**UNIVERSIDADE FEDERAL DA PARAÍBA**

CENTRO DE INFORMÁTICA

MÉTODOS DE PROJETO DE SOFTWARE

SOFTWARE PARA COMÉRCIO

DE COMUNICAÇÃO VISUAL

Diagrama de Classe

João Pessoa, dezembro de 2017

**Carlos Alberto**

[cans10cans10@gmail.com](mailto:cans10cans10@gmail.com)

**Roberto Nóbrega**

[roberto.computing@gmail.com](mailto:roberto.computing@gmail.com)

**Validação**

Prof. Raoni Kulesza

PADRÕES UTILIZADOS NO PROJETO

|  |  |  |  |
| --- | --- | --- | --- |
| Padrões | Classes envolvidas na implementação | Justificativa | Cor |
| FACATE | **Classes:** UsuarioForm\_cadatrar, FachadaValidarNomeSenha, ValidarNome e ValidarSenha | Quando for validar um cadastro de uma pessoa, tem-se que validar o login e senha, então a fachada já chama os dois métodos simultaneamente em uma só classe. | Roxa |
| ADAPTER | **Classes:** ControlImplem, ControlCliente, ControlDistribuidor, ControlFuncionario, ControlGerente, ControlPessoaJuridica, infraAdapter, UsuarioAdapterBD, ClienteAdpterBD, FuncionarioAdapterBD, ProdutoAdapterBD, VendaAdapterBD, Venda\_ProdutosAdapetrBD, UsuarioBD, ClienteBD, FuncionarioBD, ProdutoBD, VendaBD e Venda\_ProdutosBD  **Interface:** InfraAdapter | Para gravar os dados de uma Cliente, Funcionário, Gerente, Distribuidor, Administrador, Produto e Vendas, todos os dados necessários são capturados das classes do Pacote view, repassados ao as respectivas Classes Control no Pacote Business, que aciona a classe ‘InfraAdapter’, que distribuir para a classe que realiza a execução de gravar. No nosso caso, usamos apenas o Banco de Dados (BD), mas se quiser outras formas de gravação, como em arquivo de texto ou cvs, é só fazer as devidas classes de gravação e buscar os dados na classe ‘InfraAdapter’. Todo o código anterior permanece intacto. | Vermelho |
| TEMPLATE METHOD | **Classes:** UsuarioBD, ClienteBD, FuncionarioBD, ProdutoBD, VendaBD, Venda\_ProdutosBD, TemplateBD e Conexao, | Os códigos de todas as classes que necessitam operar com o Banco de Dados são bastantes semelhantes, diferenciando apenas de duas variáveis (o comando e a tabela), então construímos as classes que fazem de fato a conexão e as operações, e as demais apenas repassam essas duas variáveis e fazem as devidas utilização do Banco de Dados. Reduziu bastante a codificação dos métodos em cada classe. | Amarelo |
| FACTORY METHOD | **Classes:** ControlImplem, ControlCliente, ControlDistribuidor, ControlFuncionario e ControlGerente, ControlPessoaJuridica  **Interface:** FabricaControlInt | Na verdade fizemos duas fábricas, uma para as classes do Pacote Business.Control e Business.Model.  Facilita a codificação nas views, que não precisou da importação de todas as classes ‘Control’ e ‘Model’, não permitindo a visualização de todas as classes do Pacote Business. | Rosa |
| COMANDO | **Classes:** PagamentoCredito, PagamentoDebito, CartaoHipercard e CartaoVisa  **Interface:** TipoCartao | Para dar mais flexibilidade aos tipos de cartões que efetuam o pagamento de uma venda com cartões, seja de débito ou crédito, implementamos uma interface e os tipos de cartões aceitos, no caso apenas dois, e as classes PagamentoCredito e PagamentoDebito repassam apenas o tipo de cartão e as classes especifica de cada cartão faz a operação devida. Pode então aumentar os tipos de cartões, só criando novas classes que implementa a interface ‘TipoCartao’, sem precisar alterar o código restante. | Verde |
| MEMENTO | **Classes:** Venda\_ProdutosForm, Vendas, VendasMemento e VendasCareTaker | Em uma única venda, pode-se ter a inclusão de vários produtos, que são relacionados e guardados em um Memento. Caso seja necessário a retirada de algum produto da lista (no caso, o último produto), basta apenas retornar a situação anterior da lista e a relação está refeita. | Amarelo claro |

|  |  |  |  |
| --- | --- | --- | --- |
| STRATEGY | **Classes:** ValidarNome e ValidarSenha | Criamos duas classes distintas que realizam as operações de validar login e senha, cada, para dar mais flexibilidade quando necessário usar esse mesmo métodos em outros momentos. | Roxa (junto com o Facete) |
| MEDIATOR | **Classes:** MensagemForm, ControlEnviarMensagem, MensagemMediator, Mensagem, MensageE\_Mail, MensgemSMS e MensagemWhsatsApp  **Interface:** Mediator | Quando necessário enviar uma mensagem para os clientes, implementamos, para maior flexibilidade na criação dos meios de envio, o padrão mediator, que recebe a mensagem propriamente dita e repassa para os meios de envios desejados, que fazem de fato a execução do envio da mensagem. | Amarelo escuro |
| CHAIN OF RESPONSABILITY | **Classes:** ControlPagamento, FormasPagamento, PagamentoDinheiro, PagamentoCheque, PagamentoDebito, PagamentoCredito e PagamentoBoleto  **Enum:** PagamentoChain | Quando de um pagamento de uma venda, o cliente tem várias opções de pagamento, repassando a opção através da classe ControlPagamento, que percorre todas as formas de pagamento até achar a que trata a forma de pagamento especificada pelo cliente. | Azul escuro |
| SINGLETON | **Classes:** Vendas e VendasSingleton | Utilizamos este método em razão de que todas as vendas realizadas devem ter um número sequencial para utilização na Nota Fiscal, e assim, implementamos uma classe que gera apenas uma instância sua, porém incrementa sua numeração todas as vezes que for acionada. | Azul claro |

