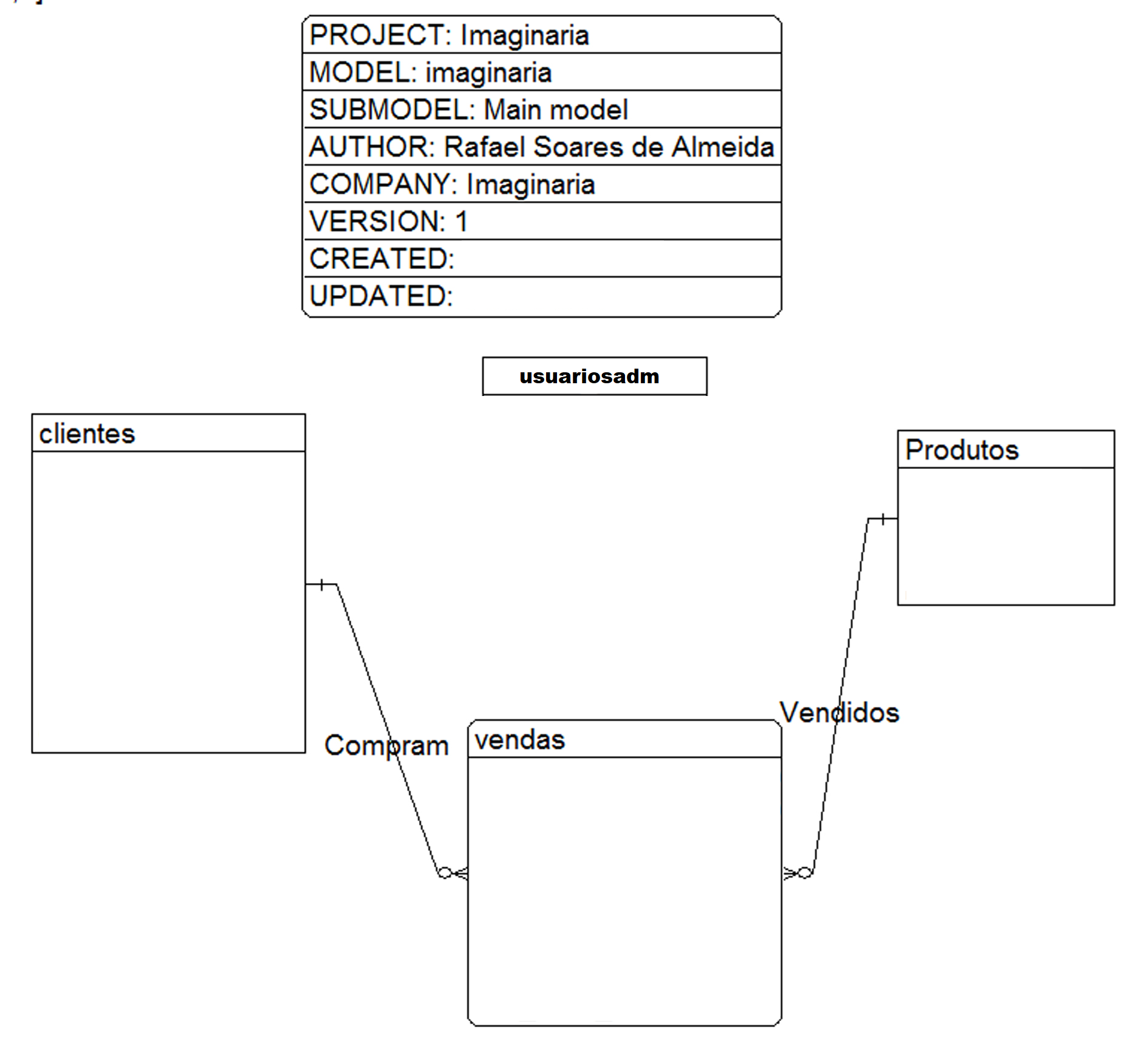
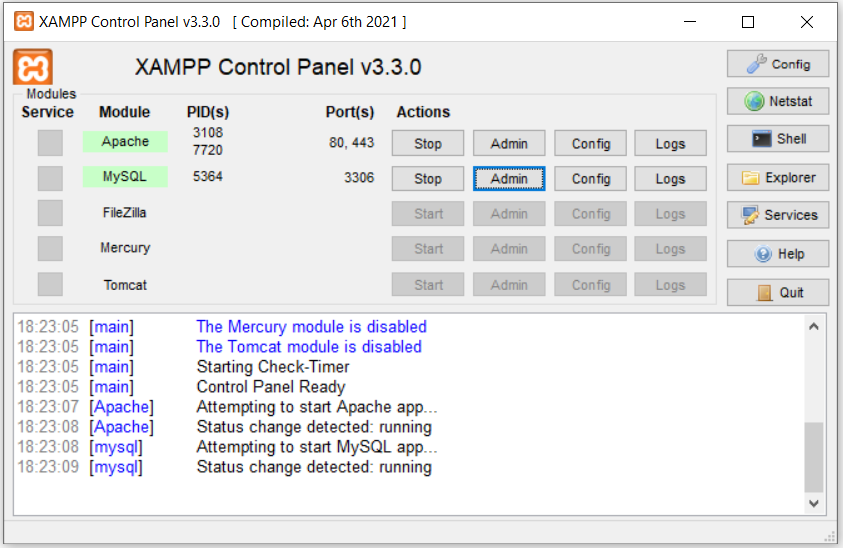
***Professor Msc Rafael Soares***

## Preparando a Estrutura da Base de Dados do MySQL

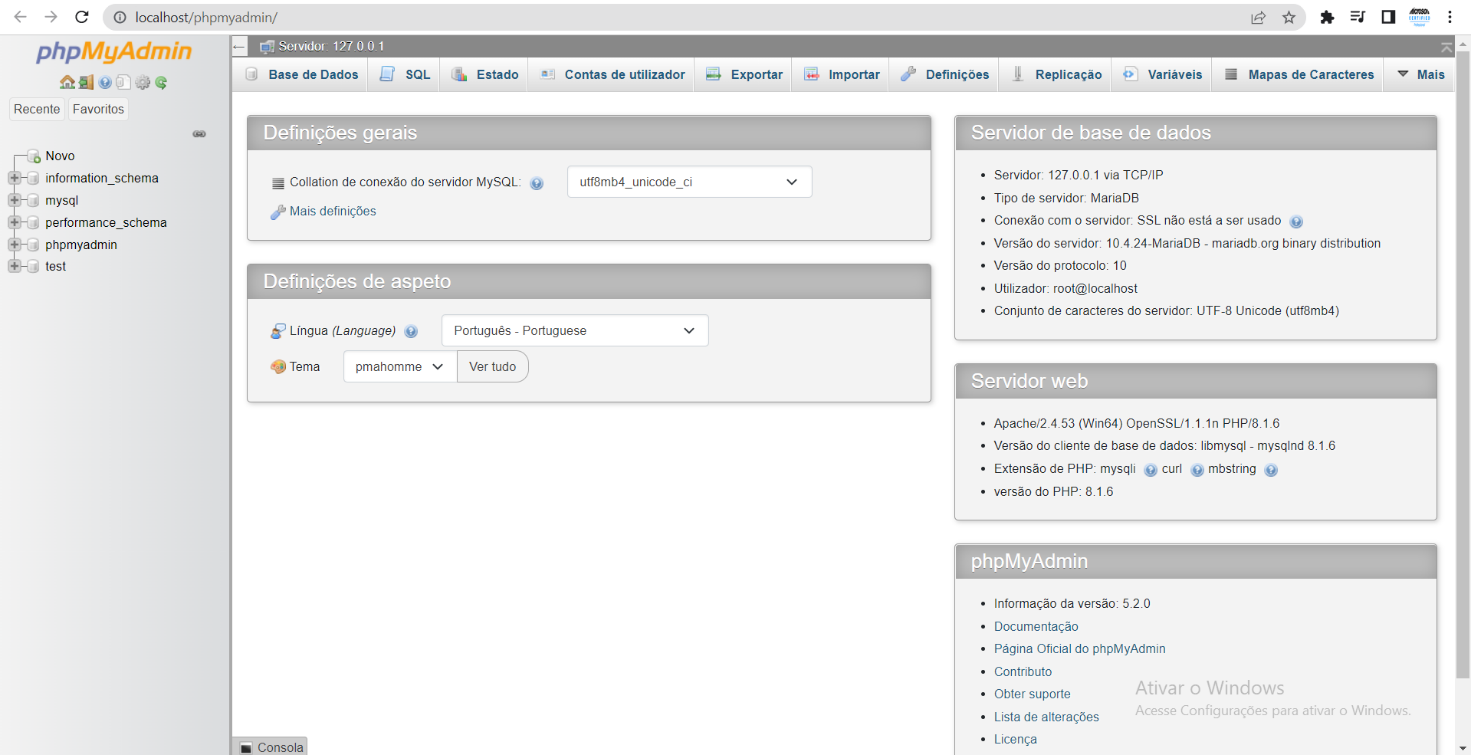
Iremos trabalhar na construção de uma aplicação para a empresa fictícia chamada **Bicicletaria** **Imaginaria Ltda - BIL**. A empresa **BIL**, precisa de uma aplicação para cadastro de clientes e produtos vendidos. O primeiro passo é definir a estrutura da aplicação, ou seja, o **Modelo Entidade e Relacionamento – MER**, das tabelas que vão compor a estrutura do banco de dados. Mais uma vez não vou entrar em detalhes de como construir este modelo. Para nossas tabelas **clientes, produtos e vendas** se relacionarem temos que construir o **MER** que vai ficar conforme figura abaixo:



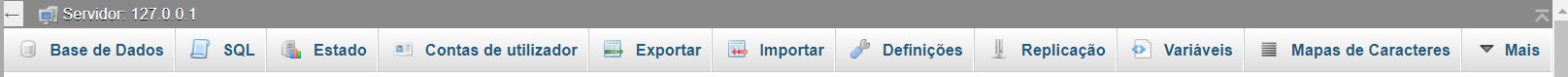
Decifrando o que o desenho representa temos um relacionamento de **Muitos** **para Muitos** ou **N:N**, entre a tabela **clientes** e **produtos**, ou seja, clientes compram um ou mais produtos e produtos são vendidos para um ou mais clientes. A tabela **vendas** é o resultado da junção dos lados **N’s** do relacionamento entre **clientes** e **produtos**, pois os bancos de dados relacionais não implementam relacionamentos **N:N**. A tabela **usuariosadm** pertence ao sistema e não se relaciona com as demais. Após esta breve explicação, estamos aptos a criar nossa base de dados com suas respectivas tabelas. Agora vamos criar uma base de dados no **MySQL** chamada **billtda** em seguida vamos criar as tabelas **clientes, produtos e vendas**. Para isto abra o **XAMPP Control Panel**, conforme figura abaixo:



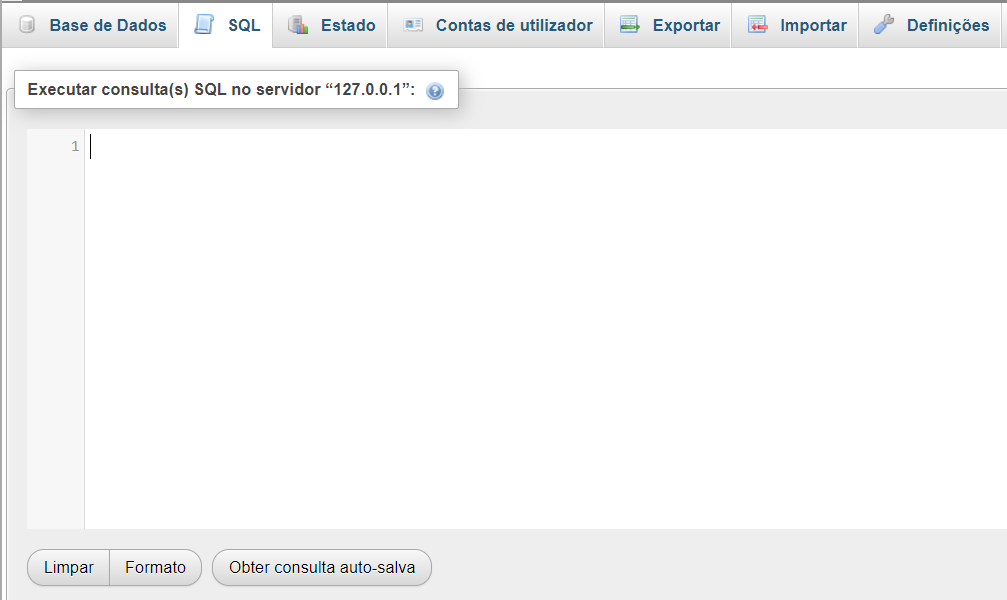
Clique no botão **Start**, para o **Apache** e o **MySQL**, em seguida no **MySQL**, clique no botão **Admin**, e vai aparecer no navegador a página **http://localhost/phpmyadmin/**, conforme figura abaixo:



No menu do lado direito, clique no item **SQL**, conforme figura abaixo:



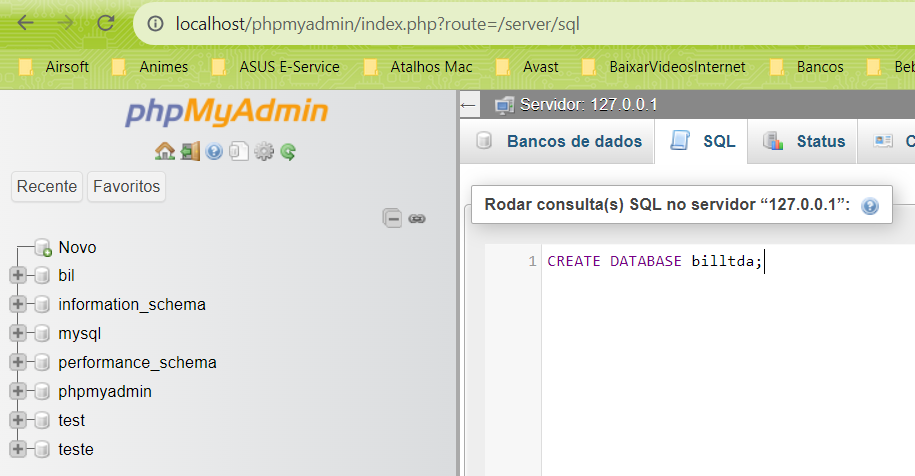
Vai abrir o editor do **SQL**, conforme figura abaixo:



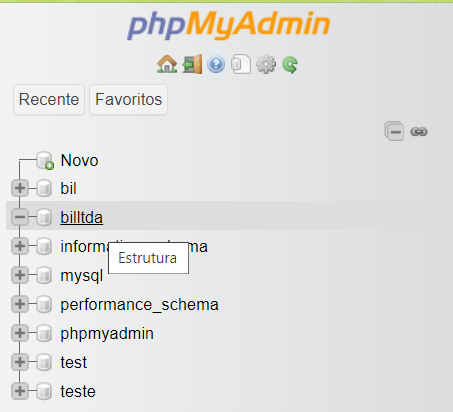
Na **linha 1**, digite o seguinte comando:

**CREATE DATABASE billtda;**

Conforme figura abaixo:



Clique no botão **Continuar(Executar)**, veja que a base de dados foi criada no painel do lado esquerdo, conforme figura abaixo:



Agora vamos criar nossas tabelas, primeiro a tabela **clientes**, para isto, clique no nome da nossa base de dados **bil**, do lado esquerdo do painel, em seguida, retorne ao item **SQL**, no painel do lado direito, e digite o código abaixo:

**Create table clientes (**

**cliID Int NOT NULL AUTO\_INCREMENT,**

**cliDtInclusao Datetime,**

**cliNome Varchar(200),**

**cliEndereco Varchar(200),**

**cliBairro Varchar(80),**

**cliEmail Varchar(200),**

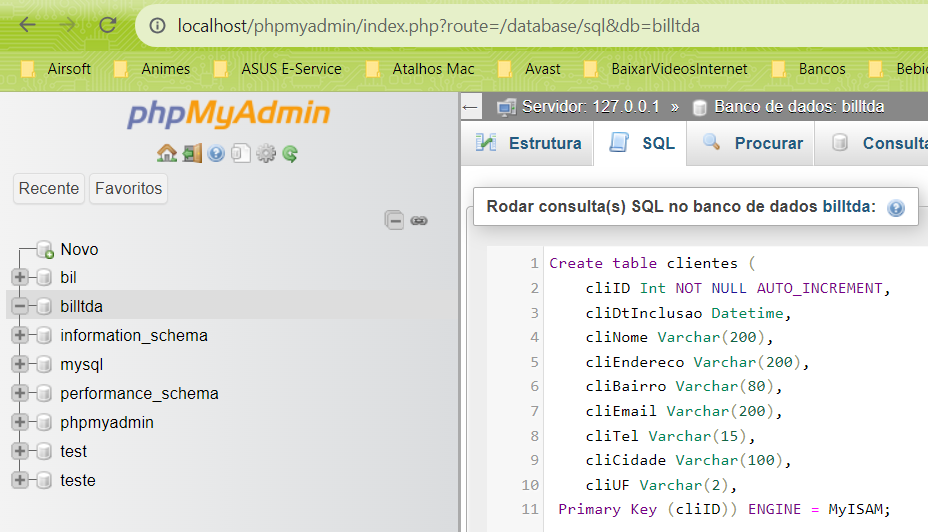
**cliTel Varchar(15),**

**cliCidade Varchar(100),**

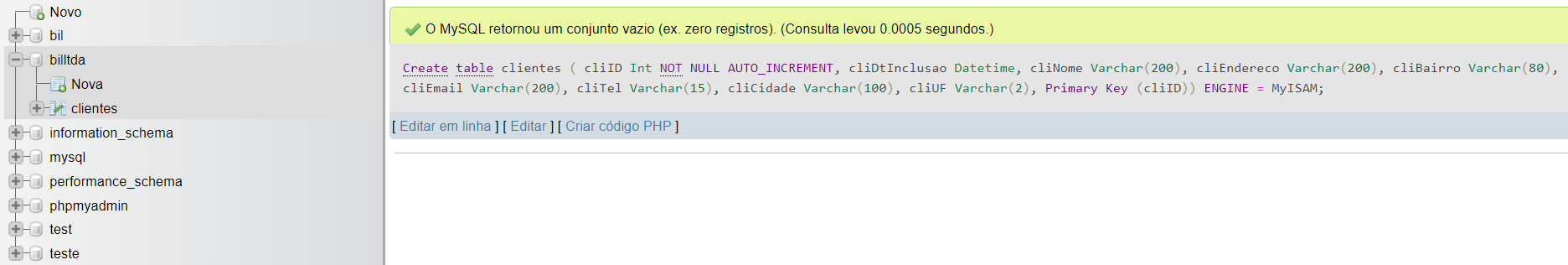
**cliUF Varchar(2),**

**Primary Key (cliID)) ENGINE = MyISAM;**

Veja como ficou, conforme figura abaixo:



Clique no botão **Continuar(Executar)**, e veja que na nossa base de dados **bil**, agora tem uma tabela **clientes**, conforme figura abaixo:



Agora vamos criar nossa tabela **produtos**, para isto, clique no nome da nossa base de dados **billtda**, do lado esquerdo do painel, em seguida, retorne ao item **SQL**, no painel do lado direito, e digite o código abaixo:

**Create table produtos (**

**proID Int NOT NULL AUTO\_INCREMENT,**

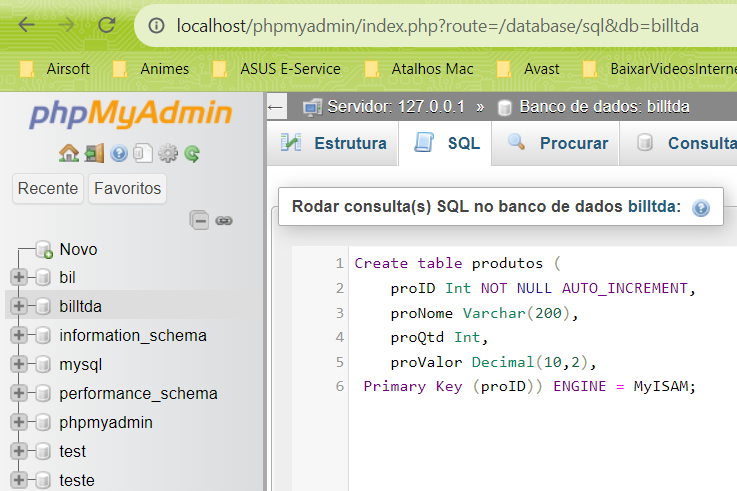
**proNome Varchar(200),**

**proQtd Int,**

**proValor Decimal(10,2),**

**Primary Key (proID)) ENGINE = MyISAM;**

Veja como ficou, conforme figura abaixo:



Clique no botão **Continuar(Executar)**. Agora vamos criar nossa tabela **vendas**, para isto, clique no nome da nossa base de dados **billtda**, do lado esquerdo do painel, em seguida, retorne ao item **SQL**, no painel do lado direito, e digite o código abaixo:

**Create table vendas (**

**cliID Int NOT NULL,**

**proID Int NOT NULL,**

**venID Int NOT NULL AUTO\_INCREMENT,**

**venData Date,**

**venQtdUnitario Int,**

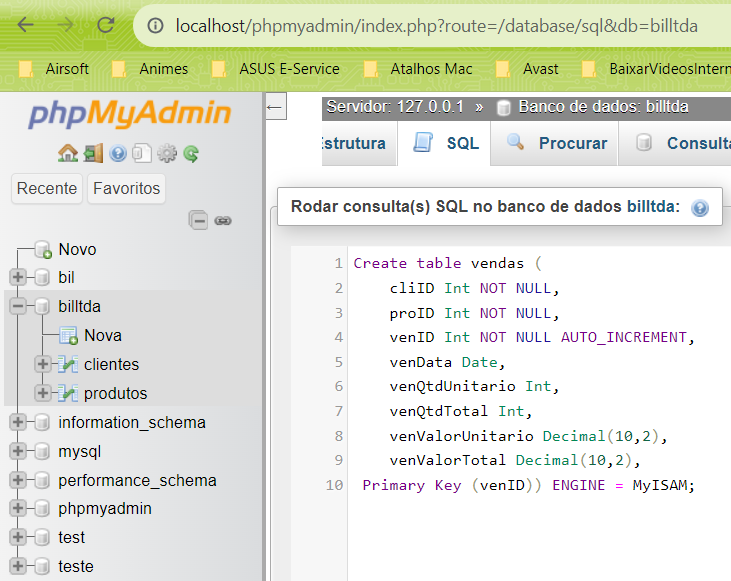
**venQtdTotal Int,**

**venValorUnitario Decimal(10,2),**

**venValorTotal Decimal(10,2),**

**Primary Key (venID)) ENGINE = MyISAM;**

Veja como ficou, conforme figura abaixo:



Clique no botão **Continuar(Executar)**. Agora vamos criar nossa tabela **usuariosadm**, para isto, clique no nome da nossa base de dados **billtda**, do lado esquerdo do painel, em seguida, retorne ao item **SQL**, no painel do lado direito, e digite o código abaixo:

**Create table usuariosadm (**

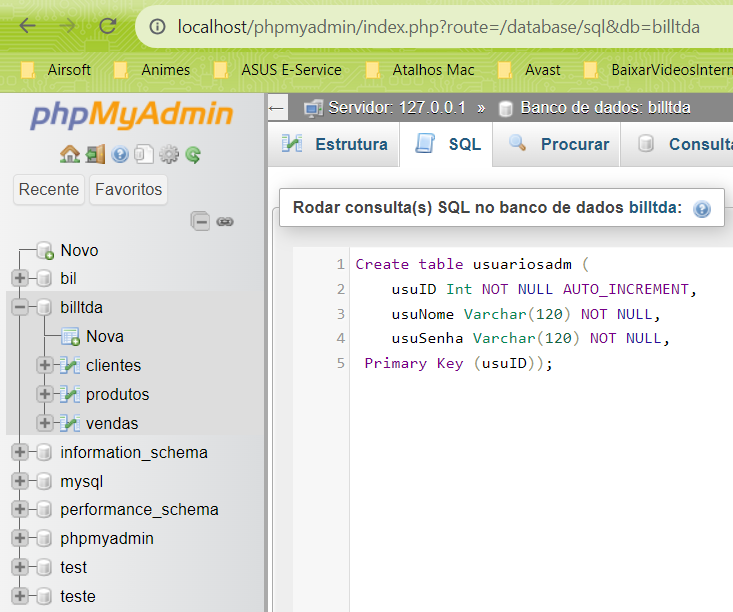
**usuID Int NOT NULL AUTO\_INCREMENT,**

**usuNome Varchar(120) NOT NULL,**

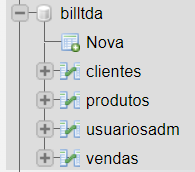
**usuSenha Varchar(120) NOT NULL,**

**Primary Key (usuID));**

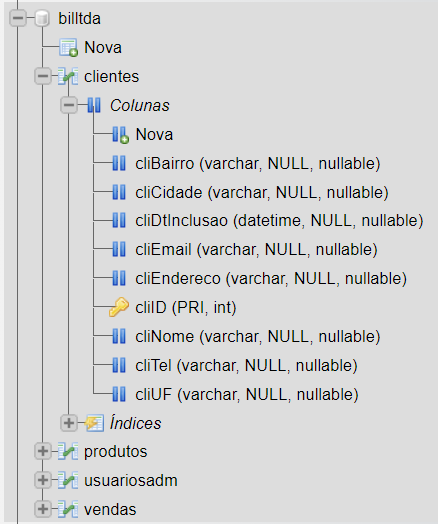
Veja como ficou, conforme figura abaixo:



Clique no botão **Continuar(Executar)**. Pronto, veja que no painel do lado esquerdo, na base de dados **billtda**, estão todas as nossas tabelas, conforme figura abaixo:



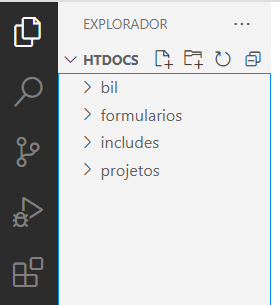
Para verificarmos se todos os campos de nossas tabelas foram criados, basta expandir cada uma delas clicando na setinha com o sinal de **+**, que fica ao seu lado, em seguida clique no sinal de **+**, no item **Colunas**, conforme figura abaixo:



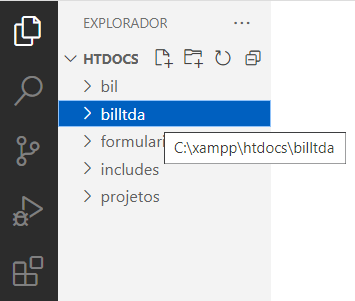
Pronto! Conferida todas as nossas tabelas estamos prontos para criar nosso aplicativo no PHP.

## Criando o Aplicativo para a Empresa Fictícia Bicicletaria Imaginária - LTDA no PHP

O primeiro passo e criarmos a pasta que vai receber nossos scripts **PHP** referente ao nosso aplicativo, lembrando que não é um aplicativo de e-commerce, ou seja, vamos trabalhar com recursos da internet utilizando também a intranet. Para isto crie uma pasta chamada **billtda**, dentro da pasta web do servidor web que estiver utilizando, no nosso caso, estamos usando o servidor **Apache** o caminho será: **C:\xampp\htdocs\billtda**, abra o **VS Code**, em seguida clique na pasta **htdocs**, conforme figura abaixo:



Em seguida clique no ícone da pastinha cm o sinal de +, e digite **billtda**, e pressione a tecla **Enter**, vejam como ficou, conforme figura abaixo:



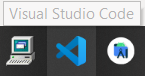
Pronto, já estamos aptos s começar a desenvolver nosso aplicativo com **PHP** e **MySQL**.

## Conectando um Script PHP com MySQL

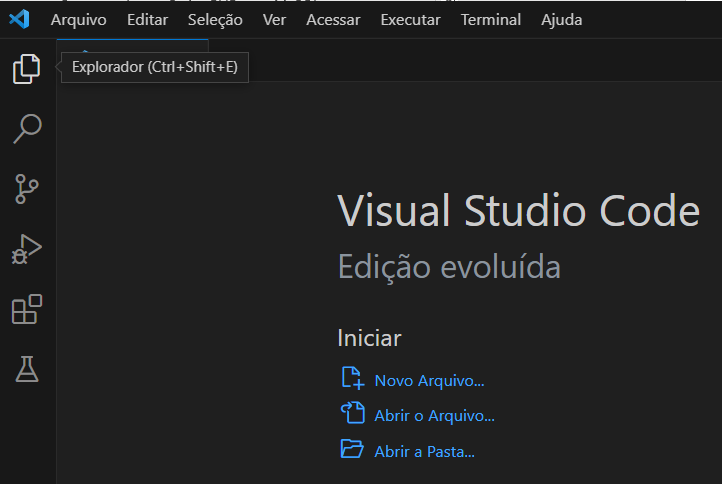
**MySQLi** significa **MySQL Improved**. Trata-se de uma extensão exclusiva para o **MySQL** que adiciona novos recursos para uma interface de database **MySQL**. O **MySQLi** é tanto processual quanto orientada por objeto – este último atributo foi herdado da versão mais antiga do **MySQL**.

## Como Usar o MySQLi para Conectar um Script PHP com o MySQL

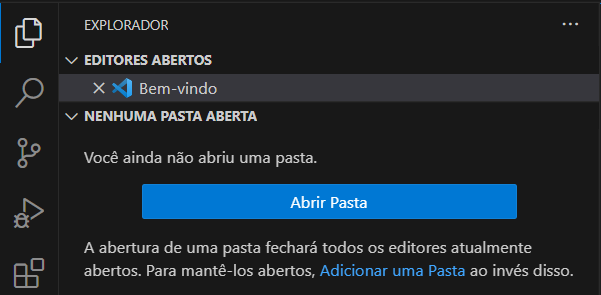
O **MySQL** original divide uma tarefa em procedimentos lineares passo a passo, o que faz com que seja difícil de modificá-la. Afinal, você precisa editar o código desde o começo. Enquanto isso, o **MySQLi** vê os dados como um conjunto de objetos intercambiáveis com funções, permitindo que os usuários adicionem ou removam dados com facilidade. Vamos agora implementar a conexão com a nossa base de dados **MySQL**, que chamamos de **billtda**, utilizando o **MySQLi**, abra o **Visual Studio Code**, basta clicar no ícone do programa que se encontra na barra de ferramentas do **Windows**, conforme figura abaixo:



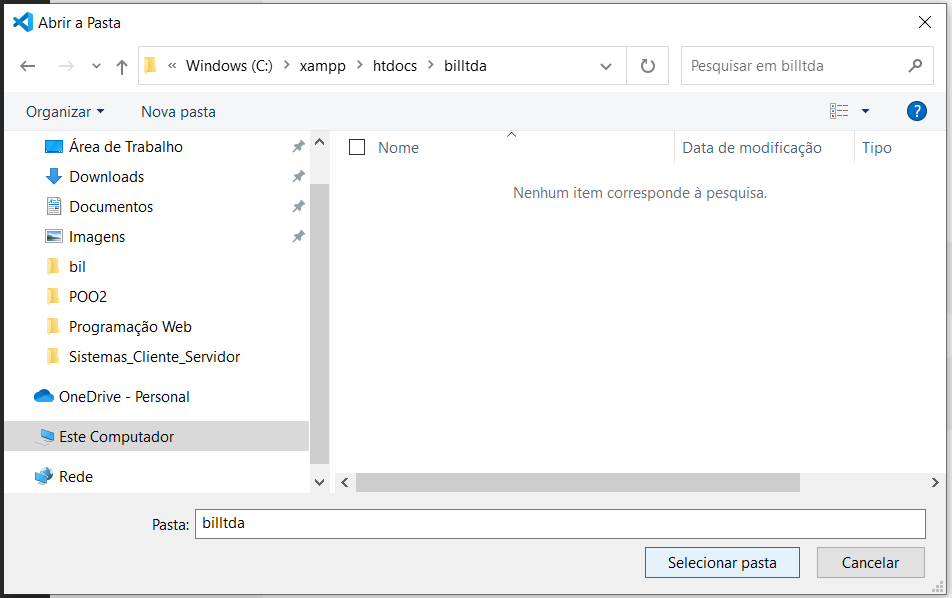
Vai abrir o **Visual Studio Code**, conforme figura abaixo:



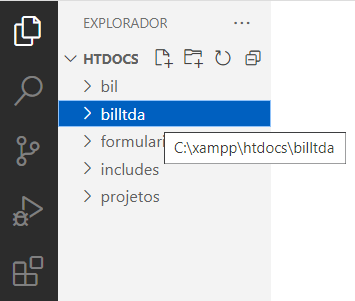
Em seguida clique no item **Explorador(Ctrl+Shift+E)**, vai abrir outro item, conforme figura abaixo:



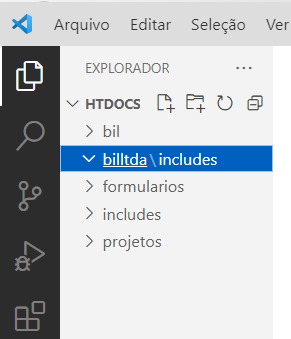
Clique no botão Abrir Pasta, ou no item **Iniciar**, clique sobre o item **Abrir a Pasta...**, vai abrir o **Explorador de arquivos**, procure o seguinte caminho: **C:\xampp\htdocs\billtda**, conforme figura abaixo:



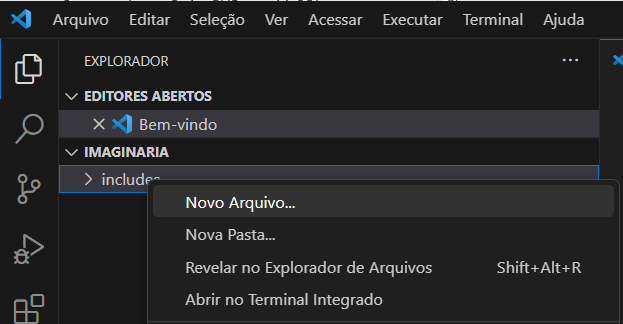
Clique no botão **Selecionar pasta**, e vejam como ficou, conforme figura abaixo:



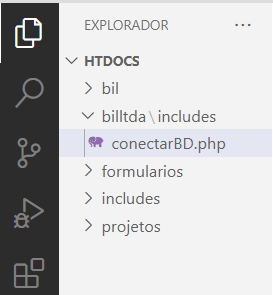
Clique com o mouse sobre a pasta **billtda**, em seguida clique com o mouse sobre o ícone de uma pasta com o sinal de +, e vamos criar uma pasta chamada **includes**, pressione a tecla **Enter**, e vejam como ficou, conforme figura abaixo:



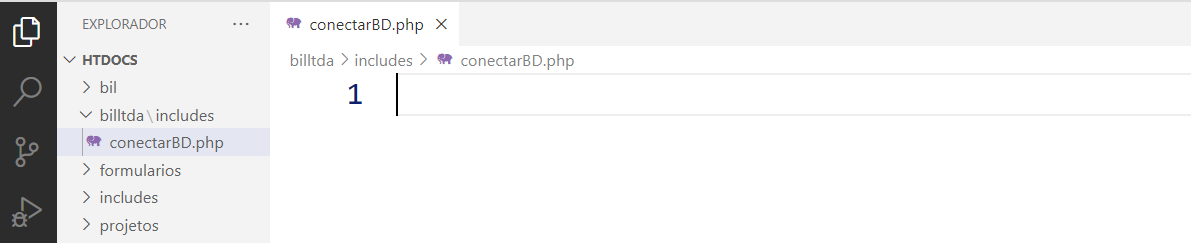
Clique com o botão do mouse sobre a pasta **includes**, e selecione o item **Novo Arquivo...**, conforme figura abaixo:



No local onde o cursor do mouse está piscando, digite **conectarBD.php**, conforme figura abaixo:



Pressione a tecla **Enter**, e vejam que o cursor do mouse vai ficar piscando na **linha 1**, no editor que vai abrir do lado direito da tela, conforme figura abaixo:



Neste local, digite o código abaixo:

**<?php**

**/\* Este script conecta um banco de dados MySQL conforme parâmetros enviados\*/**

**$nomeServidor = "localhost";**

**$nomeBD = "billtda";**

**$usuario = "root";**

**$senha = "";**

**$conexao = mysqli\_connect($nomeServidor, $usuario, $senha, $nomeBD);**

**// Fazer a Conexão**

**@$conexao = mysqli\_connect($nomeServidor, $usuario, $senha, $nomeBD);**

**// Testar a Conexão**

**if(mysqli\_connect\_error())**

**{**

**echo "<p>ERRO: (" . mysqli\_connect\_errno($con) . ") " . mysqli\_connect\_error($con) . "</p>";**

**exit;**

**}**

**else**

**{**

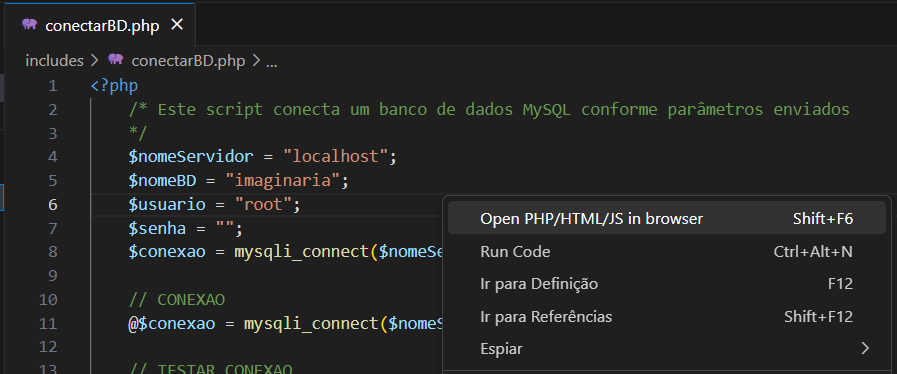
**echo "<p>Conexão com a Base de Dados do MySQL $nomeBD Estabelecida com Sucesso !!!!.</p>";**

**mysqli\_close($conexao);**

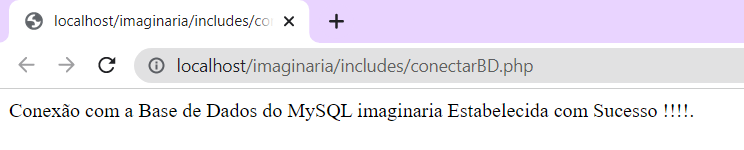
**}**

**?>**

Agora, vamos fazer o teste da nossa conexão, para isto clique com o botão esquerdo do mouse na área do código **PHP**, e selecione o item **Open PHP/HTML/JS in browser**, conforme figura abaixo:



Vão aparecer as opções de browsers, selecione o seu preferido, em seguida deve aparecer o seu navegador e observe como ficou o resultado na figura abaixo:



Se correu tudo bem, deve aparecer a mensagem, conforme figura acima. O principal método usado neste script, na **linha 7**, é o **mysqli\_connect()**. Essa é uma função **PHP** interna para estabelecer uma nova conexão com um servidor **MySQL**. No começo do nosso código, vamos ver algumas declarações de variáveis e valores atribuídos a essas variáveis. Normalmente, precisamos de quatro deles para estabelecer uma conexão de banco de dados apropriada, na **linha 3**, **$nomeServidor**, na **linha 4**, **$nomeBD**, na **linha 5**, **$usuario** e na **linha 6**, **$senha**. No código, vamos definir os detalhes do nosso banco de dados como os valores para aquelas variáveis, para que eles possam ser passados para a função. Na **linha 13**, temos o teste para verificar se a conexão não foi bem-sucedida, isso basicamente mata o nosso script e nos apresenta o erro de mensagem de conexão que definimos. Por padrão, o erro de conexão **MySQL** vai dizer se **@$conexao**, na **linha10**, falhou seguido de uma mensagem de erro exata descrevendo o problema. Por outro lado, se a conexão com o **MySQL** foi bem-sucedida, o código vai imprimir a seguinte frase: **Conexão com a Base de Dados do MySQL billtda estabelecida com sucesso !!!!**. A última parte do código é a função **mysqli\_close**, na **linha 21**, que simplesmente vai fechar a conexão com o banco de dados de modo manual. Se isso não for especificado, as conexões **MySQL** vão fechar por conta própria quando o script for encerrado.

## Considerações Sobre os Scripts de Conexão

Um dos recursos mais importantes que ambos suportam é o de **prepared statements**, que acelera o tempo necessário para o **MySQL** executar a mesma solicitação múltiplas vezes. Ele também é usado para evitar ataques de **injeção SQL** na hora de promover mudanças para o banco de dados. Independentemente do método que você estiver utilizando, você vai precisar da informação correta para que possa se conectar com o banco de dados **MySQL** que você criou. Você também vai precisas do nome do servidor ou do **hostname** correto para a configuração. Estamos usando o **localhost** como o **hostname** para o nosso servidor **MySQL**. As funções **mysql\_** originais estão obsoletas e não devem mais ser usadas, já que elas não oferecem segurança e não podem mais ser mantidas ou desenvolvidas. Para nossas aplicações reais, vamos utilizar uma conexão que não mostre nenhuma mensagem na tela. Para isto abra o **Visual Studio Code**, basta clicar no ícone do programa que se encontra na barra de ferramentas do **Windows**, conforme vimos na figura xxx. Vai abrir o **Visual Studio Code**, conforme vimos na figura xxx. Clique com o mouse sobre a pasta **billtda**, em seguida clique com o mouse sobre a pasta **includes**, selecione o item **Novo Arquivo...**, conforme vimos na figura xxx. No local onde o cursor do mouse está piscando, digite **conectarBD2.php**, conforme vimos na figura xxx. Pressione a tecla **Enter**, e vejam que o cursor do mouse vai ficar piscando na **linha 1**, no editor que vai abrir do lado direito da tela, conforme vimos na figura xxx. Neste local, digite o código abaixo:

**<?php**

**define("HOST","localhost");**

**define("USUARIO","root");**

**define("SENHA","");**

**define("NOMEBD","billtda");**

**$conexao = new mysqli(HOST,USUARIO,SENHA,NOMEBD);**

**if($conexao->error)**

**{**

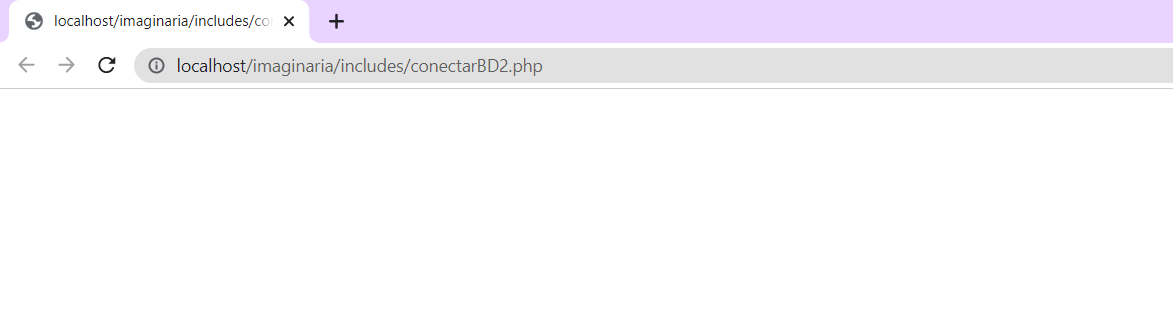
**//A função die, significa que vamos matar o processo, encerrar o processo**

**die("<pre>"."Não foi possível conectar-se ao MySQL. Favor contactar o Administrador !!!!". $conexao->error);**

**}**

**?>**

Agora, vamos fazer o teste da nossa conexão, para isto clique com o botão esquerdo do mouse na área do código **PHP**, e selecione o item **Open PHP/HTML/JS in browser**, conforme vimos na figura xxx. Em seguida deve aparecer o seu navegador e observe como ficou o resultado na figura abaixo:



Se correu tudo bem, deve aparecer uma página vazia, conforme figura acima. O script de conexão com o **MySQL** que criamos acima, é para que possamos trabalhar localmente no nosso computador. A função **mysql\_connect**, vai abrir uma conexão com o banco de dados **MySQL**, caso não consiga ele vai exibir a mensagem: **Não foi possível conectar-se ao MySQL. Favor contactar o Administrador**. Em caso positivo, a página aparece vazia.

## Um pouco sobre o uso de controle de sessão no PHP

## O que é controle de sessão?

Já ouvimos falar que "HTTP é um protocolo sem informações de estado". Isto significa dizer que o protocolo não tem nenhuma maneira predefinida de manter o estado entre duas transações, ou seja, quando um usuário solicita uma página, seguida de outra, o protocolo HTTP não fornece uma maneira de dizer se as duas solicitações vêm do mesmo usuário.

Por este motivo a idéia de controle de sessão é ser capaz de monitorar um usuário durante uma única sessão em um script, seja na intranet ou em um Web site. Desta forma, poderemos facilmente suportar o login de um usuário e mostrar o conteúdo de acordo com seu nível de autorização ou preferências pessoais. Como dito anteriormente, desde a versão 4, o **PHP** inclui funções de controle de sessão nativas. Com a introdução das variáveis superglobais, em especial a superglobal **@\_SESSION**, a abordagem para controle de sessão mudou um pouco.

## Conhecendo a funcionalidade de sessão básica

Sessões no **PHP** têm por definição um **ID** ou **identificador** único de sessão, um número criptograficamente aleatório. Esse **ID** de sessão é gerado pelo próprio PHP e vai ser armazenado no lado cliente pelo tempo de vida de uma sessão.

Neste caso, ele pode ser armazenado no computador de um usuário em um cookie ou passado junto com URLs. Este **ID** de sessão vai a atuar como uma chave que permite registrar variáveis particulares como as chamadas variáveis de sessão. O conteúdo dessas variáveis é armazenado no servidor. O **ID** de sessão é a única informação visível no lado cliente, ou seja, se no momento de uma conexão particular com seu sistema ou site, o **ID** de sessão for visível por um cookie ou pelo URL, poderemos acessar as variáveis de sessão armazenadas no servidor para essa sessão. Por default, as variáveis de sessão são armazenadas em arquivos simples no servidor. Observe se aparece uma string de dados de aparência aleatória no seu URL, é possível que ela seja alguma forma de controle de sessão.

## Como Implementar sessões simples

Vejamos os passos básicos para utilizar as sessões, conforme abaixo:

* Iniciar uma sessão;
* Registrar variáveis de sessão;
* Utilizar variáveis de sessão;
* Remover registros de variáveis e destruir a sessão.

Veja que todos esses passos não acontecem necessariamente no mesmo script e alguns deles acontecerão em vários scripts.

### Iniciar uma sessão

Observe que antes de utilizar a funcionalidade da sessão, vamos precisar iniciar realmente uma sessão. Existem algumas maneiras de fazer isso. A primeira e mais simples, é iniciar um script com uma chamada para a função **session\_start( )**: Esta função vai verificar se existe um **ID** de sessão atual. Se não existir, essa função criará um. Se já existir um, vai ser carregada as variáveis registradas de sessão para que possamos utilizá-Ias. Sempre faça a chamada de **session\_start( )** no início de todos os scripts que utilizam sessões. A segunda maneira de começar uma sessão é configurar o **PHP** para iniciar uma automaticamente quando um usuário logar no nosso sistema. Neste caso, poderemos fazer isso com a opção **session.auto\_start** que se encontra no arquivo **php.ini,** mais adiante veremos como configurar este arquivo.

### Registrar variáveis de sessão

As variáveis de sessão são armazenadas no array superglobal chamado **$\_SESSION,** para criar uma variável de sessão, devemos simplesmente configurar o array **$\_SESSION**, como mostra a sintaxe abaixo:

**$\_SESSION['variavel'] = 5;**

Acabamos de criar a variável de sessão que será monitorada até que a sessão seja finalizada ou até que seja desconfigurada manualmente, ou seja, destruída.

### Utilizar variáveis de sessão

Para utilizarmos as variáveis de sessão, devemos colocar as mesmas em escopo de modo que possam ser utilizadas, primeiro inicie uma sessão utilizando **session\_start( )**. Em seguida, poderemos acessar a variável via array superglobal **$\_SESSION,** no nosso exemplo, como **$\_SESSION['variavel']**.

Ao utilizar um objeto como variável de sessão, é importante incluir a definição de classe antes de chamar **session\_start( )** para recarregar as variáveis de sessão, pois desta forma, o **PHP** sabe como reconstruir o objeto da sessão.

Se você estiver utilizando a opção **register\_globals** ativado no arquivo **php.ini**, ao codificar, lembre-se de que uma variável de sessão não pode ser sobrescrita por dados **GET** ou **POST**, o que é um bom recurso de segurança.

Em contrapartida, você precisa ser cuidadoso ao verificar se as variáveis de sessão foram configuradas via comando **isset( )** ou **empty( )**. Não esqueça que as variáveis podem ser configuradas pelo usuário via **GET** ou **POST.** Podemos verificar se é uma variável de sessão registrada através do array **$\_SESSION,** observe a sintaxe abaixo:

**if (isset($\_SESSION['variavel']))...**

### Remover registro de variáveis e destruir uma sessão

Ao terminarmos de utilizar uma variável de sessão, podemos desconfigurá-la. Para que isto ocorra, devemos desconfigurar diretamente o elemento apropriado do array **$**\_**SESSION**, como mostra a sintaxe abaixo:

**$\_SESSION = array( );**

Ao terminar o uso de uma sessão, devemos primeiro remover o registro de todas as variáveis e então chamar o array **session\_destroy( )**, desta forma estaremos limpando o **ID** da sessão.