

## === Login de Acesso

Vamos criar, agora, os scripts que serão necessários e responsáveis por validar na base de dados (tabela usuário) a existência daquele “usuário” e daquela “senha” para então permitir acesso ao menu de opções.

### 1º - Alteração na base de dados:

- Inclusão da tabela “usuário”:

```

28
29
30 CREATE TABLE `produtos` (
31   `id` int(11) NOT NULL,
32   `nome` varchar(50) NOT NULL,
33   `estoque` int(11) NOT NULL
34 ) ENGINE=MyISAM DEFAULT CHARSET=utf8mb4;
35
36
37 -- Extrair dados da tabela `produtos`
38
39 INSERT INTO `produtos` (`id`, `nome`, `estoque`) VALUES
40 (1, 'Pão', 203),
41 (2, 'Arroz', 324),
42 (3, 'Chocolate', 342),
43 (4, 'Dolly', 213),
44 (5, 'Uva', 21);
45
46
47 -- Estrutura da tabela `usuario`
48
49
50
51 CREATE TABLE `usuario` (
52   `Login` varchar(5) NOT NULL,
53   `Senha` int(11) NOT NULL
54 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
55
56
57 -- Extrair dados da tabela `usuario`
58
59
60 INSERT INTO `usuario` (`Login`, `Senha`) VALUES
61 ('a', 123),
62 ('b', 456);
63

```

Figura 1- Código fonte da criação do banco

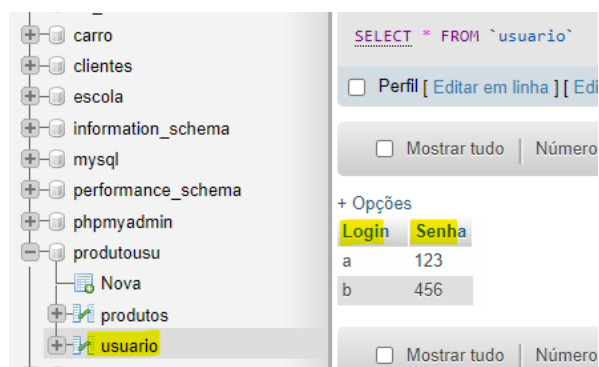
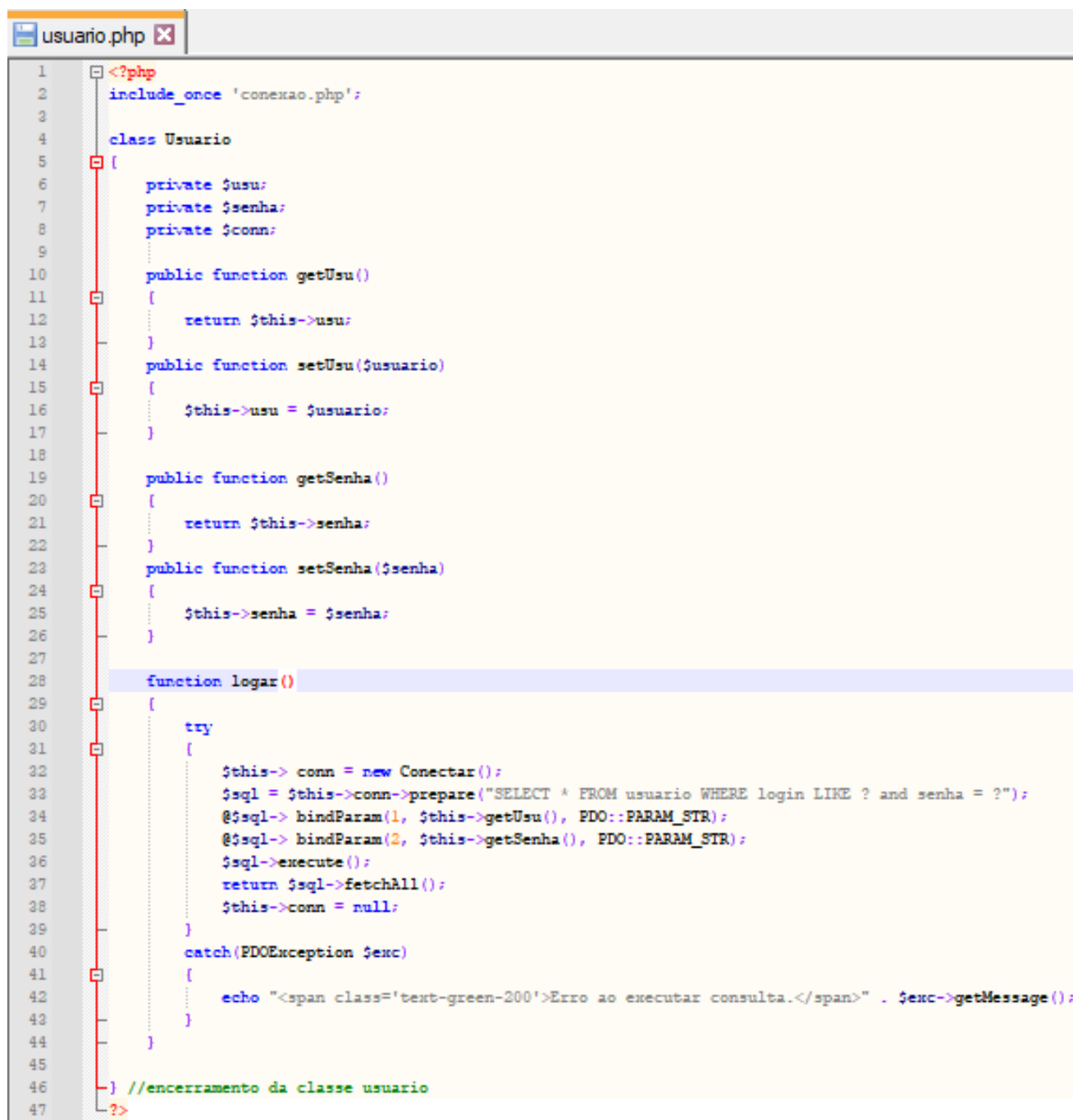


Figura 2- Layout do banco

**2º - Criação da classe de modelagem “Usuario”:**

```
1 <?php
2 include_once 'conexao.php';
3
4 class Usuario
5 {
6     private $usu;
7     private $senha;
8     private $conn;
9
10    public function getUsu()
11    {
12        return $this->usu;
13    }
14    public function setUsu($usuario)
15    {
16        $this->usu = $usuario;
17    }
18
19    public function getSenha()
20    {
21        return $this->senha;
22    }
23    public function setSenha($senha)
24    {
25        $this->senha = $senha;
26    }
27
28    function login()
29    {
30        try
31        {
32            $this->conn = new Conectar();
33            $sql = $this->conn->prepare("SELECT * FROM usuario WHERE login LIKE ? and senha = ?");
34            @$sql->bindParam(1, $this->getUsu(), PDO::PARAM_STR);
35            @$sql->bindParam(2, $this->getSenha(), PDO::PARAM_STR);
36            $sql->execute();
37            return $sql->fetchAll();
38            $this->conn = null;
39        }
40        catch(PDOException $exc)
41        {
42            echo "<span class='text-green-200'>Erro ao executar consulta.</span>" . $exc->getMessage();
43        }
44    }
45
46 } //encerramento da classe usuario
47 ?>
```

*Figura 3 - Classe de Modelagem Usuario***3º - Criação do formulário para entrada de dados dos campos do login de acesso:**



The image shows a login form titled "Login de acesso". It has a light green background with a darker green border. The form contains two input fields: "Usuario:" and "Senha:". Below the "Senha:" field is a button labeled "Acessar". The form is set against a dark green background.

Figura 4 - Tela para acesso

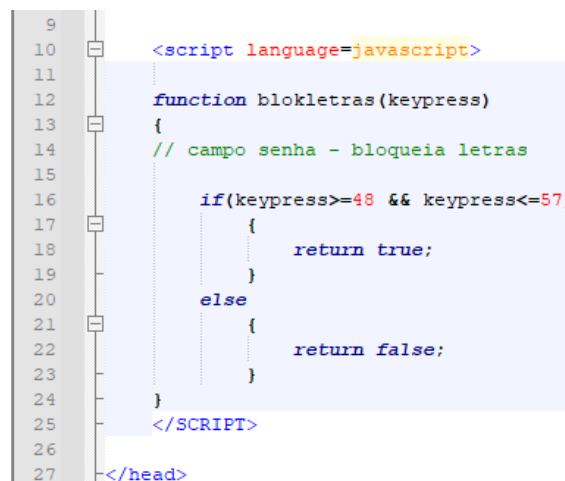
Nas caixas de texto foi setada a propriedade **required** pois obriga que algo seja digitado no campo, impedindo assim que o mesmo fique vazio antes de ir para o próximo campo.

Na caixa de texto "txtsenha" também foi setada a propriedade **maxlength = "3"** para que aceite a digitação de no máximo 3 caracteres, limite deste campo no banco de dados.

Além das duas propriedades anteriores, na caixa de texto "txtsenha", foi incluído um método "Java Script" (dentro da seção HEAD) para aceitar enquanto se digita somente "números" nesta caixa, e foi acionada através do evento **onkeypress="return blokletras(window.event.keyCode)":**

```
name="txtsenha"
maxlength = "3"
required
onkeypress="return blokletras(window.event.keyCode) ">
```

Figura 6- Propriedades caixa txtsenha



The image shows a code editor with a line number margin on the left (9 to 27). The code is as follows:

```

9
10 <script language=javascript>
11
12 function blokletras(keypress)
13 {
14 // campo senha - bloqueia letras
15
16 if(keypress>=48 && keypress<=57)
17 {
18     return true;
19 }
20 else
21 {
22     return false;
23 }
24 }
25 </SCRIPT>
26
27 </head>
```

Figura 5 - Método para bloquear entrada de letras na digitação

Este método consiste em checar na tabela ASC II se o código da última tecla digitada (keypress) corresponde a algum código pertencente a "categoria numérica" (códigos de 48 a 57 – de 0 a 9).

Abaixo da criação do formulário, após o </form>, digite o código PHP a seguir.

```

67 <?php
68 extract($_POST, EXTR_OVERWRITE);
69 if(isset($btnconsultar))
70 {
71     include_once 'usuario.php';
72     $u = new Usuario();
73     $u->setUsu($txtnome);
74     $u->setSenha($txtsenha);
75     $pro_bd=$u->logar();
76
77     $existe = false;
78     foreach($pro_bd as $pro_mostrar)
79     {
80         $existe = true;
81         ?>
82         <br><b> <?php echo "Bem vindo! Usuário: ".$pro_mostrar[1]; ?></b> <br><br>
83
84         <center>
85             <input type="button" name="btntentrar"> <a href="menu.html">Entrar</a>
86         </center>
87         <?php
88     }
89     if($existe==false)
90     {
91         header("location:loginInvalido.html");
92     }
93 }
94 ?>

```

Figura 6 - Código fonte para checagem do acesso

**Linha 77:** foi criado um “flag” para fluxo, uma variável booleana atribuída a “false”

**Linha 80:** caso exista no banco o usuário e senha digitados então o “flag” passa a ser “true

**Linha 82:** apresenta o nome do usuário que logou, logo abaixo da tela de login:

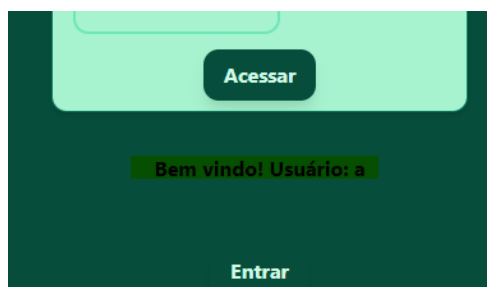


Figura 7 - Mensagem de bem vindo

Ao clicar no botão “Entrar” a tela inicial do sistema (menu de opções) será aberta:



Figura 8 - Tela Inicial do sistema

**Linha 89:** caso “não” exista no banco o usuário e senha digitados, então o “flag” continua a ser “false”

**Linha 91:** então através do comando php `header("location:arquivo")` é chamado (linkado) o arquivo que apresenta a tela de “login inválido”:

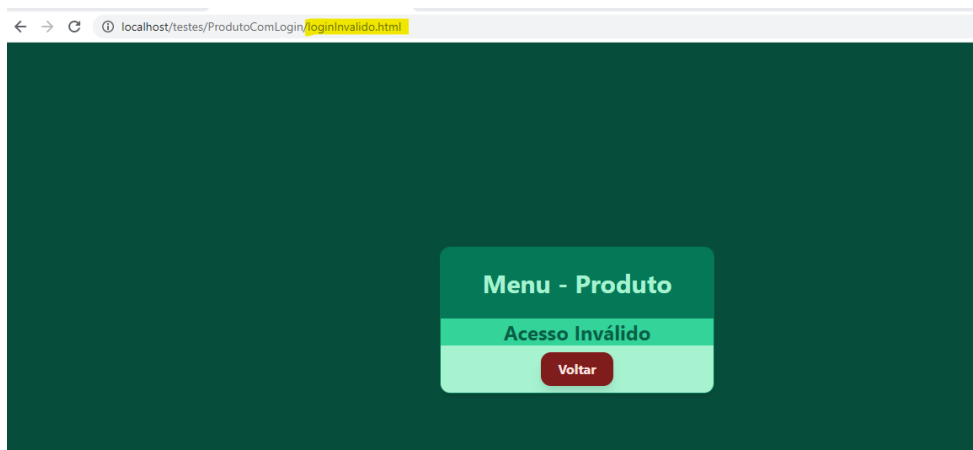


Figura 9 - Tela para "Login Inválido"