|  |  |
| --- | --- |
| **Nome do projeto** | ToDo App |
| **Nome do componente** | Cucumber |
| **Autores do documento** | Bruno, Gian, Igor Issao, Igor Pancheski, Murilo |
| **Data** | XX/XX/XXXX |
| **Número de páginas** | X |

**1 Introdução**

  Este trabalho tem por objetivo explanar o funcionamento da ferramenta de realização de testes “Cucumber”.

O cucumber nada mais é do que uma ferramenta de linha de comando utilizada para automação de testes de softwares. No projeto em questão o objetivo será elaborar planos de teste, criando features, que são as funcionalidades e detalhando dentro dessas os cenários, que consistem na descrição ou etapas para que tal teste seja realizado.

 Após criado e detalhado todos os planos de testes, deve se criar uma interface para demonstração dos mesmos, para isso serão implementados nas plataformas necessárias para então, se obter respostas.

1. **Sumário de alto nível**

* Aprender a ferramenta Cucumber
  + Visão geral
  + Funcionalidades
  + Como implementar

* Implementação

* Features (Característica/Funcionalidade)
  + Descrição

* Detalhar Cenários
  + Dados (steps)
  + Condições
  + Resultados esperados

* Definir interfaces
  + A interface do projeto foi definida e será realizada apenas por linhas de comando.

* Definir plataformas a serem utilizadas na criação do projeto
  + Visual Studio Code
  + Cucumber
  + Javascript
  + Node js
  + Mongo DB
  + Express
  + Axios

**3 Ambiente e Configuração dos Testes**

3.1 A Ferramenta Cucumber e o Ambiente de execução dos testes:

  Para a materialização de uma ideia/software,deve-se haver uma boa comunicação entre cliente e desenvolvedor, ou seja, deve haver entendimento entre as partes interessadas. Isto é necessário pois diversas vezes quem idealiza um software não tem as habilidades necessárias para desenvolvê-lo. A solução para isso seria implementar o "Cucumber" pois ele tanto é utilizado para aplicação de testes quanto pra fazer esta comunicação.

  Mas o que é cucumber? Cucumber é uma ferramenta desenvolvida na linguagem de programação ruby que suporta o desenvolvimento orientado pelo comportamento, o cucumber interpreta as frases utilizando seu analisador de idioma chamando Gherkin e as transforma em “STEPS” (passos), cada frase vira um bloco de código e dentro de cada bloco pode-se colocar um trecho de código.

  Para realização dos testes com a ferramenta cucumber e execução dos planos de teste, fez-se necessário criar uma aplicação/webpage para que eles fossem demonstrados, com isso a equipe chegou a uma conclusão e criou o "ToDo App".

Para criação da aplicação, O visual Studio Code seria no caso o melhor ambiente para realizar esta implementação, visto que suporta  a linguagem de programação definida (JavaScript) e serve como extensor para diversas ferramentas, o que inclui a própria ferramenta cucumber.

**4**  **Recursos**

Esta seção apresenta os recursos recomendados para o teste do protótipo de arquitetura do C-Registration, suas principais responsabilidades e seu conhecimento ou configuração de habilidades.

**4.1**  **Funções**

Esta tabela mostra as principais funções e a responsabilidade cada integrante do grupo.

|  |  |  |
| --- | --- | --- |
| **Função** | **Recursos Mínimos Recomendados** | **Responsabilidades Específicas/Comentários** |
| Gerente de Testes | Bruno, Igor Issao, Murilo | ·         Relatório  ·         Adquirir recursos apropriados |
| Testador do Sistema | Igor Pancheski | ·         Executar testes  ·         Registrar resultados  ·         Corrige os erros  ·         Documentar defeitos |
| Administrador do Sistema de Teste | Igor Pancheski | ·         Administrar o sistema de testes. |
| DBA | Igor Pancheski, Gian, Bruno | ·         Administrar os dados de teste (banco de dados) |
| Designer | Bruno, Gian, Igor, Igor Issao | ·         Identifica e define a(s) classes de teste |
| Implementador | Gian | ·         Cria as classes de testes e implementa no sistema. |

**5 Objetivo do Teste**

**Objetivo 1 – Testar as principais funcionalidades do sistema.**

- **Condição do Teste** – O usuário deve conseguir criar, alterar, excluir, finalizar e listar as tarefas.

- **Abordagem** – Executar todas as funções citadas anteriormente.

- **Critério de aceitação** – Todas as funções citadas devem estar funcionando corretamente.

**Casos de teste**

**Caso de teste 1 – Criar tarefa.**

**- Descrição** – Verificar se o usuário consegui criar uma tarefa corretamente.

**- Critério de saída** – A tarefa será criada.

**- Objetivo mapeado** – Objetivo 1.

**Caso de teste 2 – Editar tarefa.**

**- Descrição** – Verificar se o usuário consegue alterar uma tarefa já criada.

**- Critério de saída** – A tarefa será alterada.

**- Objetivo mapeado** – Objetivo 1.

**Caso de teste 3 – Excluir tarefa.**

**- Descrição** – Verificar se o usuário consegue deletar uma tarefa.

**- Critério de saída** – A tarefa será deletada.

**- Objetivo mapeado** – Objetivo 1.

**Caso de teste 4 – Finalizar tarefa.**

**- Descrição** – Verifica se o usuário está tentando finalizar uma tarefa prevista para uma data superior a data atual.

**- Critério de saída** – A tarefa será finalizada ou aparecerá uma mensagem dizendo que a tarefa não poderá ser finalizada.

**- Objetivo mapeado** – Objetivo 1.

**Caso de teste 5 – Listar tarefa.**

**- Descrição** – Verificar se a listagem com filtros está funcionando corretamente.

**- Critério de saída** – As tarefas serão listadas conforme o filtro utilizado.

**- Objetivo mapeado** – Objetivo 1.

**Execução dos Testes**

**Cobertura**

As funcionalidades que foram testadas são: Permitir que a usuário possa criar tarefas, alterar as tarefas já criadas, deletar alguma tarefa caso a mesma não possa mais ser executada e listar as tarefas utilizando um filtro.