Tutorial para configurar o repositório git no WampServer

Nota1: Este tutorial destina-se apenas à Unidade Curricular de Programação para a Web – Servidor (PWS).

Nota2: O nome que devem dar à pasta APP que consta na estrutura do repositório como explicado no enunciado de MDS deve ser, no caso de PWS, o nome da *framework*, isto é, weblogicmvc. **Mudar o nome da pasta APP para weblogicmvc.**

A *framework* WebLogicMVC requer obrigatoriamente que a pasta que contém a mesma seja a pasta raiz logo a seguir à pasta www. Deste modo, é necessário proceder a algumas alterações no que diz respeito ao mecanismo/funcionamento normal do git.

Problema:

Dado que o repositório contém 2 pastas, tal como é definido no enunciado de Metodologias de Desenvolvimento de Software e é necessário proceder à criação e posterior clone do repositório, torna-se no caso de PWS obrigatório que esse processo seja executado na pasta www para que o projeto possa ser executado/testado.

Portanto, ao fazerem o comando git clone na pasta www, o que acontece é que dentro da pasta www irão ter uma pasta com o nome do vosso repositório e que por sua vez apenas lá dentro é que terão outra pasta com o conteúdo do projeto/*framework* e outra pasta (DOC) com a documentação associada.

Exemplo: Figura 1

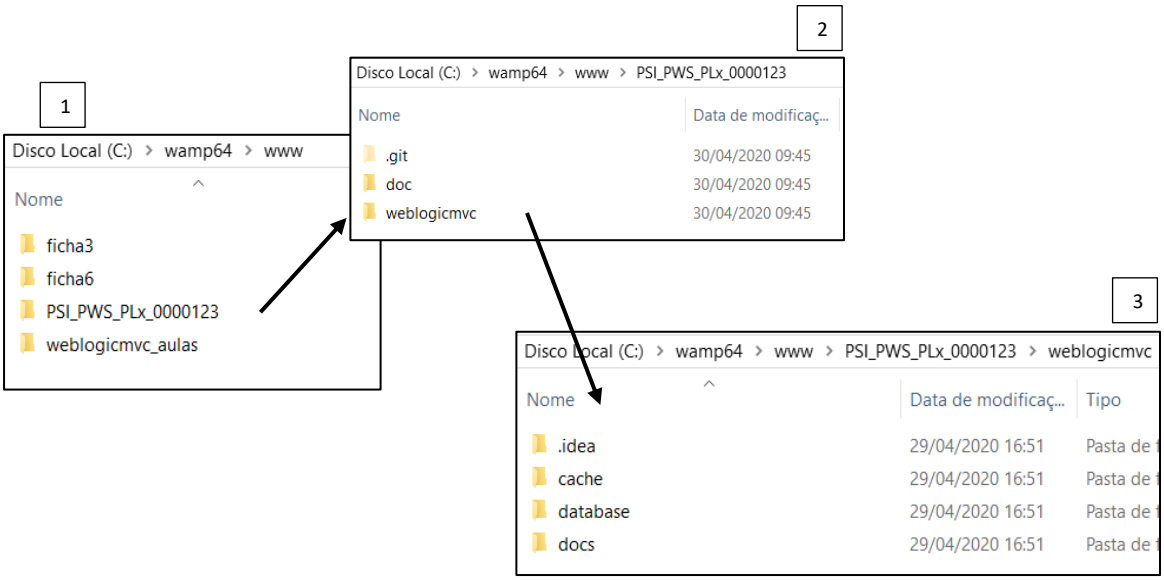

Figura 1 - Estrutura de pastas com o problema inicial

Assim, tem-se uma pasta antes da pasta que contém a *framework*, o que fará com que a mesma ao ser executada apresente um erro, normalmente cor-de-laranja, apontando para um ficheiro *frontcontroller*.

Atenção: Não devem mexer nos ficheiros da *framework* mas sim seguir os passos abaixo.

Solução:

Para solucionar o problema identificado acima devem seguir os passos abaixo, quer para a criação do repositório, quer para o clone do mesmo.

Dado que a pasta www será a pasta git que irá ter todo o conteúdo do nosso repositório, é importante que o nosso ficheiro .gitignore seja bem definido para que ao fazer *add*, *commit* e *push* não se envie nada que esteja na pasta www que não pertença ao projeto, tais como, resoluções de fichas anteriores, etc…

- **<u>Para quem cria o repositório:</u>**

1. `git init` na pasta www
2. `git remote add origin <repo_url>`
    - exemplo:

   git remote add origin https://github.com/username/PSI_PWS_PLx_0000123.git
3. Criar o ficheiro .gitignore na pasta www e colocar as seguintes linhas:

```
#para o git ignorar tudo
/*


#menos as seguintes pastas e ficheiros
!/weblogicmvc
!/.gitignore
!/DOC


#no entanto, queremos que ele ignore também a
#pasta vendor da weblogicmvc, pois não nos
#interessa que vá para o git, nem a pasta que
#contém as configurações do IDE
/weblogicmvc/vendor
/weblogicmvc/.idea
```

Quadro 1 - Ficheiro .gitignore

4. Colocar a framework na pasta www, bem como, criar a pasta doc também na pasta www.
5. Verificar o que está *untracked/unstaged* e verificar se coincide com os ficheiros que queremos enviar e apenas esses;
    - Para quem usa linha de comandos:
        - `git status`
    - Para quem usa um cliente gráfico do git (ex. GitKraken):
        - É apenas verificar se aparece alguma coisa *unstaged*.

    Os ficheiros que deverão aparecer são:

    weblogicmvc, **doc** e **.gitignore**.

6. Adicionar os ficheiros/pastas com o comando `git add .`
7. Realizar o commit inicial:

    `git commit -m "Commit inicial. Inclui configuração do .gitignore e estrutura de pastas."`
8. Efetuar o push para o repositório remoto.

    `git push origin master`

Após todos estes passos deverão ficar com uma estrutura de pastas idêntica à ilustrada pelas figuras abaixo (Figura *2*, Figura 3, Figura *4*).

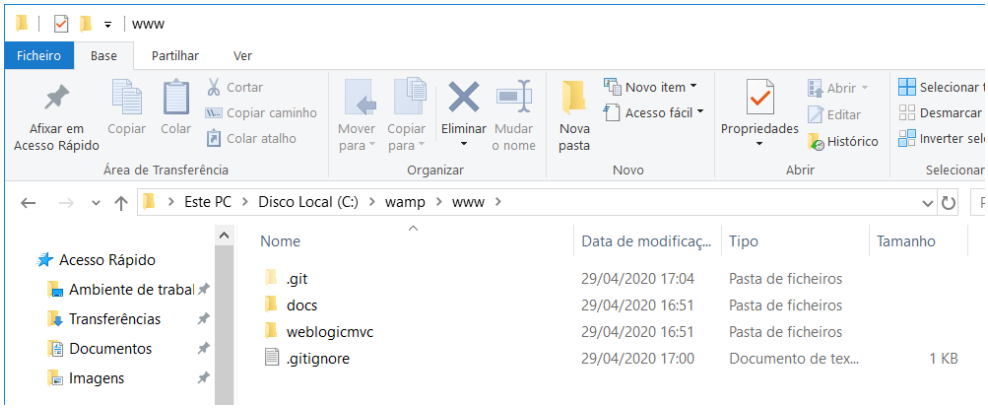

Figura 2 - Pasta www após passar a ser o repositório

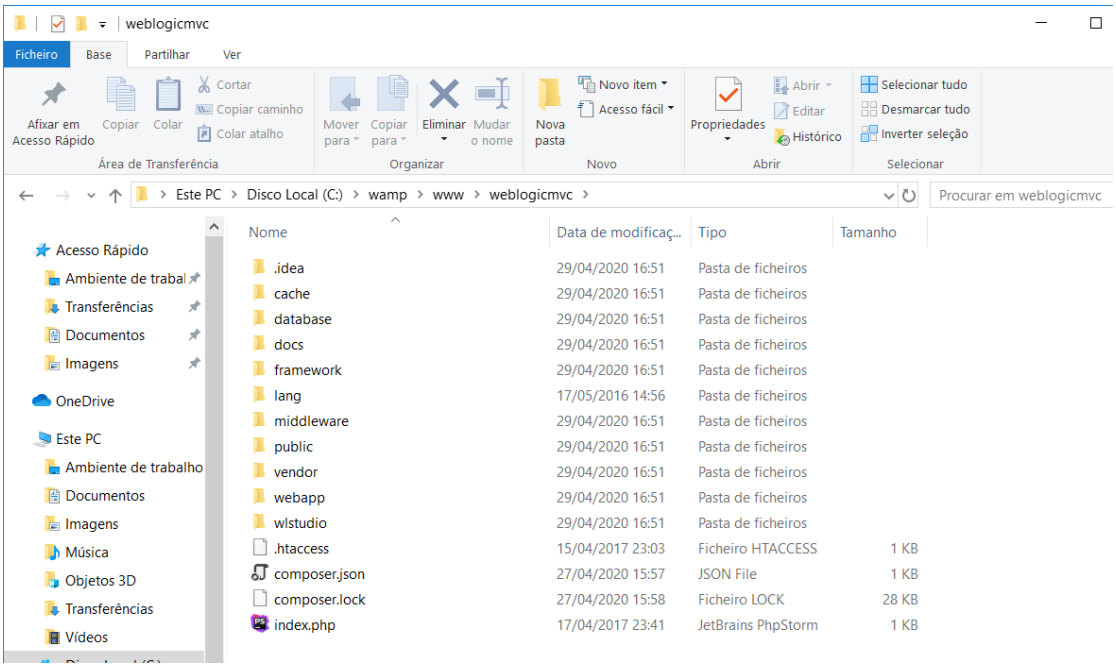

Figura 3 - Pasta weblogicmvc dentro do repositório

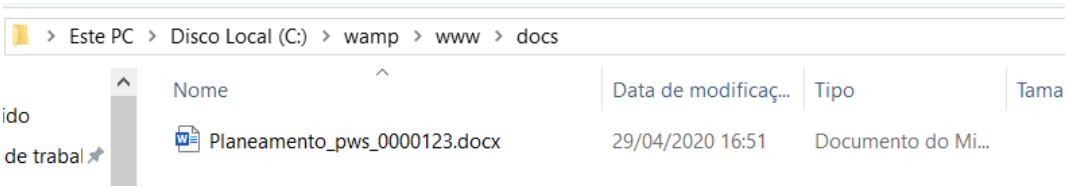

Figura 4 - Pasta docs dentro do repositório (www)

- **<u>Para quem pretende fazer clone do repositório:</u>**

    1. Ir para a pasta www;
    2. `git init`
    3. `git remote add origin <repo_url>`
    4. Realizar o pull;
       `git pull origin master`
    5. Testar/Executar/Correr o projeto no localhost, fazendo como nas aulas de PWS
       `localhost/weblogicmvc`
       e verificar se tudo funciona correctamente;
    6. Agora podem trabalhar normalmente no vosso projeto/código, abrindo o projeto no PHPStorm como tem feito ao longo nas aulas.
    7. Sempre que pretenderem fazer *commit* e *push* de uma alteração, basta:
        - Para quem usa um cliente gráfico do git (ex. GitKraken):
            - Abrir repositório e depois localizam a pasta www;
            - Verificar se o(s) ficheiro(s) alterado(s) está(ão) no *unstaged*;
            - Colocá-lo(s) *staged*;
            - Realizar os *commits* e *push* normalmente.
        - Para quem usa a linha de comandos:
            - Ir para a pasta www do wamp;
            - Realizar o `git status` para verificar qual(is) o(s) ficheiro(s) alterado(s);
            - Fazer `git add <ficheiroalterado>` ou `git add .`
            - Fazer `git commit -m "mensagem"`
            - `git push origin master`

Bom trabalho!