`foto`,`lat`,`lon`,`lat2`,`lon2`,`username`,`grupo_oid`,`teste`)
VALUES (4,'antonio@gmail.com','nova','António
Carvalho','http://cdn1.iconfinder.com/data/icons/softwaredemo/PNG/256x256/DrawingPin1_
Blue.png',40.824948183814335,-8.73284912109375,41.65107744417234,-
7.77154541015625,'antonio',1,'upload/images/1/IMG20110905_001.jpg');

INSERT INTO `utilizador`
(`oid`,`email`,`password`,`nome`,`foto`,`lat`,`lon`,`lat2`,`lon2`,`username`,`grupo_oid`,`teste`)

```

```

VALUES (6,'omnimonmaster@hotmail.com','ruilei','rui pereira','http://www.g-
volutiondance.com/Images/GV08home.jpg',34.044373582597856,-
15.67071533203125,47.30225321039508,-
0.28985595703125,'states',1,'upload/images/1/IMG20110905_001.jpg');

INSERT INTO `utilizador`
(`oid`,`email`,`password`,`nome`,`foto`,`lat`,`lon`,`lat2`,`lon2`,`username`,`grupo_oid`,`teste`)
VALUES (7,'rui@gmail.com','ruirui','Rui
Pereira','http://multimedia.fnac.pt/multimedia/PT/images_produits/PT/grandes150/4/5/8/40
44156023854.gif',40.21319374398207,-8.67242431640625,41.87029512726479,-
6.74981689453125,'ruipapa',1,'upload/images/1/IMG20110905_001.jpg');

INSERT INTO `utilizador`
(`oid`,`email`,`password`,`nome`,`foto`,`lat`,`lon`,`lat2`,`lon2`,`username`,`grupo_oid`,`teste`)
VALUES
(8,'luis.5.aa@hotmail.com','luis','Luis','http://data.whicdn.com/images/16658538/405194_460
s_v1_thumb.jpg',34.044373582597856,-15.67071533203125,47.30225321039508,-
0.28985595703125,'luis',1,'upload/images/1/IMG20110905_001.jpg');

-- tipos de eventos

INSERT INTO `tipo` (`idtipo`,`nome`) VALUES (1,'regional');
INSERT INTO `tipo` (`idtipo`,`nome`) VALUES (2,'desportivo');
INSERT INTO `tipo` (`idtipo`,`nome`) VALUES (3,'religioso');
INSERT INTO `tipo` (`idtipo`,`nome`) VALUES (4,'festivo');
INSERT INTO `tipo` (`idtipo`,`nome`) VALUES (5,'astronómico');
INSERT INTO `tipo` (`idtipo`,`nome`) VALUES (6,'político');
INSERT INTO `tipo` (`idtipo`,`nome`) VALUES (7,'artístico');
INSERT INTO `tipo` (`idtipo`,`nome`) VALUES (8,'feiras');

-- eventos

INSERT INTO `evento`
(`idevento`,`dtfim`,`dtinicio`,`titulo`,`descricao`,`lat`,`lon`,`proxnewsletter`,`tipo_idtipo`,`utiliza
dor_oid`) VALUES (1,'19:44:25','19:44:27','Festa da Nossa Senhora','A realizar em Salamonde
Vieira do Minho.',41.10385002812584,-8.3251953125,'0',4,1);

INSERT INTO `evento`
(`idevento`,`dtfim`,`dtinicio`,`titulo`,`descricao`,`lat`,`lon`,`proxnewsletter`,`tipo_idtipo`,`utiliza
dor_oid`) VALUES (9,'09:30:00','09:30:00','Festa do Além Mar','Festa com muita dança e
companheirismo a realizar na Georgia.',42.3868351423178,41.61202636718747,'0',4,1);

INSERT INTO `evento`
(`idevento`,`dtfim`,`dtinicio`,`titulo`,`descricao`,`lat`,`lon`,`proxnewsletter`,`tipo_idtipo`,`utiliza
dor_oid`) VALUES (5,'23:27:55','23:27:51','Feira em Honra de Nossa Senhora','Nesta feira serrerá
possível comprar de tudo inclusive computadores e latas de conserva.',41.54439989652676,-
8.433385848999023,'0',3,1);

```

6.2. Código Javascript – Google Maps API

```
<!--API da google que dá suporte para a utilização de mapas -->
<script type="text/javascript" src="http://maps.googleapis.com/maps/api/js?sensor=true">
</script>
<!--API usada para converter o XML em JSON, facilitando
    O tratamento da resposta do servidor -->
<script type="text/javascript"
src="http://www.terracoder.com/scripts/saXMLUtils.js"></script>
<!--Script construído realiza os pedidos de eventos ao servidor e os insere no mapa -->
<script type="text/javascript">
function inserirNoMapa(json)
{
    for(var i = 0 ; i < json.Entity[0].Instance.length ; i++){
        var idEvento = json.Entity[0].Instance[i].Attribute[0].Text;
        // só insere o evento no mapa se o mesmo não estiver lá
        if(!marcas[idEvento]){
            var id = "i"+idEvento;
            var pos = new google.maps.LatLng(
                json.Entity[0].Instance[i].Attribute[2].Text,
                json.Entity[0].Instance[i].Attribute[1].Text
            );
            var titulo = json.Entity[0].Instance[i].Attribute[3].Text;
            var tipo = json.Entity[0].Instance[i].Attribute[4].Text;
            // vai buscar imagem para o tipo do evento
            var foto = getImagemTipo(tipo);
            // criação de uma nova marca para inserção
            // no mapa
            marcas[id] = new google.maps.Marker({
                position: pos,
                map: map,
                title: titulo,
                icon: foto,
                conteudo:titulo+'<br /><a href="page15.do?kcond3.att19='
                    +idEvento+'">aqui</a>'
            });
            marcas[id].setDraggable(false);
            marcas[id].setClickable(true);
            marcas[id].info = new google.maps.InfoWindow();
            marcas[id].idev = id;
            // registo do comportamento para o click numa marca
            // do mapa
            google.maps.event.addListener(marcas[id],"click",
                function(mouse){
                    marcas[id].info.setContent(this.conteudo);
                    marcas[id].info.open(map,this);
                });
        }
    }
}
```

```

function novosEventos()
{
    var xmlhttp;
    // identifica fronteiras visíveis no mapa no momento
    var ne = map.getBounds().getNorthEast();
    var sw = map.getBounds().getSouthWest()
    // constrói pedido REST
    var pedido = "EventosService/mostraeventos/Area.do?LatMenor="
                + sw.lat() + "&LatMaior=" + ne.lat() + "&LonMenor="
                + sw.lng() + "&LonMaior=" + ne.lng();
    if (window.XMLHttpRequest)
    { // code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp=new XMLHttpRequest();
    }
    else
    {
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
    // indica o que fazer quando chegar a resposta do
    // servidor com o XML
    xmlhttp.onreadystatechange=function()
    {
        if (xmlhttp.readyState==4 && xmlhttp.status==200)
        {
            // converte XML resposta do servidor
            // para JSON, pois é mais fácil de tratar
            // com javascript
            var resp = xmlhttp.responseText;
            var json = XMLObjectifier.xmlToJSON(XMLObjectifier.textToXML(resp));
            // chama função que insere os eventos no mapa
            inserirNoMapa(json);
        }
    }
    // realiza pedido HTTP
    xmlhttp.open("GET",pedido,true);
    xmlhttp.send();
}

```


6.3. Código do Template da *Power Index Unit* alterado

```
#?delimiters [%, %], [%=, %]
<wr:LayoutParameter label="Show Header" name="show-header"
type="boolean" default="true">
Defines the rendition of the header on the top.
Allowed values are: true (default) or false.
</wr:LayoutParameter>
<wr:LayoutParameter label="Show Attributes Name" name="show-
attribute-name" type="boolean" default="false">
Defines the rendition of the attributes names.
Allowed values are: true or false(default).
</wr:LayoutParameter>
<wr:LayoutParameter label="Show Bullet" name="show-bullet"
type="boolean" default="true">
Defines the rendition of a bullet next to each line.
Allowed values are: true or false(default).
</wr:LayoutParameter>
<wr:LayoutParameter label="Use Alternate Rows" name="use-alternate"
type="boolean" default="false">
Defines the usage of alternates colours for each line.
Allowed values are: true or false(default).
</wr:LayoutParameter>
<wr:LayoutParameter label="Links Position" name="link-position"
type="enum" values="left|right|on_bullet|on_row" default="right">
Defines the rendition of the sub levels links.
Allowed values are:
- right (default): places the links on the right
- left: places the links on the left
- on_bullet: places the first link on the bullet
- on_row: places the first link on the attributes
</wr:LayoutParameter>
<wr:LayoutParameter label="Scrolling Position" name="scrolling
position" type="enum" values="top|bottom|both" default="top">
Defines where to position the scroll links.
Allowed values are:
- top (default): places the links at the top
- bottom: places the links at the bottom
- both: places the links at the top and bottom
</wr:LayoutParameter>
<wr:LayoutParameter label="Buttons Position" name="button-position"
type="enum" values="top|bottom|both" default="bottom">
Defines where to position the buttons to render submit links.
Allowed values are:
- top: places the buttons at the top
- bottom (default): places the buttons at the bottom
- both: places the buttons at the top and bottom
</wr:LayoutParameter>
<wr:LayoutParameter label="Show Jump Links" name="show-jump-links"
type="boolean" default="true">
Defines whether to show the jumps links of the scroller or not.
Allowed values are: true(default) or false.
</wr:LayoutParameter>
<wr:LayoutParameter label="Select All" name="select-all"
type="boolean" values="true|false" default="false">
Defines the rendition of a link that allows
to select or deselect all the checkboxes of
the unit.
Allowed values are: true or false(default).
```

```

</wr:LayoutParameter>
<wr:LayoutParameter label="Use Empty Unit Message" name="use-empty-
unit-message" type="boolean" default="false">
Defines the usage of a message for empty units.
Allowed values are: true(default) or false.
</wr:LayoutParameter>
<wr:LayoutParameter label="Empty Unit Message" name="empty-unit-
message" type="string" default="emptyUnitMessage">
Defines the key of the message to use if the unit is empty.
Default value: emptyUnitMessage
</wr:LayoutParameter>

[%import org.w3c.dom.Attr

import org.apache.commons.lang.StringUtils
import org.apache.commons.lang.math.NumberUtils

setHTMLOutput()

def unitId = unit["id"]
def blockFactor = unit["blockFactor"]
def isSortable = unit["sortable"]
def isScrollable = blockFactor != "" ? "true": "false"

def links =
unit.selectNodes("layout:Link").findAll{it["_exp"]!='t'}
def unitLink = links.empty ? null : links[0]
def unitLinkIsAjax = unitLink != null ? (unitLink["ajaxEnabled"]
=="true" && isAjaxPage(page)) : false
def link = unitLink?.valueOf("@link")
def myIndex = unit.hashCode()

def showHeader = params["show-header"]
def showAttributeName = params["show-attribute-name"]
def useAlternate = params["use-alternate"]
def showBullet = params["show-bullet"]
def linkPosition = params["link-position"]
def buttonPosition = params["button-position"]
def showJumpLinks = params["show-jump-links"]
def scrollingPosition = params["scrolling-position"]
def selectAll = params["select-all"]
def useEmptyUnitMessage = params["use-empty-unit-message"]
def emptyUnitMessage = params["empty-unit-message"]

def firstLink =
unit.selectSingleNode("layout:Link[contains(@link, 'First')]")
def previousLink =
unit.selectSingleNode("layout:Link[contains(@link, 'Previous')]")
def blockLink =
unit.selectSingleNode("layout:Link[contains(@link, 'Block')]")
def nextLink =
unit.selectSingleNode("layout:Link[contains(@link, 'Next')]")
def lastLink =
unit.selectSingleNode("layout:Link[contains(@link, 'Last')]")
def isAutoExpandAjax = isAjaxPage(page)
def checkable = "true" == unit["checkable"] && "true" !=
unit["distinct"]

//returns true if the given link must be rendered as a form button
def isButton(link){
def navLink = getById(link["link"])

```

```

    if(navLink == null){
        return false
    }
    def linkParams =
navLink.selectNodes("LinkParameter").collect{it["source"]}
    if(linkParams.isEmpty()){
        linkParams =
getAutomaticLinkParameters(navLink).collect{it["source"]}
    }
    for(param in linkParams){
        if(param?.endsWith("Checked")){
            return true;
        }
    }
    return false
}

def buttonsCount =
unit.selectNodes("layout:Link").findAll{ isButton(it) }.size();
%]

[%
def getIconPath(linkLayout) {
    folder = linkLayout.parameters["icon-folder"]
    name = StringUtils.defaultIfEmpty(linkLayout.parameters["icon-
name"], linkLayout.contextElement?.attributeValue("name"))
    extension = linkLayout.parameters["icon-extension"]
    return getFilePath(folder, name, extension)
}
%]

[% if (useEmptyUnitMessage != "true") { %]
<c:if test="\${not(empty <wr:UnitId/>) and (<wr:UnitId/>.dataSize gt
0)}">
[% } else { %]
<c:choose>
<c:when test="\${not(empty <wr:UnitId/>) and (<wr:UnitId/>.dataSize
gt 0)}">
[% } %]
    <wr:Frame>
        <div class="plain <wr:StyleClass/>">
            <div class="plain PowerIndexUnit">
                <table cellspacing="1" cellpadding="2">

                    <!--Ínicio dos Comentários -->
                    <c:forEach var="current" varStatus="status"
items="\${<wr:UnitId/>.data}">
                        <c:set var="index"
value="\${status.index}"/>

                                <wr:Iterate var="attr"
context="unit" select="layout:Attribute">
                                    <tr>

                                        <wr:Visible
position="'index'">

                                            [% if ((linkPosition ==
"on_row") && (!links.empty)) {%]

                                                <td
class="<wr:StyleClass context="unitLink"/> link<c:if
test="\${<wr:UnitId/>.currentIndex eq index}">Current</c:if>[% if
(useAlternate == "true") { %]<c:if test="\${index mod 2 eq

```



```

0}">Alternate</c:if>[% } %] [%= attr["type"]%]">
                                <wr:Visible
context="unitLink" position="'index'">
                                <a
href="<wr:URL context="unitLink"/>" class="<wr:StyleClass
context="unitLink"/> link<c:if test="\${<wr:UnitId/>.currentIndex eq
index}">Current</c:if>[% if (useAlternate == "true") { %}<c:if
test="\${index mod 2 eq 0}">Alternate</c:if>[% } %]"
onclick="<wr:AjaxURL context="unitLink" />" [%if
(unitLink?.attributeValue("newWindow") == "true") { %}
target="_blank" [% } %]>
                                </wr:Visible>
                                <wr:Value/>
                                <wr:Visible
context="unitLink" position="'index'">
                                </a>
                                </wr:Visible>
</td>
[% } else { %]

                                [%
if( attr.attributes()[index].value == "att17") { %]
                                <td
style="background:#F1F1F1" class="<wr:StyleClass/> value<c:if
test="\${<wr:UnitId/>.currentIndex eq index}">Current</c:if>[% if
(useAlternate == "true") { %}<c:if test="\${index mod 2 eq
0}">Alternate</c:if>[% } %] [%= attr["type"]%]">

                                <wr:Value />

</td>

                                [% } else
if( attr.attributes()[index].value == "t"){ %]
                                <td
style="background:#F1F1F1" class="<wr:StyleClass/> value<c:if
test="\${<wr:UnitId/>.currentIndex eq index}">Current</c:if>[% if
(useAlternate == "true") { %}<c:if test="\${index mod 2 eq
0}">Alternate</c:if>[% } %] [%= attr["type"]%]">

                                <wr:Value />
                                </td>

                                [% } else { %]
                                <td
rowspan="3" class="<wr:StyleClass/> value<c:if
test="\${<wr:UnitId/>.currentIndex eq index}">Current</c:if>[% if
(useAlternate == "true") { %}<c:if test="\${index mod 2 eq
0}">Alternate</c:if>[% } %] [%= attr["type"]%]">

                                <wr:Value />

                                </td>

                                [% } %]

                                [% } %]
                                </wr:Visible>

```

```


                                </tr>
                                </wr:Iterate>

                                </c:forEach>

                                <!-- Fim dos comentários --->
                                </table>
                                </div>
                                </div>
                                </wr:Frame>
                                [% if (useEmptyUnitMessage != "true") { %]
                                </c:if>
                                [% } else { %]
                                </c:when>
                                <c:otherwise>
                                <wr:Frame>
                                <div class="plain <wr:StyleClass/>">
                                <div class="plain PowerIndexUnit">
                                <table>
                                <tr>
                                <td><bean:message
                                key="[%printJSPTagValue(emptyUnitMessage) %]" /></td>
                                </tr>
                                </table>
                                </div>
                                </div>
                                </wr:Frame>
                                </c:otherwise>
                                </c:choose>
                                [% } %]

```

6.4. Screenshots da Aplicação



Think it, Get it!

[Pesquisar Eventos](#) | [Autenticação](#)

> Autenticação

Login

Username

Password

Figura 17 Página de Autenticação.

Evento

Título Festa do Além Mar

Descrição Festa com muita dança e companheirismo a realizar na Georgita.

Tipo do Evento festivo

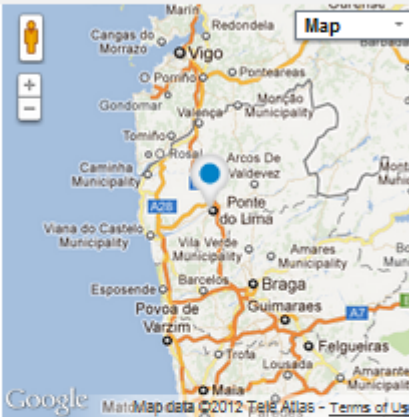
Início 1/1/70 9:30:00 AM

Fim 1/1/70 9:30:00 AM


Latitude 41.763

Longitude -8.59


[Marcar Como Favorito](#) [Editar](#) ou [Remover Evento](#)



Comentários



Muito Interessante.



Gostava Imenso de passar por lá!!!

Novo Comentário

Título


Corpo

Figura 18 Página que permite ver um evento detalhado para utilizadores autenticados.

Pesquisar Eventos Favoritos

> Gestão de Eventos > Criar Evento

Novo Evento



Título

Descrição

Tipo do Evento

Início

Fim

Insira as coordenadas gps ou seleccione um ponto no mapa.

Latitude

Longitude

ProximaNewsletter

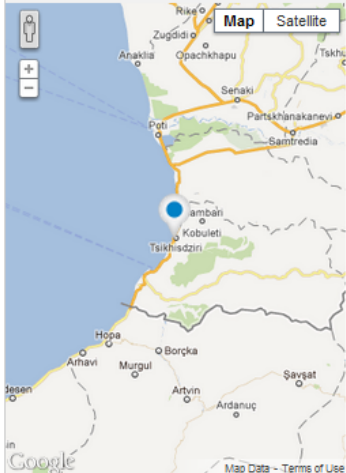
☐ yes ☒ no

Figura 19 Página de Criação de Evento.

Pesquisar Eventos Favoritos

> Gestão de Eventos > Página Edição de Evento

EditarEvento



Título

Descrição

Tipo do Evento

Início

Fim

Insira as coordenadas gps ou seleccione um ponto no mapa.

Latitude

Longitude

Figura 20 Página de Edição de Evento.

Pesquisar Eventos

Favoritos

> Favoritos

Favoritos

Título	Descrição	Início	Fim	Latitude	Longitude	
<input type="checkbox"/> Festa do Além Mar	Festa com muita dança e companheirismo a realizar na Georgia.	1/1/70 9:30:00 AM	1/1/70 9:30:00 AM	42.387	41.812	Desmarcar Favorito Ver Evento Detalhado

Figura 21 Página Favoritos.

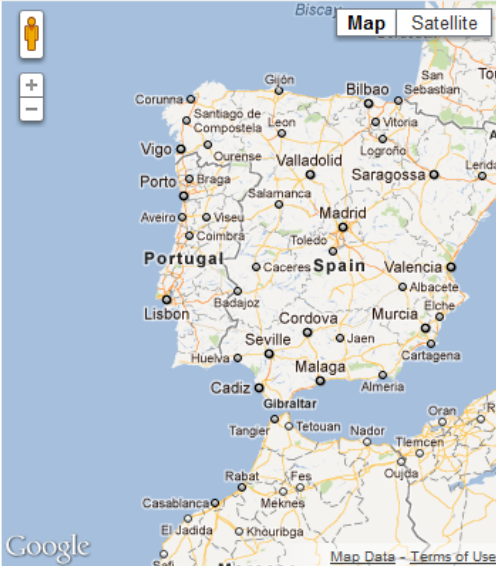
Pesquisar Eventos		Autenticação			
> Página de Registo					
Novo Utilizador					
Nome	<input type="text"/>				
Username	<input type="text"/>				
Email	<input type="text" value="rafael"/>				
Password	<input type="password" value="****"/>				
Reintroduza a Password	<input type="password"/>				
Localize zona de interesse no Mapa					
Latitude Menor	<input type="text" value="31.86906525977223"/>				
Longitude Menor	<input type="text" value="-14.22052001953125"/>				
Latitude Maior	<input type="text" value="45.5147369053"/>				
Longitude Maior	<input type="text" value="1.16033935546875"/>				
Fotografia	<input type="button" value="Choose File"/> No file chosen				
<input type="button" value="Sign Up"/>					

Figura 22 Página de Registo na Aplicação Web.

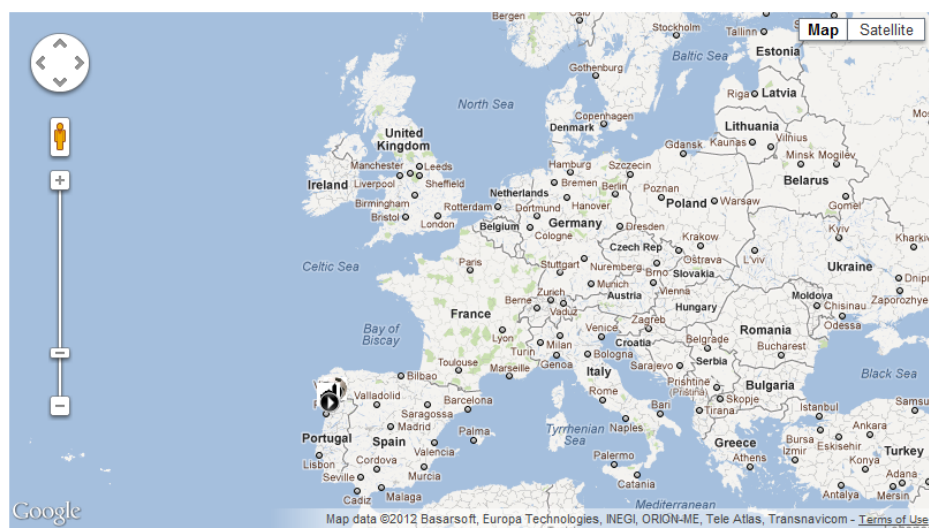


Figura 23 Página principal para utilizadores registados.

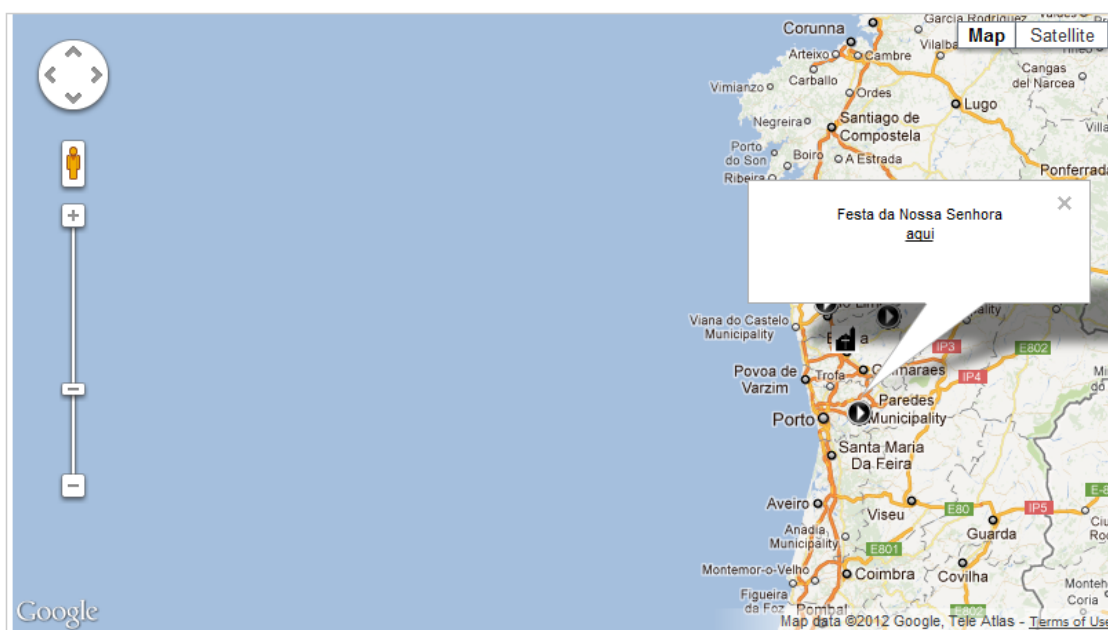


Figura 24 Página principal para Visitantes.