

ANÁLISE E DESENVOLVIMENTO  
DE SISTEMAS

AVALIAÇÃO 3

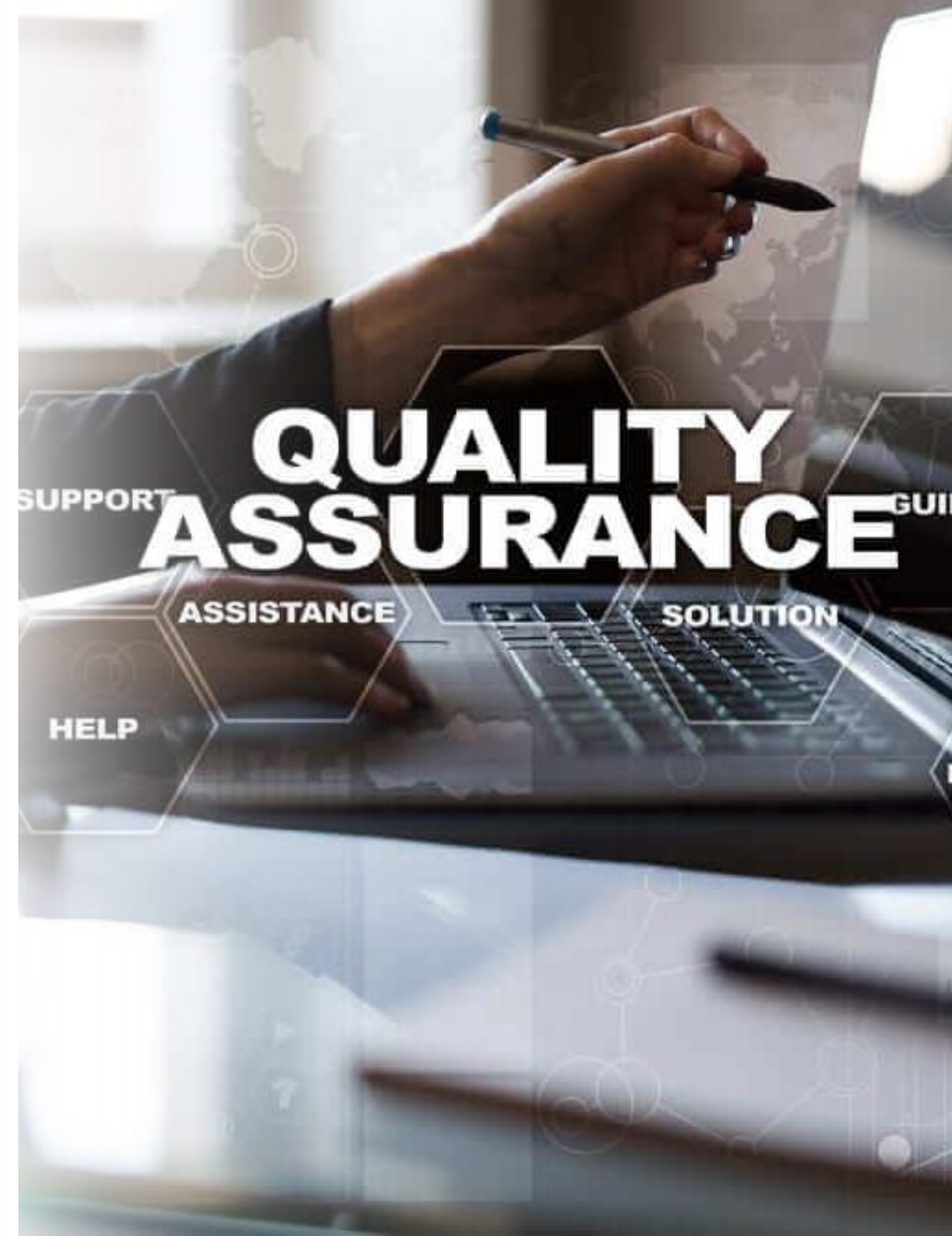
# GESTÃO E QUALIDADE DE SOFTWARE

---



# Introdução

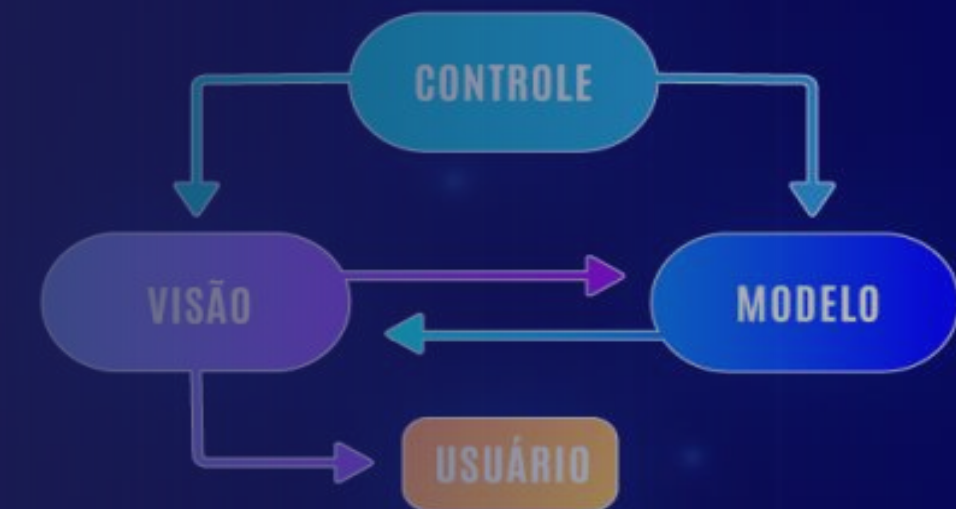
No dinâmico cenário do desenvolvimento de software, onde a inovação e a eficiência são fundamentais, a constante busca pelo aprimoramento técnico e metodológico se torna imprescindível. Este projeto surge como resposta a essa demanda, propondo uma oportunidade valiosa para trabalhar em equipe, treinar e aprimorar nossos conhecimentos em Gestão e Qualidade de Software.





# Motivação

A motivação central deste projeto é impulsionar nossa equipe a alcançar novos patamares de excelência, consolidando e ampliando nossas habilidades em Gestão e Qualidade de Software e programação.



## CRUD

A implementação de um sistema *CRUD*, seguindo os princípios do padrão *MVC*, e integrados a um banco de dados, não apenas desafia nossas capacidades técnicas, mas também proporciona uma plataforma robusta para aplicação prática dos conhecimentos adquiridos durante esse semestre.



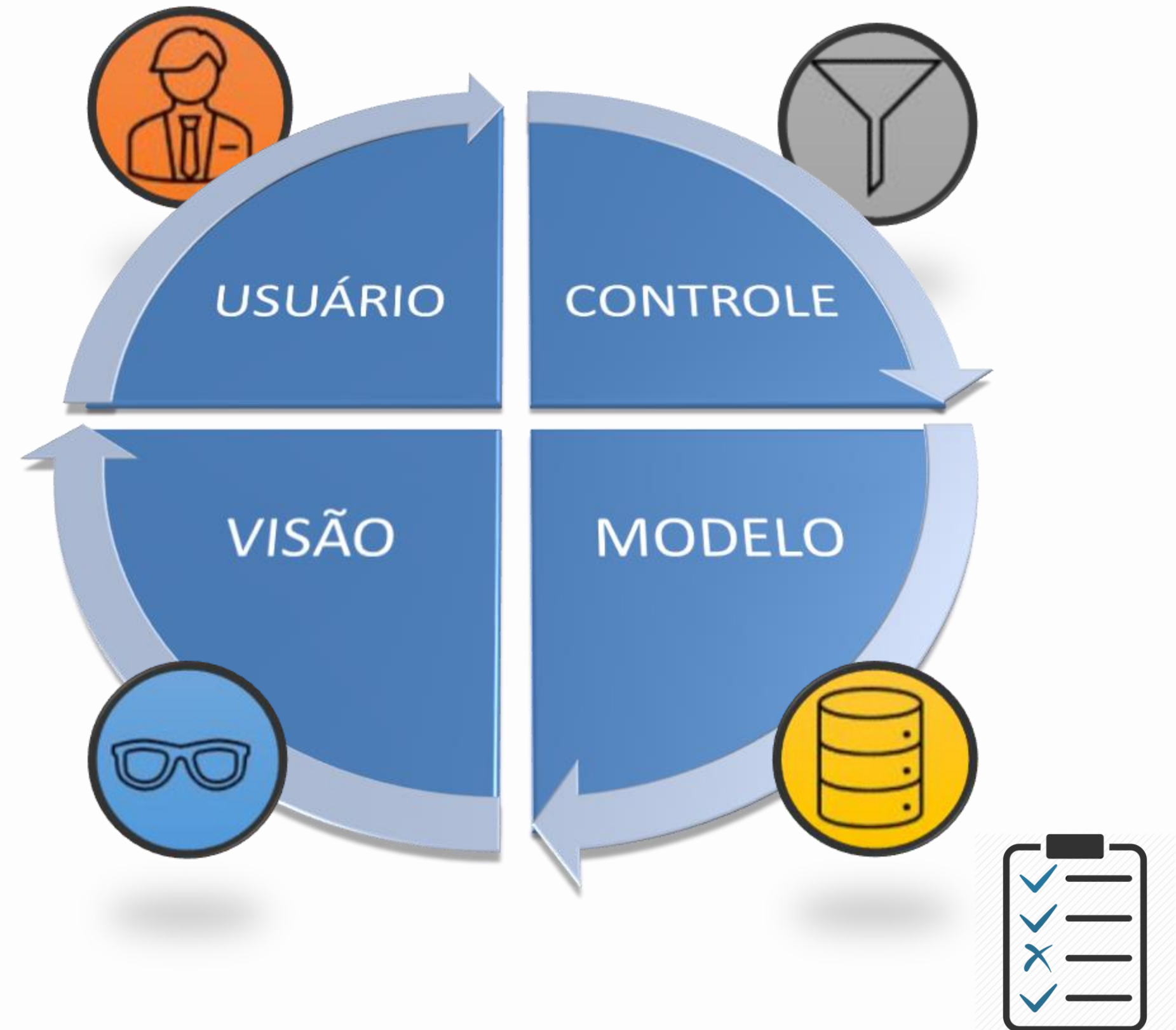
Além disso, a incorporação de ferramentas de testes como *GHERKIN*, *JUNIT*, *CUCUMBER*, *METRICS* e *COVERAGE* no desenvolvimento do projeto visa enriquecer nossa abordagem, garantindo não apenas funcionalidade, mas também elevados padrões de qualidade, mensuráveis por meio de métricas e cobertura de testes.



# DESENVOLVIMENTO

---

Ao longo do processo de desenvolvimento, adotamos uma abordagem meticulosa, aplicando as melhores práticas de Gestão e Qualidade de Software. A arquitetura do sistema, baseada no padrão MVC, foi cuidadosamente planejada, permitindo a modularidade e escalabilidade necessárias para futuras expansões.





```
maven > src > main > java > dao > DBConnection.java > ...
1 package dao;
2 import java.sql.Connection;
3 import java.sql.DriverManager;
4 import java.sql.SQLException;
5 /**
6  *
7  * @author Sâmeck
8  */
9 public class DBConnection {
10
11     private static DBConnection instance = null;
12     private Connection conn = null;
13     private String url = "jdbc:mysql://localhost:3306/";
14     private String dbName = "marketplace";
15     private String driver = "com.mysql.jdbc.Driver";
16     private String driverManager = null;
17
18     private DBConnection() {
19         try {
20             conn = DriverManager.getConnection(url + dbName, "root", "root");
21         } catch (SQLException e) {
22             System.err.println("Erro ao conectar ao banco de dados:");
23             e.printStackTrace();
24         }
25     }
26
27     public static DBConnection getInstance() {
28         if (instance == null) {
29             instance = new DBConnection();
30         }
31         return instance;
32     }
33
34     public Connection getConnection() {
35
36         try {
37             if (conn == null || conn.isClosed()) {
38                 conn = DriverManager.getConnection(url + dbName, "root", "root");
39             }
40         } catch (SQLException e) {
41             // Tratar a exceção de forma apropriada
42             System.err.println("Erro ao obter conexão com o banco de dados:");
43             e.printStackTrace();
44         }
45         return conn;
46     }
47
48     public void closeConnection() {
49         try {
50             if (conn != null && !conn.isClosed()) {
51                 conn.close();
52             }
53         } catch (SQLException e) {
54             // Tratar a exceção de forma apropriada
55             System.err.println("Erro ao fechar a conexão com o banco de dados:");
56             e.printStackTrace();
57         }
58     }
59 }
60
```

AINDA SOBRE O DESENVOLVIMENTO...

A criação das classes e a integração com o banco de dados foram realizadas com o intuito de refletir não apenas a complexidade técnica do projeto, mas também a aplicação coerente dos conceitos aprendidos durante nosso treinamento.



A elaboração do roteiro de teste, utilizando as linguagens GHERKIN, JUNIT e CUCUMBER, fortaleceu a abordagem de desenvolvimento orientado a testes, garantindo uma maior confiabilidade e robustez ao sistema. A análise de métricas e cobertura proporcionou uma visão abrangente da qualidade do código, permitindo correções proativas e garantindo a estabilidade do sistema.



Visão abrangente da qualidade do código;



Permite correções proativas;



Garantia de estabilidade do sistema.





## Testes utilizando Cucumber



maven > src > test > java > JUnit > TesteAplicacao.java > TesteAplicacao > CadastrarNovoCliente()

```
4 import dao.ProdutoDAO;
5 import dao.VendedorDAO;
6 import org.junit.Test;
7 import static org.junit.Assert.*;
8
9 import model.Cliente;
10 import model.Produto;
11
12 import java.util.List;
13
14 import aplicacao.aplicacao;
15
16 public class TesteAplicacao {
17
18     @Test
19     public void CadastrarNovoCliente() {
20         ClienteDAO clienteDAO = new ClienteDAO();
21         int tamanhoAntes = clienteDAO.getClientes().size();
22
23         aplicacao.cadastrarNovoCliente();
24
25         int tamanhoDepois = clienteDAO.getClientes().size();
26         assertEquals(tamanhoAntes + 1, tamanhoDepois);
27     }
28 }
```

PROBLEMAS 7 SAÍDA RESULTADOS DE TESTE TERMINAL CONSOLE DE DEPUÇÃO PORTAS

Teste executado em 30/11/2023, 21:49:23

- ✓ CadastrarNovoCliente()
- ✓ CadastrarNovoClienteComInformacoesInvalidas()
- ✓ CadastrarNovoProduto()
- ✓ CadastrarNovoVendedor()
- ✓ ExibirClientes()
- ✓ ExibirProdutos()
- ✓ ExibirVendedores()
- ✓ GetClientes()
- ✓ GetProdutos()
- ✓ SalvarProdutoComPrecoZero()
- ✓ SaveProduto()
- ✓ SaveProdutoComNomeNulo()

%TESTE 8, SalvarProdutoComPrecoZero(JUnit.TesteAplicacao)

%TESTS 9, GetClientes(JUnit.TesteAplicacao)

%TESTE 9, GetClientes(JUnit.TesteAplicacao)

%TESTS 10, ExibirClientes(JUnit.TesteAplicacao)

%TESTE 10, ExibirClientes(JUnit.TesteAplicacao)

%TESTS 11, CadastrarNovoProduto(JUnit.TesteAplicacao)

%TESTE 11, CadastrarNovoProduto(JUnit.TesteAplicacao)

%TESTS 12, CadastrarNovoVendedor(JUnit.TesteAplicacao)

%TESTE 12, CadastrarNovoVendedor(JUnit.TesteAplicacao)

%TESTS 13, SaveProduto(JUnit.TesteAplicacao)

%TESTE 13, SaveProduto(JUnit.TesteAplicacao)

%RUNTIME1013

## Testes unitários usando Junit



maven > src > test > java > steps > RunCucumberTest.java > RunCucumberTest

```
1 package steps;
2
3 import io.cucumber.junit.Cucumber;
4 import io.cucumber.junit.CucumberOptions;
5 import org.junit.runner.RunWith;
6
7 @RunWith(Cucumber.class)
8 @CucumberOptions(features = "src/test/resources", glue = "steps")
9 public class RunCucumberTest {
10 }
11
```

PROBLEMAS 11 SAÍDA RESULTADOS DE TESTE TERMINAL CONSOLE DE DEPUÇÃO PORTAS

(Cadastro de novo cliente),false,1,false,-1,Tentar cadastrar um cliente com informações inválidas(Cadastro de novo cliente),, %TESTS 3,Cadastrar um novo produto com sucesso(Cadastro de Produto)

%TESTE 3,Cadastrar um novo produto com sucesso(Cadastro de Produto)

%TESTS 4,Falha ao cadastrar um produto com informações inválidas(Cadastro de Produto)

%TESTE 4,Falha ao cadastrar um produto com informações inválidas(Cadastro de Produto)

%TESTS 6,Consultar a lista de vendedores(Cadastro de Vendedor)

%TESTE 6,Consultar a lista de vendedores(Cadastro de Vendedor)

%TESTS 7,Cadastrar vendedor com CNPJ inválido(Cadastro de Vendedor)

%TESTE 7,Cadastrar vendedor com CNPJ inválido(Cadastro de Vendedor)

%TESTS 9,Cadastrar novo cliente com sucesso(Cadastro de novo cliente)

%TESTE 9,Cadastrar novo cliente com sucesso(Cadastro de novo cliente)

%TESTS 10,Tentar cadastrar um cliente com informações inválidas(Cadastro de novo cliente)

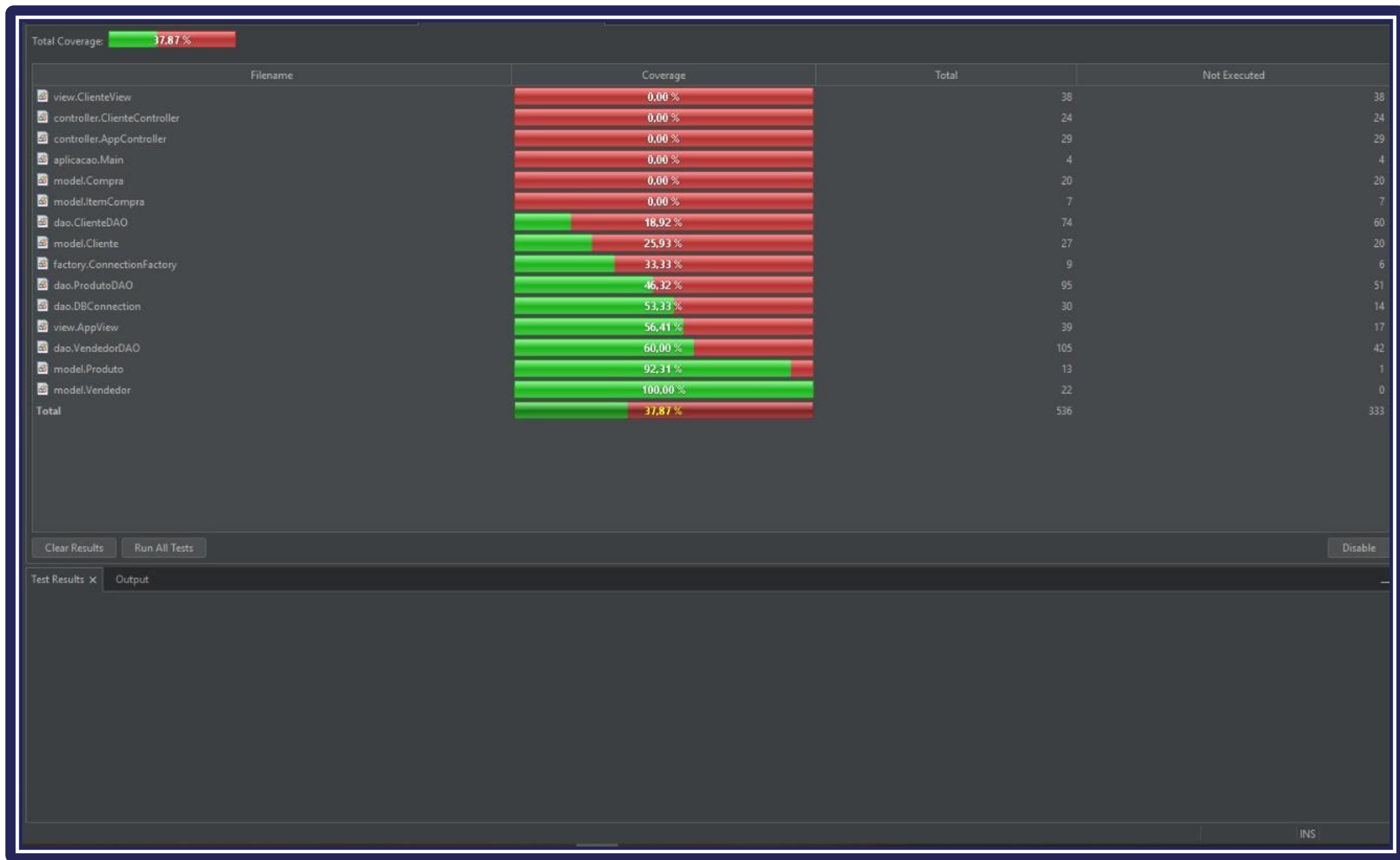
%TESTE 10,Tentar cadastrar um cliente com informações inválidas(Cadastro de novo cliente)

%RUNTIME912

Teste executado em 05/12/2023, 20:19:54

- ✓ Cadastrar um novo produto com sucesso
- ✓ Falha ao cadastrar um produto com informações inválidas
- ✓ Consultar a lista de vendedores
- ✓ Cadastrar vendedor com CNPJ inválido
- ✓ Cadastrar novo cliente com sucesso
- ✓ Tentar cadastrar um cliente com informações inválidas



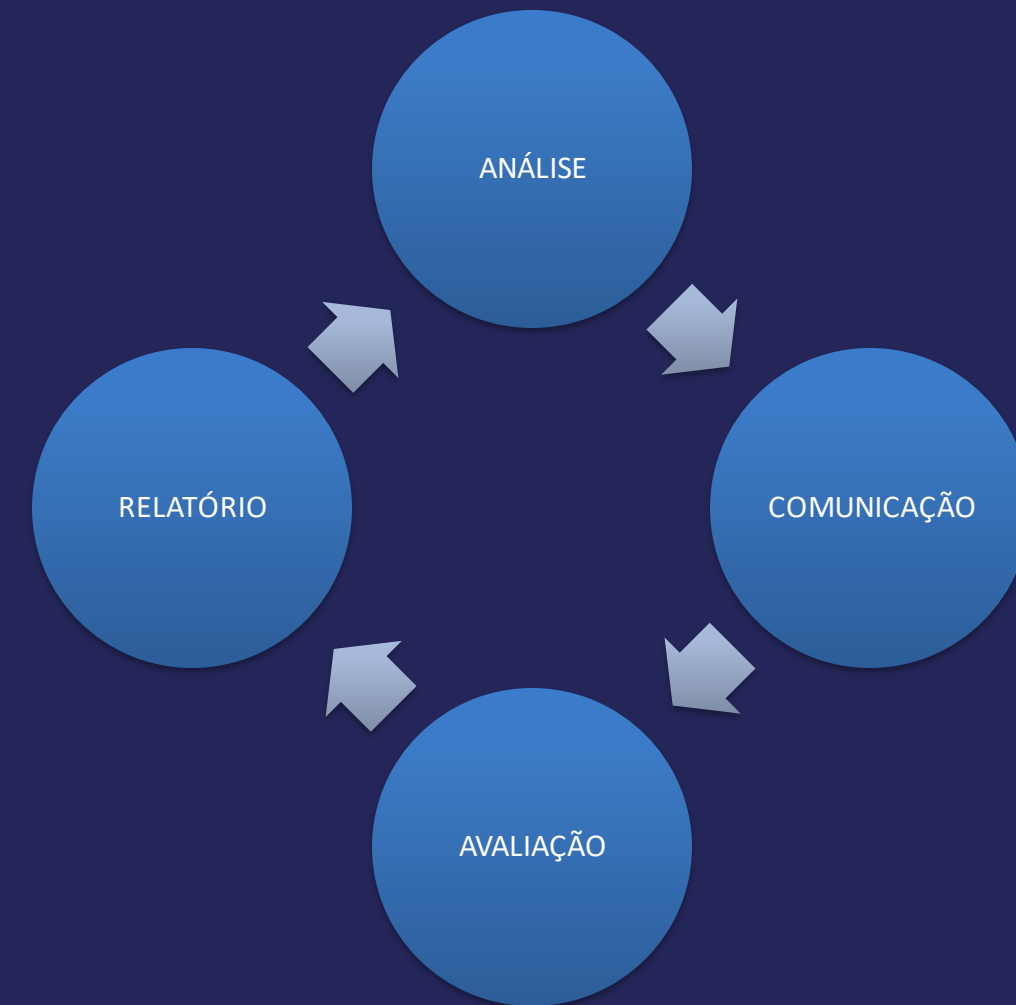


Testes utilizando Code Coverage



# Resultados

Os resultados obtidos demonstram não apenas a conclusão bem-sucedida do projeto, mas também o aprimoramento significativo das habilidades da equipe em Gestão e Qualidade de Software. A implementação do sistema CRUD, aliada às práticas de teste e às métricas de qualidade, resultou em um produto final sólido e confiável.



A análise de métricas e cobertura revelou índices que superaram as expectativas, indicando a eficácia das práticas adotadas. O roteiro de teste, com o uso conjunto de GHERKIN, JUNIT e CUCUMBER, não apenas validou a funcionalidade do sistema, mas também proporcionou uma documentação clara e acessível para futuras manutenções.





# Considerações

---

Em conclusão, este projeto se destaca como um marco em nosso percurso de desenvolvimento profissional. A experiência adquirida não apenas fortaleceu nossos conhecimentos em Gestão e Qualidade de Software, mas também reflete o comprometimento e a capacidade de nossa equipe em enfrentar desafios complexos e solucionar problemas.

A implementação do sistema CRUD, aliada às práticas de teste e à análise de métricas, não apenas atendeu aos requisitos estabelecidos, mas superou as expectativas, evidenciando nosso compromisso com a excelência técnica. Este projeto não é apenas um exercício isolado, mas um catalisador para o contínuo aprimoramento e crescimento de cada integrante do time, contribuindo para o sucesso de futuros empreendimentos em Gestão e Qualidade de Software.



# Nossa Equipe

---

CURSO	NOME	RA	E-MAIL
<b>ANÁLISE E DESENVOLVIMENTO DE SISTEMAS</b>	Jean Carlos Oliveira da Silva	32210226	jeancarlos_bh@outlook.com
	Gabriel Henrique de Oliveira Moura	322115099	gabriel.moura@outlook.com.br
	Almo Contim	322118509	almo.contim@hotmail.com
	Sâmeck Zanela Costa	322212223	sameck.szc@gmail.com
	Lucas Henrique	32228309	lucasupate@gmail.com
	Carlos Caique Silvestre Moreno	12523219226	caiquesmoreno@gmail.com