# iBody
# Plano de Testes

## Versão 1.0

# Plano de Testes

## 1. Introdução

### 1.1 Propósito

O propósito deste documento é coletar todas as informações necessárias para planejar e controlar o esforço de teste em uma interação. Ele descreve as abordagens de teste de software.

Este plano de testes suporta os seguintes objetivos:

- Rodar o teste em site hosteado pelo GitHub Pages
- Observar a navegabilidade do site, através de hiperlinks internos
- Utilizar-se de script em JavaScript com bibliotecas do Selenium WebDriver e driver do Google Chrome, rodando o teste automático pelo Node.JS

### 1.2 Escopo

As áreas que serão excluídas serão hiperlinks e área de inserção de e-mail, localizada no rodapé da página index.

### 1.3 Audiência pretendida

A audiência pretendida por este projeto é professores e alunos para fins educativos.

### 1.4 Terminologia e Acrônimos

**JS –** JavaScript

**JSON –** JavaScript Object Notation

**Node –** Node.JS

### 1.5 Referências

Projeto iBody – Especificações de Requisitos – **v1.0**

Selenium WebDriver – Documentação JavaScript – **(sem conhecimento da versão da documentação)**

Projeto iBody – Visão – **v1.0**

Projeto iBody – Diagrama de Componentes – **v1.0**

## 2. Missão e Motivação dos Testes

Esclarecer erros, bugs e outros problemas que o site possa ter, corrigindo-os assim que descobertos.

Otimização de tempo e recursos, além de maior eficiência nas buscas de bugs.

- Encontrar possíveis bugs.
- Encontrar importantes problemas e avaliar seus riscos à qualidade do projeto.
- Verificar as saídas para valores esperados.

Os elementos chaves que motivam os testes incluem os elementos gráficos, requerimentos funcionais, não-funcionais e riscos na qualidade.

## 3. Itens Alvos de Testes

A lista abaixo identifica os elementos que são alvos de teste (ou seja, representam os itens que serão testados).

- Menu interativo ao usuário, que contém o guia de páginas do projeto.
- Calculador de IMC na página principal, verificando sua eficácia.

## 4. Esboço dos Testes Planejados

### 4.1 Esboço dos Testes Incluídos

Serão executados os testes da calculadora de IMC e navegabilidade do menu.

### 4.2 Esboço dos Testes Excluídos

- Não será realizado o teste da inserção do e-mail, por não haver recursos suficientes para realização de um teste functional.

## 5. Abordagem de Testes

Desenvolver uma função para inserir dentro de campos que aceitam números, valores randômicos para testar diferentes resultados em diversos testes.

Desenvolver função para verificar a navegabilidade entre as páginas.

## 5.1 Técnicas e Tipos de Testes

### *5.1.1 Testes Funcionais*

| | |
|---|---|
| Objetivo da Técnica: | Verificar o calculador de índice de massa corporal |
| Técnica: | • Geração randômica de um valor para inserção nos campos que devem ser preenchidas para cálculo |
| Oráculos: | • Desenvolver uma função de geração randômica de valores tendo um limitador de máximo e mínimo. Com isso, espera-se a resposta do arquivo de JavaScript com diversos valores e verificar possíveis erros na geração da resposta |
| Ferramentas Necessárias: | • Node.JS<br>• Selenium WebDriver<br>• Driver do Google Chrome |
| Critérios de Sucesso: | • Calculador de IMC |
| Considerações Especiais: | O alert do navegador impacta diretamente no resultado do calculador, sendo ele o responsável pela saída da resposta dos valores randômicamente gerados pelo teste |

### *5.1.2 Testes de Interface de Usuário*

| | |
|---|---|
| Objetivo da Técnica: | • Verificar a navegabilidade entre as páginas do projeto |
| Técnica: | Será feito uma varredura no menu, verificando se os hiperlinks estarão funcionando corretamente |
| Oráculos: | Fazer chamadas utilizando o ID de cada tag <a> com “href” para observar se o resultado será bem executado. |
| Ferramentas Necessárias: | • Node.JS<br>• Selenium WebDriver<br>• Driver do Google Chrome |
| Critérios de Sucesso: | Todas as páginas terem sido abertas e retorno para a página principal |
| Considerações Especiais: | Função de inserir e-mail estando inativa, sem funcionalidade atual no programa |

## 6. Entregáveis

- Plano de Teste
- Script de Teste Funcional Automatizado

## 7. Necessidades de Ambiente

Deve estar em posse de uma IDE que possa gerar um arquivo JavaScript, bem como sua manipulação. Recomenda-se a utilização do VSCode, por sua facilidade na manipulação. Deve utilizer-se também do Node.JS para rodar o teste, o Selenium WebDriver para fornecer a biblioteca necessária para formular o teste e o driver do navegador Google Chrome, que será utilizado para rodar o teste.