UNIVERSIDADE
LUSÓFONA

Concepção e Implementação de um Website com parte editável por CMS

Trabalho Final de curso

Nome do Aluno: Gonçalo Condeça

Nome do Aluno: Rafael Guerreiro

Nome do Orientador: Pedro Alves

Trabalho Final de Curso | LEI | 30/01/2017

Direitos de cópia

Contents

Índices de Imagens e Código Fonte

Resumo

Os websites hoje em dia são algo banal. Visitamos dezenas de websites diariamente, desde a simples leitura da capa de jornal online até à consulta do e-mail. Vivemos numa era digital e, como tal, temos acesso à informação mais facilmente. Não precisamos de nos deslocar a uma biblioteca para saber mais sobre algum tema, não precisamos de esperar pelas notícias da televisão, é realmente um facilitismo. Com isto, qualquer empresa ou associação quer estar online e ter a sua informação divulgada na internet de modo a abranger mais público.

Para a realização desta dissertação, que consistiu na Conceção e Implementação de um Website com parte editável por CMS, procurou-se fazer uma ligação entre a edição de conteúdo de um website com o facto de poder ser feito por pessoas com menos conhecimentos informáticos. Estes leigos conseguirão sem qualquer complexidade criar, editar, gerenciar e publicar conteúdo. Focámo-nos mais com o propósito destas funcionalidades serem realizadas pelo administrador do website. Com isto em mente decidimos criar o website para um clube de futebol, o Águias de Camarate. Tivemos ordem do Presidente do clube para avançarmos com este projeto.

De seguida, escolhemos a melhor maneira de conceber e implementar este website. Queremos ter dois tipos de páginas, estáticas e dinâmicas. Escolhemos dividir o projeto nestes dois tipos de páginas pois é nosso objetivo que o administrador edite tudo o que é texto através do Wordpress. Chamamos a estas páginas de estáticas. Relativamente ao facto de termos páginas dinâmicas, o que queremos realizar é a possibilidade do administrador conseguir inserir os resultados de cada jornada de todas as equipas da liga correspondente ao Águias de Camarate e com os dados introduzidos automaticamente gerar uma tabela de classificações ordenada por pontos e por diferença de golos. Para esta introdução de resultados implementámos o CRUD para uma maior flexibilidade de funcionamento por parte do administrador.

Decidimos optar por construir uma aplicação Web MVC com Framework Spring para a construção do website e das páginas dinâmicas. Relativamente às páginas estáticas optámos por usar um CMS que fosse amigável para a sua utilização por parte do administrador. Utilizámos o Wordpress como CMS por ser uma ferramenta conhecida, com muita informação, com a possibilidade de um plugin JSON que seria bastante útil na nossa transferência de conteúdo do Wordpress para o nosso website. Para implementarmos isto transformámos as páginas de Wordpress em JSON para podermos aceder ao seu conteúdo que guardamos como sendo um JSON Object e enviamos para o website através de WebService onde irá publicar o conteúdo na jsp correspondente. Demos então início à implementação do nosso website.

Palavras-chave: Website, CMS, páginas estáticas, páginas dinâmicas, JSON, Web MVC.

Abstract

Website nowadays are something so usual. We visit dozens of websites daily. From the simple newspaper cover to checking our email. We live in a digital era and, as such, we have easy access to information. We don't need to go out to a library to learn more about one topic, we don't need to wait for the news to come on television, it's really facilitating. With this, every company or association wants to be online and have their info shared by the web in a way to have more people with knowledge about their services.

For the realization of this dissertation, which consisted in the Conception and Implementation of a Website with an editable part through CMS, we tried to connect the edition of a website content with the fact of not being hard to do for a person with less informatic knowledge. This people will be able without any complexity to create, edit, manage and publish content. We focused with the purpose to make these functionalities able to the website admin. With this in mind we decided to create the website for a football team, Grupo Desportivo Águias de Camarate. We had the permission of the President of this institution to go through with this project.

Next, we choose the best way to concept and implement this website. We choose to divide this project into two types of pages because it's our goal that the administrator can edit everything that is text trough Wordpress. We call this pages static. Relatively to the fact that we have dynamic pages, what we hope to achieve is that the administrator has the possibility to insert the results of each journey of every team in the league that correspond to Águias de Camarate and with the data introduced, generate automatically ordered by points and difference of goals. For this introduction of results, we implemented the CRUD for a bigger flexibility of the administrator.

We wanted to have two types of pages, statics and dynamics. Our choice was to build a Web MVC application with Spring Framework for the build of our website and our dynamic pages. Entering the static pages topic, we decided that a CMS was the friendliest way to use by the admin. Wordpress was the chosen one since it is a well-known tool, with plenty of information, with the possibility to add a JSON plugin which would be very useful for our content transfer from Wordpress to our website. To implement this we transform our Wordpress pages into JSON so we can access their content which we will save as being a JSON Object and then sent to the website through WebService where it will be put into the correspondent jsp. We then started implementing our website.

Keywords: Website, CMS, static pages, dynamic pages, JSON, Web MVC.

1. Introdução

Começamos então por apresentar os principais aspetos do trabalho desenvolvido no âmbito da Conceção e Implementação de um website com parte editável por CMS. Apresentaremos desde o enquadramento do tema até aos métodos que utilizámos, chegando por fim aos resultados e conclusões sobre este trabalho.

O que nos motivou à realização deste projeto foi um aglomerado de razões. A principal razão foi o facto de querermos fazer algo que seja um contributo para alguma instituição e não só mais um Trabalho Final de Curso que após concluído se guarde no arquivo e seja esquecido. Como tal, queremos dar continuidade ao funcionamento deste website. Outra das razões foi o facto de estarmos familiarizados com o Águias de Camarate e a descontinuidade e falta de atualizações do seu antigo website. Por fim, mas não menos importante, será o tema que mais despertou o nosso interesse ao longo deste curso de Engenharia Informática e uma das nossas cadeiras mais interessantes ter sido Engenharia de Software onde realizámos também um website, embora com menos funcionalidades e menos complexo comparado com este trabalho.

Na realização deste projeto deparámo-nos com algumas adversidades. Adversidades essas que nos puseram a pensar sobre o funcionamento precário e a falta de informação disponibilizada por algumas ferramentas e seus criadores. Começámos por nos informar sobre os diferentes tipos de CMS. Depois de alguma pesquisa recebemos um bom feedback sobre o Alfresco, plataforma de CMS que contém muitas outras funcionalidades. A habituação ao funcionamento do Alfresco foi demorada, e após alguma troca de emails com elementos da empresa e depois de muita pesquisa nos fóruns ficámos com a ideia de que o Alfresco tinha uma documentação muito precária, o que nos complicou o trabalho. Pensámos em alternativas e chegámos à conclusão que o Wordpress podia ser uma forte escolha, já que o mesmo é uma plataforma muito simples de usar para o administrador fazer as suas edições às páginas estáticas.

Ficámos com as páginas dinâmicas feitas como aplicação Web MVC e com as páginas estáticas feitas no Wordpress que iriam ser importadas de alguma forma para o nosso site. Para aumentar a complexidade do problema concluímos que podíamos passar as páginas estáticas, neste caso o seu conteúdo, para JSON. JSON é um serviço que traduz texto em linguagem humana para transmitir objetos de dados. O que nós queremos após algum estudo sobre JSON é obter o content da página e mostrá-lo no nosso site, ou seja, uma página era criada no Wordpress, onde podia ser editada e tudo o que foi referido acima, e o seu conteúdo seria mostrado na página correspondente do nosso site.

Para além do que foi referido supra, outros dos nossos requisitos do produto final serão:

- Consulta pública de informação de caráter geral sobre o clube (história, órgãos sociais, contactos, localização, como se tornar sócio;
- Consulta pública de planteis, resultados e classificação no campeonato dos escalões seniores, juniores, juvenis e iniciados;
- Introdução, modificação e eliminação de resultados por parte do administrador em forma de formulário;
- Autenticação de administrador;
- Criação de um projeto em Wordpress com a finalidade de suportar as páginas que iremos considerar como estáticas;
- Introdução e alteração da informação de caráter geral, através de uma interface amigável (tipo editor de texto);
- Manter minimamente semelhantes o aspeto do nosso Website e das páginas no Wordpress.

2. Enquadramento teórico

JSON [1] é o acrônimo de JavaScript Object Notation. Originalmente foi concebido para trabalhar com javascript mas atualmente é possível de ser utilizado em mais 20 linguagens de programação. É um formato leve para intercâmbio de dados computacionais. No nosso caso usamos o JSON nas nossas páginas de Wordpress para "agarrarmos" o conteúdo duma página, ou seja, o seu body para enviarmos para a nossa aplicação web.

PRETTY GSON [2,3] ou simplesmente GSON é uma Java API desenvolvida pela Google usada para converter objetos Java em objetos JSON. Também pode ser utilizado para converter uma string JSON em objeto Java que é o que acontece na nossa situação. Transformamos a nossa string JSON num objeto.

SPRING Framework [4,5] é uma plataforma de Java que fornece um suporte de infraestruturas para o desenvolvimento de aplicações em Java. Foi construída com base na ideia de dependency injection e inversion of control. Numa linguagem mais comum em vez de termos classes a criarem-se umas às outras e passarem-se de um lado para outro, temos um controller. Cada classe do controller declara as suas dependências e o Spring container resolve estes requisitos ligando tudo automaticamente.

WEB MVC (slides Engenharia de Software) contém vários parâmetros como o Model, View e Controller. No model temos os dados e o processamento dos mesmos. O view apresenta os resultados das operações. Finalmente o controller trata as ações do utilizador, como a consulta de uma página ou a submissão de um formulário. Tanto o controller como o model são classes java enquanto o view são jsp's.

CSS [6,7] (slides Engenharia de Software) é o acrônimo de Cascding Style Sheets e contém as regras para a apresentação do html no ecrã. Poupa muito trabalho pois consegue controlar o layout de várias páginas de Web de uma só vez. Veio resolver um grande problema do html que não foi concebido para conter tags de formatação da página web mas sim para a descrição do seu conteúdo. Os CSS podem ser incluídos de 3 maneiras Inline em que vamos dando instruções a cada linha como a queremos ver desenhada, Embebido serve para quando vamos usar Styles repetidamente para uma página mas que não são necessários para a maioria do nosso website e, através de ficheiro externo onde se encontra os Styles que vamos usar repetidamente em várias páginas do nosso website. No nosso projeto iremos utilizar os 3 tipos de CSS.

Bootstrap [8] é uma framework de html, css e javascript que pode ser usado como base para criar websites ou aplicações web. A sua maior qualidade é o facto de ser responsive, ou seja, adapta-se ao tamanho do ecrã e mantém o mesmo aspeto em todos os browsers. Ainda previne repetição entre projetos, adiciona consistência ao design e ao código entre projetos e programadores.

Wamp [9] ou WampServer é a compilação do web server Apache, PHP e MySQL. Basicamente precisamos do Wamp para contruirmos o nosso server local e instalarmos o wordpress.

Wordpress [10] é uma ferramenta online, open Source de criação de websites escrito em php. É provavelmente o CMS mais poderoso e mais simples existente hoje em dia. É no wordpress que criamos as páginas que queremos que sejam editáveis de uma forma simplista.

IntelliJ IDEA [11] é um IDE para o desenvolvimento de software computacional. Foi desenvolvido pelo JetBrains antigamente conhecido por IntelliJ.

Alfresco [12,13,14] é um CMS poderoso com um framework web portal para gerir conteúdos. Tem uma interface CIFS que dispõe de uma compatibilidade de ficheiros de sistema em sistemas

operativos Windows, web CMS que consegue virtualizar websites estáticos e aplicações Web através de Activity workflow, Apache Tomcat e Lucene.

Tomcat [15,16] é uma aplicação do Apache Software Foundation que executa Java Servlets e desenha as páginas Web que inclui código Java Server Page. Pode ser usado como produto sozinho com o seu web server interno ou junto com outros web servers incluindo Apache, Netscape Enterprise Server, Microsoft Internet Information Server e Microsoft Personal Web Server. O tomcat requer um Java Runtime Enterprise Environment.

MySQL [17] é o maior sistema de gestão de base de dados do mundo. Utiliza linguagem SQL como interface. Tem vindo a crescer cada vez mais graças à sua portabilidade, compatibilidade, excelente desempenho e estabilidade, a facilidade do seu uso e, a pouca exigência relativamente à quantidade de recursos de hardware que requer.

Derby [18], também conhecido como Apache Derby, é um subprojecto da Apache DB. É uma base de dados relacional, open source, implementada inteiramente em Java. Em comparação ao MySQL podemos dizer que é muito mais leve o que o leva a ser usado para coisas mais simples e rápidas.

Maven [19] é uma ferramenta que gere dependências e ciclos de vida de projetos de software. No fundo, facilita a compilação do código, unifica e automatiza o processo de geração do sistema, centraliza informações organizadas do projeto, incluindo dependências, e reforça as boas práticas de desenvolvimento. Falando um pouco sobre a estrutura de um projeto Maven temos sete diretórios que são, src/main/java onde se localiza o código fonte do sistema, src/main/resources em que encontramos arquivos auxiliares como propriedades, XML's e configurações, src/main/webapp contém jsp, html, javascript e css visto que estamos a falar de uma aplicação web, src/test/resources tem mas classes com testes unitários, src/test/resources está relacionado com o anterior visto que contém os arquivos auxiliares usados nos testes, ainda temos o pom.xml que é o arquivo que concentra as informações do nosso projeto e, por fim, o target que é um diretório onde fica tudo o que é gerado, todos os JARs, WARs, JavaDoc, entre outros.

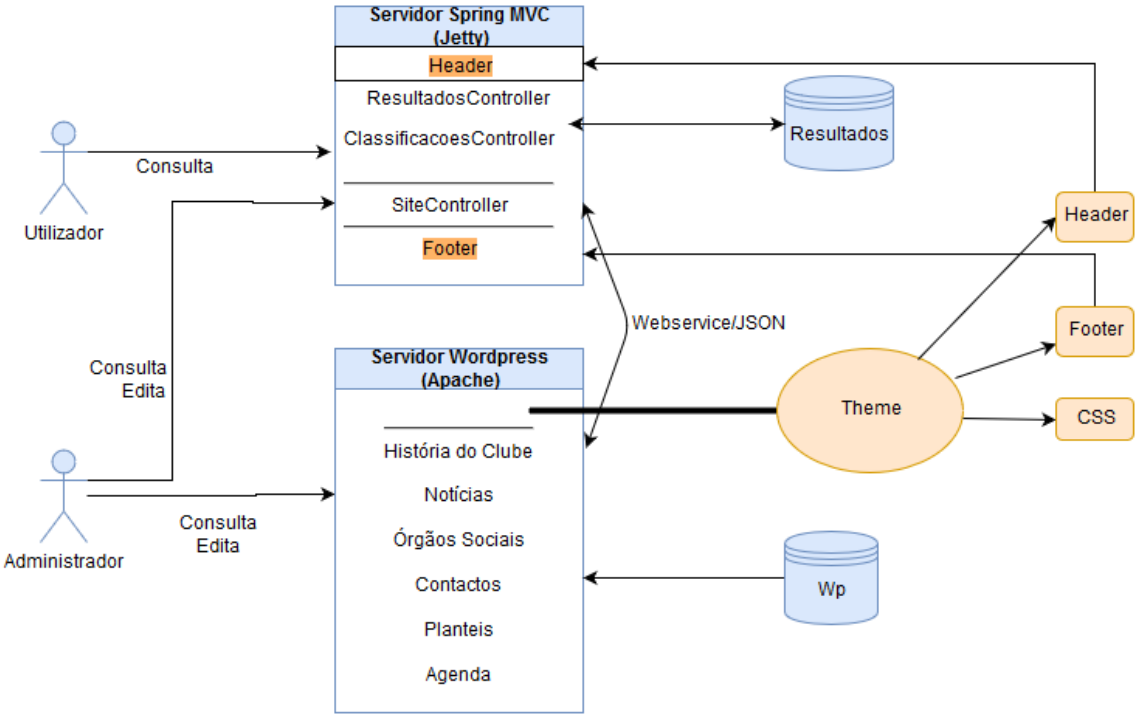
Spring Security [20] trabalha a segurança através de declarações baseadas em papéis. Não necessita de chamar nenhum método para realizar uma autenticação ou autorização. Através dos papéis definidos, podemos transmitir à aplicação web que recursos podem, ou não ser acedidos ou restringidos a um determinado utilizador.

3. Método

3.1 Planeamento

- Spring/Visualização (Páginas Dinâmicas) - 13 Outubro até 30 Outubro
- Wordpress/Edição (Páginas Estáticas) - 31 Outubro até 6 Novembro
- JSON/Webservices;
  Realizar ultimas edições;
  Aspeto Site = Aspeto Wordpress (theme) - 7 Novembro até 4 Dezembro
- Relatório - Dezembro até 23 Dezembro
- Revisão do site e relatório - 2 Janeiro até 15 Janeiro
- Preparar Apresentação (aspetos a realçar) - 16 Janeiro até 31 Janeiro

3.2 Arquitetura



O planeamento da arquitetura deste projeto faz-nos ter uma ideia mais concreta do que está a acontecer. Temos dois servidores a correr em simultâneo. Temos um servidor com Spring MVC onde estará alojado o nosso website e um servidor wordpress onde podemos encontrar as páginas estáticas e editá-las.

No que toca ao servidor com Spring MVC temos as páginas dinâmicas como resultados e classificações, e ainda a componente que vai buscar o conteúdo das páginas de wordpress para publicar nas páginas correspondentes. Este servidor está conectado com a base de dados criada no mysql. No servidor de wordpress estão contidas várias páginas a que designamos de estáticas. Estas páginas como a História do Clube e Contactos poderão ser editadas pelo administrador.

Como o wordpress para além de criar as páginas, cria-as num website e como tal existe uma opção de pré-visualização. Nisto pode-se escolher a framework a exibir embora este site de wordpress só esteja disponível para visualização por parte do administrador. Esta framework é como um theme onde podemos encontrar e editar o footer, header e todas as componentes deste site. Como tal, queremos deixar semelhante ao website original de forma a que o administrador consiga pré-visualizar as suas alterações sem ter de ir ao website, consultando assim somente o wordpres.

Ambos os servidores estão conectados um com o outro com a finalidade de o website ir buscar o conteúdo das páginas dinâmicas ao servidor de wordpress. Para isto utilizamos um webservice que vai efetuar o processamento e envia os dados para a nossa aplicação no formato JSON. Assim o cliente faz o pedido ao site para a visualização de uma página estática, o servidor Spring MVC recebe o pedido e vai requisitar o conteúdo da página ao servidor wordpress pelo webservice, que vai responder ao pedido com o conteúdo correspondente à página pedida pelo cliente num formato JSON que posteriormente será colocado no body da jsp que vai desenhar a página com o seu conteúdo já num formato texto entendido pelo utilizador.

3.3 Implementação

Comecemos por falar sobre a implementação das páginas dinâmicas e a elaboração do formulário de inserção de resultados e o tratamento da sua informação. Temos uma tabela equipas criada em excel que contém todas as equipas com uma coluna do escalão em que se encontra e com um id para cada equipa. Esta tabela é importada para o MySQL.

- Consulta pública de informação de caráter geral sobre o clube (história, órgãos sociais, contactos, localização, como se tornar sócio).

Esta consulta pública pode ser feita por qualquer utilizador acedendo ao nosso website. Estas páginas são criadas em Wordpress visto tratarem-se de páginas estáticas que podem ser editáveis pelo administrador do site. Dado isto, iremos posteriormente falar como são criadas estas tais páginas.

- Consulta pública de planteis, resultados e classificações no campeonato dos escalões seniores, juniores, juvenis e iniciados.

Estas são as páginas que consideramos dinâmicas visto se tratarem de páginas que serão mostradas conforme a introdução dos resultados por parte do administrador. Os planteis são estáticas pelo que serão criadas em wordpress. A consulta destes parâmetros todos estará disponível para o utilizador comum, exceto os resultados que só serão consultados e editados pelo administrador.

- Autenticação de administrador.

Este processo de autenticação e autorização é realizado no dispatcher-servlet.xml. Aqui situam-se todas as ferramentas necessárias, tanto os dados de login (user_name password) como o redireccionamento e autorização de acesso a páginas. Para todo este serviço de autenticação e autorização funcionar corretamente são necessárias dependências, que se situam no ficheiro pom.xml. O Spring Security usa também o Spring Security Filter, que é responsável pela interceção de alguns padrões URL. Este Filter tem como objetivo aplicar a autenticação e a autorização conforme o ficheiro de configuração de segurança do Spring.

- Inserção, modificação e eliminação de resultados por parte do administrador em forma de formulário e respetivas classificações.

A implementação de resultados é realizada só e apenas pelo administrador após a sua autenticação. O administrador consegue gerir todos os dados relativos a resultados e, por consequente, as classificações.

O menu "Administrador" possui três opções distintas: Inserir Resultados, Listar Resultados, Listar Classificações.

Esta páginas são criadas através dos Controller.java. É aqui que existem os métodos GET, que permitem obter dados do servidor (por exemplo, quando acedemos a URL diretamente ou clicamos num link), e POST, que permitem escrever dados no servidor (por exemplo, quando submetemos um formulário).

A criação das páginas "Inserir Resultados" e "Listar Resultados" estão presentes no ResultadosController.java. Aqui são criados: formulários e principais restrições desses mesmos formulários, a tabela "Resultados" e respetivas funcionalidades (Editar e Remover resultados), e todos os cálculos relativamente às Classificação existentes.

Estes dois últimos parâmetros são realizados neste controller.java dado que quando é inserido o primeiro resultado na Base de Dados, as tabelas "Resultados" e "Classificações" são criadas com as respetivas equipas. Se for adicionado novamente um resultado, já não existe criação de novas tabelas, mas sim a atualização das mesmas. Se algum resultado precisar de ser atualizado, os dados das tabelas "Resultados" e "Classificações" serão também atualizados.

O formulário existente para "Inserir Resultados" é criado no ResultadosController.java, assim como a tabela "Listar Resultados". É nas jsp's "escalaoForm" e "resultadosForm" que se localizam os formulários. No "resultadosList" é mostrada a tabela ao Administrador.

A criação da página "Listar Classificações" está presente, por sua vez, no ClassificacoesController.java. É aqui onde é criada a tabela "Classificação", mas é na jsp "classificacaoList" onde são mostradas todas as colunas necessárias.

Esta tabela requer uma atenção especial, pois para o seu conteúdo foram realizados alguns cálculos. Estes cálculos localizam-se no ClassificacoesController.java e possibilitam não só uma classificação com dados corretos (número de jogos, pontos, vitórias, golos, etc.), como também nos indica a posição exata de cada equipa consoante estes mesmos dados.

Quando o administrador insere um resultado, o formulário existente possui três campos. São eles escalão, equipas e resultados. Quando é escolhido o escalão, apenas aparecem as equipas correspondentes a esse escalão. Este processo é realizado através de um Query (Source Code - 5) que acede à nossa tabela "Equipas" presente na Base de Dados, onde passamos o parâmetro correspondente ao escalão pretendido. O não preenchimento correto de todos estes campos, levará ao aparecimento de mensagens de erro.

Para a alteração de um resultado é realizado também um Query (Source Code - 6) que permite aceder à tabela "Resultados", mais propriamente ao resultado pretendido, e alterá-lo. Esta alteração é realizada também através de um formulário, tal como quando inserimos um novo resultado. Após o correto preenchimento de todos os campos, a alteração relativamente a esse resultado é efetuada e são atualizadas, tanto a tabela "Resultados" como também a tabelas "Classificações".

Para apagar um resultado é ligeiramente semelhante. Realizamos exatamente o mesmo Query (Source Code - 6) mas a finalidade é totalmente diferente, isto é, acedemos à tabela "Resultados" e, ao invés de optarmos pela sua alteração, prosseguimos com a sua eliminação. Isto leva a que as tabelas "Resultados" e "Classificações" sejam atualizadas, ou seja, resultado eliminado e classificação revertida, respetivamente.

Nas jsp's criadas, usamos diferentes métodos de implementação de CSS conforme vamos necessitando. Nestes casos usámos o método inline para instruções simples. Como podemos ver no source code 4, alinhamos ao meio, adaptamos a largura de label's, entre outras alterações.

- Criação de um projeto em Wordpress com a finalidade de suportar as páginas que iremos considerar como estáticas.

Para a criação do projeto em wordpress tivemos que o instalar localmente através do wamp (9). Instalamos o wamp, vamos no browser ao phpMyAdmin onde criamos a base de dados que o wordpress vai usar. Essa base de dados é logo importada para o mySQL. Acedemos ao http://localhost/wordpress visto que wordpress foi o nome que atribuímos. Aí configuramos o wordpress, tal como a base de dados criada previamente, o user e password que desejamos. Após isto acedemos ao wordpress onde podemos começar a criar páginas, menus, entre outros.

- Introdução e alteração da informação de caráter geral, através de uma interface amigável (tipo editor de texto).

Para as páginas estáticas limitamo-nos a criar a página no SiteController.java que como se percebe pelo nome é o nosso controller referido na explicação de MVC. Fazemos o controller que contém a função para a criação da página. Como o método para a criação de páginas estáticas é semelhante para todas e a única coisa que se altera é o url do wordpress e o nome da jsp, vamos então apresentar um exemplo genérico que vai servir para todas. Para a obtenção da nossa página wordpress num formato JSON que irá ser falado mais à frente, só é possível devido ao wordpress disponibilizar a introdução de um plugin denominado JSON API [https://pt.wordpress.org/plugins/json-api/].

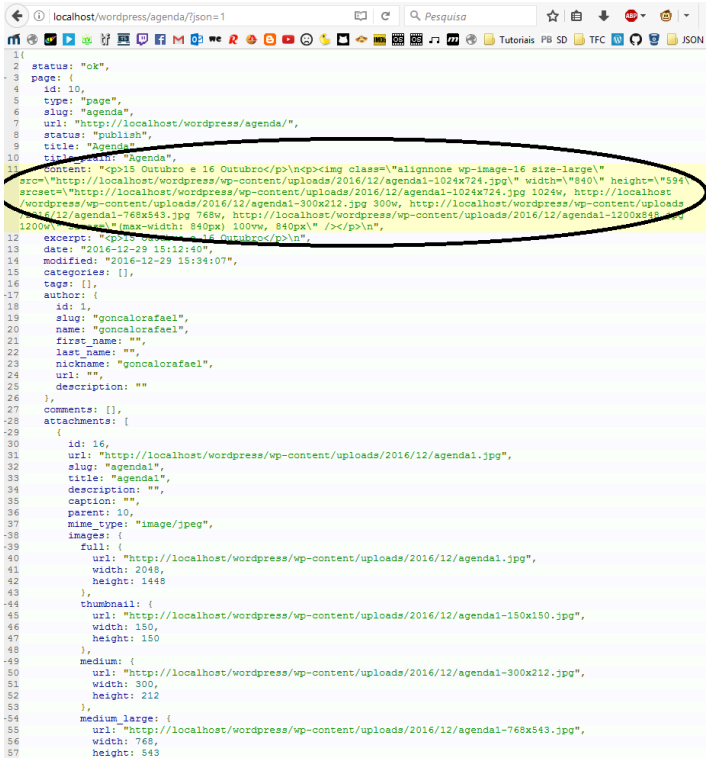


Imagem 1 – Exemplo de Página em JSON

Após a criação do nosso controller como visto no Source Code 3, onde definimos a nossa variável que vai ser uma String chamando a nossa classe GetPage.java onde está definida a função getContent que guarda o url como variável JSONObject e a passa para String, transformando essa String JSON num preetyGSON para melhor perceção e caminhando dentro do nosso objeto até chegar ao content. Damos return desse content e no nosso controller fazemos um model.put("content", content) onde metemos o content no sitio correto. Retornamos a nossa jsp que desenha a nossa página colocando o content no body desta. A criação da jsp demonstrada no Source Code 2 desenha uma página geral para todo o site. Com o mesmo footer, o mesmo header e só muda o title e o content de página para página. No caso das páginas estáticas temos o mesmo layout e definimos o title que queremos dar à página e fazemos um ciclo que faz com que sempre que haja um content diferente publica-o na página fazendo com que cada vez que haja uma alteração de conteúdo no wordpress não seja necessário reiniciar o servidor Spring MVC. As páginas que seguem este formato serão as seguintes: História do Clube, Agenda, Notícias, Órgãos Sociais, Contactos e Plantéis de cada escalão.

Para obtermos o conteúdo JSON do servidor wordpress, referido supra, procedemos da seguinte forma. A classe GetPage.java trata de tudo. Temos um String Builder para construir a nossa String. Temos a função readJsonFromUrl() que lê como input o url, lê o texto em JSON desse url. guarda-o num objeto JSON. Objeto esse que vai ser tratado na função getContent() onde passa o texto do url para uma variável JSONObject, passa essa variável para String que é transformada de JSON para prettyGson. Dentro da nossa String temos vários campos até chegar ao contente. Para ser mais específico temos de entrar no map, depois o page, novamente map e por fim conseguimos obter o content. Retornamos esse content também como String.

- Manter minimamente semelhantes o aspeto do nosso Website e das páginas no Wordpress.

Após alguma investigação sobre frameworks de wordpress conseguimos encontrar um semelhante ao nosso website. Decidimos então utilizar o Tiny Framework. Claro que tivemos que fazer algumas alterações para manter a semelhança entre os dois. Adicionámos a imagem do header do site no header do wordpress, fizemos o mesmo para o footer que é onde se encontram os patrocínios. Mudámos a cor de fundo para a cor do nosso website. Remodelámos o menu de forma a aparecer todas as funcionalidades do nosso website embora alguns botões do menu não levem a lado nenhum. Conseguimos deixar minimamente parecido de forma a o administrador conseguir pré-visualizar as suas alterações sem sair do wordpress.

4. Resultados

Começamos por analisar os resultados finais com uma pequena demonstração do que é o aspeto do wordpress, onde trabalhámos as páginas estáticas.

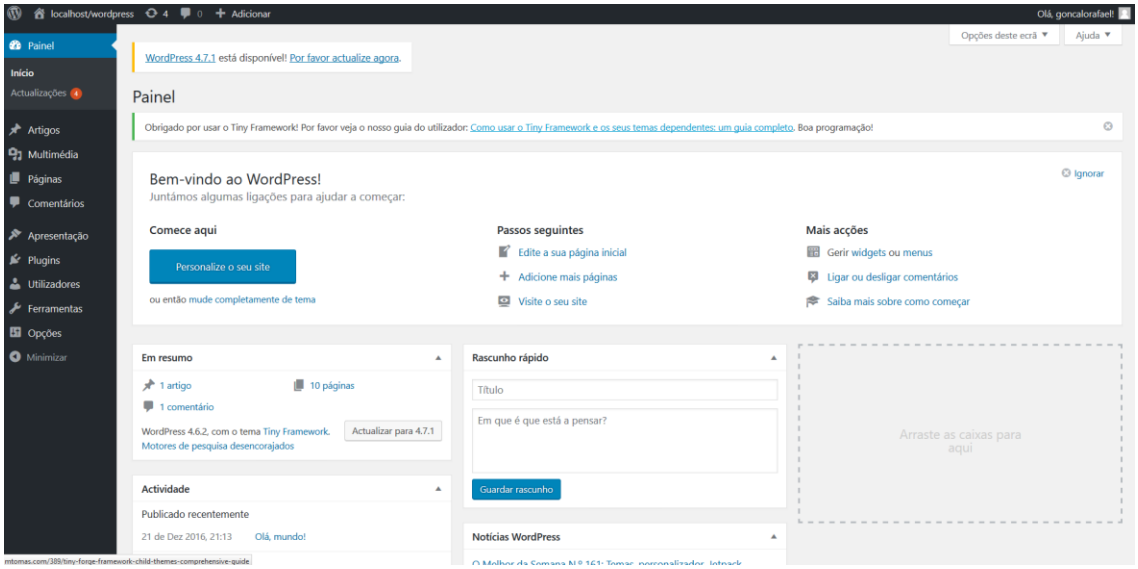


Imagem 2 – Aspeto Wordpress

Aqui podemos ver a página inicial para onde o Wordpress nos manda para o início da conceção do nosso site ou, no nosso caso, das páginas estáticas.

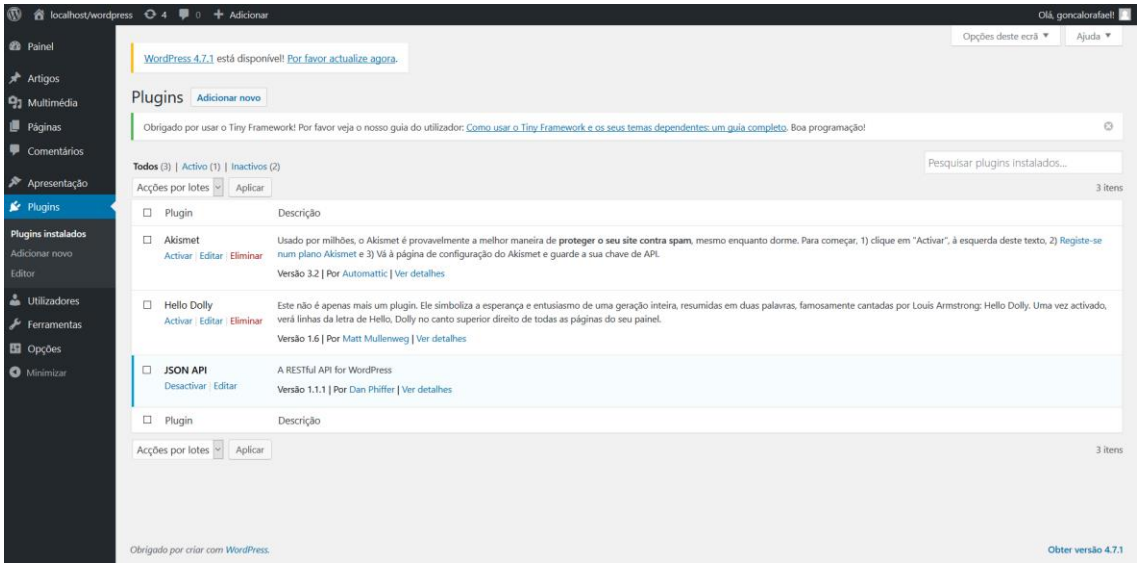


Imagem 3 – Plug in's Wordpress

Nesta segunda imagem reparamos onde é possível adicionar plug in's para dar outras funcionalidades ao nosso Wordpress.

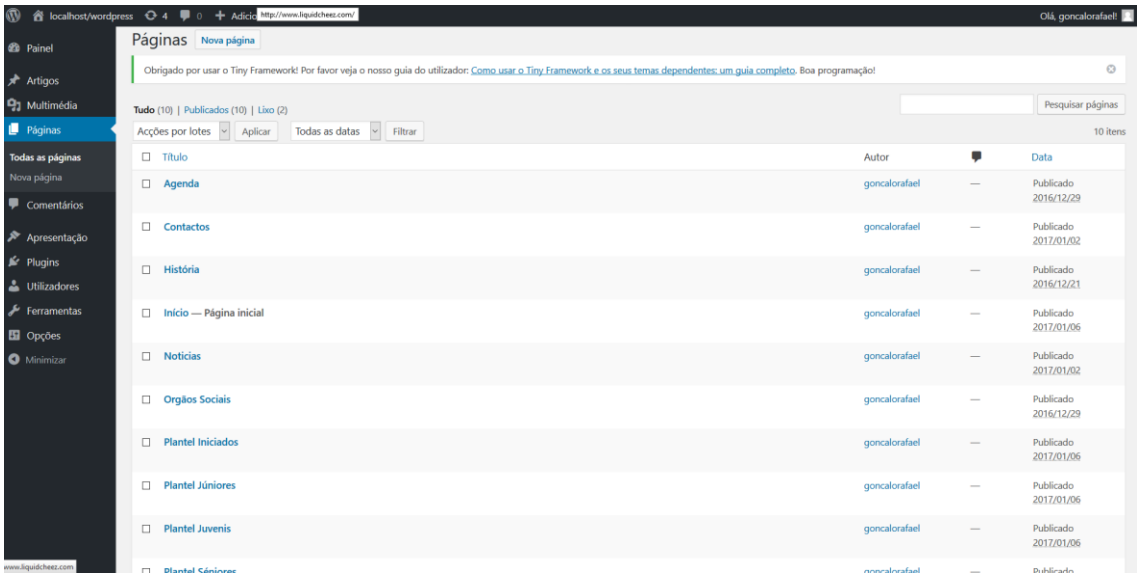


Imagem 4 – Páginas Wordpress

Na tab das páginas podemos ver todas as páginas que já criámos, onde podemos criar novas e onde as podemos editar simplesmente clicando em cima delas. Para a edição leva-nos para a figura seguinte onde se pode ver que podemos editar tudo, desde o título, ao texto, até mesmo ao url da página correspondente. Podemos ainda fazer uma pré-visualização das alterações.

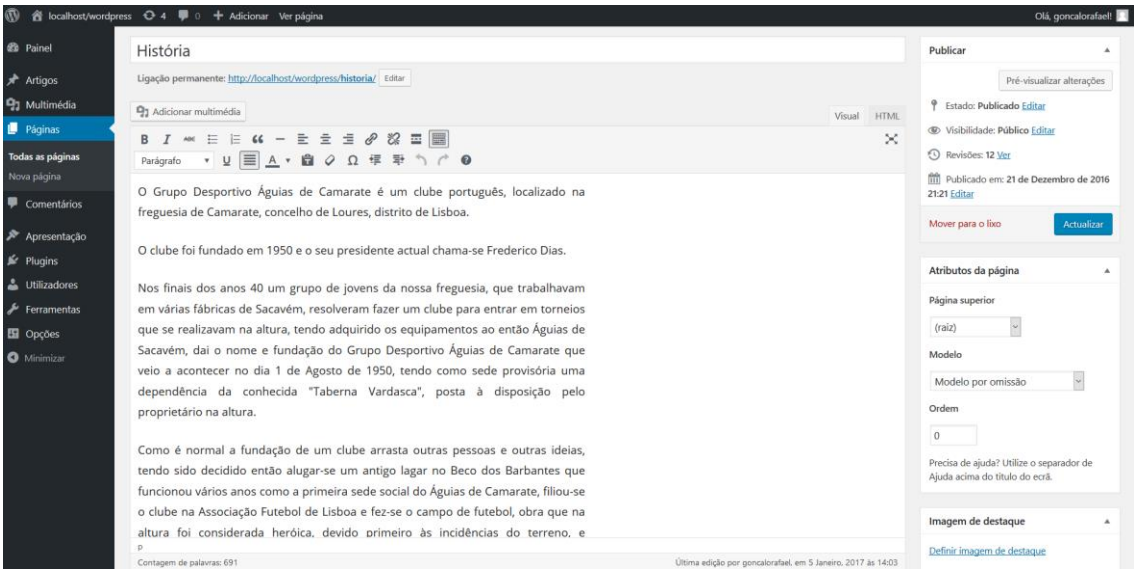


Imagem 5 – Exemplo de Página Wordpress

Imagem 6 – Aspeto das Páginas Wordpress

Com o intuito de deixarmos as nossas páginas iguais ao nosso Website de modo a quando for feita a edição pelo administrador ele possa visualizar as suas alterações com um aspeto semelhante ao do Website. Para isto tivemos de editar widgets, menus, imagens e ainda que fazer umas alterações no Framework escolhido por nós. Como se pode verificar na imagem 6 podemos alterar tudo no framework escolhido. Temos os vários templates em uso como o header.php ou o footer.php e temos a possibilidade de os editar a todos de modo a apresentarmos a pré-visualização da maneira que achamos mais semelhante ao nosso Website.

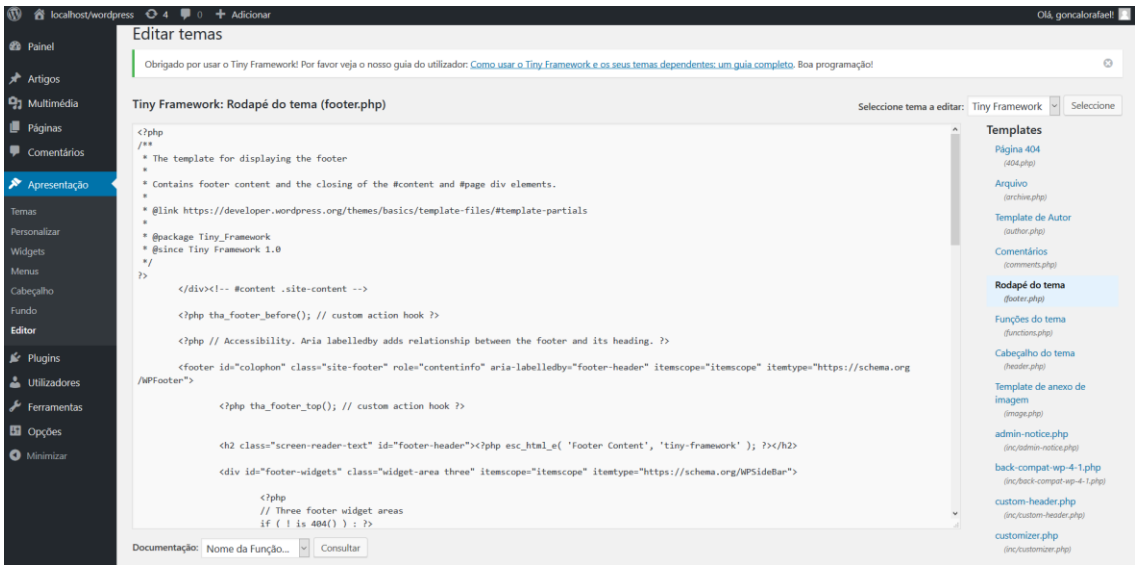


Imagem 7 – Tema do Wordpress

Agora iremos analisar o resultado final do nosso website.

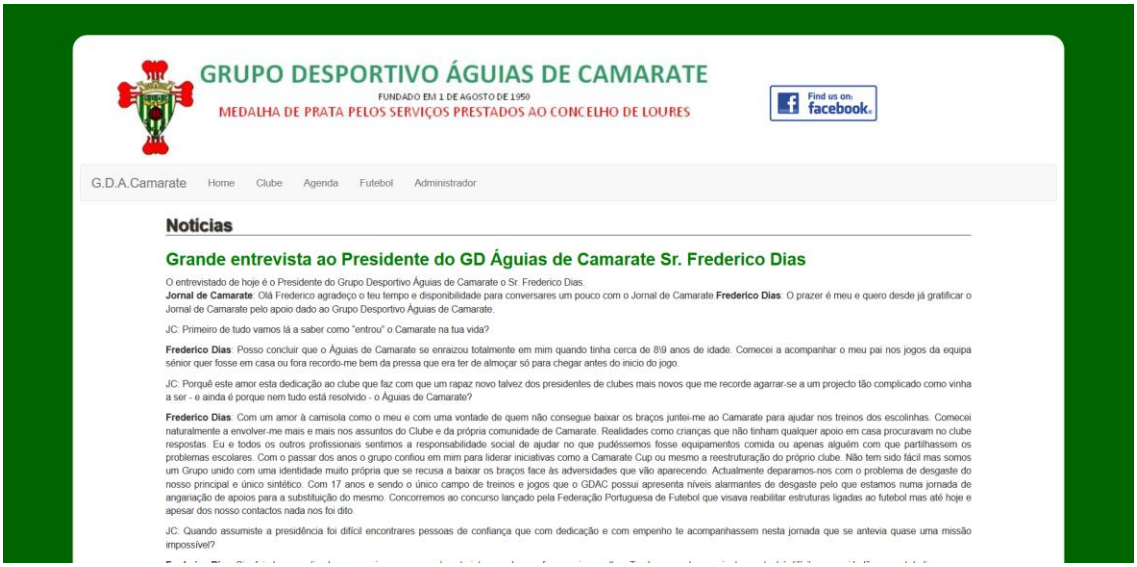


Imagem 8 – Home

Ao acedermos ao nosso site podemos verificar que nos leva logo para a página das notícias onde se encontram as ultimas notícias sobre o clube, alguns destaques de jogos, entre outras.

Imagem 9 – Autenticação

Para a inserção, edição e eliminação de resultados temos de fazer login como administrador. Só aí poderemos ter a acesso a este tipo de configurações. Como se pode ver na imagem seguinte, o administrador tem à sua disposição um menu para facilitar a sua perceção do que pode fazer dentro do Website relativamente aos resultados e às classificações.

Imagem 10 – Menu Administrador

Imagem 11 – Escolha do Escalão

Para o administrador inserir resultados primeiramente terá de escolher o escalão correspondente ao resultado que deseja inserir. Só após a submissão do escalão é que poderá introduzir as equipas do jogo que quer registar e o seu resultado. No fim irá aparecer uma mensagem de que o resultado foi introduzido com sucesso e aí poderá listar os resultados para saber o que já inseriu e os resultados que terá ainda de elaborar.

Imagem 12 – Formulário de Resultado

Após a inserção dos resultados, se acedermos no menu Futebol a um dos escalões poderemos consultar o plantel, criado em Wordpress, e ainda a classificação das equipas do escalão correspondente. Esta consulta pode ser feita não só pelo administrador como pelo público em geral.

Imagem 13 – Tabela de Classificações

5. Conclusões e trabalho futuro

Neste trabalho abordámos um problema existente num clube de futebol, o Aguias de Camarate. Este problema consistia na falta de informação e atualizações no site desta instituição desportiva. Em consenso e com a aprovação do presidente do clube criámos um novo site perfeitamente funcional e com a possibilidade de dar continuidade ao mesmo. Continuidade essa que foi um dos nossos objetivos com a realização deste projeto. Num futuro próximo iremos reunir com o presidente do Águias de Camarate para chegar a um acordo para a manutenção do website.

Cumprimos todos os objetivos que nos foram propostos e a que nós próprio nos propusemos, executando o website com a ajuda das várias ferramentas que tínhamos ao dispor, ferramentas essas que obtivemos o conhecimento através de cadeiras do nosso curso e claro, com alguma pesquisa na internet.

Foi um projeto muito interessante de se realizar, pois foi muito importante para a evolução do nosso trabalho prático bem como para o aperfeiçoamento do nosso conhecimento sobre várias ferramentas que poderão ser uteis para o nosso futuro profissional. Esperamos com este projeto obter uma primeira experiência no mercado de trabalho através da manutenção do mesmo.

Bibliografia

Slides da cadeira de Engenharia de Software do ano letivo 2015/2016.

1 - http://desenvolvimentoparaweb.com/javascript/json-javascript-object-notation/

2 - http://www.javacreed.com/simple-gson-example/

3 - https://sites.google.com/site/gson/gson-user-guide#TOC-Overview

4 - http://docs.spring.io/spring/docs/current/spring-framework-reference/html/overview.html

5 - http://stackoverflow.com/questions/10179260/spring-framework-in-simple-terms

6 - http://www.w3schools.com/css/css_intro.asp

7 - http://www.boogiejack.com/CSS_3.html

8 - https://www.taniarascia.com/what-is-bootstrap-and-how-do-i-use-it/

9 – http://www.wpbeginner.com/wp-tutorials/how-to-install-wordpress-on-your-windows-computer-using-wamp/

10 - https://ithemes.com/tutorials/what-is-wordpress/

11 - https://en.wikipedia.org/wiki/IntelliJ_IDEA

12 - http://www.cygnet-infotech.com/blog/all-you-need-to-know-about-alfresco-content-management-system

13 - https://community.alfresco.com/

14 - http://docs.alfresco.com/3.4/concepts/gs-wcm-setup-project.html

15 - http://www.theserverside.com/definition/Tomcat

16 - https://federicoponte.wordpress.com/2013/08/02/setting-up-tomcat-7-with-intellij-idea/

17 - http://www.devmedia.com.br/introducao-ao-mysql/28438

18 - https://db.apache.org/derby/

19 - http://luizricardo.org/2014/06/instalando-configurando-e-usando-o-maven-para-gerenciar-suas-dependencias-e-seus-projetos-java/

20 - http://imasters.com.br/artigo/16693/java/seguranca-passo-a-passo-com-spring-security-30/?trace=1519021197&source=single

Anexos (todo o source code do projecto deve ser colocado aqui)

```
public class GetPage {

    private static String readAll(Reader rd) throws IOException{
        StringBuilder sb = new StringBuilder();
        int cp;
        while ((cp = rd.read()) != -1) {
            sb.append((char) cp);
        }
        return sb.toString();
    }

    public static JSONObject readJsonFromUrl(String url) throws
IOException, JSONException {
        InputStream is = new URL(url).openStream();
        try {
            BufferedReader rd = new BufferedReader(new
InputStreamReader(is, Charset.forName("UTF-8")));
            String jsonText = readAll(rd);
            JSONObject json = new JSONObject(jsonText);
            return json;
        } finally {
            is.close();
        }
    }
    public static String getContent(String myURL) throws
JSONException, IOException {
        JSONObject json = readJsonFromUrl(myURL);
        json.toString();
        System.out.println(json + "\n");

        Gson prettyGson = new
GsonBuilder().setPrettyPrinting().create();
        String prettyJson = prettyGson.toJson(json);
        System.out.println(prettyJson + "\n");

        final JSONObject obj = new JSONObject(prettyJson);
        final Object content =
obj.getJSONObject("map").getJSONObject("page").getJSONObject("ma
p").get("content");
        return (content).toString();
    }
}
```

Source Code 1 - GetPage.java

```
<%@ page contentType="text/html;charset=UTF-8" language="java"
%>
<%@ taglib prefix="tiles" uri="http://tiles.apache.org/tags-
tiles" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

```
<tiles:insertDefinition name="defaultTemplate">
    <tiles:putAttribute name="body">
        <title>Exemplo</title>
        <div style="background-color: transparent; margin: 0px
50px 0px 50px; font-size: small" align="middle">

            <div style="margin: 0px 80px 0px 80px" align="left">
                <h3 style="border-style: solid; border-width:
0px 0px 1px 0px; text-shadow: black 1px 1px
1px"><strong>Exemplo</strong></h3>
                <div style="height: 80%; width: 80%">
                    <c:forEach var="content" items="${content}">
                        ${content}
                    </c:forEach>
                </div>
            </div>
        </div></br>
    </tiles:putAttribute>
</tiles:insertDefinition>
```

Source Code 2 – exemploForm.jsp

```
@RequestMapping(value = "/exemploForm", method = RequestMethod.GET)
public String exemplo(ModelMap model) throws IOException,
JSONException {
    String content =
GetPage.getContent("http://localhost/wordpress/exemplo/?json=1");
    model.put("content", content);
    return "exemploForm";
}
```

Source Code 3 – SiteController.java

```
<div align="middle">
    <div class="input-group" style="width: 350px">
        <span class="input-group-addon"><i class="glyphicon glyphicon-
user"></i></span>
        <input id="name" type="text" required="required" class="form-
control" name="j_username" placeholder="Utilizador">
    </div></br>
    <div class="input-group" style="width: 350px">
        <span class="input-group-addon"><i class="glyphicon glyphicon-
lock"></i></span>
        <input id="password" type="password" required="required"
class="form-control" name="j_password" placeholder="Password">
    </div><br>
    <p align="center">
        <input style="width: 100px" class="btn btn-default"
type="submit" value="Login"/></br>
    </p>
</div>
```

Source Code 4 – loginPage.jsp

A1. Anexo de Manual de utilizador (administrador) da aplicação

Criação de um resultado

Para criar um resultado tem de se logar como administrador e no menu administrador podemos selecionar a opção de inserção de um novo resultado. Depois é só preencher os campos de acordo com o resultado que se pretende.

Alteração e eliminação de um resultado

Para a alteração de um resultado já existente, acedemos à lista de resultados e posteriormente às informações desse mesmo resultado. Nesta página, estarão presentes duas opções. São elas o editar e o apagar.

Edição de páginas estáticas

Para editar páginas estáticas acedemos ao localhost/wordpress/wp-admin e aqui acedemos à tab lateral das páginas e escolhemos a página a ser editada. Esta edição é feita através de um editor de texto com uma interface amigável. Após a edição concluída é só publicar as alterações. Uma das funcionalidades é o facto de podermos criar uma hiperligação. É só selecionarmos o texto que queremos que seja uma hiperligação, clicamos no botão para inserir nove hiperligação, introduzimos o sítio da web para onde queremos levar e clicar em aplicar.

Glossário

CMS – Content Management System.

CRUD – Create Read Update Delete.

CSS – Cascading Style Sheets.

Gson – Biblioteca Java para serializar e descirializar objetos.

IDE – Integrated Development Environment.

HTML5 – Hypertext Markup Language versão 5.

HTTP – Hyper Text Transfer Protocol.

Jetty – Servidor HTTP e Servlet Container.

JSON – JavaScript Object Notation.

JSP – JavaServer Pages.

PHP – Hypertext Processor.

POM – Project Object Model.

SQL – Structured Query Language.

WAMP – Acrónimo para Windows, Apache, MySQL, PHP-Perl-Python.

Web MVC – Model View Controller.

WWW – World Wide Web.