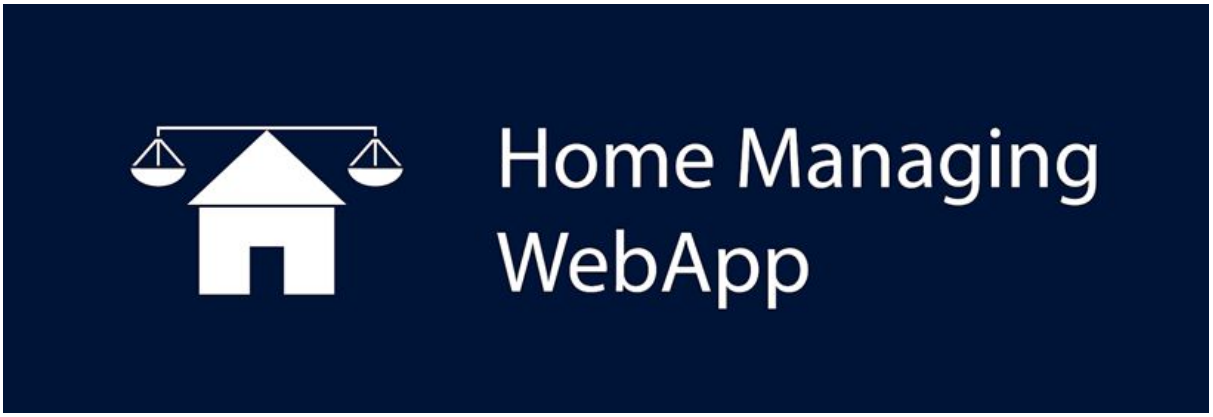
Projeto Prático

Programação II



Ema Guedes, nº 11280
Manuel Morais, nº 11337

Multimédia
Instituto Superior Miguel Torga

Para testar

Como **Administrador**

- Número de identificação: **15**
- Senha: **admin**

Como **morador**

- Número de identificação da Casa: **25**
- Senha: **adolfo**

Tema

A nossa aplicação web, Home Managing WebApp, tem como propósito ajudar na gestão de casas, principalmente de casas partilhadas. Quando falamos em “casas”, referimo-nos a todas as construções destinadas a habitação.

Esta aplicação foi primeiramente pensada para os estudantes em casas partilhadas. Esta seria uma aplicação que ajudaria os estudantes a saber quanto e quando é que têm de pagar as despesas. Também veriam as informações do administrador, caso o precisassem de contactar ou fazer transferência bancária. Já o administrador poderia apenas introduzir o valor das facturas das despesas e a aplicação faria automaticamente a conta de quanto é que cada um teria de pagar. Para além disso, poderia confirmar se uma pessoa já pagou ou não. Todos saberiam se já pagaram, enquanto o administrador veria se alguém ainda lhe está a dever o dinheiro das despesas.

Esta era a nossa ideia inicial, que fomos polindo e completando ao longo do caminho. Este é um trabalho que para estar super completo precisaria de mais tempo e trabalho.

Funcionamento

O único que precisa de criar uma conta é o administrador, enquanto os moradores apenas precisam do número de identificação da casa e da senha da mesma, que tem de ser previamente entregue ao morador. Ao entrar na sua conta, o administrador vê as casas que adicionou. Ao entrar nas casas ele pode adicionar quartos e ao entrar nos quartos pode adicionar pessoas. Ele pode adicionar os valores das despesas correspondentes às faturas, sendo somadas e divididas pelo número de pessoas da casa. O dinheiro que cada pessoa terá que pagar é o resultado dessa conta mais a despesa do quarto em que a pessoa está. Tanto os moradores, quanto o administrador conseguem ver uma tabela com as despesas da casa. Lá estão os valores das faturas associadas à casa , enquanto na pessoa já aparecem os valores divididos de cada valor. Na tabela, também está a data de quando as despesas foram publicadas e até quando é que os moradores devem pagar ao senhorio. Para além disso, o administrador tem um botão na tabela para confirmar se uma pessoa já pagou ou não as despesas daquela data, enquanto na tabela dos moradores aparece se eles já pagaram ou não. A informações do

administrador, da casa, do quarto e da pessoa podem ser editadas. Casa, quartos e pessoas podem ser eliminados.

Cores e estética

O nosso logo é de duas cores: branco e azul, pois é uma combinação estimulante que predispõe à simpatia e dá a sensação de segurança e estabilidade.

Utilizamos o amarelo para chamar a atenção do utilizador para os números de identificação da casa e do administrador, sendo o segundo apenas visível para o administrador.

Basicamente usámos variantes de azul, branco e preto, dando um aspecto *clean*, tal como uma aplicação destinada a gestão deve ter. A cor amarela foi a única que se desviou dessa paleta de cores.

Código

```php
//QUANTAS PESSOAS ESTÃO NUMA CASA
$npc = $db->query("SELECT COUNT(*) as n_linhas FROM quartos WHERE id_casas = ".$casa_id."")->fetch();
$n_quartos = $npc['n_linhas'];
$n_pessoas_casa=0;

for ($i=1; $i <= $n_quartos; $i++) {

    $infoquartos = $db->query("SELECT * FROM quartos WHERE id_casas = ".$casa_id." AND n_quarto_casa = ".$i."")->fetch();

    //n pessoas no quarto
    $n_pessoas = $db->query("SELECT COUNT(*) as n_pessoas FROM pessoas WHERE id_quartos = ".$infoquartos['id_quartos']." AND eliminado = 0")->fetchColumn();


    for ($x=1; $x <= $n_pessoas; $x++) {

        $n_pessoas_casa++;

    }
}
```

Este pedaço de código é usado muitas vezes pois é frequente ser preciso saber quantas pessoas existem numa casa.

```php
$npc = $db->query("SELECT COUNT(*) as n_linhas FROM casas WHERE id_admin = ".$id."")->fetch();
$n_casas = $npc['n_linhas'];

if ($n_casas == 0) {
    $n_final = 1;

} else {
    $n_final = ++$n_casas;
}
```

O sistema que criamos com as colunas de quantos quartos existem em casas e quantas pessoas existem nos quartos é uma das bases do site e a inserção funciona como se fosse um id diferente apenas para um certo tipo de linhas.

```php
$var = [
    'morada' => $_REQUEST['morada'],
    'pass' => $_REQUEST['pass'],
    'n_casas' => $n_final
];
$sql = "INSERT INTO casas(morada,senha_casa,id_admin,n_casa_admin) VALUES(:morada,:pass,".$id.",:n_casas)";
$stmt= $db->prepare($sql);
$stmt->execute($var);
```

Todos os Insert, Select e Update são básicos para o funcionamento da WebApp.

```php
$despesa_total_pessoa = round($total / $n_pessoas_casa,2);
$despesa_agua_pessoa = round($_REQUEST['agua'] / $n_pessoas_casa,2);
$despesa_eletr_pessoa = round($_REQUEST['eletricidade'] / $n_pessoas_casa,2);
$despesa_gas_pessoa = round($_REQUEST['gas'] / $n_pessoas_casa,2);
$despesa_net_pessoa = round($_REQUEST['internet'] / $n_pessoas_casa,2);
$despesa_segur_pessoa = round($_REQUEST['seguranca'] / $n_pessoas_casa,2);
$despesa_limp_pessoa = round($_REQUEST['limpeza'] / $n_pessoas_casa,2);
```

O cálculo das despesas (divisão pelo número de pessoas da casa (não eliminadas))

```php
$despesa_total_pessoa += ($infoquartos['preço'] / $n_pessoas_neli);
```

A adição do preço do quarto à despesa total (dividido pelo número de pessoas no quarto)

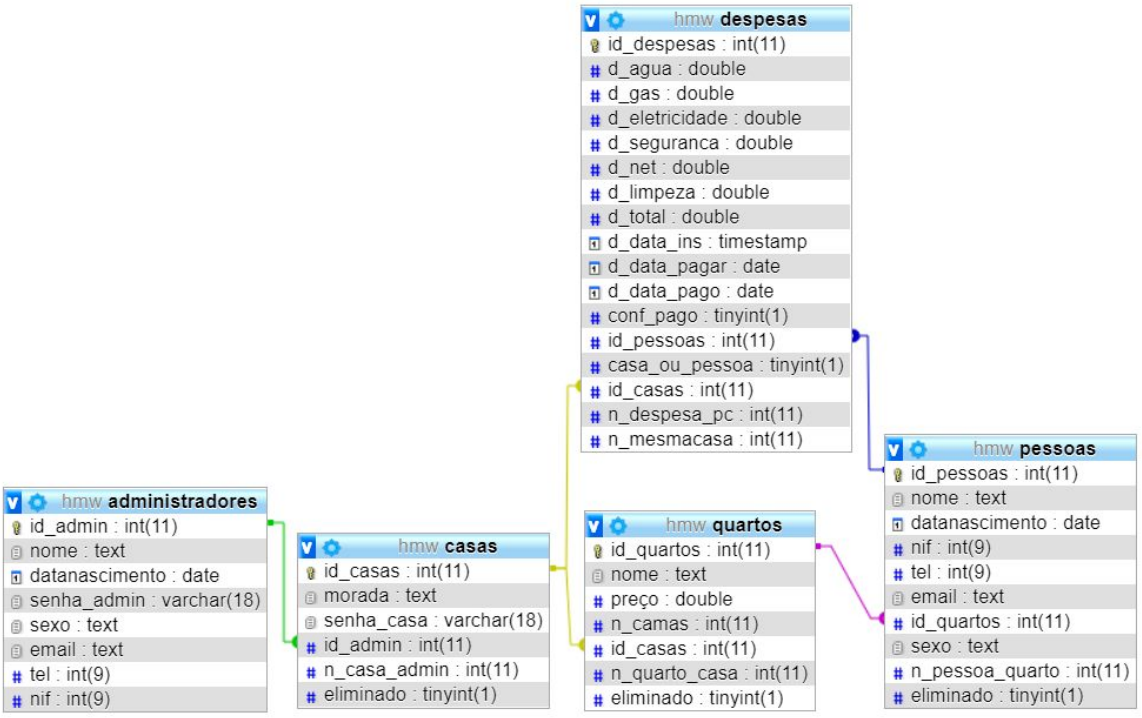
```php
for ($i=1; $i <= $linhas; $i++) {

    <?php
    $infoquartos = $db->query("SELECT * FROM quartos WHERE id_casas = ".$casa_id." AND n_quarto_casa = ".$i."")->fetch();
    $n_pessoas = $db->query("SELECT COUNT(*) as n_pessoas FROM pessoas WHERE id_quartos = ".$infoquartos['id_quartos']." AND eliminado = 0")->fetch();

    if ($infoquartos['eliminado'] == 0) {
    ?>

    <tr>
        <th scope="row">
            <form id= "<?php echo $i?>" action="quarto_entrar.php" method="POST" class="entrar" data-toggle="tooltip" data-placement="right" title="Entrar no Quarto"
                <input type="hidden" name="n_quarto" value="<?php echo $infoquartos['id_quartos'] ?>">
                <input class="btn btn-primary" role="button" type="submit" value="<?php echo $i; ?>">
            </form>
            <form action="eliminar_quarto.php" method="POST" class="close">
                <input type="hidden" name="n_quarto_e" value="<?php echo $infoquartos['id_quartos'] ?>">
                <input role="button" name="submit" type="image" aria-label="Close" src="imagens\_ionicons_svg_ios-trash.svg" alt="Eliminar Quarto" class="icon" data-
            </form >
            <form action="editar_quarto.php" method="GET" class="close">
                <input type="hidden" name="n_quarto" value="<?php echo $infoquartos['id_quartos'] ?>">
                <input role="button" name="submit" type="image" aria-label="Close" src="imagens\_ionicons_svg_md-create.svg" alt="Editar Casa" class="icon" data-togg
            </form>
        </th>
        <td><?php echo $infoquartos['preço']; ?>€</td>
        <td><?php echo $infoquartos['nome']; ?></td>
        <td><?php echo $infoquartos['n_camas']; ?></td>
        <td><?php echo $n_pessoas['n_pessoas']; ?></td>
p }};?>
```

As tabelas e a repetição de cada linha com a personalização de cada com informação única a cada casa/quarto/pessoa/despesa.

```php
<?php
            if ($infodespesas['conf_pago'] == 0) {
        ?>
        <form id= "<?php $infodespesas['id_despesas']?>" action="conf_pag.php" method="POST">
            <input type="hidden" name="n_despesa" value="<?php echo $infodespesas['id_despesas'] ?>">
            <input class="btn btn-primary" role="button" type="submit" value="Confirmar que foi pago">
        </form>
        <?php
            }else {
                echo "Despesa Paga!";
            }
        ?></td>
```

Verificação do pagamento das despesas (mostrar botão de confirmação de pagamento e mostrar a frase “Despesa Paga!”)

Base de Dados



Administradores

A tabela administradores guarda o id_admin e senha_admin que servem como nome de usuário e palavra passe para entrar na webapp e outras informações sobre os utilizadores (Nome, Data de nascimento, etc).

Casas

A tabela casas guarda o id_casas e a senha_casa que servem como nome de usuário e palavra passe para os moradores e como meio de identificação para as várias casas do administrador. A coluna n_casa_admin serve para contar cada casa pertencente a um certo administrador e a coluna eliminado serve para mostrar se a casa foi eliminada ou não..Esta também contém outras variáveis descritivas da casa.

Quartos

A tabela quartos guarda colunas descritivas e identificadoras como o id_quartos, nome e n_camas. O preço do quarto (preço) é adicionado aos residentes do mesmo (dividido se for mais que um residente). O n_quarto_casa serve para contar quantos quartos existem na mesma casa e o eliminado para verificar se este foi eliminado ou não.

Pessoas

A tabela pessoas guarda várias colunas descritivas e identificadoras (id_pessoa, nome, email, tel, etc). O n_pessoa_quarto é para contar quantas pessoas existem dentro desse mesmo quarto. O eliminado é para verificar se a pessoa foi eliminada ou não.

Despesas

A tabela despesas guarda todas as despesas das casas e das pessoas (calculadas através da despesa da casa). A coluna casa_ou_pessoa serve para verificar se a despesa pertence a uma casa ou pessoa. n_despesa_pc serve para ver quantas vezes é que cada despesa de uma certa pessoa ou casa repete e, n_mesmacasa para verificar quantas despesas são da mesma casa (incluindo pessoas diferentes e despesas apenas da casa).

A melhorar

Adicionar uma confirmação quando se pretende eliminar.

Acrescentar a opção de publicar imagens que comprovam os valores da factura de cada despesa.

As tabelas com as despesas estarem mais organizadas. Criar uma nova tabela sempre que o administrador adiciona despesas.

A data de quando foi pago ser colocada na base de dados e aparecer.

Adicionar setas na página que ao carregar numa iria para o fundo da página e a outra iria para o início da página, porque a página ficará muito longa ao decorrer do tempo. Nós tentamos fazer isso como o professor disse e como este site explica : https://www.alura.com.br/artigos/ancorando-elementos-com-html5 , mas apesar da seta de cima (que aponta para baixo e nos leva para o final da página) estar muito longe da hiperligação “Sair”, nós não conseguíamos sair. Quando tentávamos sair, simplesmente íamos para o fundo da página.

A página de erro ser mais bonita e ser mais específica em que erro.

Conclusão

O nosso projeto evoluiu muito na navegabilidade, organização de dados e também já não dá origem a erros, principalmente ao eliminar como acontecia no trabalho anterior. Acrescentámos a opção de editar e uma página com todas as despesas ao longo do tempo, tanto para o administrador quanto para os moradores. A confirmação de pagamento também foi melhorada. Para conseguirmos ter esta evolução, tivemos de alterar a base de dados anterior e acrescentar a tabela “despesas”, como foi sugerido pelo professor.

Desta vez, ficou tudo como planeámos. Sabemos que deveríamos de acrescentar a opção de publicar imagens que comprovam os valores da factura de cada despesa para deixar a aplicação mais completa, mas planeámos não o fazer, devido ao curto intervalo de tempo para entregar o projecto. O facto dos moradores entrarem apenas com o número de identificação da casa e com a sanha da mesma foi escolha nossa, pois não consideramos necessário a confirmação do administrador para o outro entrar na casa. Se um dos moradores abandonar a casa e o administrador não quiser que ele aceda mais à “casa”, apenas tem de

editar/mudar a senha da casa e enviar uma mensagem a todos os moradores da mesma, tendo em conta que ao adicionar uma pessoa, ele tem as informações suficientes para a contactar.

No final, este trabalho coincidiu com as nossas expectativas, ao contrário das versões anteriormente entregues.

Bibliografia:

https://www.w3schools.com/tags/tag_meta.asp : para sabermos que *<meta> Tags* usar no head.

https://developer.mozilla.org/pt-PT/docs/Utilizando_meta_tags : para sabermos que *<meta> Tags* usar para a data de criação e de última modificação.

https://getbootstrap.com/docs/4.3/extend/icons/ : onde encontrámos os links para os sites com os ícones.

https://ionicons.com/ : onde transferimos as imagens dos ícones.

https://getbootstrap.com/docs/4.3/layout/grid/ : para os *containers* e *grid*.

https://getbootstrap.com/docs/4.3/content/tables/ : para as tabelas.

https://getbootstrap.com/docs/4.3/components/badge/ : para os *spans*.

https://getbootstrap.com/docs/4.3/components/buttons/ : para os botões.

https://getbootstrap.com/docs/4.3/components/card/ : para as *cards* que usamos para fazer o footer. Foi lá que percebemos como mudar a cor das letras(*text-*) e dos fundos(*bg-*).

https://getbootstrap.com/docs/4.3/components/forms/ e https://getbootstrap.com/docs/4.3/components/input-group/ : para os *inputs*.

https://getbootstrap.com/docs/4.3/components/navs/ e https://getbootstrap.com/docs/4.3/components/navbar/ e https://getbootstrap.com/docs/4.3/examples/pricing/ : para a barra de navegação. No final, usamos praticamente o último link, mas os outros ajudaram-nos a perceber como é que se fazem as modificações que queríamos.

https://getbootstrap.com/docs/4.3/components/tooltips/ e https://getbootstrap.com/docs/4.3/components/popovers/ : para aparecer a informação correspondente ao elemento da aplicação web quando pousassemos o rato em cima, assim facilitando a navegação do utilizador.

https://getbootstrap.com/docs/4.3/utilities/close-icon/ : para quando o utilizador colocasse o rato em cima do icons, eles se destacarem, ficando mais escuros.

https://www.w3schools.com/css/tryit.asp?filename=trycss_color_names : onde encontrámos a cor "*DodgerBlue*" para equivaler à cor "*primary*" do *Bootstrap*. Ao usarmos o código do *Bootstrap*, no número de identificação da pessoa (na tabela), os ícones desciam uma linha. Esta foi a maneira encontrada para contornar essa situação.

.