**Escola Secundária de Silves**

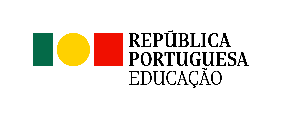
**Curso Profissional de Técnico de Informática - sistemas**

**<Título do Relatório da PAP>**

[Figura Apropriada]

<Nº 12285> <Ricardo Leandro Medina Cabrita>

**<Data>**



**Escola Secundária de Silves**

**Curso Profissional de Técnico de Informática - Sistemas**

**<data>**

**<Título do Relatório da PAP>**

**Relatório Final**

**da**

**Prova de Aptidão Profissional**

<Nº 12285> <Ricardo Cabrita>

Orientado pelos Professores: <Nome dos Professores>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Nome do Aluno) (Professores responsáveis)

**Agradecimentos**

Queria primeiramente agradecer ao meu tutor de estagio que eu acompanhei durante dois anos por ter me ajudado no desenvolvimento do projeto de aptidão profissional, ele que me ajudou na construção da aplicação.

Também queria agradecer aos meus pais pelo apoio deles.

E também queria agradecer a minha namorada por sempre me animar e de dar motivação para continuar a construir o projeto.

E queria agradecer a minha professora orientadora Mónica Franco pela ajuda e compreensão da parte dela.

**Resumo**

O projeto de aptidão profissional que realizei é um site de uma loja online para uma oficina de mecânica automóvel e agrícola sendo a função dela alem de fazer publicidade, demostra os serviços na oficina como é cada serviço, para também otimizar vendas assim um utilizador fazer a encomenda que deseja e a encomenda pode ser levantada na loja.

**Índice**

[Introdução 7](#_Toc171369006)

[Contexto 8](#_Toc171369007)

[Descrição técnica 9](#_Toc171369008)

[Ferramentas 9](#_Toc171369009)

[Linguagens de programação 10](#_Toc171369010)

[Desenvolvimento do projeto 10](#_Toc171369011)

[Análise do Sistema 26](#_Toc171369012)

[Desenvolvimento 27](#_Toc171369013)

[Avaliação 28](#_Toc171369014)

[Conclusão 29](#_Toc171369015)

[Bibliografia 30](#_Toc171369016)

[Anexo A - Listagem do material utilizado 31](#_Toc171369017)

**Notação** e **Glossário**

|  |  |
| --- | --- |
| **WWW** | World Wide Web |
|  |  |
|  |  |
|  |  |

«Esta secção apresenta os conceitos (glossário de termos) ordenados alfabeticamente, símbolos matemáticos e acrónimos **utilizados** no corpo do texto do relatório.»

# Introdução

Desenvolvi o meu projeto de aptidão profissional inspirado na oficina de carros e maquinas agrícolas do meu pai que com o objetivo de otimizar as vendas da oficina criei uma loja online de maneira a alcançar a atualidade do marketing não só vendas na loja mas também encomendas online e publicidade pela internet.

A minha motivação para o projeto foi para poder ajudar de alguma forma o meu pai na sua empresa para isso decidi criar uma loja online e uma maneira de mostrar como os serviços funcionam lá, o tema é sobre mecânica como é uma oficina mas é mais especializada na parte agrícola e é algo que me interesso bastante é a área de mecânica.

O meu objetivo com o meu projeto é um dia mais tarde se eu tiver oportunidade tornar publico na internet a loja online.

# Descrição técnica

Para o desenvolvimento da aplicação em questão utilizei um paradigma de desenvolvimento OOP-programação orientada a objetos, construindo uma framework tipo MVC

A utilização deste tipo de desenvolvimento possibilita atribuição de responsabilidades a cada parte do projeto. Numa breve descrição podemos identificar os seguintes pontos que fazer parte deste acrónimo:

* Modelo(**M)**: manipula as conexões do base de dados e passa informações para o controlador.
* View(**V**): este manipulador das visualizações do utilizador, como templates no front-end.
* Controlador(**C**): este manipulador controla toda a lógica negociável entre todas as partes, o Modelo e as Visualizações.

O MVC, resumidamente é são frameworks que seguem o padrão de, ao aceder a uma URL, a framework direciona a solicitação a um controlador específico, que então chama um método que executa uma ação correspondente, como exibir uma página ou processar um formulário.

* Controlador: A parte que manipula a lógica de fluxo da aplicação.
* Método: Função dentro do controlador que responde a uma ação específica.
* Ação: A tarefa específica que o método do controlador executa, geralmente em resposta a uma solicitação do usuário.

# Ferramentas

Para a realização do projeto, comecei por instalar as seguintes ferramentas:

* Visual Studio Code (VSC);
* Xammp;
* Repositório GitHub
* Git

O visual studio code é uma ferramenta de desenvolvimento (IDE) [1], que serve para desenvolvimento de código utilizando várias linguagens de programação. Para o desenvolvimento do meu projeto, vou utilizar a linguagem de desenvolvimento web PHP.

A princípio é uma ferramenta muito simples, mas ele possui uma loja de extensões imensa, e que continua crescendo.

Ou seja, com essa enorme coleção de extensões, podemos adicionar diversas funcionalidades ao VS Code de forma bem simples.

Como citado na referencia [2], o xammp é um pacote de software livre que consiste em um servidor web Apache, um banco de dados MySQL, PHP e Perl. O nome XAMPP é um acrônimo para as iniciais dos componentes que o compõem: X (para qualquer sistema operacional), Apache, MySQL, PHP e Perl. Ele foi desenvolvido para ser uma solução fácil de instalar e configurar, permitindo que aos utilizadores que tenham um ambiente de desenvolvimento web completo em seu computador local.

o repositório GitHub: [3] O GitHub é uma plataforma de desenvolvimento colaborativo que aloja projetos na nuvem utilizando o sistema de controle de versões chamado Git. GitHub foi onde eu guardei o meu projeto em segurança

o Git é um sistema de controle de versões para guardar as versões do projeto e quando ligado a rede enviar para a cloud.

# Linguagens de programação

No desenvolvimento do meu projeto utilizei várias linguagens de programação e scri+ting, nomeadamente:

Scri+ting:

* HTML
* CSS
* PHP
* Javascript

O Html, também conhecido por ser uma linguagem de programação mais básico na construção de Web.

O Css é igualmente útil para a estética do trabalho aplicando um estilo ao conteúdo do html .Pode ser implementado um template ,podemos deixar o front-end mais percetível para ter um aspeto mais versátil para o utilizador.

Já o javascript como descrito na referencia [4] é uma linguagem de programação que permite a você criar conteúdo que se atualiza dinamicamente, controlar multimídias, imagens animadas, e tudo o mais que há de interessante. Ok, não tudo, mas é maravilhoso o que você pode efetuar com algumas linhas de código JavaScript.

Por fim a utilização do PHP, permitiu me contruir a minha framework de raiz , onde pude estabelecer relações entre as várias partes do projeto, sendo nas views, como o controlador e no modelo. Resumidamente, foi a linguagem principal que utilizei para estabelecer a comunicação necessária entre todas as parte «s do meu projeto.

# Desenvolvimento do projeto

Inicialmente, comecei por instalar todas as ferramentas necessárias identificadas anteriormente.

Criei uma conta no pessoal no GitHub para puder alojar todo o meu projeto de uma maneira segura e puder sempre verificar todos os passos que ia dando sempre que adicionava alguma atualização ao meu projeto.

Após o repositório publico criado, que pode ser consultado em [5], comecei por puxar a versão ‘main’, para o meu computador local e dar inico ao desenvolvimento do projeto.

Toda a informação, ou documentação que ia gerando, foi sempre sido atualizada através do sistema disponibilizado de ‘commits’, que permitia assim ir documentando todo o meu projeto de uma forma automática.

Ao iniciar o desenvolvimento, criei um novo branch(ramo) do meu projeto no repositório, porque as boas praticas indicam que nunca se deve trabalhar no repositório ‘main’, pois este pode ser considerado o repositório de produção.

Assim sendo, inicie um novo branch, com um nome de ‘develop’ e comecei por criar alguns ficheiro e pastas, que passarei de momento a explicar.

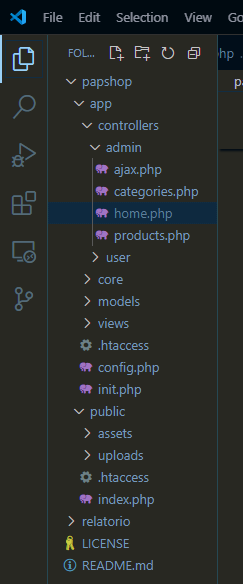


Imagem1 – Tree do projeto em desenvolvimento

A pasta com o nome “PAP” dentro dessa pasta criei o meu projeto com o nome “papshop” dentro do projeto criei duas pastas “app” (backend) e “public”(frontend).

Dentro da pasta publica criei mais dois arquivos “índex.php”,”.htaccess”.

No índex.php” é onde se inicia a sessão em nosso navegador e incluímos na pasta “app” para instanciar um objeto da nova função APP() que vai arrancardo a nossa framework.

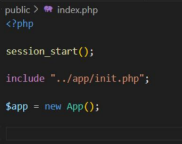


Imagem 1

No arquivo “.htaccess” é onde um conjunto de regras do manipulador direciona todos os URLs

solicitações que não correspondem a um arquivo ou diretório existente ao arquivo index.php, permitindo que o script PHP processe a URL e decida como lidar com a solicitação.

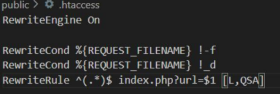


Imagem 2 – ficheiro .htaccess

Como podemos observar na tree do projeto, dentro da pasta “APP” criei 2 arquivos “init.php”,”.htaccess” e quatro pastas “controllers”,”core”,”models”,”views”.

Dentro do arquivo “init.php” é onde identifico todos os caminhos para as principais pastas do meu projeto.

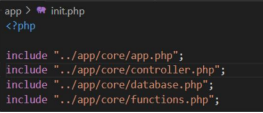


Imagem ? – init.php (app)

No arquivo “.htaccess” a opção Options -Indexes estiver presente, o servidor web não mostrará uma lista de arquivos quando alguém aceder a um diretório no navegador e não houver nenhum arquivo de índice presente. Esta configuração é uma medida de segurança comum para impedir que visitantes vejam a estrutura de diretórios do site e tenham acesso aos arquivos que não deveriam ser públicos.

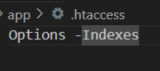


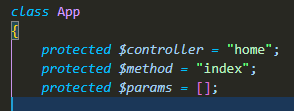
Imagem ? – .htaccess (app)

Com o propósito de iniciar os testes de conectividade e funcionalidade na minha framework, comecei por criar dentro da pasta core um ficheiro app.php. Este ficheiro irá fazer toda a manipulação de conteúdos que é escrita na url do nosso browser, como já identificado anteriormente, na explicação do MVC(Model, View, Controller)

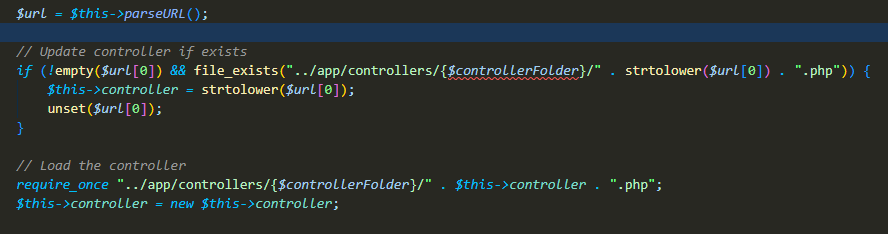
Assim sendo, com o propósito de facilitar a explicação irei proceder a uma breve explicação por partes.

Como já havia identificado anteriormente, a instanciação da classe APP é onde todo o projeto arranca.

Comecei por definir 3 variáveis, que irão ajudara manipular a url.



No código seguinte é onde irá ser feita a leitura do controlador, ou seja, inicialmente o utilizador irá carregar em um href no html e ele irá verificar se o controlador, para onde o href esta apontado existe e irá colocar em letras pequenas e agarrar em um array a primeira posição do índex da variável $url.



Com o objetivo de testar a primeira parte do código gerado, criei um controlador como nome ”Home” para conseguir fazer debug do que estava a fazer no meu projeto. Este controlador, irá servir para no futuro carregar a minha view do índex.php.

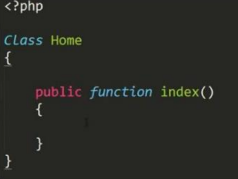
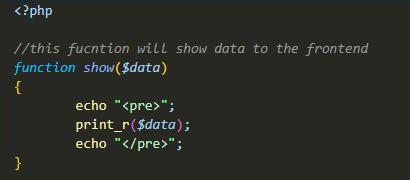


Imagem ? – Controolador da home

Para facilitar o debug, criei uma função show() dentro da pasta functions do meu projeto, que irá servir para todas as funções globais da framework.



Imagem? – função para fazer debug da aplicação no browser

O próximo passo, foi abrir o xammp e iniciar o servidor apache para testar o desenvolvido ate ao momento.

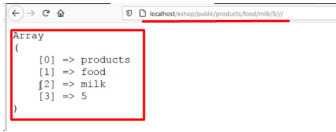
Ao inserir alguns dados na url, este foi o resultado obtido:  
  


Imagem ? – debug do que foi escrito na url

Assim, como eu tinha identificado, anteriormente, neste momento os dados da url estão a ser colocados dentro de um array e são separados em cada índex, cada vez que é identificado que existe um barra (/) ele automaticamente faz o split e insere os dados em cada posição.

O próximo passo é apanhar o método. Para isso adicionei ao código o seguinte excerto:

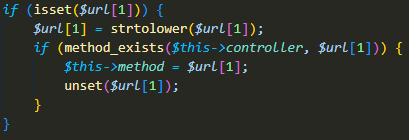


Image ? – parâmetros da url

Neste excerto de código é onde verifico se o método existe e, se existir, ler o método, caso não exista, enviará o utilizador para uma página de erro. Para isso, é criado um sistema que verifica se a URL contém um segundo segmento, que geralmente é usado para especificar um método dentro do controlador. Se o método existir no controlador, ele será definido como o método atual e o segmento correspondente da URL será removido. Isso permite chamar métodos específicos dentro de controladores com base na URL

Com todo o sistema testado e a funcionar, foi hora de introduzir as views para testar se tudo estava conectado.

Para isso dentro da pasta views, criei um ficheiro índex.php, e inseri algum texto aleatório para teste.

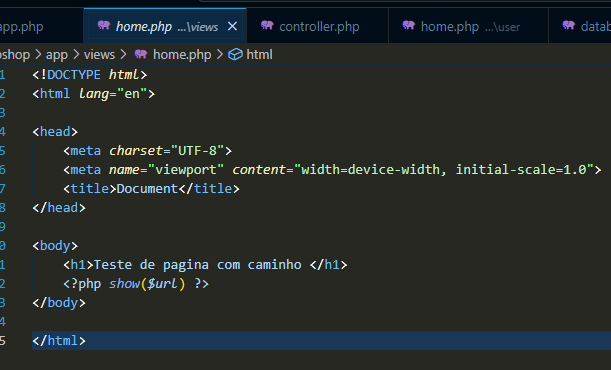


Imagem ? -Exemplo de view

Então, seguindo o paradigma de construção, todos os controladores devem se iniciar com o método índex, é é neste método onde vai ser carregada o bossa view.

Para isso, foi criado uma função publica, que irá tratar desse processo:

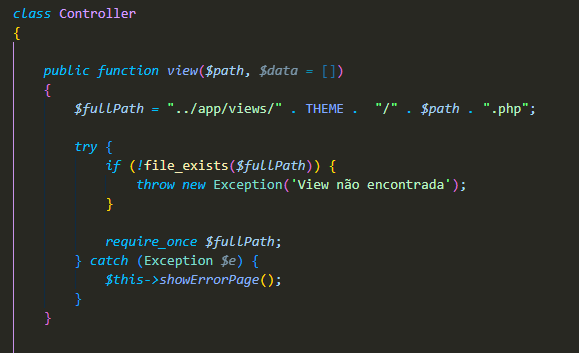


Imagem ? – Função de controlo de carregamento das views

Esta função esta contruída para carregar as views do projeto. Caso a view não seja encontrada, irá disparar um erro, que vai chamar a pagina 404.php.

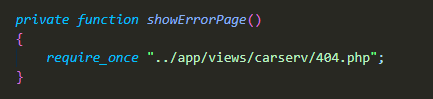


Imagem ? – Função de erro de pagina

De seguida vamos testar a função view e carregar o nosso índex.php:

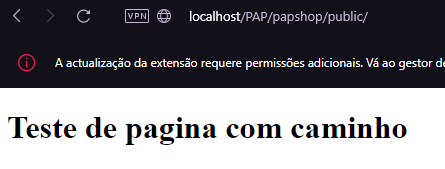


Imagem ? – pagima teste

Com todo o sistema de carregamento de dados testado e funcional, e como boa pratica e seguindo o paradigma, vsto que o projeto consite em um webstore, necessito de adicionar ao meu projeto uma área de backend onde somente o utilizador autorizado está deverá aceder para para fazer a manipulação do site, como por exemplo a introdução na secção de vendas dos produtos e categorias.

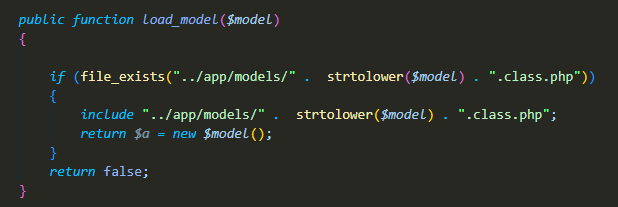
Assim tive de manipular a app, de forma a dividir o conceito para, que um dia quem viesse a agarrar neste porjeto, tenha facilicade em perceber o que esta construído e como esta contruído o mais rápido possível.

Então, o código alterado no app.php foi o seguinte:

O passo seguinte foi encontrar um template para utilizar no projeto. Escolhi um projeto que utiliza a tecnologia BOOTSTRAP para facilitar a implementação.

Para evitar cópias não autorizadas, optei por fazer um pequeno investimento e adquirir um template no site ???????

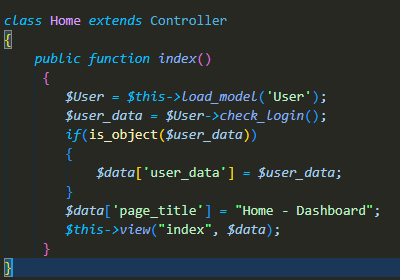
Dentro do arquivo “controller.php” inseri também a seguinte função para manipular as ligações com os modelos.



Este código recebe o caminho da view verifica a existência e carrega a view que é exigida para o utilizador ver.

em seguida no arquivo “home.php” localizado na pasta “user” fiz uma ligação OPP para extender o controlador inicial para este controlador,porque a função de índice precisa acessar a função de visualização do controlador para poder exibir as visualizações para o utilizador.

Criei um arquivo dentro da pasta “views” que dei o nome de “índex.php” esse arquivo é o que carrega a pagina principal da aplicação

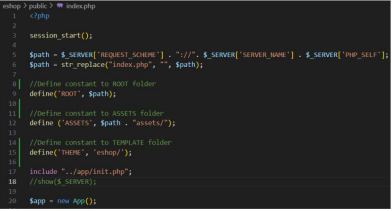
Em seguida inseri o resto do código no arquivo “home.php” :

Este código é o que controla a view da principal da aplicação para que mostre na aplicação.

Um pasta dentro das views “\_includes” dentro dessa pasta contem três arquivos “user\_footer.php”,”user\_header.php”,”user\_menu.php”,fiz isto para organizar e otimizar melhor a minha aplicação que neste caso estes arquivos servem para se alterar alguma coisa dentro deles em toda a aplicação sera alterada ao mesmo tempo sem ter que ir a cada view separadamente fazer as alterações.



A ligação é feita por essa linha de código mas é diferente em baixo e em cima em baixo é alterado de “user\_menu” para “user\_footer” assim ligação é feita com o devido arquivo.

Dentro do arquivo “índex.php” localizado na pasta “public” inseri o seguinte codigo

Este código constrói o caminho base da aplicação, define constantes para os diretórios ,e cria uma instancia da classe ‘app’ para iniciar a aplicação.

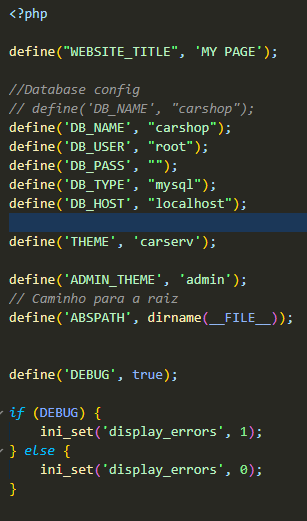
Para fazer as imagens aparecerem na aplicação substitui os caminhos defenindo uma rota direta igual ao seguinte código:



Assim fica mais fácil para lidar com a localização das imagens.

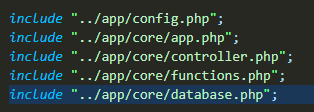
Criei dois arquivos na pasta controllers “register.php”,”login.php”

Criei um arquivo na pasta “app” com o nome “config.php” e inseri o seguinte código:

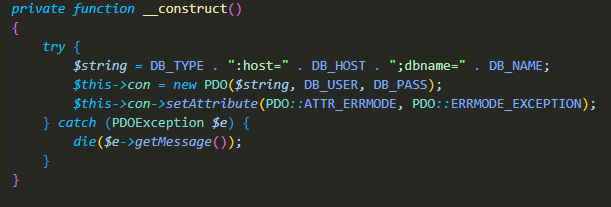


Este código faz a identificação e a conexão com a base de dados.

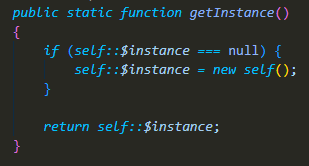
No arquivo “init.php” fiz a referencia de todos os arquivos que fazem a aplicação funcionar e o código que inseri é este:



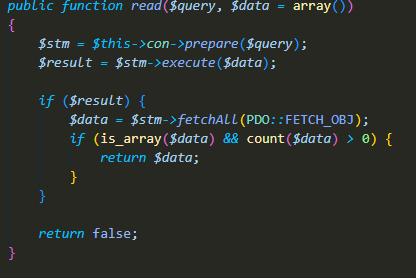
Criei a minha base de dados “carshop” para a aplicação para poder guardar os dados dos utilizadores e dos produtos que serão adicionados mais tarde.

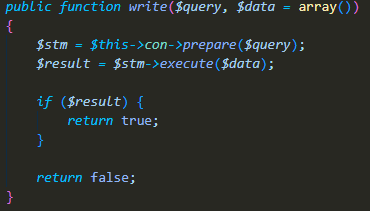
No arquivo “database.php” localizado na pasta “core” adicionei o seguinte código para fazer a ligação via PDO: 

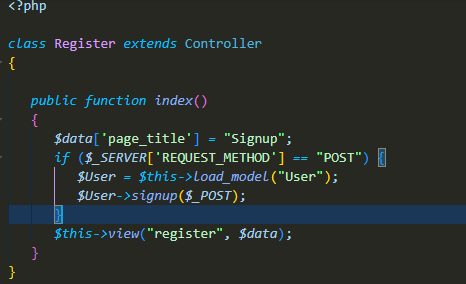
E para garantir apenas exista uma instancia do objeto de conexão exista durante todo o tempo de utilização da aplicção implementei este codigo logo a baixo da ligação PDO:



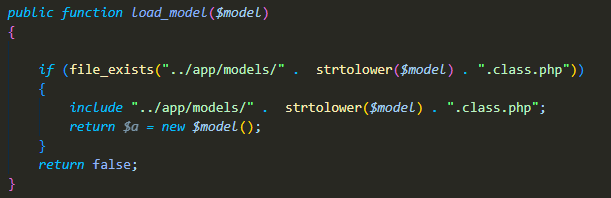
Restrigindo a criação de novas instancias e permitindo apenas uma evita a subrecarga de multiplas conexões na base de dados assim permitindo um bom desempenho da base de dados e do servidor da aplicação.

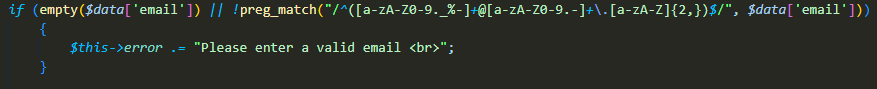
Tambem criei outra função para ler os dados da base de dados fazendo as consultas pela base de dados e recuperar dados o codigo que utilizei é foi o que demosntra na imagem: 

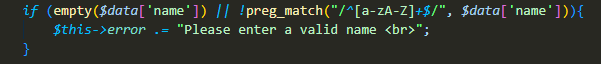
e por fim no arquivo “database.php” inseri o codigo para escrever na base de dados para que possa guardar os dados que eu desejar ou por exemplo quando um utilizador novo criar uma conta os dados serem guarados na base de dados eo codigo que utilizei foi o seguinte: 

dois arquivos na pasta controllers uma para o login eoutra para o register,dentro do “register.php” inseri o seguinte codigo que recebe um “$POST” quando o utilizador clica em registrar e cria um documento json e manda para a base de dados: 

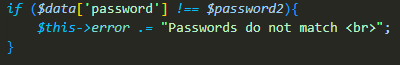
inseri tambem mais uma função no arquivo “controller.php” localizado no “core” que recupera o arquivo de “user.class.php” da pasta models para enviar os dados para a base de dados:

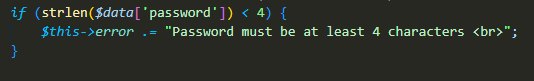


O arquivo “user.class.php” como mencionei no codigoanterior é usado para atribuir funções exemplo :validar um email como vou mostrar no codigo a função validação de email: 

Tambem a validação do nome: 

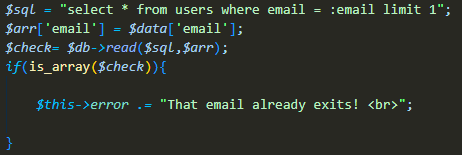
E tambem validação da palavra pass e verificação se a primeira palavrapass é igual ou não a segunda palavra pass:



Verificação se a palavra pass é mais de 4 caracteres: 

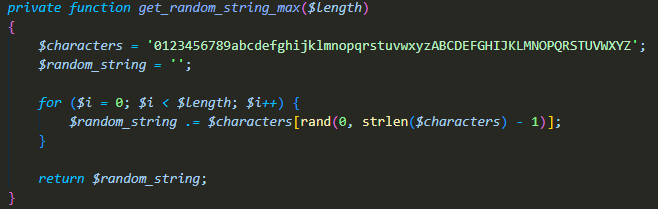
caso não seja maior que 4 caracteres o programa escreve uma mensagem de erro a informar que a palavra pass tem que ser mais de 4 caracteres.

Verifica se o email já esta a ser utilizado :



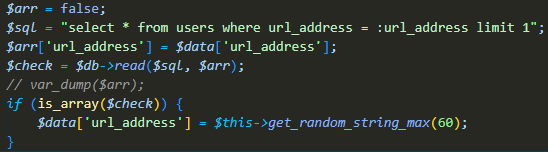
Caso esteja já em uso escreve uma mensagem de erro a informar que o email fornecido já esta em uso.

Cria um numero aleatório para quando um utilizador fazer um novo registro seja atribuído um numero como se fosse um identificação dentro da base de dados

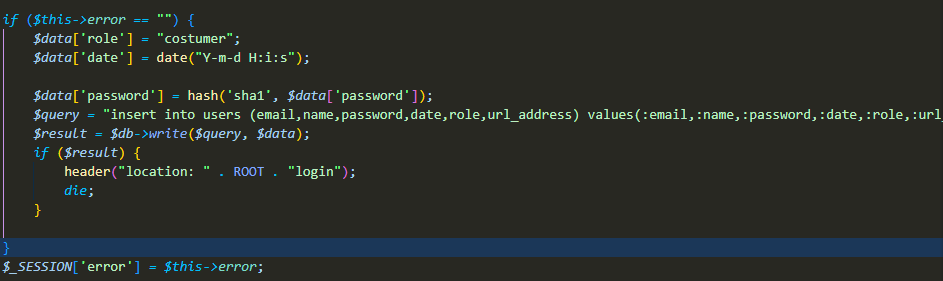


Esta linha de código atribui uma string gerada aleatoriamente à variável $data['url\_address'].



Faz a verificação da existência da URL do utilizador para não acontecer o caso de associar dois números iguais: 

O resto do codigo identifica os campos para serem preenchidos para a base de dados :

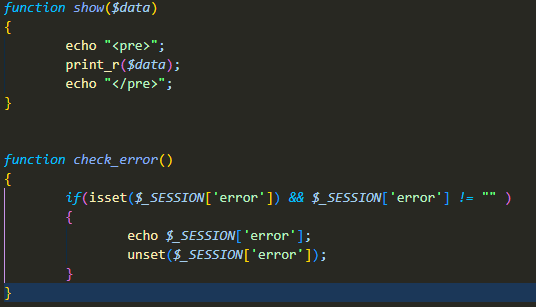


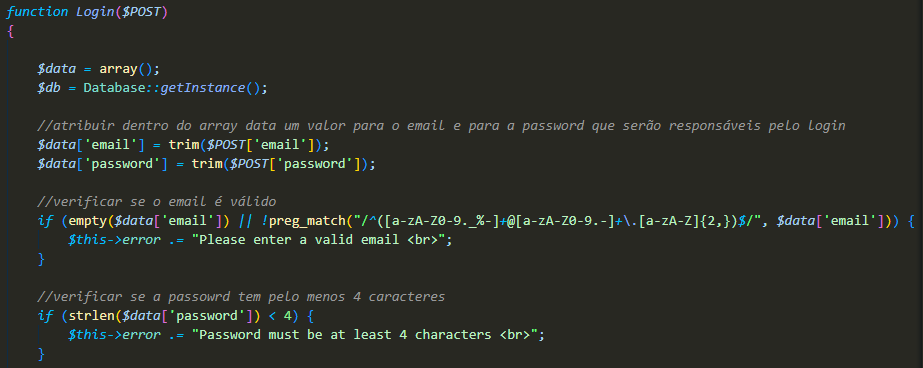
No geral, este segmento de código completa o processo de registo do utilizador, inserindo dados validados do utilizador na base de dados e tratando o sucesso ou a falha conforme necessário. Além disso, gere mensagens de erro para fornecer feedback ao utilizador, se necessário.

Caso o utilizador escreva algo errado no registro este código envia uma mensagem de erro a informar o erro para o utilizador saber qual é a falha:

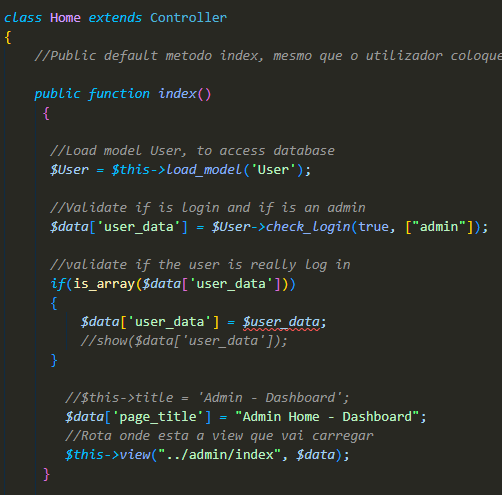


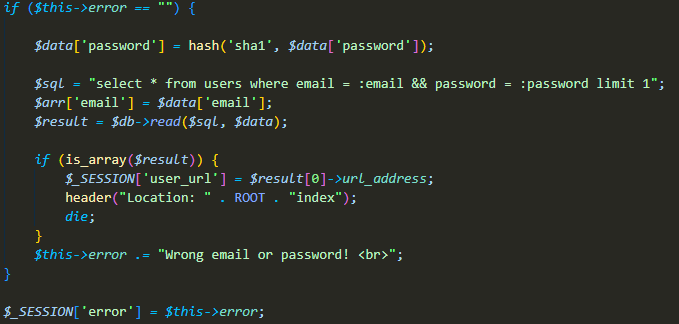
Contrui uma função erro para mostrar erro:



Validação do login se exite o perfil na base de dados se o email coincide se a palavra pass esta correta ou não: 

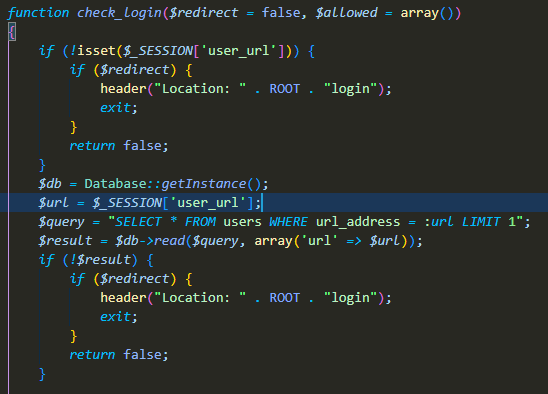
criei o um arquivo tipo controlador “home.php” localizado na pasta “admin” para mostrar a Dashboard do admin onde é a “secção admin” e inseri este:



faz a conexão a base de dados e verifica se esta tudo correto caso não ter nenhum erro o login prossegue normalmente mas caso ter algum erro é mostrado uma mensagem de erro a informar o erro no login: 

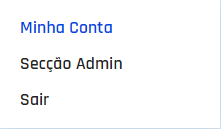
A palavra pass do utilizador é criptografada com segurança usando a criptografia do sistema.

Faz a verificação se o utilizador esta logado e caso não esteja encaminha-lo para a pagina de login:

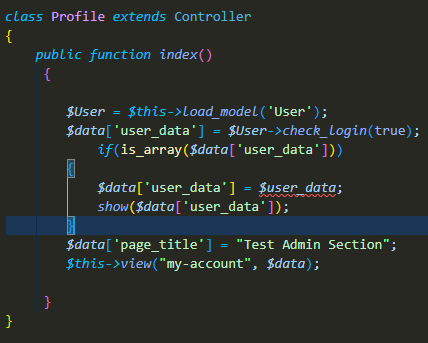


Adicionei uma pasta “admin” dentro da pasta “views” e dentro da pasta “assets” também adicionei outra pasta “admin” para gerenciar a zona admin da aplicação.

Este código caso o utilizador seja admin ira aparecer um botão na secção da conta mas se ao for admin e for um utilizador normal não ira aparecer como esta na imagem:



Para isso utilizei este código que verifica se o utilizador é o não admin:



Se houver, exibe a mensagem e remove-a da sessão.

Para facilitar a implementação da aplicação recorri alguma documentação online, visto que, este tipo de matérias não são lecionadas durante o nosso curso.

Optei por escolher este método de desenvolvimento pois assim consigo ter uma precessão melhor e estruturado do meu código, atribuindo responsabilidades a cada classe e a cada método.

A Aplicação é dividida em duas partes “app”,”public”, cada parte tem a sua função ,a zona “public”

É dividido e quatro partes começando a inicialização do programa com o ficheiro “index.php” que com o comando “session\_start();”



Inicia uma sessão PHP ou retoma a sessão atual baseada em um identificador de sessão passado via um pedido GET, POST ou um cookie. As sessões permitem armazenar informações entre diferentes páginas e pedidos.

**Define a constante ‘ROOT’**

o código constrói o URL base do projeto e define a constante ‘ROOT’ com ele.

* ‘$\_SERVER['REQUEST\_SCHEME']’: Protocolo utilizado (‘http’ ou ‘https’).
* ‘$\_SERVER['SERVER\_NAME']’: Nome do servidor (domínio).
* ‘$\_SERVER['PHP\_SELF']’: Caminho do script atual.

Depois, o código remove "index.php" do caminho para obter o diretório base e define a constante ROOT com este valor.

**Define a constante ‘ASSETS’**

constante ‘ASSETS’ como a URL do diretório de ativos do projeto, concatenando a URL base (‘$path’) com "assets/".

**o arquivo init.php**

# Análise do Sistema

Análise das especificações do problema; identificação das entidades e das suas relações, dos fluxos de dados, esquema de tabelas da base de dados, dicionário de dados, definição de interfaces e layout, etc. Desenvolvimento do modelo conceptual para resolução do problema.

# Desenvolvimento

Este capítulo descreve a implementação da solução proposta no capítulo anterior.

Neste capítulo são também descritas as especificidades de implementação de acordo com o ambiente de desenvolvimento, plataforma e linguagem escolhida para o desenvolvimento. Devem também ser reportados os problemas encontrados e a solução escolhida para os resolver.

Descrição concisa do hardware e do software utilizado.

# Avaliação

Breve descrição de como instalar, usar ou aceder ao sistema desenvolvido. Se a descrição for extensa, considerar a sua localização num anexo ou mesmo num documento à parte. Descrição dos testes realizados e os resultados experimentais obtidos; análise crítica dos resultados.

Este capítulo pode não existir em todos os projetos, dependendo da sua natureza. Em alguns casos é também aceitável que se transforme numa secção do capítulo anterior.

# Conclusão

Ao realizar este projeto pude ampliar mais os meus conhecimentos sobre programação de sites, aprendi mais sobre a segurança a ter ao construir um site online para impedir vazamento de dados, organização a ter no programa facilitando a localização de uma linha de código, tive dificuldades de inicio ao implementar algo que não foi ensinado no curso então tive dificuldade a aprender e a aplicar o conhecimento no código e realizando a construção do projeto

As conclusões finais devem focar o sucesso/insucesso do trabalho, revendo as dificuldades encontradas. Devem resumir, de alguma forma, as vantagens do produto desenvolvido. Podem também referir as facilidades e as dificuldades sentidas ao longo do projeto.

Deve também apresentar sugestões para a continuação do trabalho.

A conclusão deve dar resposta as seguintes questões:

Em que medida os objectivos foram alcançados?

Quais foram as lições aprendidas?

Quais são as ideias para trabalhos futuros?

Quais são as vantagens de desvantagens da solução apresentada, face a outras já existentes?

# Bibliografia

Esta secção lista as referências bibliográficas online citadas no texto.

**[1]VS Code - O que é e por que você deve usar?** Disponível em: <https://www.treinaweb.com.br/blog/vs-code-o-que-e-e-por-que-voce-deve-usar>.

**[2] O que é: XAMPP - Blog Escola Ninja WP**. Disponível em: <https://blog.escolaninjawp.com.br/glossario/o-que-e-xampp/>.

**[3]**LENCINA, W. **O que é GitHub: para que serve, como funciona e como utilizar**. Disponível em: <https://ebaconline.com.br/blog/o-que-e-github>.

**[4]O que é JavaScript? - Aprendendo desenvolvimento web | MDN**. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript>.

‌

# Anexo A - Listagem do material utilizado

Para a construção do meu projeto de aptidão profissional utilizei:

* repositório no GitHub;
* kit digital escolar;
* computador pessoal.