

Aplicativo para auxiliar na pratica de exercício (Levantai)

|  |  |
| --- | --- |
| **Disciplina:** | Engenharia de Software |
| **Professor:** | Amauri Antunes Filho |
| **Período:** | Noite |

|  |  |  |
| --- | --- | --- |
| **Nome:** | **Curso:** | **RA:** |
| Vitor Roberto Nunes | Engenharia da Computação | 00097747 |

Sorocaba

**Sumário**

[**1. Definições de projetos** 4](#_Toc73385680)

[**1.1 Qual o deverá ser o tipo:** 4](#_Toc73385681)

[**1.2 Descrição geral da finalidade do software:** 4](#_Toc73385682)

[**1.3 Quem utilizará o mesmo** 5](#_Toc73385683)

[**1.4 Qual plataforma (web? App? Desktop? Mais de uma?** 5](#_Toc73385684)

[**2. Desenvolvimento ágil** 6](#_Toc73385685)

[**2.1 Scrum Master** 6](#_Toc73385686)

[**2.2 Dono do produto** 6](#_Toc73385687)

[**2.3 Equipe de desenvolvimento** 6](#_Toc73385688)

[**2.4 Qual será a estrutura das daily’s** 6](#_Toc73385689)

[**2.5 Tamanho das sprints** 6](#_Toc73385690)

[**2.6 Como será documentado o backlog? Como será a validação com o cliente, ou usuários?** 6](#_Toc73385691)

[**3. Requisitos** 7](#_Toc73385692)

[**3.1 Stakeholders:** 7](#_Toc73385693)

[**3.2 Requisitos funcionais:** 7](#_Toc73385694)

[**3.3 Documento de requisitos:** 7](#_Toc73385695)

[**3.4 Requisitos não funcionais:** 8](#_Toc73385696)

[**3.5 Histórias de usuários para requisitos funcionais:** 8](#_Toc73385697)

[**3.6 Requisitos não funcionais em linguagem natural:** 8](#_Toc73385698)

[**4. Modelagem** 9](#_Toc73385699)

[**4.1 Diagrama de caso de uso** 9](#_Toc73385700)

[**4.2**  **Diagrama de classe** 9](#_Toc73385701)

[**5. Testes** 10](#_Toc73385702)

[**5.1 TestComplete Mobile** 10](#_Toc73385703)

[**5..1.1 Conceito** 10](#_Toc73385704)

[**5.1.2 Funcionamento** 10](#_Toc73385705)

[**5.1.3 Código Exemplo** 10](#_Toc73385706)

[**5.2 Selenium WebDriver** 11](#_Toc73385707)

[**5.2.1 Conceito** 11](#_Toc73385708)

[**5.2.2 Funcionamento** 12](#_Toc73385709)

[**5.3 Código Exemplo** 12](#_Toc73385710)

[**5.4 Teste Funcional Automatizado** 15](#_Toc73385711)

[**6. Qualidade** 16](#_Toc73385712)

[**6.1 Especificação de Qualidade** 16](#_Toc73385713)

[**6.1.1 Usabilidade** 16](#_Toc73385714)

[**6.1.2 Segurança e privacidade** 16](#_Toc73385715)

[**6.1.3 Confiabilidade** 16](#_Toc73385716)

[**6.1.4 Eficácia** 16](#_Toc73385717)

[**6.1.5 Capacidade de Manutenção** 17](#_Toc73385718)

[**6.1.6 Portabilidade** 17](#_Toc73385719)

[**6.1.7 Característica** 17](#_Toc73385720)

[**6.2 ISSO 9126** 17](#_Toc73385721)

[**6.3.1 Métricas avaliativas** 18](#_Toc73385722)

[**6.3.1.1 Indicadores de demanda** 18](#_Toc73385723)

[**6.3.1.2 Métrica de Arquitetura** 18](#_Toc73385724)

[**6.3.1.3 Métrica de Implementação** 18](#_Toc73385725)

[**6.3.1.4 Indicadores de teste** 19](#_Toc73385726)

[**6.3.1.5 Indicadores de desempenho** 19](#_Toc73385727)

[**6.3.1.6 Tempo para desenvolvimento e conclusão do projeto** 19](#_Toc73385728)

[**6.3.1.7 Recursos necessários para implementação de novas funções** 19](#_Toc73385729)

[**7. Segurança** 20](#_Toc73385730)

[**7.1 Analise de ativos** 20](#_Toc73385731)

# **1. Definições de projetos**

# **1.1 Qual o deverá ser o tipo:**

* Aplicações Interativas baseadas em transações. Ou seja, Aplicativos executados em computadores remotos, aplicativos que podem ser acessados ​​remotamente através de dispositivos móveis ou até mesmo computadores. Esta categoria inclui serviços em nuvem, como aplicativos da web, aplicativos de internet banking, e-mail e compartilhamento de fotos.

# **1.2 Descrição geral da finalidade do software:**

* Prevenção de LER (lesões por esforços repetitivos);
* Prevenção de dores na coluna;
* Prevenção de diabetes;
* Prevenção de obesidade;
* Prevenção de problemas circulatórios;
* Auxiliar pessoas que trabalham em Home Office, avisando de tempos em tempos a hora de levantar e praticar exercícios;
* Simples, personalizável e fácil de usar. Este temporizador para exercícios faz com que seja mais fácil do que nunca criar suas próprias rotinas de treinamento durante seu expediente de trabalho.

O Levantai é temporizador de intervalo totalmente personalizável e utilizado em todo o mundo, na qual sua finalidade é ditar quais exercícios e intervalos de tempos o usuário deve seguir. Este aplicativo tem como objetividade prevenir que você obtenha doenças ocasionadas pela repetição excessiva de movimentos e pela estagnação. O Levantai irá direcionar exercícios de acordo com sua rotina de trabalho e seu tempo, fazendo com que se exercite e melhore sua qualidade de vida. O foco do Levantai é auxiliar as pessoas que trabalham em Home Office e as que trabalham muito tempo em uma posição, notificando de tempos em tempos a hora de levantar e praticar exercícios.

* **Exercícios de repetição – Pressione Próximo**Durante o exercício, com o recurso de repetições, você pode tomar seu tempo para realizar seu exercício e quando terminar, apenas pressione próximo. Pois os exercícios podem ter uma mistura de repetições e exercícios temporizados.
* **Smartwatch**Leve seu treinamento ao próximo nível com o Levantai no seu smartwatch Android. O Levantai irá facilmente sincronizar com seu smartwatch e permitirá que você monitore suas rotinas de exercício diretamente do seu pulso.
* **Recursos que as pessoas adoram sobre Levantai:**

Salve suas predefinições para alternar rapidamente entre diferentes atividades;

Use-o em segundo plano enquanto estiver usando outros aplicativos ou com a tela bloqueada;

Obtenha áudio, vibração ou notificações silenciosas adicionais;

Salve seus treinos para uso posterior.

# **1.3 Quem utilizará o mesmo**

* Pessoas que trabalham na frente do computador e em homeOffice.

# **1.4 Qual plataforma (web? App? Desktop? Mais de uma?**

* App.

# **2. Desenvolvimento ágil**

# **2.1 Scrum Master**

* Nome do responsável pelo Scrum Master é Lara Vitória Machado Ribeiro.

# **2.2 Dono do produto**

* O nome da empresa responsável pela contratação do desenvolvimento do software é LVG - Soluções Tecnológicas. A mesma fica responsável pela identificação dos recursos ou requisitos do produto, de tal forma que ela deve prioriza-los durante o desenvolvimento do projeto e revisar constantemente o backlog do produto para confirmar que o projeto continue atendendo suas necessidades.

# **2.3 Equipe de desenvolvimento**

* Gabrielle Marisa Manrique;
* Lara Vitória Machado Ribeiro;
* Vitor Roberto Nunes.

# **2.4 Qual será a estrutura das daily’s**

* As daily’s começaram as 09:00 da manhã e terão cerca de 15 minutos de duração. Nas daily’s deverão ser respondidas objetivamente essas 3 perguntas: O que fiz ontem? O que vou fazer hoje? Algum impedimento que barre a mim ou ao time? Para que o projeto seja desenvolvido com sucesso.

# **2.5 Tamanho das sprints**

* As sprints terão uma duração de 2 semanas.

# **2.6 Como será documentado o backlog? Como será a validação com o cliente, ou usuários?**

* O backlog será documentado através da plataforma Trello, onde cada membro da equipe poderá ter acesso e checar o que cada pessoa foi designada a fazer, o que já foi feito e o que falta fazer;
* Para realizar os testes do App, a equipe irá pegar 10 pessoas que não sabem sobre o mesmo e irá pedir para elas baixarem, se cadastrarem e tentarem usar, para que assim, a equipe possa ver se há dificuldade para cadastro e uso do aplicativo, sendo possível fazer um parâmetro do que é necessário alterar e especificar.

# **3. Requisitos**

# **3.1 Stakeholders:**

* Usuário que trabalha homeoffice;
* Usuário com problema de saúde;
* Pessoa responsável pelo acompanhamento dos treinos;
* Equipes de TI responsável pela instalação e manutenção do sistema;
* Responsavel pela compra da plataforma e que implementará na empresa;

# **3.2 Requisitos funcionais:**

* Usuário deve estar cadastrado;
* O que pode ser personalizado é o tempo, o conjunto de exercícios em base de sua posição monótona, peso, idade e altura que possibilitará a utilização de algumas funçoes do app, como por exemplo calcular o IMC;
* Objetivo por dia, gameficar;
* Cadastro de exercícios que podem ser realizados;
* Personalizar tempo de inatividade;
* Salvar dados dos treinos e predefinições dos usuários;
* Cadastrar doenças ocupacionais comuns;
* Interface para smartwatch.
* Armazenamento na nuvem, neste caso os dados do usuário poderão ser sincronizados com a nuvem para que ele não perca as configurações caso troque aparelhos ou desinstale o app.

# **3.3 Documento de requisitos:**

* Usuário deve estar cadastrado;
* Deve existir uma interface para receber todas as informações necessárias;
* Deve existir uma interface para smartwatch;
* Interface para gamificação do aplicativo, com pontos e rankings;
* Os usuários serão pessoas que trabalham muito tempo em uma posição ou em Home Office;
* As informações exibidas para o usuário são salvas e variam ao longo do tempo;
* Para algumas perguntas, mais de uma resposta pode ser selecionada;
* Deve existir uma tela para o usuário com o tempo de inatividade;
* Os usuários podem alterar alguns dados cadastrais;
* O usuário terá acesso ao histórico de todos os treinos já feitos.

# **3.4 Requisitos não funcionais:**

* O app é disponibilizado apenas para a plataforma Android;
* Uso de design responsivo;
* Armazenamento na nuvem, neste caso seria o banco de dados e os servidores que irão viabilizar o sistema;

# **3.5 Histórias de usuários para requisitos funcionais:**

* Eu como usuário com problema de saúde, quando necessitar levantar e me exercitar, quero poder visualizar os exercícios que podem ser realizados através do smartphone ou pelo smartwatch, de modo em que eu seja avisado o momento em que devo me exercitar. Para que eu possa me exercitar prevenindo doenças.
* Eu como pessoa responsável pelo acompanhamento dos treinos, quando necessitar visualizar exercícios quero poder visualizar o histórico do usuário final. A exibição do histórico deve apresentar os exercícios que serão feitos e já foram feitos, o cálculo do IMC e se o mesmo possui alguma doença, para que eu possa ser orientado de forma correta a maneira de se fazer o exercício.
* Eu como responsavel pela aquisição da plataforma e que irei implementar na minha empresa, quando necessitar de um exercício específico ou alterar o exercício, quero uma interface personalizada e que possa ser possível a personalização do exercício para o meu porte físico e para as debilidades que possuo por conta do estado da minha saúde, portanto, será preciso a exibição dos exercícios que posso fazer de acordo com o histórico de saúde, para que eu consiga me exercitar sem prejudicar minha saúde ou agravar alguma doença.
* Eu como usuário que exerce o trabalho homeoffice, quando necessitar levantar e me exercitar, quero poder informar e visualizar os dados como o tempo a ser exercitado, o intervalo dos treinos, para que eu possa manter a saúde.

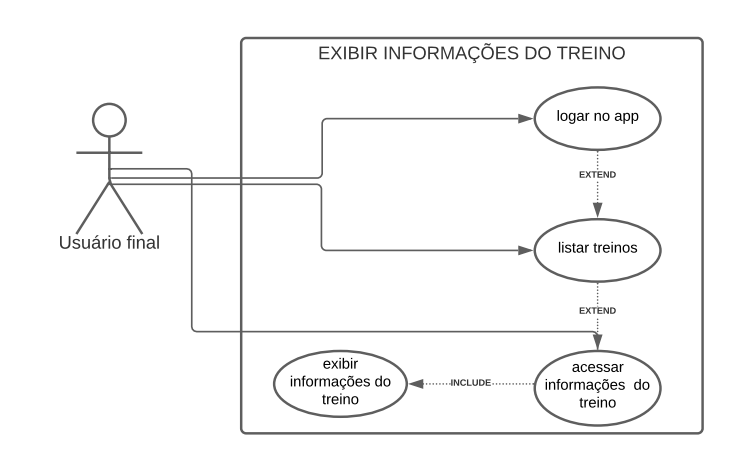
# **3.6 Requisitos não funcionais em linguagem natural:**

* O aplicativo será disponibilizado apenas na plataforma android pois o responsavel pela compra do app possui parcerias com empresas com esta plataforma, além de ter feito pesquisas e entrevistas, que lhe proporcionaram informações alegando que as pessoas que trabalham em sua empresa possuem apenas smartphones com o plataforma Android e que há uma maior quantidade de usuarios no mesmo.
* O responsavel pela compra do produto não tem o foco em comprar e possuir um servidor para armazenar os dados dos clientes/usuários/funcionários, portanto, foi apresentado a ideia de ser armazenado na nuvem, deste modo as informações serão acessadas por meio do browser.

# **4. Modelagem**

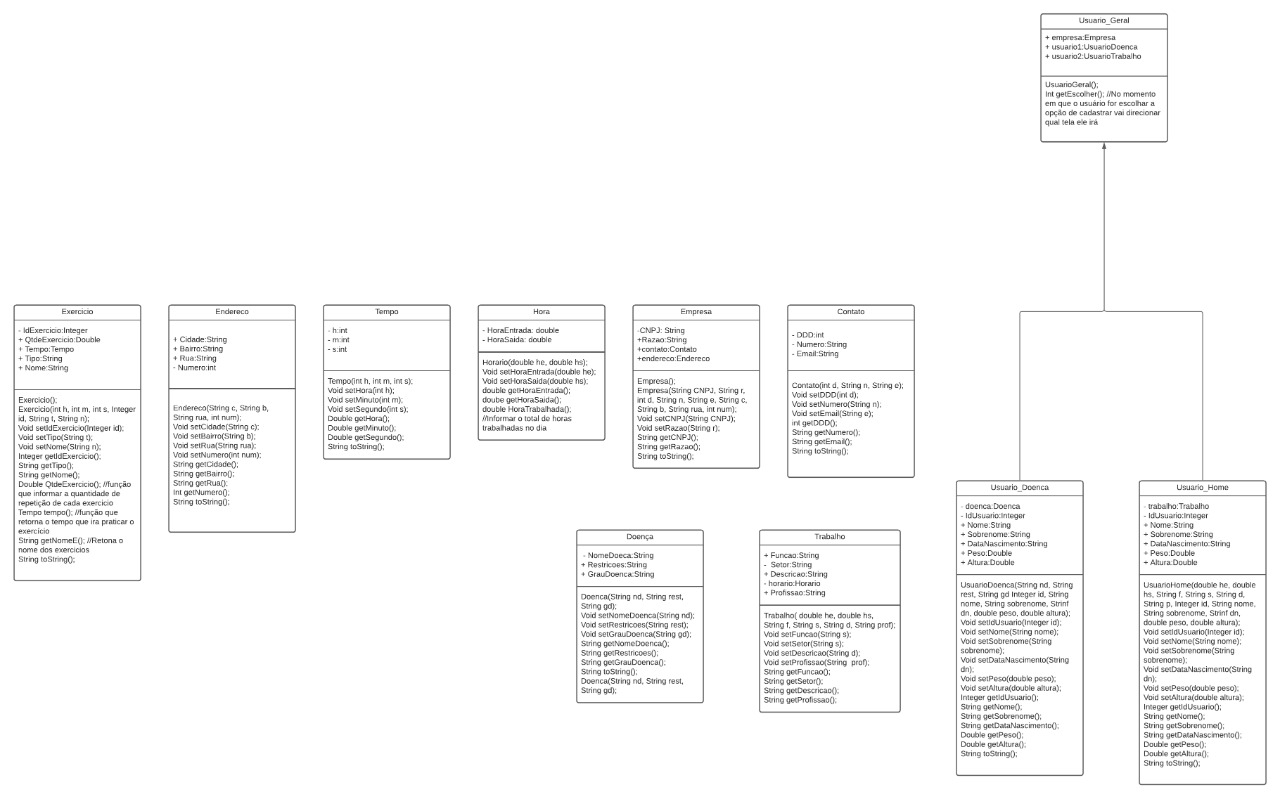
# **4.1 Diagrama de caso de uso**

A seguir temos a modelagem de caso de uso:

****

# **4.2 Diagrama de classe**

A seguir temos a modelagem de classe:



# **5. Testes**

# **5.1 TestComplete Mobile**

# **5..1.1 Conceito**

É uma plataforma de teste automatizada totalmente funcional. TestComplete permite que os testadores criem testes automatizados para aplicativos Microsoft Windows, Web, Android e iOS. Você pode usar operações orientadas por palavras-chave para registrar manualmente, escrever scripts ou criar testes e usá-los para cópia automática e registro de erros.

# **5.1.2 Funcionamento**

Testes funcionais foram criados para aplicativos iOS e Android para verificar se eles funcionam conforme o esperado. Além disso, ele também é usado para criar e automatizar muitos tipos diferentes de testes de software. A gravação de teste e a criação de reprodução irão registrar o processo de teste manual executado pelos testadores e permitir que ele seja continuamente reproduzido e mantido como um teste automático. Os testadores podem modificar os testes registrados posteriormente para criar novos testes ou aprimorar os testes existentes com mais casos de uso.

# **5.1.3 Código Exemplo**

Criar/Escrever arquivo de texto

function CriandoEEscrevendoEmUmArquivoTexto(){

//Caminho do arquivo que será gravado

var path = "C:\\teste.txt";

//Se o arquivo não existe, cria

if(!aqFile.Exists(path)){

aqFile.Create(path);

}

//Criando objeto para abertura do arquivo

var meuArquivo = aqFile.OpenTextFile(path, aqFile.faWrite, aqFile.ctANSI, true);

//Inserção de texto em cada linha do arquivo

meuArquivo.WriteLine("Linha 1");

meuArquivo.WriteLine("Linha 2");

meuArquivo.WriteLine("Linha 3");

meuArquivo.WriteLine("Linha 4");

//Fecha o arquivo

meuArquivo.Close();

}

# **5.2 Selenium WebDriver**

# **5.2.1 Conceito**

Consiste em uma ferramenta de automação de testes funcionais web, na qual compõe um pacote maior denominado “Open QA Selenium”. O Selenium é uma ferramenta open source que nasceu com o objetivo de ser uma alternativa para as ferramentas pagas que dominavam o mercado. A Selenium está dividido em três módulos diferentes:

Selenium IDE: Ferramenta criada como plug-in para o navegador Mozila Firefox, com a qual é possível registrar as operações realizadas no navegador com um script em forma de tabela, repetindo assim as operações registradas.

Selenium Remote Control (RC): Permite que você salve o conteúdo gravado em outros idiomas (como C # e Java). A ferramenta pode alterar o conteúdo gravado anteriormente, adicionando loops, funções e bibliotecas que podem ser criadas.

Selenium Grid: executando testes de maneira coordenada, os testes podem ser distribuídos em vários computadores, reduzindo assim o tempo necessário para testar vários navegadores ou sistemas operacionais.

# **5.2.2 Funcionamento**

A maioria dos testes de aplicativos da web envolve o uso de um navegador da web. O Selenium permite simular o comportamento do usuário usando um navegador da web.

Quando usamos o Selenium para testes automatizados, existem basicamente duas ferramentas que podem ser usadas:

Selenium IDE: é uma ferramenta que pode testar rapidamente o protótipo do script. Este é um plug-in do Firefox que pode registrar o comportamento do usuário: as páginas que ele visita, o texto escrito no formulário, cliques em links e botões, etc. Depois de registradas, essas ações podem ser exportadas para scripts de teste em várias linguagens de programação: Java, Python, Perl, JavaScript, etc.

Selenium WebDriver: é uma ferramenta que fornece uma API que permite escrever scripts de teste de forma mais eficiente e organizada. Esta é uma escolha natural quando queremos usar o Selenium para escrever testes automatizados para aplicativos da web. Os exemplos neste artigo usam WebDriver.

O Selenium WebDriver usa o suporte de automação nativa de cada navegador para fazer chamadas diretamente para o navegador. Portanto, o teste escrito com WebDriver é muito realista, porque não usa seu próprio mecanismo JavaScript (como fazia no Selenium 1), mas chama diretamente o navegador.

# **5.3 Código Exemplo**

[21:01, 10/05/2021] Iara: ant+shift+c = cancelar a execução

click(foto) = para clicar em determinado lugar

type("") = para digitar(não aceita caracter especial

keyDown(Key.TECLA) = para clicar em determinada tecla

wait(segundos de espera) = tempo de espera

wait([imagem]) = espera a imagem

nomevariavel = valor

nomevariavel = imagem -> guardando imagem na variavel

paste("") -> para colar texto, quando for usar caracter especial -> paste(unicd(""))

Ativar o TAB

type(Key.TAB)

Abrir arquivo somente para leitura

arquivo = open(r"C:\estrategia\filmes.txt", "r")

Ler uma linha do arquivo

linha = arquivo.readline()

Um LOOP para ler todas as linhas do arquivo

while linha:

//Código

linha = arquivo.readline()

print("fim")

Linha de Texto:

Harry Potter Bruxo 120 20/10/2020

Comando split (Igual ao Explode do PHP)

Quebrar string pelo TAB

partes = linha.split("\t")

titulo = partes[0]

partes[1] irá retornar Diretor

partes[2] irá retornar Duracao

partes[3] irá retornar Lancamento

#rstrip - para cancelar enter

[21:01, 10/05/2021] Iara: abrir chrome

openApp(r"C:\Program Files (x86)\Google\Chrome\Application\chrome.exe")

'esperar 2 segundos

Função Wait

wait(2)

wait([IMAGEM])

Criação de variável

nomevariavel = valor

nomevariavel = imagem

Clicar em determinada área da tela

click([IMAGEM])

Colar Texto

paste("Exemplo")

titulo = "Senhor dos Anéis"

Colar Texto que contenha caractere especial

paste(unicd("TEXTO"))

Ativar o TAB

type(Key.TAB)

Abrir arquivo somente para leitura

arquivo = open(r"C:\estrategia\filmes.txt", "r")

Ler uma linha do arquivo

Um LOOP para ler todas as linhas do arquivo

linha = arquivo.readline()

while linha:

print(linha)

linha = arquivo.readline()

print("fim")

Linha de Texto:

Harry Potter Bruxo 120 20/10/2020

Comando split (Igual ao Explode do PHP)

Quebrar string pelo TAB

partes = linha.split("\t")

titulo = partes[0]

paste(unicd(titulo))

partes[1] irá retornar Diretor

partes[2] irá retornar Duracao

partes[3] irá retornar Lancamento

# **5.4 Teste Funcional Automatizado**

Teste Automatizado é o uso de um software para controlar a execução do teste de software, comparando os resultados esperados com os resultados reais.

A automação de testes traz diversas vantagens: é mais rápido, diminui a chance de erro humano (mas é preciso atentar para que os scripts não apresentem falhas) e reduz o esforço com tarefas repetitivas, liberando os testadores para atividades que exijam raciocínio humano.

# **6. Qualidade**

# **6.1 Especificação de Qualidade**

A especificação de qualidade do projeto é o local mais adequado para o projeto, como tempo de resposta aos usuários, usabilidade, portabilidade etc. Dentre outras diversas categorias, enfatizamos estas em nosso projeto.

# **6.1.1 Usabilidade**

Usabilidade refere-se à capacidade de usar software e ser fácil para qualquer pessoa o operar. A maioria dos aplicativos atuais funciona em dois ou mais idiomas, pode ser customizada e possui um layout intuitivo, que ajuda o usuário a entender a situação ao utilizar a ferramenta. Quanto mais alto o índice de usabilidade, melhor o software é apresentado às pessoas que o operam todos os dias.

# **6.1.2 Segurança e privacidade**

Se não for dada atenção suficiente aos níveis de segurança e privacidade dos dados, todas as soluções técnicas podem ser afetadas pela invasão e roubo de dados. A maioria das soluções desenvolvidas em computação em nuvem já vem com certificados de segurança, criptografia de dados e outros sistemas de controle de acesso que tornam a ferramenta mais confiável.

# **6.1.3 Confiabilidade**

Se o software não tiver defeitos, ele é confiável, mas é considerado viável. Confiabilidade se refere à capacidade de manter um certo nível de desempenho ao operar em um determinado ambiente de uso.

# **6.1.4 Eficácia**

A eficiência inclui o tempo de execução e os recursos envolvidos (se forem compatíveis com o nível de desempenho do software).

# **6.1.5 Capacidade de Manutenção**

Corresponde à capacidade de manutenção do software. A função de manutenção está relacionada à facilidade de modificação de produtos de software. Este recurso é de particular interesse para desenvolvedores e não deve ser confundido com a possibilidade de configuração de software. As modificações incluem, por exemplo, correções em produtos ou adaptações às mudanças de requisitos (como mudanças nos regulamentos).

# **6.1.6 Portabilidade**

Representa a capacidade do software de se mover de um ambiente para outro.

Mostra a possibilidade de usar código-fonte em diferentes plataformas de execução. Em outras palavras, assume-se que o programa pode ser projetado para ser executado em um ambiente com características diferentes. )

# **6.1.7 Característica**

A função refere-se à função do software quando solicitada pelo usuário, como a exibição de dados na tela. Essa característica se refere à capacidade de realizar tarefas.

# **6.2 ISSO 9126**

Uma norma que tem como objetivo avaliar a qualidade, bem como as suas características, sub características e atributos. Este é o objetivo da ISO 9126, mais especificamente a NBR 13596, que padroniza a avaliação da qualidade do software.

Um produto final com qualidade é uma necessidade, uma vez que desde o levantamento de requisitos até a fase de produção, os fatores que atendem ao que o cliente precisa devem estar em evidência, bem como o feedback para melhoria contínua dos processos da organização.

O modelo de qualidade estabelecido pela ISO/IEC 9126 segue os seguintes itens:

* Processo de desenvolvimento, onde a qualidade afeta a qualidade do software gerado.
* Produto, compreendendo as características de qualidade do software gerado.
* Qualidade em uso, onde trata a comparação da qualidade do software em cada contexto específico de usuário.

# 

# **6.3.1 Métricas avaliativas**

Métricas são métricas quantificáveis ​​utilizadas para analisar os resultados de um determinado processo, operação ou estratégia, ou seja, são métricas de desempenho. Uma métrica é um número que representa a métrica que você deseja seguir para entender se o projeto é eficaz. Dessa forma, somente por meio de medição, por exemplo, é possível avaliar o andamento do processo de software. A partir disso, seguimos com os principais indicadores que iremos introduzir no projeto.

# **6.3.1.1 Indicadores de demanda**

Nesta métrica, iremos medir o software da perspectiva do usuário, considerando como o software irá interagir com o software e os resultados esperados de cada interação. Deve especificar regras de negócios e regras de interface entre o software e o usuário. Ao final da contagem, medirá quanto o usuário solicitou a partir da demanda e / ou quanto recebeu. Para medir este indicador, serão analisados ​​os pontos de função e o NESMA.

# **6.3.1.2 Métrica de Arquitetura**

Nesta métrica, mediremos a capacidade da arquitetura com base no número de funções empacotadas fornecidas para designers, desenvolvedores e testadores. A medição deste indicador será realizada por meio da análise do número de componentes abstratos e interfaces dos serviços não funcionais que atende.

# **6.3.1.3 Métrica de Implementação**

Nesta métrica, mediremos fisicamente o tamanho do código implementado. A métrica pode ser o tamanho físico do número de linhas de cada arquivo do código-fonte ou pode ser a lógica de usar o número de comandos presentes em cada linha do código-fonte. A medição dessa métrica será feita por meio de LOC (linha de código), que contém uma técnica para medir o número de linhas de código. Existem duas variantes desta técnica, SLOC (linha de código) e LLOC (linha de código lógico), que medem o número de linhas físicas e o número de comandos por linha de código, respectivamente, conforme descrito no parágrafo anterior.

# **6.3.1.4 Indicadores de teste**

Mesmo se a quantidade mínima de código que a tecnologia selecionada deve cobrir for documentada e acordada, essa métrica medirá a quantidade de código coberta pelo teste. A medição dessa métrica será feita por meio de "cobertura de código".

# **6.3.1.5 Indicadores de desempenho**

Está métrica verifica o que foi documentado e estabelecido nos requisitos mínimos e máximos de desempenho para o correto desenvolvimento da arquitetura de software e seleção de tecnologias e frameworks. O método de medição dessa métrica é informar a velocidade média em milissegundos, e a velocidade média responde a cada função de negócio que responde a cada operação do usuário do tipo de dispositivo concebido na arquitetura atende às expectativas.

# **6.3.1.6 Tempo para desenvolvimento e conclusão do projeto**

Este indicador inclui o tempo total da equipe trabalhando no projeto, o tempo para um desenvolvedor de software específico executar uma parte específica do aplicativo e o prazo para desenvolvimento e conclusão do projeto. processar.

# **6.3.1.7 Recursos necessários para implementação de novas funções**

Este indicador inclui o tempo extra que a equipe deve despender no projeto para implementar certas funções, a quantidade total de trabalho por dia do homem e recursos de computador.

# **7. Segurança**

Neste trabalho, devemos aplicar a tecnologia de segurança para apoiar o desenvolvimento e manutenção de aplicações, podendo resistir a ataques maliciosos que visam destruir o sistema ou os dados nele contidos.

# **7.1 Analise de ativos**

|  |  |  |
| --- | --- | --- |
| **Ativo** | **Valor** | **Exposição** |
| Cadastro e registro de usuário. | Alto. no caso de a informação da senha do usuário vazar perderíamos muito com isto, pois o usuário perderia informações restritas e assim acarretando em que teríamos perca de usuários e seriamos levado a justiça por conta do mesmo. | Alto. Possível perda financeira de usuários futuros que não procurarão o aplicativo para se exercitar. |

|  |  |  |
| --- | --- | --- |
| **Ativo** | **Valor** | **Exposição** |
| Próprio sistema de software. | Alto. Informações pessoais dos usuários, podendo causar constrangimento e podendo deixar os usuários desconfortáveis o que causaria repudia dos usuários, consequentemente perca de usuários. | Alto. Possível perda financeira de usuários futuros que não procurarão o aplicativo para uso novamente. |