PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

Bacharelado em Sistema de Informação

Introdução à pesquisa

Hugo Marcos Dias de Oliveira

Leomar Evangelista Duarte

Lucas Ponciano de Souza

Paulo José Ribeiro Silva

Poliana Ramos dos Anjos

TRABALHO FINAL FUNDAMENTO DE TESTE DE SOFTWARE

Parte 1 - Teste de Unidade

Belo Horizonte

2016

Hugo Marcos Dias de Oliveira

Leomar Evangelista Duarte

Lucas Ponciano de Souza

Paulo José Ribeiro Silva

Poliana Ramos dos Anjos

TRABALHO FINAL FUNDAMENTO DE TESTE DE SOFTWARE

Parte 1 - Teste de Unidade

Trabalho interdisciplinar de Fundamento de Teste apresentado ao Curso de Sistemas de Informação da Pontifícia Universidade Católica de Minas Gerais.

Professor: Claudiney Vander Ramos

Belo Horizonte

2016

LISTA DE FIGURAS

SUMÁRIO

INTRODUÇÃO

O objetivo deste trabalho da disciplina ministrada pelo professor Claudiney Ramos, segue com a ideia de entender e compreender a importância da funcionalidade do teste em um sistema realizado pelos programadores, que em meados de 30 anos atrás, esta área era totalmente deixada de lado. Atualmente, ganha-se muito em tempo de aprendizado e entendimento do programa, diminuindo o tempo e aumentando a lucrabilidade, uma vez que o número de erros decresce quase que exponencialmente com os testes realizados. Deve-se deixar claro que, os testes que serão mostrados no trabalho a seguir, não garantem com 100% de eficácia que não ocorra nenhum tipo de erro, todavia esses testes garantem uma legabilidade alta no programa e agrega valor ao seu desenvolvimento.

PLANO DE TESTE

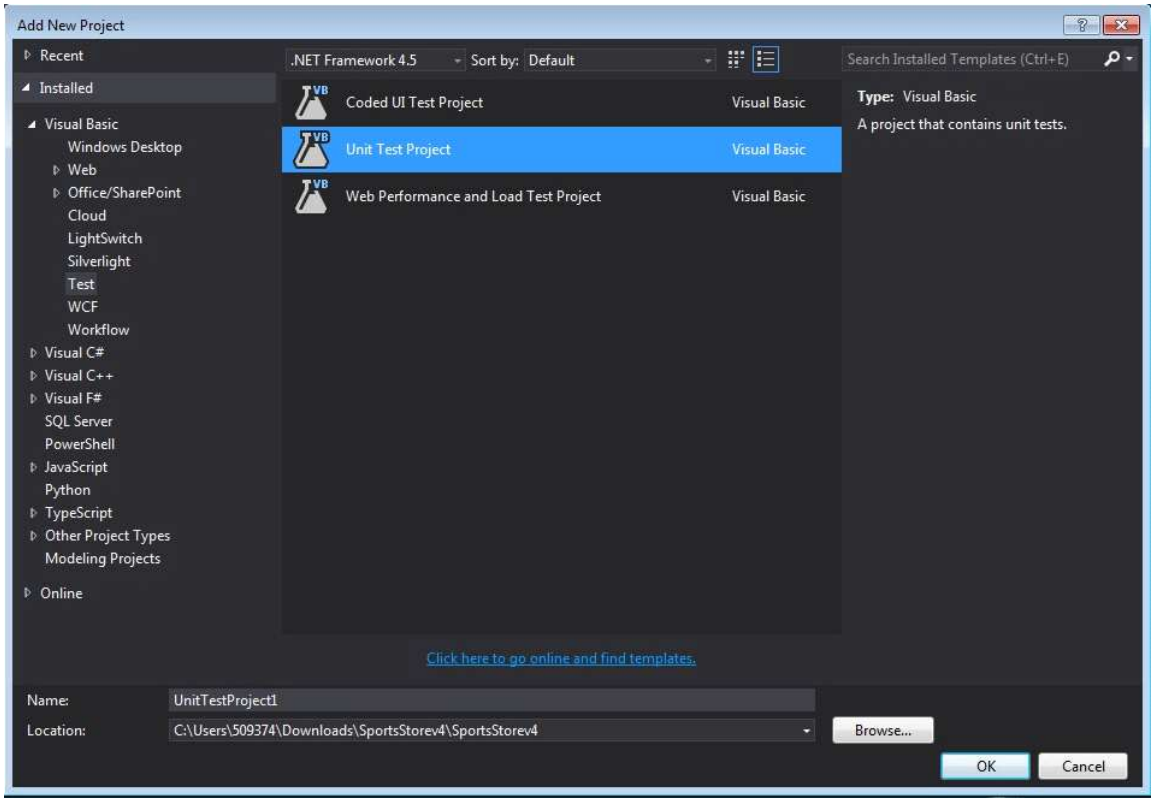
O plano de teste consiste em uma documentação descrito na IEEE 829, que nada mais é que uma norma que especifica a forma e o conjunto de artefatos no teste de software. Esse plano é constituído em três fases:

1. **Definir cronograma de atividades:** Serão apresentadas as atividades que devem ser realizadas, as etapas que devem ser seguidas e a ordem cronológica da execução;

No nosso trabalho, o programa desenvolvido a ser executado é nomeado de "SportsStore", um programa desenvolvido na linguagem C# que simula de forma básica uma loja online de esportes. Segue abaixo, uma demonstração do sistema online:

2. **Fazer alocação de recursos:** Definir quem realiza as atividades e quais ferramentas/recursos a serem utilizados;

Foi apresentado na disciplina, algumas ferramentas para a realização dos casos de testes de unidade para a linguagem C# como NUnit e CSUnit, entre outros. Entretanto, os membros do grupo optaram por utilizar a ferramenta própria do Visual Studio para realizar a tarefa solicitada. O projeto Unit Test, fica disponível para ser adicionado no projeto de Testes. Um atalho para sua utilização é: **Botão direito na Solution / Add / New Project / "Linguagem a ser utilizada (VB, C#, etc.)" / Test / Unit Test Project.** Segue imagem:



3. **Definir marcos do projeto:** Estabelecer marcos, ou milestones, a serem alcançados com objetivo de se fazer o acompanhamento.

CASOS DE TESTE

<Caso de Teste ()> - <Teste de Fazer Login>

Descrição: O usuário deve ser capaz de efetuar login no sistema, informando LOGIN e SENHA para poder acessar a aplicação. O objetivo esperado será de FALHA apenas para mostrar o uso da ferramenta Visual Studio com o projeto de Unit Test falhar.

Pré-condições: Sistema disponível e operante.

Pós-condições: Login falhar, apresentando na ferramenta uma mensagem onde o usuário possa interpretar que ocorreu a falha.

Dados necessários: Login e Senha.

<Caso de Teste 1> - <Create a new cart>

Descrição: Criar um novo carrinho de compras.

Pré-condições: Não se aplica.

Pós-condições: O teste irá verificar se em todos os locais do código em que chamam a função de criar um novo carrinho de compras, se atende as expectativas.

Dados necessários: Instanciar objeto da classe “Cart”.

<Caso de Teste 2> - <Create some test products>

Descrição: Será testado a criação de alguns produtos.

Pré-condições: Não se aplica.

Pós-condições: Verificação se o método permite a criação de alguns objetos da classe Product.

Dados necessários: Ser informado o ID e o Name do objeto da classe Product.

<Caso de Teste 3> - <Add some products to the cart>

Descrição: Adicionar alguns produtos ao carrinho de comprar online.

Pré-condições: Não se aplica.

Pós-condições: Verifica se é permitido a chamada do método “AddItem” a uma variável do tipo Cart.

Dados necessários: Instanciar um objeto do tipo Cart, chamar esse método passando os parâmetros Produto e quantidade (seguindo essa ordem).

<Caso de Teste 4> - <Create the mock repository>

Descrição: Cria uma simulação de repositório.

Pré-condições: Não se aplica.

Pós-condições: Verificar se foi possível instanciar a lista de repositório do tipo Mock.

Dados necessários: Não se aplica.

<Caso de Teste 5> - <Create the controller>

Descrição: Criar o controlador.

Pré-condições: Não se aplica.

Pós-condições: Verifica se foi possível instanciar um objeto do tipo CartController.

Dados necessários: Passagem de parâmetro do objeto do tipo “mock”.

<Caso de Teste 6> - <Add a product to the cart>

Descrição: Adicionar um produto ao carrinho de compras online.

Pré-condições: Não se aplica.

Pós-condições: Verificar se foi possível chamar o método “AddToCart” da classe “CartController”.

Dados necessários: Objeto do tipo Cart, ID e o tipo de retorno esperado (caso exista).