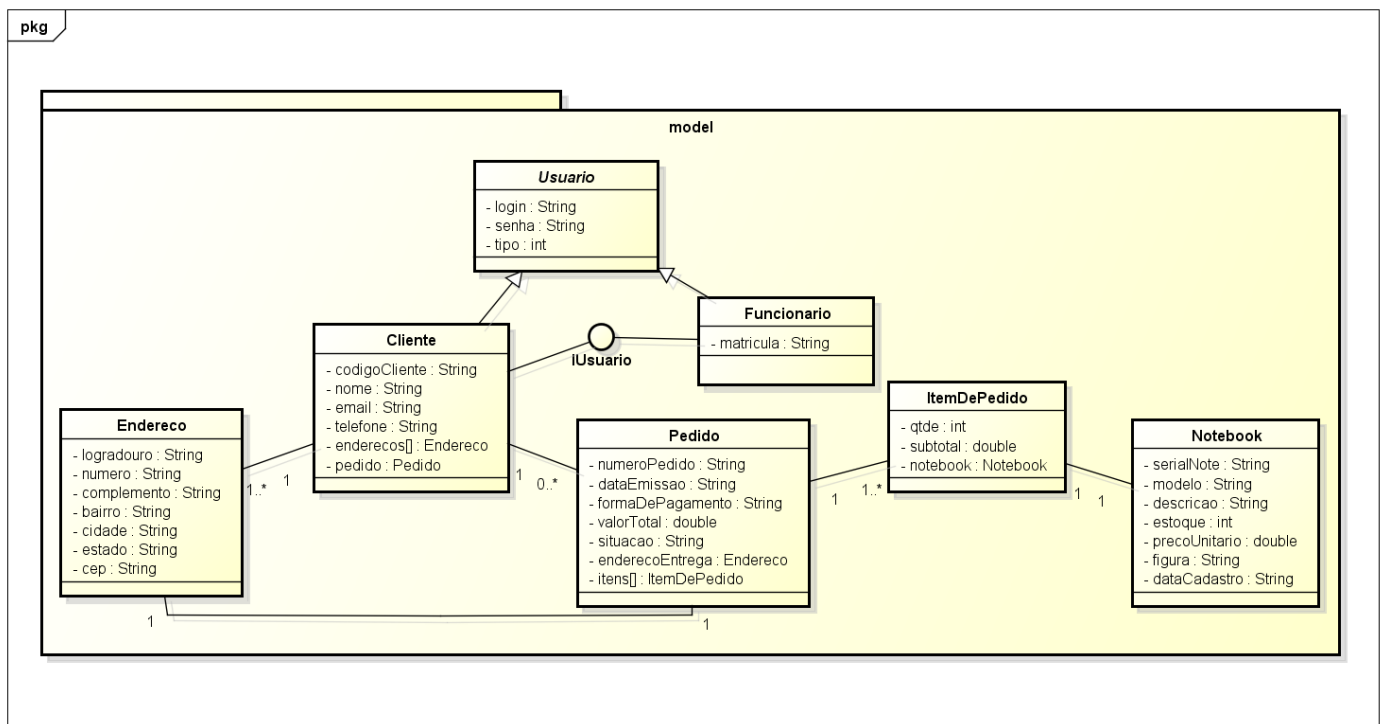


# Projeto *InfoNote\_13*



powered by astah®

## Execute este passo a passo:

- 1 – Copiar e colar o projeto **InfoNote\_12** e renomear para **InfoNote\_13**.
- 2 – Criar no **mysql** e ativar o banco **InfoNote**.
- 3 – Criar as tabelas **Usuario**, **Cliente**, **Funcionario** e **Endereco**, com os atributos referentes às Classes deste projeto.
- 4 – Criar conexão com o banco de dados.
  - 4.1 – Abrir o arquivo **config.ini** e configurar o local da database, o arquivo de driver, o usuário da database e senha do usuário:

(Abaixo da linha ajuda=ajuda.txt)

*url = jdbc:mysql://localhost:3306/infonote*

*driver=com.mysql.jdbc.Driver*

*login=root*

*senha=alunolab*

#### 4.2 – Abrir a Classe **Configurador**:

##### 4.2.1 – Declarar as 4 variáveis:

(Após a linha *private String arquivoAjuda;*)

```
private String url;  
private String driver;  
private String login;  
private String senha;
```

##### 4.2.2 – Carregar as respectivas propriedades e cada atributo.

(Após a linha *arquivoAjuda = prop....*)

```
url = prop.getProperty("url");  
driver = prop.getProperty("driver");  
login = prop.getProperty("login");  
senha = prop.getProperty("senha");
```

##### 4.2.3 – Completar os métodos **get** que estão faltando no fim da Classe (Lembrar que nesta Classe não existem métodos set)

4.3 – Copiar a Classe **Conexao** que está no pacote util do projeto **17\_ConexaoBD** e colar dentro do pacote **util** deste Projeto **InfoNote**.

#### 4.4 – Carregar o **conector** do driver **mysql** neste projeto.

5 – Criar o pacote **model.DAO** dentro do pacote **model**.

6 – Criar a Classe **UsuarioDAO** dentro do pacote **model.DAO**

##### 6.1 – Inicializar os atributos para conexão ao banco de dados:

```
Configurador config = new Configurador();
```

```
String url;  
String driver;  
String login;  
String senha;
```

```
public UsuarioDAO() {  
  
    url = config.getUrl();  
    driver = config.getDriver();  
    login = config.getLogin();  
    senha = config.getSenha();  
}
```

## 6.2 – Criar o método *inserir()*:

```
public static Usuario inserir(String login, String senha, int tipo) {

    Usuario usuario = null;
    UsuarioDAO userDAO = new UsuarioDAO();

    try {
        // Criação do insert
        String sql = "insert into usuario (login, senha, tipo) values (?, ?, ?)";

        // Obter a conexão com o banco de dados
        Conexao conex = new Conexao(userDAO.url, userDAO.driver,
                                     userDAO.login, userDAO.senha);

        // Abrir a conexão
        Connection con = conex.obterConexao();

        // Preparar o comando para ser executado
        PreparedStatement comando = con.prepareStatement(sql);

        comando.setString(1, login);
        comando.setString(2, senha);
        comando.setInt(3, tipo);

        // Comando executado
        comando.executeUpdate();

    } catch (Exception e) {
        System.out.println(e.getMessage());
    }

    return usuario;
}
```

7 – Como a Classe **ClienteDAO** grava os dados somente do Cliente (sem endereço), faz-se necessária a criação de um terceiro método construtor na Classe **Cliente**, sem as funcionalidades de Endereço:

```
public Cliente(String login, String senha, int tipo, String codigoCliente,
               String nome, String email, String telefone) {
    super(login, senha, tipo);
    this.codigoCliente = codigoCliente;
    this.nome = nome;
    this.email = email;
    this.telefone = telefone;
}
```

8 – Criar a Classe **ClienteDAO** dentro do pacote **modelDAO**.

```
package model.DAO;

import java.sql.Connection;
import java.sql.PreparedStatement;

import model.Cliente;
import util.Conexao;
import util.Configurador;
```

```

public class ClienteDAO {

    Configurador config = new Configurador();

    String url;
    String driver;
    String login;
    String senha;

    public ClienteDAO() {

        url = config.getUrl();
        driver = config.getDriver();
        login = config.getLogin();
        senha = config.getSenha();
    }

    public static Cliente inserir(String login, String senha, int tipo,
        String codigoCliente, String nome, String email, String telefone) {

        Cliente cliente = null;
        ClienteDAO cliDAO = new ClienteDAO();

        try {
            // Criação do insert
            String sql = "insert into cliente " +
                "(codigocliente, nome, email, telefone, fklogin) " +
                "values (?, ?, ?, ?, ?)";

            // Obter a conexão com o banco de dados
            Conexao conex = new Conexao(cliDAO.url, cliDAO.driver,
                cliDAO.login, cliDAO.senha);

            // Abrir a conexão
            Connection con = conex.obterConexao();

            // Preparar o comando para ser executado
            PreparedStatement comando = con.prepareStatement(sql);

            comando.setString(1, codigoCliente);
            comando.setString(2, nome);
            comando.setString(3, email);
            comando.setString(4, telefone);
            comando.setString(5, login);
            //não é fklogin, porque aqui a referência é da Classe

            // Comando executado
            comando.executeUpdate();

        } catch (Exception e) {
            System.out.println(e.getMessage());
        }

        cliente = new Cliente(login, senha, tipo, codigoCliente, nome,
            email, telefone );
        return cliente;
    }
}

```

## 9 – Criar a Classe **EnderecoDAO** dentro do pacote **model.DAO**:

```
package model.DAO;

import java.sql.Connection;
import java.sql.PreparedStatement;

import model.Endereco;
import util.Conexao;
import util.Configurador;

public class EnderecoDAO {

    Configurador config = new Configurador();

    String url;
    String driver;
    String login;
    String senha;

    public EnderecoDAO() {

        url = config.getUrl();
        driver = config.getDriver();
        login = config.getLogin();
        senha = config.getSenha();
    }

    public static Endereco inserir(String logradouro, String numero,
        String complemento, String bairro, String cidade,
        String estado, String cep, String codigoCliente) {

        Endereco endereco = null;
        EnderecoDAO endDAO = new EnderecoDAO();

        try {
            // Criação do insert
            String sql = "insert into endereco " +
                "(logradouro, numero, complemento, bairro, cidade, " +
                "estado, cep, fkcodigocliente)" +
                " values (?, ?, ?, ?, ?, ?, ?, ?)";

            // Obter a conexão com o banco de dados
            Conexao conex = new Conexao(endDAO.url, endDAO.driver,
                endDAO.login, endDAO.senha);

            // Abrir a conexão
            Connection con = conex.obterConexao();

            // Preparar o comando para ser executado
            PreparedStatement comando = con.prepareStatement(sql);

            comando.setString(1, logradouro);
            comando.setString(2, numero);
            comando.setString(3, complemento);
            comando.setString(4, bairro);
            comando.setString(5, cidade);
            comando.setString(6, estado);
            comando.setString(7, cep);
            comando.setString(8, codigoCliente);
            //não é fkcodigocliente, porque aqui a referência é da Classe

            // Comando executado
            comando.executeUpdate();

        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

```

    }

    endereco = new Endereco(logradouro, numero, complemento,
                             bairro, cidade, estado, cep);
    return endereco;
}
}

```

10 – Abrir a Classe **InfoNote** e realizar as seguintes alterações:

10.1 – No método **cadastrarUsuario()**, inserir a linha que identifica o tipo correspondente a Cliente:

(Após a linha String senha – Teclado....)

```
int tipo = 0; // porque todo cliente é tipo zero
```

10.2 – Ainda no método **cadastrarUsuario()**, **APAGAR** as seguintes linhas de código:

```

Endereco endereco = new Endereco(logradouro, numero, complemento,
                                   bairro, cidade, estado, cep);

cliente = new Cliente(login, senha, 0, codigoCliente, nome, email,
                      telefone, endereco);

//Aqui acima o tipo recebeu valor zero, para no futuro,
//ocorrer uma associação zero para cliente e um para administrador.

```

10.3 – Digitar no local onde foram removidas as linhas anteriores:

```

usuario = UsuarioDAO.inserir(login, senha, tipo);

cliente = ClienteDAO.inserir(login, senha, tipo, codigoCliente, nome,
                             email, telefone);

Endereco endereco = EnderecoDAO.inserir(logradouro, numero, complemento,
                                          bairro, cidade, estado, cep, codigoCliente);

```

10.4 – Lá no início da Classe, criar objeto **usuario** como objeto global:

```
Usuario usuario;
```

10.5 – As 2 últimas linhas do método **cadastrarUsuario()** devem ser:

```

System.out.println(cliente);
System.out.println(endereco);

```

11 – Na Classe Cliente, no método **toString()**, **APAGAR** a linha:

```
"Endereco: " + enderecos[0];
```

12 – Na Classe **ClienteDAO**, criar o método **buscarPorLoginSenha()**:

```
public static Cliente buscarPorLoginSenha (String login, String senha){
    Cliente cliente = null;

    ClienteDAO cliDAO = new ClienteDAO();

    try{
        String sql = "Select * from usuario u, cliente c " +
            "where u.login=c.fklogin and u.login = ? " +
            "and u.senha = ?";

        Conexao conex = new Conexao(
            cliDAO.url, cliDAO.driver, cliDAO.login, cliDAO.senha);

        Connection con = conex.obterConexao();

        PreparedStatement comando = con.prepareStatement(sql);

        comando.setString(1,login);
        comando.setString(2, senha);

        ResultSet rs = comando.executeQuery();

        if(rs.next()){
            cliente = new Cliente(
                rs.getString("login"),
                rs.getString("senha"),
                rs.getInt("tipo"),
                rs.getString("codigocliente"),
                rs.getString("nome"),
                rs.getString("email"),
                rs.getString("telefone"));
        }

        rs.close();
        comando.close();
        con.close();

    } catch (Exception e){
        System.out.println(e.getMessage());
    }
    return cliente;
}
```

13 – Na Classe **InfoNote**, alterar o método **efetuarLogin()**, deixando-o desta forma:

```
public void efetuarLogin() {
    String login, senha;
    login = Teclado.lerTexto("Digite o login: ");
    senha = Teclado.lerTexto("Digite a senha: ");

    cliente = ClienteDAO.buscarPorLoginSenha(login, senha);

    if (cliente != null) {
        logado = cliente.validarLogin(login, senha);
    }else{
        if (logado) {
            System.out.println("Login efetuado com sucesso!");
        } else {
            System.out.println("Usuário ou senha inválido.");
        }

        int opcao2 = 3;
        do {
            System.out.println("Digite:");
            System.out.println("1 - Para efetuar Login");
            System.out.println("2 - Para cadastrar-se");
        } while (opcao2 != 1 && opcao2 != 2);
    }
}
```

```

        System.out.println("3 - Para sair do sistema");
        opcao2 = Teclado.lerInt("");

        switch (opcao2) {
            case 1:
                efetuarLogin();
                break;
            case 2:
                cadastrarUsuario();
                break;
            case 3:
                System.out.println("Obrigado e volte sempre!");
                break;

            default:
                System.out.println("Opção inválida");
        }
    } while (!logado);
}
}
}

```

14 – Na Classe **Cliente**, deixar o método **validarLogin()** exatamente assim:

```

@Override
public boolean validarLogin (String login, String senha){
    Cliente cliente = ClienteDAO.buscarPorLoginSenha(login, senha);
    if(cliente != null){
        return true;
    }
    return false;
}

```

15 – Para que um funcionário possa fazer login no sistema (Área administrativa):

15.1 – Na Classe **InfoNote**, incluir a constante global **AREA\_ADMINISTRATIVA** e alterar a **SAIR**:

```

private static final int AREA_ADMINISTRATIVA = 9;

private static final int SAIR = 10;

```

15.2 – Criar a Classe **FuncionarioDAO** dentro do pacote **model.DAO**

```

package model.DAO;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

import model.Funcionario;
import util.Conexao;
import util.Configurador;

public class FuncionarioDAO {

    Configurador config = new Configurador();

    String url;
    String driver;
    String login;
    String senha;

    public FuncionarioDAO() {

```



```

        url = config.getUrl();
        driver = config.getDriver();
        login = config.getLogin();
        senha = config.getSenha();
    }

    public static Funcionario buscarPorLoginSenha (String login, String senha){
        Funcionario funcionario = null;

        FuncionarioDAO funcDAO = new FuncionarioDAO();

        try{
            String sql = "Select * from usuario as u, funcionario as f " +
                "where u.login = ? and u.senha = ?";

            Conexao conex = new Conexao(
                funcDAO.url, funcDAO.driver, funcDAO.login,
                funcDAO.senha);

            Connection con = conex.obterConexao();

            PreparedStatement comando = con.prepareStatement(sql);

            comando.setString(1, login);
            comando.setString(2, senha);

            ResultSet rs = comando.executeQuery();

            if(rs.next()){
                funcionario = new Funcionario(
                    rs.getString("login"),
                    rs.getString("senha"),
                    rs.getInt("tipo"),
                    rs.getString("matricula"));
            }

            rs.close();
            comando.close();
            con.close();

        } catch (Exception e){
            System.out.println(e.getMessage());
        }
        return funcionario;
    }
}

```

15.3 – Na Classe **InfoNote**, incluir o case **AREA\_ADMINISTRATIVA** depois do case **AJUDA**:

```

        case AREA_ADMINISTRATIVA:
            info.areaAdministrativa();
            break;

```

15.4 – Adaptar as opções do método **mostrarMenu()**:

```

System.out.println("9 - Área Administrativa");
System.out.println("10 - Sair");

```

15.5 – Na Classe **InfoNote**, criar o método **efetuarLoginAdm()**:

```
public void efetuarLoginAdm() {
    String login, senha;
    login = Teclado.lerTexto("Digite o login: ");
    senha = Teclado.lerTexto("Digite a senha: ");

    Funcionario funcionario = FuncionarioDAO.buscarPorLoginSenha(login, senha);

    if (funcionario != null) {
        logado = funcionario.validarLogin(login, senha);
    } else {
        if (logado) {
            System.out.println("Login efetuado com sucesso!");
        } else {
            System.out.println("Usuário ou senha inválido.");
        }
    }
}
```

15.6 – Na Classe **Funcionário**, alterar o método **validarLogin()**, para que fique exatamente assim:

```
@Override
public boolean validarLogin (String login, String senha){
    Funcionario funcionario = FuncionarioDAO.buscarPorLoginSenha(login, senha);
    if(funcionario != null){
        return true;
    }
    return false;
}
```

15.7 – Na Classe **InfoNote**, criar o método **areaAdministrativa()**:

```
public void areaAdministrativa() {

    InfoNote info = new InfoNote();

    efetuarLoginAdm();

    System.out.println("Opções Administrativas\n");
    System.out.println("1 - Cadastrar Notebook");
    System.out.println("2 - Mostrar Notebooks");
    System.out.println("3 - Editar Notebook");
    System.out.println("4 - Excluir Notebook");
    System.out.println("5 - Sair");

    int opcao = 5;
    do {
        opcao = Teclado.lerInt("Digite sua opção: ");

        switch (opcao) {

            case 1:
                info.cadastrarNotebook();
                break;

            case 2:
                info.mostrarNotebooks();
                break;

            case 3:
                info.editarNotebook();
                break;
        }
    }
}
```

```

        case 4:
            info.excluirNotebook();
            break;

        case 5:
            System.out.println("Saída do Sistema");
            break;

        default:
            System.out.println("Opção inválida!");
    }
    Teclado.lerTexto("Pressione uma tecla para continuar...");
} while (opcao != 5);
}

```

16 – No BD deste projeto no MySQL, criar a tabela **Notebook**, com as colunas correspondentes à Classe Notebook deste projeto.

17 – Criar a Classe **NotebookDAO** dentro do pacote **model.DAO**.

18 – Na Classe **NotebookDAO** criar o método **inserirNotebook()** (Sugestão: Copiar este método de outra Classe deste pacote e depois realizar as adaptações necessárias)

19 – Na Classe **NotebookDAO** criar o método **buscarTodos()**

20 – Na Classe **NotebookDAO** criar o método **excluir()**

21 – Na Classe **NotebookDAO** criar o método **atualizar()**, onde poderão ser atualizados todos os campos, exceto **serialnote** e **modelo**.

22 – Na Classe **InfoNote**, criar o método **cadastrarNotebook()**.

23 – Na Classe **InfoNote**, editar o método **buscarNotebooks()** para que ele passe a trabalhar com a tabela no BD.

DICA: Este método ficará igual ao método **mostrarNotebooks()**.

24 – Na Classe **InfoNote**, criar o método **editarNotebook()**.

25 – Na Classe **InfoNote**, criar o método **excluirNotebook()**.