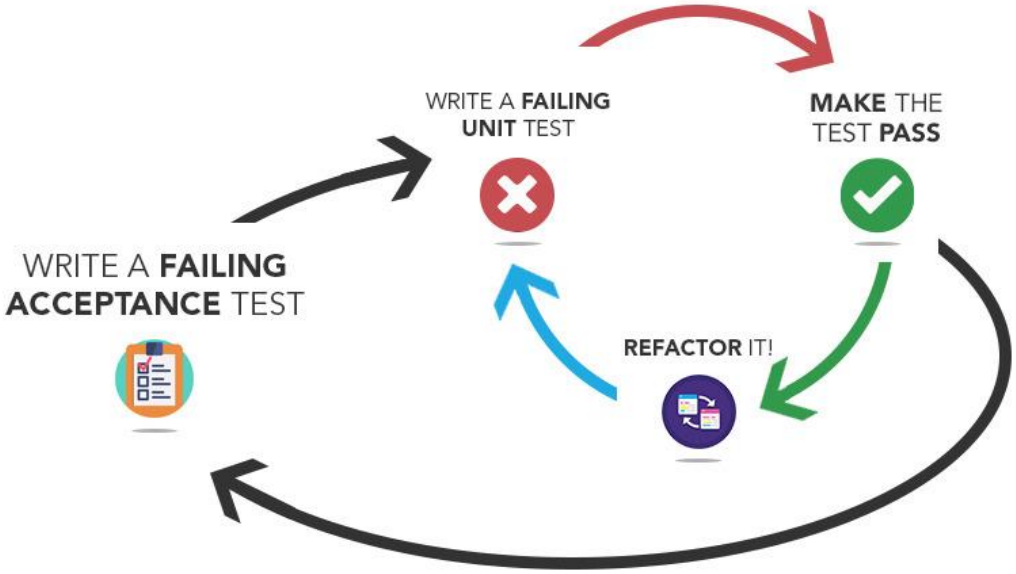
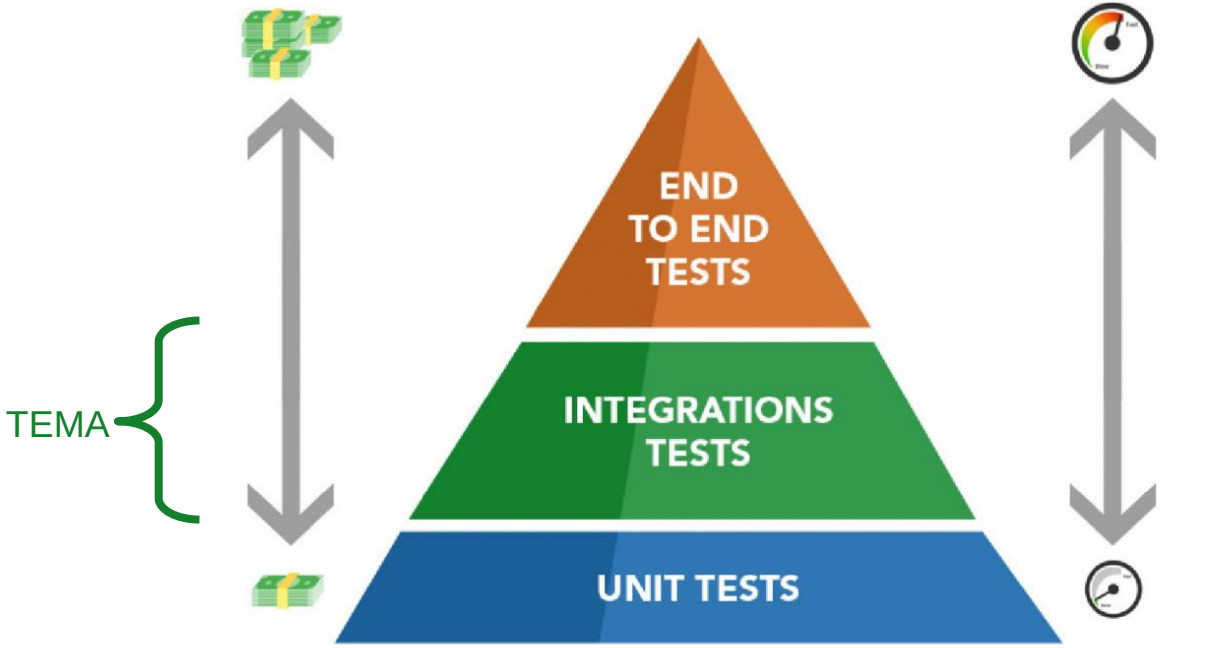

eximiaCo
EXCELÊNCIA TECNOLÓGICA POTENCIALIZANDO NEGÓCIOS

REVIEW - **METODOLOGIA**



REVIEW - **PIRÂMIDE**



TESTES DE UNIDADE

VERIFICA UMA UNIDADE DE COMPORTAMENTO

Deve verificar o garantir o funcionamento de uma **unidade**, que pode ser composta por "***small piece of code***" ou um **comportamento** de negócio.

EXECUTA RÁPIDO

A execução deve ser rápida para não atrapalhar o *flow* de trabalho.

ISOLAMENTO

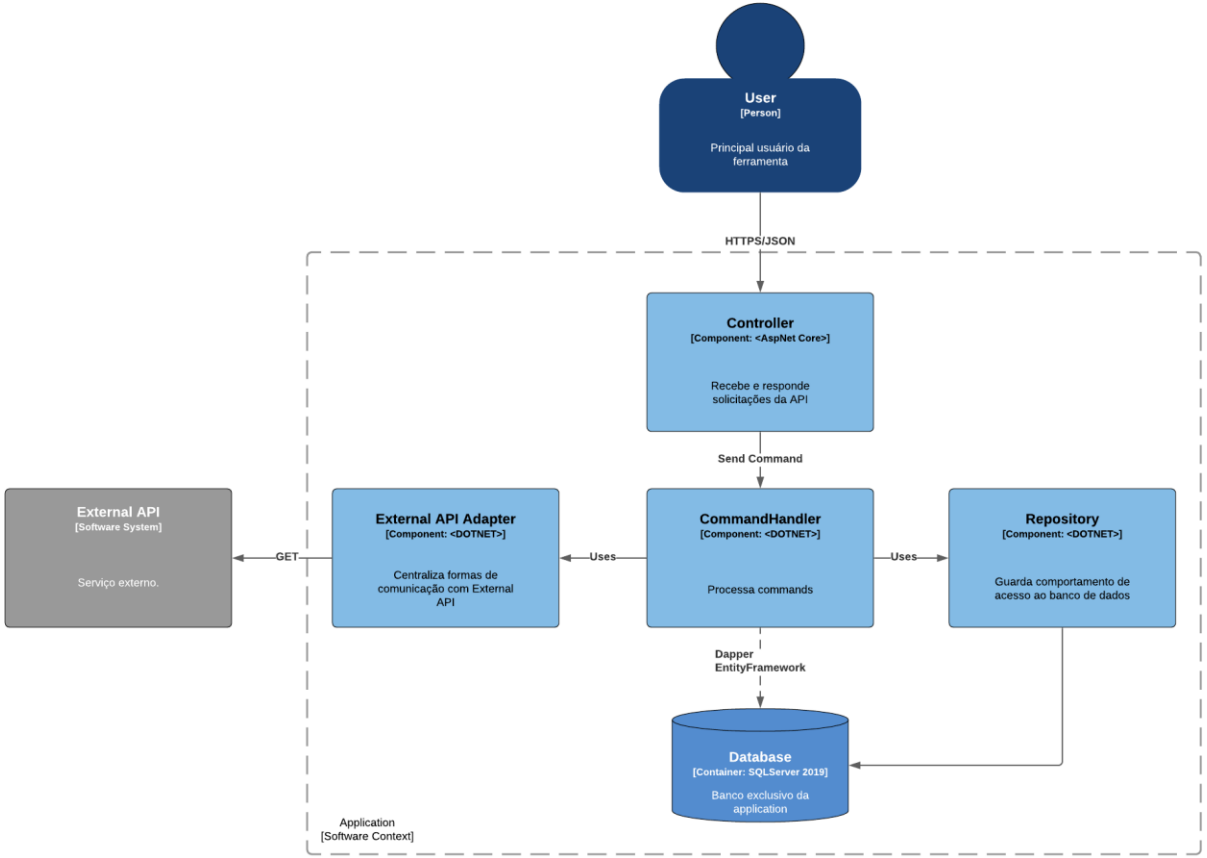
A execução deve ser isolada, em relação a outros testes e componentes externos

TESTES DE INTEGRAÇÃO

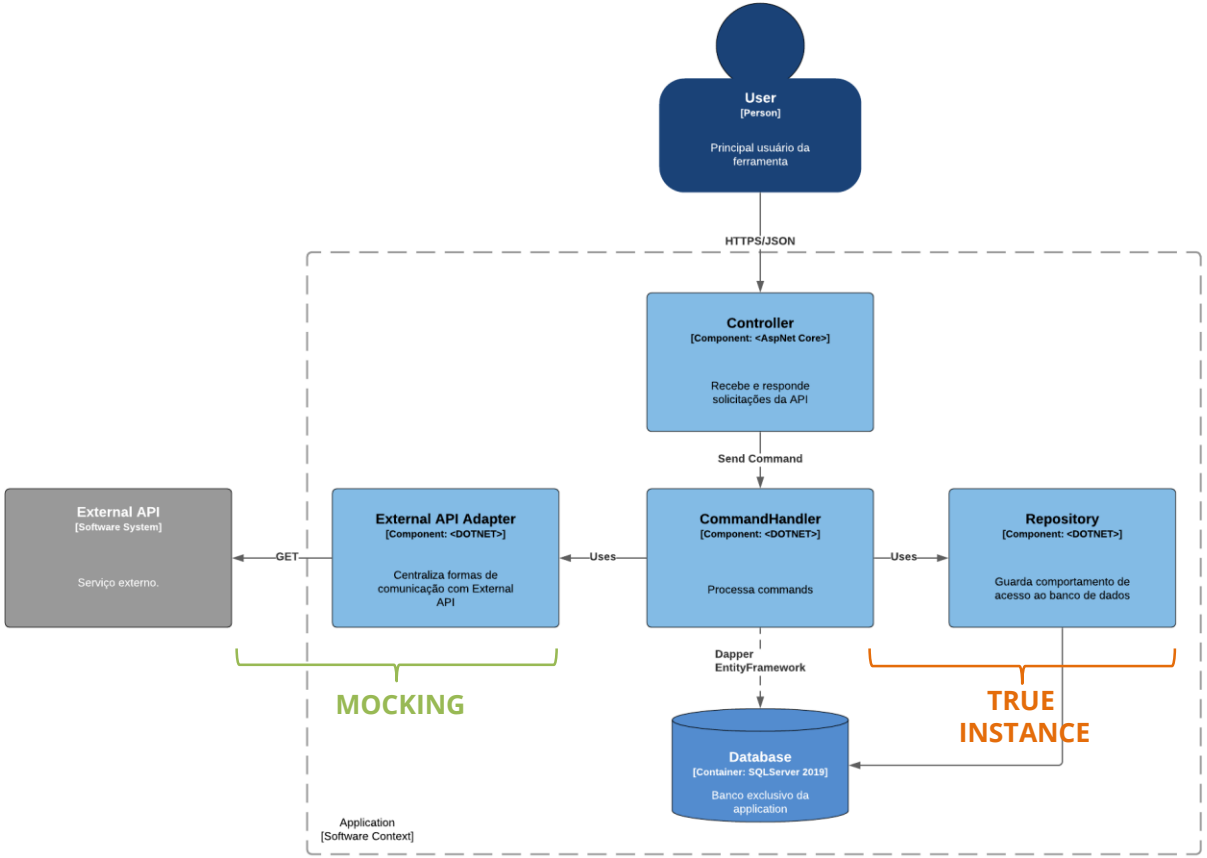
QUALQUER TESTE QUE NÃO É UM TESTE DE UNIDADE.

Teste de integração é um termo muito amplo e, normalmente, causa dificuldade na compreensão.

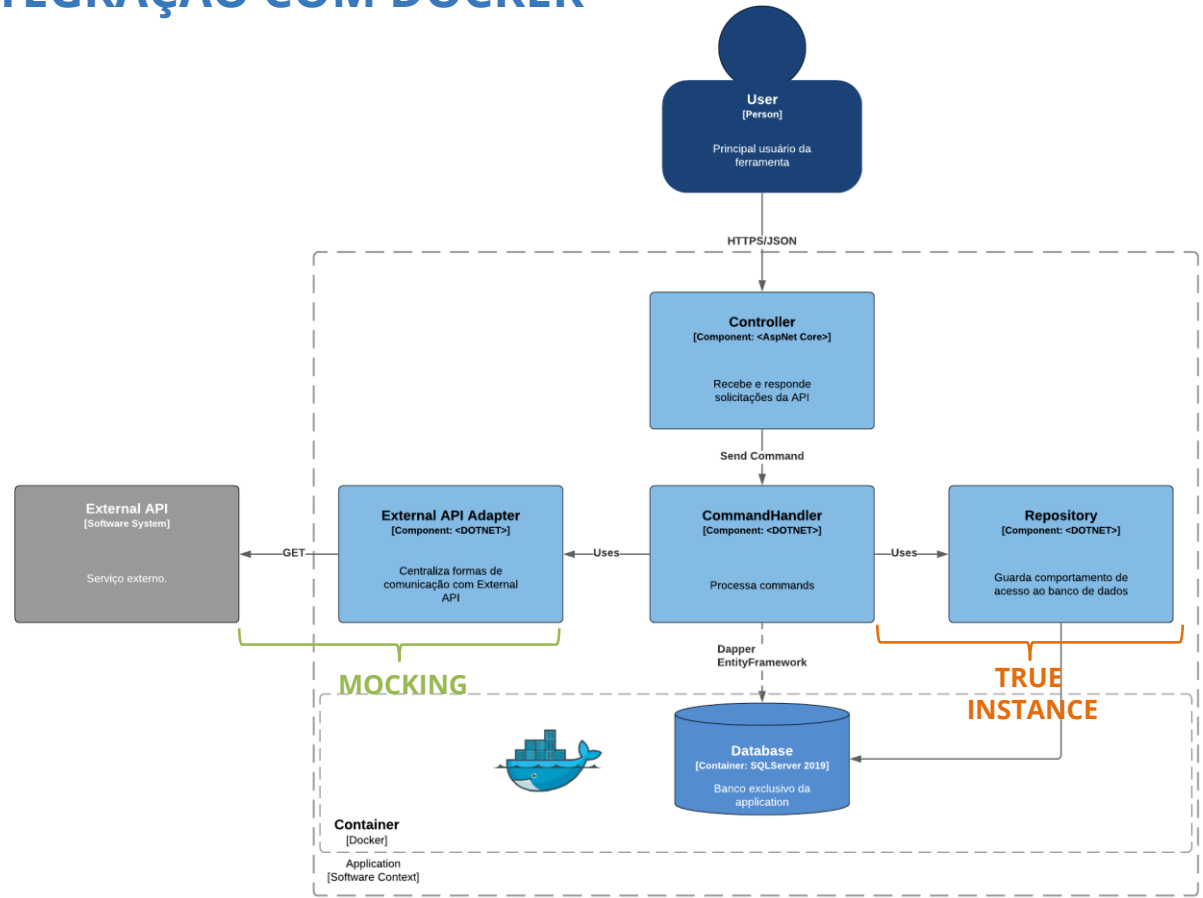
CENÁRIO - COM BANCO DE DADOS



CENÁRIO - COM BANCO DE DADOS



TESTE INTEGRAÇÃO COM DOCKER



TESTE INTEGRAÇÃO COM DOCKER

```
public abstract class SqlServerIntegrationTestBase
            : IAsyncLifetime, IAssemblyFixture<SqlServerDockerCollectionFixture>
{
    protected readonly SqlServerDockerCollectionFixture _fixture;
    protected EntityFrameworkContextTestHelper _testHelper;

    protected SqlServerIntegrationTestBase(SqlServerDockerCollectionFixture fixture)
    { _fixture = fixture; }

    public virtual async Task InitializeAsync()
    {
        var sqlConnectionString = _fixture.GetSqlConnectionString();
        _testHelper = new EntityFrameworkContextTestHelper(sqlConnectionString);
        await _testHelper.InitializeDatabase();
    }

    public virtual async Task DisposeAsync() =>
            await _testHelper.CleanupTestsAndDropDatabaseAsync();
}
```

CRIA O CONTAINER 1X POR EXECUÇÃO

ARRANGE: SEED

ASSERT: DROP DATABASE INSTANCE

EximiaCo

TESTE INTEGRAÇÃO COM DOCKER

```
public class ChangeCustomerNameTests : SqlServerIntegrationTestBase
{
    public ChangeCustomerNameTests(SqlServerDockerCollectionFixture fixture) : base(fixture) { }

    [Fact]
    public async Task Change_Customer_Name_Should_Be_Ok()
    {
        // Setup
        LocalFixture.Seed(this.GetContext);

        // Arrange
        using var context = this.GetContext();
        var handler = new ChangeCustomerNameCommandHandler(context);
        var command = CreateCommand();

        // Act
        var actual = await handler.Handle(command, CancellationToken.None);

        // Assert
        Assert.True(actual.IsSuccess);
        AssertDatabase(command);
    }
}
```

EXECUTA O NECESSÁRIO PARA CRIAR E REMOVER SQL SERVER CONTAINER

CHECA DIRETAMENTE NO BANCO

EximiaCo


eximiaCo
EXCELÊNCIA TECNOLÓGICA POTENCIALIZANDO NEGÓCIOS