UNIVERSIDADE ESTADUAL DE MARINGA

**S.A.E.I.   
Sistema de Administração para Empresas Imobiliárias**

|  |  |
| --- | --- |
| Integrantes | R.A. |
| **Airton Orlandini Junior** | **78582** |
| **Héverton Furlan Rissato** | **78680** |
| **Luís Gustavo Beligante** | **80385** |
| **Victor Hugo Pavoni** | **78499** |

Sumário

[Recursos: Tecnicos e Humanos 3](#_Toc401608874)

[Pessoas: 3](#_Toc401608875)

[Ferramentas 3](#_Toc401608876)

[Teste de Unidade 3](#_Toc401608877)

[Teste de Integração 4](#_Toc401608878)

[Tecnicas de Teste 4](#_Toc401608879)

[Caixa Preta 4](#_Toc401608880)

[Classe de Equivalencia <Serviço de Calculo de Juros > 4](#_Toc401608881)

[Classe de Equivalencia <Simulacao de Produto > 5](#_Toc401608882)

[Caixa Branca 8](#_Toc401608883)

[Caminho básico<Serviço de Calculo de Juros> 8](#_Toc401608884)

[Caminho básico<Simulacao de Produto > 11](#_Toc401608885)

[Frameworks 16](#_Toc401608886)

[Apache Struts 16](#_Toc401608887)

[Apache Tiles 16](#_Toc401608888)

[Apache Torque 16](#_Toc401608889)

[Apache Maven 17](#_Toc401608890)

[jQuery 17](#_Toc401608891)

[Bootstrap 17](#_Toc401608892)

[Padrões de Projeto 18](#_Toc401608893)

[Singleton 18](#_Toc401608894)

[Memento 18](#_Toc401608895)

[Diagrama MVC 19](#_Toc401608896)

# Recursos: Tecnicos e Humanos

## Pessoas:

Para o desenvolvimento deste projeto, serão necessárias pessoas com conhecimentos em banco de dados mysql, em liguaguem JAVA com enfase em teste de software e desenvolvimento web e em frameworks de teste, será também necessário conhecimento em apache maven.

## Ferramentas

Durante o desenvolvimento deste projeto, foram utilizadas as seguintes ferramentas:

* Eclipse Luna: Eclipse é uma IDE de desenvolvimento para linguagem java, ela conta com uma serie de recursos que agilizam e facilitam o desenvolvimento do software baseado em JAVA.
* Sublime Text: É um editor de texto de varios tipos de linguagem, também oferece uma serie de recursos interessantes que agilizam o desenvolvimento de software. Em nosso projeto foi utilizado para desenvolver paginas HTML, estilos CSS e scripts em JavaScript.
* Mysql Workbench: Esta IDE é utilizada para a manipulação do banco de dados, sua interface simples facilita a interação com a base de dados, também foi utilizada para o desenvolvimento de scripts em SQL

## Teste de Unidade

Nossa aplicação conta com uma serie de funcionalidades, as funcionalidades escolhidas para integrarem a os teste de unidade foram a funcionalidade de Venda e a funcionalidade de Gerenciamento de Intenção de Compra de produto (por parte do cliente) e Venda ( por parte da empresa).

Mais especificamente testes foram realizados para assegurarem a integridade dos dados dessas operações, neste caso, algumas classes de serviço foram geradas. Estas classes implementam de fato as regras de negocio interresantes para a empresa e foram estas as escolhidas para serem testadas de forma unitaria.

# Teste de Integração

A escrita de testes de integração se fes desnecessária para as funcionalidades escolhidas, mesmo assim, não significa que a integração entre as classes não seja testada, durante a faze de validação da aplicação esta integração pode ser vista e validada.

# Tecnicas de Teste

## Caixa Preta

Classe de Equivalencia <Serviço de Calculo de Juros >:

Este serviço faz parte da funcionalidade de simulação, que faz parte do fluxo de venda de produto do sistema. Sua função é calcular a partir de uma serie de configurações o percentual de juros para um dado valor e uma quantidade de parcelas. A configuração de cobrança de juros funciona da seguinte forma:

1. Ela possui um valor total: esse valor, indica que se uma compra for igual ou superior a mesma, esta configuração pode ser aplicada na operação, porém a condição da quantidade de parcelas deve ser considerada.
2. Ela possui uma quantidade de parcelas: esse atributo, indica que se a compra for parcelada em uma quantidade superior ou igual a este atributo, esta configuração pode ser aplicada.
3. Ela possui um percentual de juros: Este percentual é o valor que esta configuração armazena, este valor é retornado quando as condições 1 e 2 são satisfeitas.

O Objetivo deste serviço, é retornar a quantidade de juros mais adequada ao valor do produto e a quantidade de parcelas em que a compra será feita. As restirções quanto a inicialização são: As configurações não podem ser nulas, não podem ter atributos inválidos (como valor total menor que zero), não podem ser ambiguas (como: duas configurações com o mesmo valor total e o mesmo numero de parcelas).

A seguir, os casos de teste para cada uma das restrições apontadas anteriormente

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Instanciação do Serviço** | | | | |
| **Caso de teste** | **CONFIGURAÇÕES** | | | **Saida** |
| Valor Total | Parcelas | Percentual Juros |
| 1 | null | null | Nulll | Excesseção de Instanciacao |
| 2 | R$ -1,00 | 2 | 120.00% | Excesseção de Instanciacao |
| 3 | R$ 3,00 | -1 | 0.00% | Excesseção de Instanciacao |
| 4 | R$ 3,00 | -1 | 0.00% | Excesseção de Instanciacao |
| 5 | R$ 3,00 | 4 | -100.00% | Excesseção de Instanciacao |
| 6 | R$ 3,00 | 0 | 200.00% | Excesseção de Instanciacao |
| 7 | R$ 4,00 | 1 | 100.00% | Excesseção de Instanciacao |
| R$ 4,00 | 1 | 100.00% |
| 8 | R$ 3,00 | 1 | 200.00% | Objeto Instanciado |
| 9 | R$ 3,00 | 1 | 200.00% | Objeto Instanciado |
| R$ 3,00 | 5 | 210.00% |
| 10 | R$ 3,00 | 1 | 200.00% | Objeto Instanciado |
| R$ 10,00 | 2 | 300.00% |

Após testada a instanciação, precisamos definir os testes a respeito dos comportamentos que este serviço deve ter para uma serie de configurações e os varios formatos de entrada dos atributos

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metodo de Calculo de Juros** | | | | | | |
| **Caso de teste** | **CONFIGURAÇÕES** | | | **Parametros** | | **Saida (Percentual de Juros)** |
| Valor Total | Parcelas | Percentual Juros | Valor Total Parametro | Qtde Parcelas |
| 1 | R$ 3,00 | 1 | 2.0% | R$ 0, 00 | 0 | 0.0% |
| 2 | R$ 3,00 | 1 | 2.0% | R$ 0, 00 | -1 | 0.0% |
| 3 | R$ 3,00 | 1 | 2.0% | R$ -1, 00 | 0 | 0.0% |
| 4 | R$ 3,00 | 1 | 2.0% | R$ -1, 00 | -1 | 0.0% |
| 5 | R$ 3,00 | 1 | 2.0% | R$ 5,00 | 1 | 2.0% |
| 6 | R$ 3,00 | 1 | 2.0% | R$ 1,00 | 1 | 0.0% |
| R$ 3,00 | 5 | 2.1% |
| 7 | R$ 3,00 | 1 | 2.0% | R$ 3,00 | 1 | 2.0% |
| R$ 3,00 | 5 | 2.1% |
| 8 | R$ 3,00 | 1 | 2.0% | R$ 10,00 | 1 | 0.0% |
| R$ 3,00 | 5 | 2.1% |
| 9 | R$ 3,00 | 1 | 2.0% | R$ 10,00 | 6 | 2.1% |
| R$ 3,00 | 5 | 2.1% |

Classe de Equivalencia <Simulacao de Produto >

Este serviço está inserido no fluxo de venda de produto do Sistema. Sua funcionalidade é calcular, a partir de configurações passadas, o valor das parcelas e o valor total com o incluso dos juros.

1. Valor Original: É o valor total do produto, o qual foi previamente cadastrado no Sistema. Esse valor é indispensável no cálculo de simulação, e deverá ser válido e maior que o valor de entrada.
2. Juros: Valor da porcentagem de juros que será usado para calcular os juros nos valores das parcelas na simulação. Valor indispensável, o fluxo de serviço de classe de juros é responsável por calcular o valor mais adequado.
3. Valor Entrada: Valor que o cliente se disponibiliza a dar de entrada para a aquisição do produto. Esse valor é indispensável e deverá ser valido e menor que o valor original do produto.
4. Parcelas: Número de parcelas para pagamento que o cliente gostaria de realizar na aquisição do produto. Valor necessário para o cálculo.

O objetivo desse serviço é a partir das entradas, calcula o valor das parcelas e o valor final do produto já com os juros calculados. Com o intuito de simular a venda, concebendo ao cliente a oportunidade de estudar diversos modos de pagamentos (diferentes valores de entrada, e número de parcelas) para a sua melhor escolha.

A seguir, os casos de teste para cada uma das restrições apontadas anteriormente.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Instanciação do Serviço** | | | | | |
| **Caso de teste** | **PARAMETROS** | | | | **Saida** |
| Valor Original (R$) | Juros | Valor Entrada (R$) | Parcelas |
| 1 | Null | 2.1 | 2.3 | 2 | Excesseção de Instanciacao |
| 2 | R$ 0 | 2.1 | 2.3 | 2 | Excesseção de Instanciacao |
| 3 | R$ -2.0 | 2.1 | 2.3 | 2 | Excesseção de Instanciacao |
| 4 | R$ 2.1 | 2.1 | Null | 2 | Excesseção de Instanciacao |
| 5 | R$ 2.1 | 1.2 | 1.3 | 0 | Excesseção de Instanciacao |
| 6 | R$ 1.1 | 1.0 | 1.2 | 2 | Excesseção de Instanciacao |
| 7 | R$ 2.1 | 0.3 | 1.2 | 2 | Objeto Instanciado |
| 8 | 1000000 | 1.2 | 100999 | 30 | Objeto Instanciado |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metodo de Calculo de Juros** | | | | | |
| **PARAMETROS** | | | | **Saida (Valor Parcela)** | **Saida (Valor Total)** |
| Valor Original (R$) | Juros | Valor Entrada (R$) | Parcelas |
| 1000000 | 1.2 | 100999 | 30 | 30326.30 | 1010788.00 |
| 1000000 | 1.2 | 100999 | 44 | 20677.02 | 1010787.88 |
| 500777 | 1.2 | 100999 | 30 | 13485.85 | 505574.50 |
| 500777 | 1.2 | 100999 | 44 | 9194.90 | 505574.60 |
| 1000000 | 1.2 | 77013 | 30 | 31135.43 | 1011075.90 |
| 1000000 | 1.2 | 77013 | 44 | 21228.70 | 1011075.80 |
| 500777 | 1.2 | 77013 | 30 | 14294.99 | 505862.70 |
| 500777 | 1.2 | 77013 | 44 | 9746.58 | 505862.52 |
| 1000000 | 0.4 | 100999 | 30 | 30086.57 | 1003596.10 |
| 1000000 | 0.4 | 100999 | 44 | 20513.57 | 1003596.08 |
| 500777 | 0.4 | 100999 | 30 | 13379.24 | 502376.20 |
| 500777 | 0.4 | 100999 | 44 | 9122.21 | 502376.24 |
| 1000000 | 0.4 | 77013 | 30 | 30889.30 | 1003692.00 |
| 1000000 | 0.4 | 77013 | 44 | 21060.89 | 1003692.16 |
| 500777 | 0.4 | 77013 | 30 | 14181.97 | 502472.10 |
| 500777 | 0.4 | 77013 | 44 | 9669.53 | 502472.32 |

## Caixa Branca

### Caminho básico<Serviço de Calculo de Juros>

Abaixo temos o grafo de fluxo do metodo de calculo de juros e logo em seguida o trecho de codigo que representa sua implementação:

### 

### a.png

|  |  |  |
| --- | --- | --- |
| **Linhas** | | **Nó no grafo** |
| DE | ATÉ |
| 1 | 4 | 1 |
| 5 | 5 | 8 |
| 7 | 7 | 2 |
| 8 | 10 | 3 |
| 11 | 11 | 4 |
| 14 | 14 | 5 |
| 15 | 15 | 6 |
| 19 | 19 | 7 |

01 - BigDecimal melhorConfiguracao = BigDecimal.***ZERO***;

02 - BigDecimal configuracaoAtual = **null**;

03 -

04 - **if**(valorTotal == **null** || quantidadeParcelas < 0){

05 - **return** melhorConfiguracao;

06 - }

07 - **for**(JurosParcelamento configuracao : configuracoes){

08 - configuracaoAtual = **null**;

09 - **if**(configuracao.getValorTotal().compareTo(valorTotal) <= 0

10 - && configuracao.getQuantidadeParcelas() <= quantidadeParcelas){

11 - configuracaoAtual = configuracao.getPorcentagemJuros();

12 - }

13 -

14 - **if**(configuracaoAtual != **null**){

15 - melhorConfiguracao = configuracaoAtual;

16 - }

17 - }

18 -

19 - **return** melhorConfiguracao;

Complexidade ciclomática:

* Ramos: 9
* Nós: 8
* Complexidade Ciclomatica é: 9 - 8 + 2 = 3

Caminhos Escolhidos:

1. 1, 8
2. 1,2,3, 4, 5, 6, 2, 7
3. 1, 2, 3, 5, 2, 7

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Analize Casos de Teste** | | | | | | |
| **Caminho** | Caminho no Grafo | **CONFIGURAÇÕES** | | | **Parametros** | | **Resultado Esperado** |
| Valor Total | Parcelas | Percentual Juros | Valor Total Parametro | Qtde Parcelas |
| 1 | 1,8 | R$ 3,00 | 1 | 2.0% | NULL | 0 | 0.0% |
| 2 | 1, 2, 3, 4, 5, 6, 2, 7 | R$ 3,00 | 1 | 2.0% | R$ 3, 00 | 1 | 2.0% |
| 3 | 1, 2, 3, 5, 2, 7 | R$ 3,00 | 4 | 2.0% | R$ 3, 00 | 1 | 0.0% |

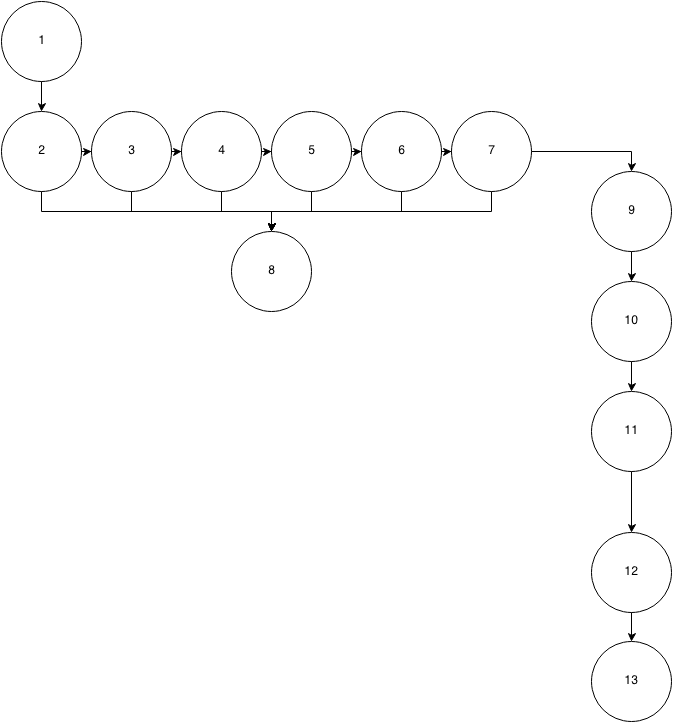
|  |  |  |
| --- | --- | --- |
| **Depuracao Caminho 1** | | |
| **NO** | **Variaveis** | |
| melhorConfiguracao | configuracaoAtual |
| 1 | 0 | NULL |
| 8 | 0 | NULL |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Depuracao Caminho 2** | | | | | |
| **NO** | **Variaveis** | | | | |
| melhorConfiguracao | configuracaoAtual | **Configuracao** | | |
| Valor Total | Parcelas | Percentual Juros |
| 1 | 0% | NULL | NULL | NULL | NULL |
| 2 | 0% | NULL | NULL | NULL | NULL |
| 3 | 0% | NULL | R$ 3,00 | 1 | 2.0% |
| 4 | 0% | NULL | R$ 3,00 | 1 | 2.0% |
| 5 | 0% | 2.0% | R$ 3,00 | 1 | 2.0% |
| 6 | 2.0% | 2.0% | R$ 3,00 | 1 | 2.0% |
| 2 | 2.0% | 2.0% | R$ 3,00 | 1 | 2.0% |
| 7 | 2.0% | 2.0% | NULL | NULL | NULL |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Depuracao Caminho 3** | | | | | |
| **NO** | **Variaveis** | | | | |
| melhorConfiguracao | configuracaoAtual | **Configuracao** | | |
| Valor Total | Parcelas | Percentual Juros |
| 1 | 0% | NULL | NULL | NULL | NULL |
| 2 | 0% | NULL | NULL | NULL | NULL |
| 3 | 0% | NULL | R$ 3,00 | 4 | 2.0% |
| 5 | 0% | NULL | R$ 3,00 | 4 | 2.0% |
| 2 | 0% | NULL | R$ 3,00 | 4 | 2.0% |
| 7 | 0% | NULL | NULL | NULL | NULL |

### Caminho básico<Simulacao de Produto >

|  |  |  |
| --- | --- | --- |
| **Linhas** | | **Nó no grafo** |
| DE | ATÉ |
| 1 | 1 | 1 |
| 3 | 33 | 2 |
| 37 | 37 | 3 |
| 41 | 41 | 4 |
| 45 | 45 | 5 |
| 49 | 49 | 6 |
| 53 | 53 | 7 |
| 34,38,42,46,50,54 | | 8 |
| 3 | 6 | 9 |
| 8 | 13 | 10 |
| 15 | 17 | 11 |
| 19 | 25 | 12 |
| 27 | 27 | 13 |



*17- checkAtributes*(valorOriginal, juros, entrada, quantidadeParcelas);

18-

19- BigDecimal valorJuros = valorOriginal

20- .subtract(entrada)

21- .divide(***CEM***, BigDecimal.***ROUND\_HALF\_UP***)

22- .multiply (juros.setScale(2, BigDecimal.***ROUND\_HALF\_UP***));

23-

24- BigDecimal valorParcela = valorOriginal

25- .subtract(entrada)

26- .add(valorJuros)

27- .divide(**new** BigDecimal(quantidadeParcelas),

28- BigDecimal.***ROUND\_HALF\_UP***)

29- .setScale(2, BigDecimal.***ROUND\_HALF\_UP***);

30-

31- BigDecimal valorTotal = valorParcela

32- .multiply(**new** BigDecimal(quantidadeParcelas))

33- .add(entrada);

34-

35- Simulacao simulacao = **new** Simulacao();

36- simulacao.setJuros(juros.setScale(2, BigDecimal.***ROUND\_HALF\_UP***));

37- simulacao.setEntrada(entrada.setScale(2, BigDecimal.***ROUND\_HALF\_UP***));

38- simulacao.setQuantidadeParcelas(quantidadeParcelas);

39- simulacao.setValorFinal(valorTotal.setScale(2, BigDecimal.***ROUND\_HALF\_UP***));

40- simulacao.setValorOriginal(valorOriginal.setScale(2, BigDecimal.***ROUND\_HALF\_UP***));

41- simulacao.setValorParcelas(valorParcela.setScale(2, BigDecimal.***ROUND\_HALF\_UP***));

42-

43- **return** simulacao;

44- }

45-

46- **private** **static** **void** checkAtributes(BigDecimal valorOriginal,

47- BigDecimal juros, BigDecimal entrada, **int** quantidadeParcelas) {

48-

49- **if**(valorOriginal == **null**){

50- **throw** **new** IllegalArgumentException("O Valor Original não pode ser nulo");

51- }

52-

53- **if**(valorOriginal == **new** BigDecimal(0)){

54- **throw** **new** IllegalArgumentException("O Valor Original não pode ser 0");

55- }

56-

57- **if**(valorOriginal.compareTo(**new** BigDecimal(0)) < 0){

58- **throw** **new** IllegalArgumentException("O Valor Original não pode ser menor que 0");

59- }

60-

61- **if**(entrada == **null**){

62- **throw** **new** IllegalArgumentException("O valor de entrada não pode ser nulo");

63- }

64-

65- **if**(quantidadeParcelas <= 0){

66- **throw** **new** IllegalArgumentException("O numero de parcelas não pode ser menor que 0");

67- }

68-

69- **if**(entrada.compareTo(valorOriginal) > 0 ){

70- **throw** **new** IllegalArgumentException("O valor de entrada não pode ser maior que o valor original(total) do produto");

71- }

72-

73- }

Complexidade ciclomática:

* Ramos: 17
* Nós: 13
* Complexidade Ciclomatica é: 17 - 13 + 2 = 6

Caminhos Escolhidos:

1. 1, 2,8
2. 1,2,3,4,5,6,7,8
3. 1,2,3,4,5,6,7,9,10,11,12,13

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Analize Casos de Teste** | | | | | | | |
| **Caminho** | Caminho no Grafo | **Parametros** | | | | **Resultado esperado** | |
| Valor Original (R$) | Juros | Valor Entrada (R$) | Parcelas | Valor da parcela (R$) | Valor total (R$) |
| 1 | 1, 2, 8 | Null | 1.2 | 39000,00 | 30 | Exception | Exception |
| 2 | 1, 2, 3, 4, 5, 6, 7, 8 | 500000,00 | 1.2 | 600000,00 | 30 | Exception | Exception |
| 3 | 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13 | 1000000,00 | 1.2 | 100999,00 | 30 | 30326.30 | 1010788.00 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Depuracao Caminho 1** | | | | |
| **NO** | **Variaveis** | | | |
| Valor Original | Juros | Entrada | Quantidade de Parcelas |
| 1 | Null | 1.2 | 39000 | 30 |
| 2 | Null | 1.2 | 39000 | 30 |
| 8 | Null | 1.2 | 39000 | 30 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Depuracao Caminho 2** | | | | |
| **NO** | **Variaveis** | | | |
| Valor Original | Juros | Entrada | Quantidade de Parcelas |
| 1 | 500000 | 1.2 | 600000 | 30 |
| 2 | 500000 | 1.2 | 600000 | 30 |
| 3 | 500000 | 1.2 | 600000 | 30 |
| 4 | 500000 | 1.2 | 600000 | 30 |
| 5 | 500000 | 1.2 | 600000 | 30 |
| 6 | 500000 | 1.2 | 600000 | 30 |
| 7 | 500000 | 1.2 | 600000 | 30 |
| 8 | 500000 | 1.2 | 600000 | 30 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Depuracao Caminho 3** | | | | | | | |
| **NO** | **Variaveis** | | | | | | |
| Valor Original | Juros | Entrada | Quantidade de Parcelas | Valor Juros | Valor Parcela | Valve Total |
| 1 | 1000000 | 1.2 | 100999 | 30 | Null | Null | Null |
| 2 | 1000000 | 1.2 | 100999 | 30 | Null | Null | Null |
| 3 | 1000000 | 1.2 | 100999 | 30 | Null | Null | Null |
| 4 | 1000000 | 1.2 | 100999 | 30 | Null | Null | Null |
| 5 | 1000000 | 1.2 | 100999 | 30 | Null | Null | Null |
| 6 | 1000000 | 1.2 | 100999 | 30 | Null | Null | Null |
| 7 | 1000000 | 1.2 | 100999 | 30 | Null | Null | Null |
| 9 | 1000000 | 1.2 | 100999 | 30 | 10788.01 | Null | Null |
| 10 | 1000000 | 1.2 | 100999 | 30 | 10788.01 | 30326.30 | Null |
| 11 | 1000000 | 1.2 | 100999 | 30 | 10788.01 | 30326.30 | 1010788.00 |
| 12 | 1000000 | 1.2 | 100999 | 30 | 10788.01 | 30326.30 | 1010788.00 |
| 13 | 1000000 | 1.2 | 100999 | 30 | 10788.01 | 30326.30 | 1010788.00 |

# Frameworks

## Apache Struts

Struts é um framework de desenvolvimento da camada controladora (na realidade ele intermedia a visão com o controle). Em um aplicativo de web padrão Java EE, o cliente normalmente envia informações ao servidor através de um formulário da web. Em seguida, a informação ou é entregue a um Java Servlet que a processa, interage com um banco de dados e produz uma resposta formatada em HTML, ou é entregue a um JavaServer Page (JSP) que mixa código HTML e Java para obter o mesmo resultado. Ambas as abordagens são consideradas inadequadas para grandes projetos porque eles misturam a lógica do aplicativo com apresentação e dificultam a manutenção.

O objetivo do Struts é separar o model (modelo - lógica de aplicativo que interage com um banco de dados) do view (visualização - páginas HTML apresentadas para o cliente) e do controller (controlador - instância que transmite informações entre a exibição e o modelo). Struts fornece o controlador/controller (um servlet conhecido como ActionServlet) e facilita a escrita de moldes padronizados para a camada de visualização/view (normalmente em JSP, mas XML/XSLT e Velocity também são suportados). O programador de aplicativo da web é responsável por escrever o código do modelo/model, e por criar um arquivo de configuração centralizador (struts-config.XML) que une modelo, visualização e controlador.

Solicitações do cliente são enviadas para o controller em forma de Actions (ações) definidas no arquivo de configuração. O controller, ao receber uma solicitação, chama a classe Action correspondente a Programação Orientada a Objeto, e esta interage com o código do model específico. O código do model retorna um "ActionForward", uma cadeia de caracteres que informa ao controller qual página de resultado deve ser enviada para o cliente. As informações são passadas entre model e view em forma de JavaBeans especial. Uma poderosa biblioteca personalizada de palavras-chave (tags - metadata) permite ler e gravar o conteúdo (beans) da camada de apresentação sem a necessidade de qualquer código Java embutido.

## Apache Tiles

Apache Tiles é um framework que auxilia na parte da visão (em aplicações MVC). Sua função principal é a de fornecer um mecanismo para criação de templates de pagina, onde existe uma pagina esqueleto, e um arquivo xml que preenche esse esqueleto com paginas jsp de acordo com alguns parametros.

## Apache Torque

Apache Torque é um framework de mapeamento Objeto Relacional para a linguagem JAVA, este framework permite realizar operações num banco de dados utilizando classes JAVA gerada a partir de uma configuração inicial. Nesta configuração se incluem um XML contendo as tabelas e seus respectivos atributos de um banco de dados e outro arquivo contendo as urls e senhas de acesso ao banco de dados.

## Apache Maven

Apache Maven é uma ferramenta de automação de compilação utilizada primariamente em projetos Java. Ela é similar à ferramenta Ant, mas é baseada em conceitos e trabalhos diferentes em um modo diferente. Também é utilizada para construir e gerenciar projetos escritos em C#, Ruby, Scala e outras linguagens.

O Maven utiliza um arquivo XML (POM) para descrever o projeto de software sendo construído, suas dependências sobre módulos e componentes externos, a ordem de compilação, diretórios e plug-ins necessários. Ele vem com objetivos pré-definidos para realizar certas tarefas bem definidas como compilação de código e seu empacotamento.

O Maven baixa bibliotecas Java e seus plug-ins dinamicamente de um ou mais repositórios, como o Maven 2 Central Repository, e armazena-os em uma área de cache local.3 Este cache local de artefatos baixados pode também ser atualizado com artefatos criados por projetos locais. Repositórios públicos podem também ser atualizados.

## jQuery

jQuery é uma biblioteca JavaScript cross-browser desenvolvida para simplificar os scripts client side que interagem com o HTML. A sintaxe do jQuery foi desenvolvida para tornar mais simples a navegação do documento HTML, a seleção de elementos DOM, criar animações, manipular eventos e desenvolver aplicações AJAX. A biblioteca também oferece a possibilidade de criação de plugins sobre ela. Fazendo uso de tais facilidades, os desenvolvedores podem criar camadas de abstração para interações de mais baixo nível, simplificando o desenvolvimento de aplicações web dinâmicas de grande complexidade.

## Bootstrap

Bootstrap é um framework CSS, desta forma, este se encaixa nos frameworks que fazem parte da camada de visão em aplicações MVC. Bootstrap fornece uma serie de classes de seletores CSS que facilitam a criação e estilização de paginas web. Garantindo ao desenvolvedor que para qualquer browser que um usario acesse a visão não será afetada.

# Padrões de Projeto

## Singleton

Será utilizado neste projeto, o Padrão de Projeto Singleton, que é classificado como Padrão de Projeto de Criação. Ele tem como função garantir a existência de apenas uma instância de uma classe, mantendo um ponto global de acesso ao seu objeto.O padrão Singleton funciona tendo um método especial usado para instanciar o objeto desejado, ao ser chamado esse método confere se o objeto já foi instanciado. No caso de tê-lo sido o método simplesmente retorna uma referencia a ele, senão, instancia o objeto e retorna uma referência para ele.Este padrão foi escolhido porque se aplicaria, por exemplo, no impedimento de dois usuários com o mesmo login se conectarem em locais diferentes, o que, em nossa aplicação, será algo proibido.

## Memento

Para o desenvolvimento desse projeto, optamos por incluir, (além do Singleton) o Padrão de Projeto Memento, classificado como um Padrão de Projeto Comportamental. Sua principal função é permitir armazenar o estado interno de um objeto em um determinado momento, para que seja possível retorná-lo a este estado, caso necessário.

Este padrão foi escolhido por que o sistema deverá permitir uma possível volta para o estado anterior, ou seja, deverá desfazer o que foi feito e voltar para o estado inicial que estava. Um exemplo seria, quando uma avaliação do produto é iniciada, o funcionário pode atribuir o status “avaliado” ao produto que já foi literalmente avaliado, esperando assim a resposta do fornecedor. Caso o fornecedor solicite uma nova avaliação, o funcionário poderá alterar este statuspara “agendado”, retornando assim para o estado anterior. Em um segundo exemplo é visto que, o funcionário poderá marcar o status “concluído” e, após a avaliação ser realizada e o funcionário solicitar uma nova avaliação, o status ‘volta’ a ser “agendado” ou até mesmo “aguardando contato”.

# Interface Grafica

## Intenção de Compra De Produto pelo Cliente

## Venda de Produto