Desenvolvimento para a Internet e Aplicações Moveis

ISCTE-IUL 23/24

A close up of food

Description automatically generated

Pastel de Nata

Salvador Palma, Tiago Palma e Martim Saldanha

**Base de Dados - Estrutura**

A diagram of a computer

Description automatically generated

[Observações]

* Tanto a classe Client como Fotos possuem campos que representam imagem, Picture e href respetivamente, os valores destes campos apresentam o caminho que leva as imagens correspondentes presentes na pasta static/media. De seguida encontram-se os formatos pelas quais estão guardadas as imagens:

Client.Picture: “Avatar{ClientID}”

Fotos.href: “Picture{EnterpriseID}-{generate\_random\_filename(20)}”

* A classe Enterprise possui uma função para definir se essa Enterprise se encontra valida para ser apresentada ao publico. Uma empresa que não cumpra os requerimentos necessários não se encontrara disponível na grelha de empresas do menu principal.
* A classe Enterprise possui dois campos, rating\_amount e rating\_average, que funcionam como campos calculados. Isto permite agilizar o processo de exibição na grelha principal, visto que não necessitamos de percorrer todas as reviews dessa companhia no instante.

**Users e as suas Funções**

**Anonimo/Desconhecido:** *(Conta para teste: sem conta)*

Um utilizador que não esteja logado poderá observar todas as companhias, as suas reviews e o perfil dos utilizadores que as fizeram. Todos os outros tipos de utilizador poderão também exercer estas funções.

**Individual:** *(Conta para teste: criar uma no momento)*

Um utilizador que esteja registado como individual poderá escrever e apagar a sua própria review. Ao escrever uma review para uma empresa que ja tenha avaliado, a review antiga vai ser substituída pela mais recente. Para alem disso, um utilizador pode editar o seu perfil, adicionando um avatar, biografia e descrição.

**Moderador:** *(Conta para teste: salvador.w.palma@gmail.com / paodequeijo123)*

Um moderador, para alem de exercer as mesmas funções de um Individual, pode apagar reviews de qualquer utilizador, com a finalidade de controlar utilizadores malcomportados. Quando e apresentado o username de um moderador, este terá a flag “[MOD]” a antecedê-lo

**Empresa:** *(Conta para teste: moleiro@gmail.com / moleiro)*

Uma empresa pode editar a sua própria página, adicionando fotos, descrição e morada. Note que uma empresa não poderá escrever reviews para outras empresas.

[Observações]

Na página principal, existe um botão (na divisória ondulada castanha) que possui um comportamento variável perante o tipo de user. Para uma empresa, este botão ira redirecionar para a página da companhia com maior rating no seu distrito. Já para um cliente individual, o botão ira assumir o distrito do utilizador (baseado nas suas reviews) e redirecionar para a página da companhia com maior rating nesse distrito que o cliente ainda não tenha avaliado.

**A Destacar/Notar**

* **Paginator:** No menu principal, existe uma grelha de empresas onde qualquer utilizador pode visualizar, pesquisar e filtrar empresas. Para reduzir a carga no cliente, apenas mostramos 8 resultados de cada vez, com a ajuda dum paginator. Este paginator também desativa os botões de passar para páginas anteriores e seguintes quando não é possível.
* **AJAX e Template Auxiliar:** Para realizar a atualização da grelha com base nas condições do utilizador começamos por implementar uma solução Ingénua: renderizar a template index.html novamente, passando os novos filtros e número de página como contexto. No entanto, esta solução fazia com que, para alem de re-renderizar certos elementos desnecessariamente (todas as secções alem da grelha), fosse pouco satisfatório para o utilizador, visto que este teria a sensação de ser redirecionado para o topo da página sempre que interagiria com a tabela. Para resolver isto, decidimos usar ajax para chamar uma template auxiliar (company\_table.html) e renderiza-la dentro do elemento #company-table sempre que existe uma interação com as condições da tabela.
* **Decoradores:** Em vez de bloquear funcionalidades com decoradores, como por exemplo, requerer que um user seja um individual logado para escrever uma review, escondemos os pontos de interação destas tarefas nas templates django. Com isto, queremos informar que não achamos necessário o uso de decoradores devido a natureza do nosso website, e que os usar tornar-se-ia desnecessário e redundante.
* **CSS e Ampersand (&):** No nosso ficheiro CSS, utilizamos o concatenador ampersand (&) de modo a podermos criar nested statments, melhorando assim, a leitura do código
* **CSS, JS e Transicoes:** Para alem de transições primarias como as onHover, também utilizamos JavaScript para criar uma maior relação entre os elementos e o utilizador. Isto pode ser observado no nosso header e background, que atualizam a sua estilização mediante a posição vertical na página, ou na página de login/registar, onde temos um painel deslizante com textos variáveis de acordo se nos encontramos no painel de login ou registar
* **Coesão com o design inicial:** De modo a facilitar o processo de formatação CSS, guiamo-nos por um design inicial feito em figma [1], de onde surgiu um resultado bastante fidedigno com o suposto

[1] Figma Prototype Link <https://www.figma.com/file/svpoMNsWNbRZTTCFPeT3b6/Pastel-de-Nata?type=design&node-id=0%3A1&mode=design&t=hcyGm3fppw16Yctf-1>