UNIP EaD
Andreia Domingues da Silva RA: 2023727
Bruna Rossanesi RA: 2016910
Douglas Pinheiro RA: 2014309
Mayara Verônica Oliveira Silva RA: 1889899
Natalia Fabricio Costa RA: 2006807
Pedro Expedito De Oliveira RA: 2028118

Projeto Integrado Multidisciplinar IV

Santa Fé-PR

2020

UNIP EaD
Andreia Domingues da Silva RA: 2023727
Bruna Rossanesi RA: 2016910
Douglas Pinheiro RA: 2014309
Mayara Verônica Oliveira Silva RA: 1889899
Natalia Fabricio Costa RA: 2006807
Pedro Expedito De Oliveira RA: 2028118

Projeto Integrado Multidisciplinar IV

Projeto Integrado Multidisciplinar para obtenção do título de tecnólogo em Análise e Desenvolvimento de Sistemas apresentado à Universidade Paulista – UNIP EaD.
Orientador(a): Marcelo Henrique dos Santos.

Universidade Paulista – Unip

Faculdade de Analise e Desenvolvimento Sistemas

Projeto Integrado Multidisciplinar

Santa Fé-PR

2020

Resumo

Este projeto integrado Multidicisplinar V, do Curso analise e desenvolvimento de sistemas tem o objetivo de desenvolver um software de alta qualidade e simplificando o armazenamento de dados de cada paciente, gerando um arquivo de texto para salvar as informações referentes aos pacientes e a lista dos pacientes no grupo de risco que será enviada para secretaria da Saúde. Contendo apenas as informações de CEP e Idade do paciente, mantendo em sigilo todas as outras informações e seguindo a risca todos os requisitos e funcionalidades apontadas no manual fornecido pela instituição UNIP. Sendo o projeto inteiramente desenvolvido em Linguagem C com a utilização apenas de ferramentas livres como GIT, Neovim, Mingw e GNU Debugger sendo indispensáveis para realização do projeto.Vale destacar o Mingw como principal ferramenta capaz de compilar Linguagem C para código de máquina que pode executar no sistema operacional Windows da Microsoft que não oferece ferramenta livres para desenvolver em sua própria plataforma, deixando o desenvolvedor dependente de suas ferramentas e cerceando a sua liberdade. Utilizando estas ferramentas é possível ter uma melhor qualidade final no produto e mantendo os direitos sobre o sistema desenvolvido e acima de tudo a liberdade.

Palavras-chave: Covid-19, Linguagem C, hospitais.

Abstract

This integrated project Multidicisplinar V, of the course analyze and develop systems has the objective of developing high quality software and simplifying the data storage of each patient, generating a text file to save as information related to patients and the list of patients in the risk group that will be sent to the Secretary of Health. Containing only the information of the patient's CEP and Age, keeping all information confidential and strictly following all the requirements and characteristics indicated in the manual provided by the UNIP institution. Being the developed project developed in Language C using only free tools such as GIT, Neovim, Mingw and GNU Debugger being indispensable for the realization of the project. Mingw stands out as the main tool capable of compiling Language C for machine code that can run on Microsoft's Windows operating system that doesn't offer free tools to develop on its own platform, leaving the developer dependent on his tools and curtailing his freedom. Using these tools it is possible to have a better final quality in the product and keep the rights on the developed system and above all freedom.

Keywords: Covid-19, C lang , Hospital.

Sumário

1 INTRODUÇÃO

Com o grande crescimento de casos de corona vírus houve a necessidade de softwares capazes de contribuir com a administração dos casos, ou seja um software capaz de facilitar os cadastros economizando tempo dos profissionais de saúde e com a finalidade de manter um acompanhamento progressivo para inclusive , auxiliar nas tomadas de decisões e ações preventivas.

E para atender as necessidades dos profissionais de saúde em poder cadastrar novos portadores do vírus COVID-19, o presente projeto desenvolveu um software capaz de autenticar os profissionais responsáveis pelo registro, cadastrar novos pacientes e salvar as informações em um arquivo de texto puro, para que na próxima execução do programa os dados se mantenham. Junto a isso, também é gerado outro arquivo de texto puro para ser enviado para secretaria da saúde do município no qual vai conter apenas os dados da idade e cep do paciente, mantendo todas as outras informações em sigilo.

Uma das qualidades do software é que ele possui validação de campo para dificultar erros no cadastro de informações como nome, cpf, data de nascimento entre outras informações, também não permite a duplicação de pacientes, economizando recursos computacionais. Neste trabalho é explicado todas as ferramentas utilizadas, o que é engenharia de software, desenvolvimento de software, framework scrum além de possuir um manual de como utilizar o software para novos usuários.

2 ENGENHARIA DE SOFTWARE

A definição sobre engenharia de software tem como objetivo reforçar a importância dos critérios a serem considerados na construção de um programa e para entender melhor este objetivo vale destacar alguns conceitos:

- O que é engenharia de software?

  > Se caracteriza por um desenvolvimento de software prático , ordenado e medido para produzir sistemas satisfatórios aos usuários e que respeitem prazos e orçamentos. (PEDRICZ, 1995).

- O que é software?

  > É um elemento de sistema lógico com instruções (programas de computador), que quando executadas produzem a função e o desempenho desejados e são também estruturas de dados que possibilitam que os programas manipulem adequadamente a informação. (PEDRICZ, 1995)).

Ainda sobre este conceito, o software não se desgasta por justamente não ser um elemento físico se comparado com um hardware, no entanto exige-se que sejam feitas constantemente melhorias e manutenção do código e isso gera consequentemente erros e falhas que necessitam ser corrigidas todas as vezes que novas mudanças ocorrem.

Sendo assim um software tem a função de atender as necessidades de um usuário e que neste trabalho utilizamos a conotação de cliente para melhor entendimento.

3 COMPONETES DO SOFTWARE

Até aqui é possível compreender que um software é criado com base nas exigências do cliente e diante disso são definidos qual o tipo de linguagem de programação que será adotada para estruturar os dados e as especificações do projeto. Logo, essa linguagem é processada e convertida para que o computador possa ler e executar o código e assim o projeto vai ganhando utilidade e sendo ajustado conforme necessário.

Neste sentido, fica claro que uma das características mais importantes de um software é a escolha da linguagem de programação adequada para o desenvolvimento do programa e isso não significa que será exatamente a linguagem mais utilizada no mercado e sim a que for apropriada ao negócio deste cliente e para aquilo que ele realmente precisa.Todavia é importante destacarmos alguns aspectos que compõem uma boa estrutura de software como por exemplo:

Facilidade de manutenção: com a constante mudança e variação na forma de codificação fica evidente e fundamental que ao escrever o código ele seja simples e fácil para acompanhar as evoluções do mercado e as adaptações que forem importantes para melhor qualidade do produto. Padronizado: apesar de que um software possa ser reutilizável , ou seja , a sua sintaxe pode ser modificada, ele precisa seguir um padrão para que tenha uniformidade sendo inclusive incorporável. Executável: primordial que o programa seja executado de acordo com as instruções indicadas e que esteja funcionando adequadamente. Documentado: registrar com detalhes explicando como o código foi escrito e como utilizar este software, assim facilita nos casos em que houver a necessidade de se realizar alguma manutenção , correção por algum desenvolvedor que não criou mas precisa efetuar reparos ou adaptações.

Embora não exista um único conceito definitivo sobre componentes de software, alguns autores como o Szyperski traz a seguinte definição:

> Um componente de software é uma unidade de composição com interfaces especificadas e dependências de contexto explícitas apenas. Um componente de software pode ser implantado de forma independente e está sujeito à composição por terceiros.” (SZYPERSKI, 2002)

Após a elucidação de Szyperski ,(D’SOUZA, 1999) declara que os componentes de software reúnem vários artefatos tais como: um código fonte executável, projetos e especificações , teste e por fim a documentação. Portanto, estas importantes propriedades formam um modelo ou um conjunto de operações independentes mas que se complementam para que o software execute a sua função.

4 DESENVOLVIMENTO DE SOFTWARE

O processo de desenvolvimento de software não é uma atividade tão simples, existem alguns desafios inerentes do processo que precisam ser considerados com muita atenção para que o produto seja eficiente dentro do padrão e da forma em que foi estabelecido entre o cliente solicitante do software e a equipe de desenvolvedores, programadores e analistas, que precisaram tirar a ideia do papel e colocá-la em prática, ou seja, software funcionando , testado e sem apresentar erros. Porquanto é primordial entregar um produto com a qualidade que o cliente espera e isso não significa apenas uma entrega , mas sim que agregue valor para o cliente , através de um trabalho bem executado e com o propósito bem definido.

Partindo desta premissa , atualmente muitas empresas necessitam adquirir novos softwares para satisfazerem as demandas do negócio, se manterem competitivas no mercado, inovando com o uso de novas tecnologias em um mundo que vive em uma corrida constante e feroz pelo uso intenso de novas aplicações e por recursos que gerem lucro e impactem positivamente a sociedade e as organizações.

Ao longo dos anos alguns princípios e técnicas foram estabelecidos para dar suporte a coordenação das etapas de construção de um software e podemos chamar esse conjunto de práticas como engenharia de software, que para Sommerville define engenharia de software como:

> É uma disciplina da engenharia, segundo a qual os engenheiros de software utilizam métodos e teorias da ciência da computação, e aplicam tudo isso de modo eficaz em relação aos custos, a fim de solucionar problemas difíceis, [...] e se ocupam de todos os aspectos da produção, desde os estágios iniciais de especificação, até a manutenção, depois que o produto entrou em operação.(SOMMERVILLE, 2019)

Desta forma , é importante destacar que existem três elementos fundamentais para o processo de criação de um software: métodos, ferramentas e procedimentos, e essa tríplice precisa ser bem definida para que o desenvolvimento de software inicie com qualidade e incluindo todas as etapas e atividades de maneira ordenada.

Portanto , a escolha do processo exige muita responsabilidade e deverá ser adequada para que seja relevante na estruturação e preparo da equipe que integrará toda a execução do produto até que seja concluído e disponível ao usuário.

Segundo (PEDRICZ, 1995) a criação de um produto envolve etapas de gerenciamento de projetos entretanto sob a ótica da engenharia de software, que nesta condição representa o início do processo de desenvolvimento de software, compreendendo as atividades de medição, estimativa, análise de erros , programação de

atividades, monitoração e controle.

Para os autores (JUANA-ESPINOSA, 2011) e (KERZNER, 2002) , projetos podem ser diferentes de acordo com o propósito estabelecido e por serem um empreendimento possuem início e fim, onde as atividades devem ser organizadas de forma lógica considerando custo, prazo e qualidade. Gerenciar projetos de software com eficiência é uma tarefa que representa um grande desafio, no entanto se torna indispensável para o sucesso e qualidade do produto. E por isso devemos considerar algumas etapas que precisam ser seguidas:

- Escopo Escopo: é a definição das funcionalidades e dos requisitos necessários feitos de forma detalhada, a fim de deixar claro quais os objetivos operacionais e de negócios que ele precisa atender ou não atender
- Recursos levantamento dos recursos que serão aplicados para suportar todo o desenvolvimento do software , incluindo recursos humanos e ferramentas por exemplo:
  - Recursos Humanos habilidades técnicas exigidas, duração das tarefas de cada envolvido, data de início destas pessoas, quantos profissionais serão necessários.
  - Ferramentas podem conter várias ferramentas como de análise e métricas de acompanhamento para gestão, sistemas e programas para escrever o código, fazer testes dentre outros itens.
- Custo definir o orçamento para que o valor a ser investido possa ser bem administrado e isso se torna um fator crucial , porque o objetivo é gerir os gastos e investimentos de forma a não gerar prejuízos ou gastos excedentes com decisões mal tomadas.
- Riscos Riscos: é importante considerar e avaliar os riscos que podem surgir durante o desenvolvimento do software, com um planejamento baseado nas melhores práticas se torna possível limitar os riscos que são toleráveis e os que podem ser previstos.

Portanto, ao definir o processo com todas as atividades que deverão ser desempenhadas por cada membro da equipe , fases do projeto, observando prazos , custos , riscos, estimativas e sem perder o foco no propósito para que ao entregar o produto com qualidade, atendendo as expectativas das partes interessadas, cliente ou do usuário final deste software, todos os envolvidos possam ter resultados positivos, de um desenvolvimento baseado nas melhores práticas , no planejamento e na utilização das ferramentas corretas.

Existem alguns modelos de ciclo de vida que servem como base e são muito importantes durante o desenvolvimento de software e que dependendo do objetivo de um nicho específico, podem adotar diferentes modelos e estratégias. Os modelos mais comuns são segundo (ABRAN; MOORE, 2001)

- Modelo Clássico (da d'água) ele é sequencial pois engloba projeto, codificação, teste e posteriormente a manutenção
- Prototipação desenvolvimento incremental/iterativo: este modelo possui uma característica importante , o refinamento dos requisitos, ou seja , o time de desenvolvimento junto com o cliente precisam fazer um levantamento do que será necessário para criar o produto e com isso simular as importantes interfaces e funções do sistema, podendo ser até testado pelo cliente.
- Espiral, modelo de reutilização:Prevê maior interação com o cliente e maior frequência em relação a verificação de riscos, possuindo quatro atributos: Planejamento, Análise dos riscos, Engenharia e Avaliação do cliente.

A partir destes conceitos , é possível perceber que cada vez mais aumentam as demandas por softwares mais complexos , com recursos mais sofisticados e entregues em menor tempo possível. E para promover um desenvolvimento mais focado na entrega de valor, o Scrum possui grande relevância no auxílio e ao fluxo de construção do softwares.

5 DESENVOLVIMENTO DE SOFTWARE COM SCRUM

O manifesto ágil contém os princípios da metodologia Scrum, um framework para desenvolver e manter produtos complexos sendo fundamentado em três pilares que apoiam o controle do processo empírico, sendo eles: transparência, inspeção e adaptação. Então podemos perceber que o objetivo do scrum é ser leve, simples de entender e extremamente difícil de dominar. Não é algo novo, já é utilizado desde o início de 1990 por diversas empresas, um dos destaques do Scrum é que ele não é um processo ou técnica de construção de produtos, mas sim um framework no qual pode ser empregado diversos processos.

O time scrum visa a flexibilidade, criatividade e produtividade, sendo composto pelo:

- Product Owner ou seja dono do produto que é o responsável por maximizar o valor do produto. O time de desenvolvimento é composto por profissionais que são auto-organizados não dependendo que pessoas de fora os organizem, sendo

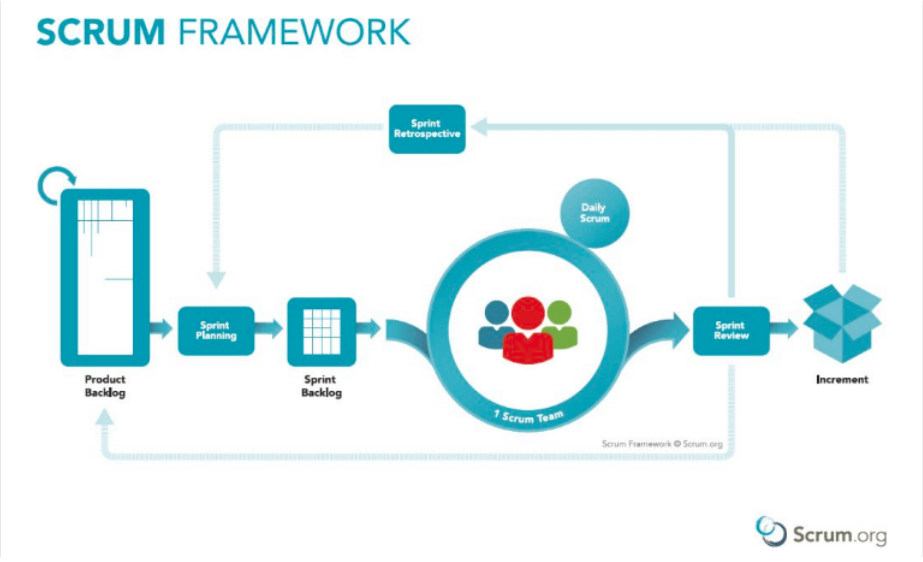
também multifuncionais, possuindo habilidades necessárias para desenvolver o produto.

- O Scrum master é o responsável para fazer a ponte entre o Scrum Owner e o time de desenvolvimento fazendo com que o projeto seja entendido e aplicado,também tem como função ajudar a eliminar qualquer impedimento que possa existir para que o prazo estabelecido seja cumprido em suma o Scrum master é servolíder para o **Time Scrum.**

No Scrum existe o sprint que é como o coração do método, podendo ter a duração de até um mês, durante o qual uma versão “pronta” de um software incremental é utilizável é criado, o novo sprint inicia após o finalização do sprint funcionando como um loop, já que scrum não é um sistema linear. O sprint pode ser cancelado apenas pelo product Owner, caso as condições e o rumo da tecnologia mudarem, geralmente o sprint deve ser cancelado dados as circunstâncias, mas devido o curto ciclo de vida de um sprint muito raramente é cancelado.

As reuniões rápidas denominadas de Daily Scrum, que devem ter uma curta duração, sendo feita para que o time por completo saiba em que ponto está o desenvolvimento, se existe algum impedimento e se terá algum atraso no prazo estipulado, o Scrum Master toma a liderança e efetua 3 perguntas: “O que foi feito ontem?”, “O que fará hoje?” e se ” Existe algum impedimento?”, com isso o time fica a par da situação do projeto.

Figura 1 – Método Scrum



Fonte: Voitto (Scrum Framework, 2020).

Mais informações referentes ao framework Scrum pode ser encontrado em (SUTHERLAND, 2013)

6 IMPLEMENTAÇÃO

O objetivo em questão foi desenvolver um software para a área da saúde (Hospitais) em que colaboradores tais como enfermeiros / médicos pudessem cadastrar os pacientes diagnosticados com o COVID-19. A equipe foi estruturada de forma a possibilitar maior integração e troca de informações para que o software fosse desenvolvido não somente de forma técnica mas contendo todos os requisitos importantes para que funcionasse corretamente registrando dados para o acompanhamento dos casos.

Na tabela abaixo, mostramos um exemplo de como o time foi estruturado considerando enfermeiros responsáveis por áreas diferentes dentro do hospital , para contribuir com informações sobre o cadastro de pacientes nos setores que atuam. O desenvolvedor que assume as atividades de programação e escrita do código, os analistas de sistemas fazendo o levantamento de requisitos e mapeamento do processo , analista de teste acompanhando os resultados e funcionamento do produto , o Product Owner e o Scrum Master.

Figura 2 – Team Master

Cargo	Responsabilidade	Área	Horas de atuação
1 Enfermeiro	Contribuir com informações para o registro de casos de pacientes com covid-19.	UTI , Clínica Médica e Pronto Socorro - (Apoio)	8 hs semanais
1 Desenvolvedor	Programar e escrever o código	Desenvolvimento	full time
2 Analistas de sistemas	Levantamento de requisitos mapeamento de processos .	Desenvolvimento	full time
1 Analista de Testes (QA)	Montar casos de testes exigidos, monitorar o processo de teste em detalhes e os resultados em cada ciclo e avaliar a qualidade geral do produto.	Desenvolvimento	full time
1 DBA	Gerenciar, configurar, atualizar e monitorar o sistema de banco de dados.	Desenvolvimento	full time
Product Owner:	Fazer a interfase com o time e TI , Contribuir com o backlog , define as sprints e faz alterações se necessário .	Desenvolvimento	full time
Scrum Master	Facilitador do Daily Scrum / Solucionador de problemas e ensinar cada etapa do scrum a equipe .	Desenvolvimento	full time

Fonte: Os autores

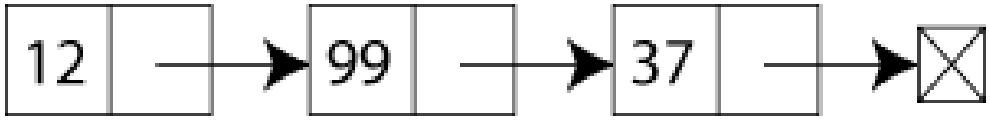
Agilidade adquirida com a implementação do Scrum contribuiu positivamente

para o desenvolvimento do software , respeitando a organização e o gerenciamento durante todo o processo de criação até a entrega final do produto funcionando. Portanto conclui-se que, o Scrum é um framework livre para organização de projetos podendo aliar-se com diversos processos de desenvolvimento, já que suas regras não são imutáveis sendo possível implementar apenas partes do Scrum e não o conjunto completo. E por ser simples e pequeno funciona bem em qualquer contexto, projeto ou produto , porém precisa ser implementado seguindo critérios de metodologias ágeis , considerando que o Scrum é um meio e não o fim, capaz de transformar o gerenciamento e o desenvolvimento de softwares.

7 ESTRUTURA DE DADOS

A estrutura de dados utilizada neste programa foi lista encadeada simples possui o campo e um específico que aponta para o próximo item da lista como mostra a figura abaixo:

Figura 3 – Lista Simplesmente Encadeada



Fonte: Os autores

Nesta figura é apresentado uma lista de inteiros decimais para o campo dedos que outro campo que aponta para o próximo item até chegar no final que aponta normalmente para nulo.

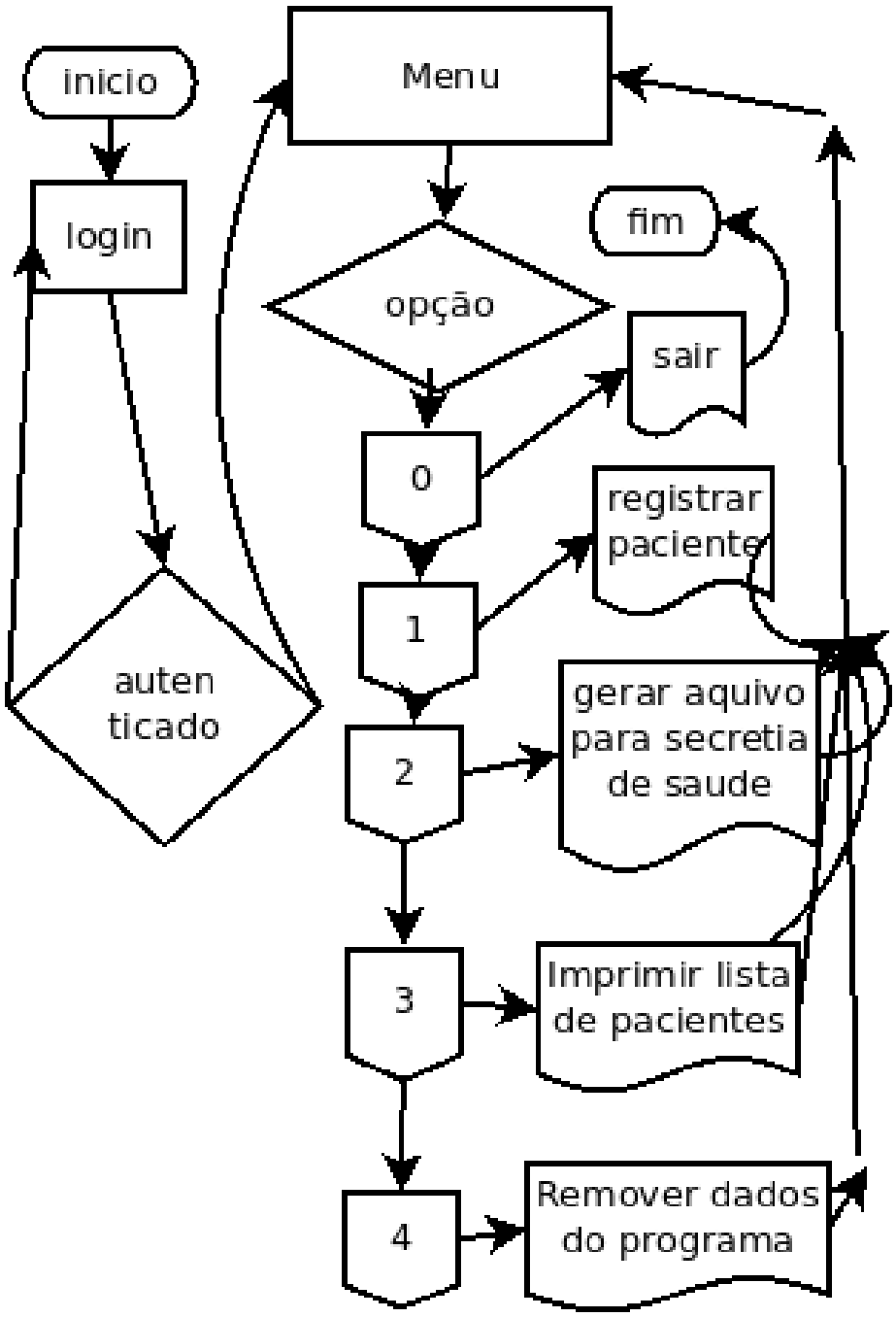
A escolha desta estrutura de dados para armazenar o cadastro de pacientes foi a simplicidade e facilidade de manipulação de dados, sendo apenas um pouco mais complexa que um vetor, porém mais simples que estruturas como árvore binária, e tendo a vantagem de ficar limitado apenas pela capacidade de hardware do computador que irá executar o software. Uma base é um computador de porte mediano pode armazenar informações de pacientes em memória cerca de 2361 pacientes diferentes sem problema algum de desempenho. E ocupando uma taxa irrisória de memória de memória não volátil, porém de ponto negativo ocupa em demasia memória volátil.

8 FLUXO

O fluxo de um programa é a parte crucial, é responsável por determinar para qual parte do binário deve ser executada. C por ser uma linguagem estruturada podemos

saltar para blocos de código sem muita dificuldade executado da forma que nos convém. Pelo programa ser feito por partes ou seja componentes as condicionais podem simplesmente chamar o bloco de código correspondendo ao que o usuário deseja. A figura abaixo explica o fluxo do software utilizando fluxograma:

Figura 4 – fluxograma



Fonte: Os autores

9 LEVANTAMENTO DE REQUISITOS

Levantamento de Requisitos é uma estratégia de desenvolvimento de software que funciona como ponto de partida para definir as etapas do desenvolvimento. Normalmente é criado uma lista de requisitos obrigatórios no qual é usado a sigla (RF) e para os requisitos obrigatórios e (RNF) para requisitos não obrigatórios. Sendo descritos todos os requisitos que o sistema deve possuir.

Neste projeto foi levantado os seguintes requisitos obrigatórios:

- Informações referentes aos pacientes devem ser salvos em arquivo de texto puro
- Sistema de autenticação
- Verificar se o paciente está no grupo de risco possuindo mais que 65 anos e comorbidades
- Gerar arquivo para secretaria da saúde com as informações de CEP e idade dos pacientes no grupo de risco
- O software deve ser feito seguindo o paradigma procedural estruturado

No qual serviu como fundação para o desenvolvimento do sistema e também foram feitos a lista de requisitos não funcionais (RNF) para serem desenvolvidos caso houve tempo no cronograma:

- biblioteca mínima para manipulação de datas brasileiras
- Evitar duplicatas de pacientes

10 COMPONETIZAÇÃO

Componentização é um método de desenvolvimento muito que tem fortes influências da filosofia Unix mais especificamente a frase Doug McIlroy em 1968 “Faça com que cada programa execute bem uma coisa. Para fazer um novo trabalho, crie de novo ao invés de complicar programas antigos adicionando novos "recursos".”(GANCARZ, 1995) . A mesma ideia pode ser utilizada, dentro do mesmo programa, não a motivas de adicionar mais recursos a uma função deixando ela complexa é melhor criar outra e então cada função faz apenas uma coisa e muito bem feito.

Tratando o sistema como pequenas partes que foram um todo é possível diminuir a complexidade do programa além de facilitar na distribuição de tarefas para cada desenvolvedor.Uma dificuldade encontrada por seguir este método foi que linguagem

de programação escolhida não tem pleno suporte a programação orientada a Objetos e também um dos requisitos do sistema foi seguir o paradigma estruturado.

11 FERRAMENTAS

11.1 MinGW Compilador

O MingW é projeto que oferece compiladores minimalista e eficazes para compilação de código nativa para plataforma microsoft Windows sem a dependência de quaisquer recursos oferecido pela Microsoft como a biblioteca de execução Microsoft C. A escolha do compilador MingGW para plataforma Microsoft Windows foi escolhida pela simples fato de suas licenças onde grande parte do código está sobre domínio público ou seja livre de direitos autorais e outras sobre licenças permissivas com BSD,MIT/X11 e outras licenças também muito permissivas.

11.2 Gdb

GDB abreviação de GNU Debugger é um programa que visa facilitar a depuração de códigos escritos em C e outras linguagem é distribuído pela projeto GNU, uma de suas funcionalidades é a possibilidade de “olhar” dentro de seu programa em tempo de execução e saber em que ponto ocorreu o problema, além da opção de desmontagem ou seja você pode visualizar o código de máquina que o compilador gera. Sendo de suma importância para realização do trabalho por facilitar encontrar pontos de falha de segmentação, saber valores de variáveis em tempo de execução, executar o programa linha a linha.

11.3 Git

Git é um software livre e open source amplamente utilizado para o versionamento de softwares por ser simples e eficiente e muito fácil de aprender. Foi desenvolvido inicialmente por Linus Torvalds para suprir a demanda de contribuições em seu kernel conhecido como Linux. Em suma o git grava pontos na história do programa podendo o usuário voltar para tais pontos a qualquer momento, também pode criar novos ramos para o desenvolvimento, possuindo também ferramentas de integração para tratamento de conflitos.

11.4 Neovim

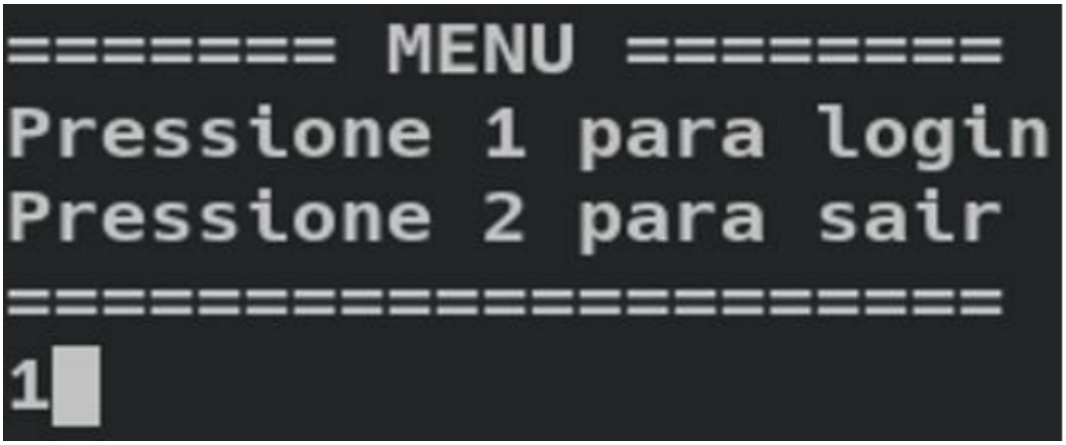
Para o desenvolvimento de uma programa em C é necessário um editor de texto puro, e a ferramenta escolhida para esta tarefa foi o Neovim que é um editor hiperextensíveis baseado no vim com suporte nativo a LSP(Language Server Protocol) no qual possibilidade verificar erros de sintaxe antes do processo de compilação, além de possuir uma licença GPLv3 e MIT para seu código, podendo o usuário da ferramenta ter uma melhor garantia de segurança e saber o'que o software faz em seu computador.

12 MANUAL DO USUÁRIO

12.1 Login

Para que o usuário tenha acesso às funcionalidades do programa antes ele precisa realizar a autenticação entrando com as informações de usuário e senha sendo ambos "admin" ou seja no campo usuário preencher com "admin" sem aspas e no campo senha preencher com "admin" sem aspas e após clicar na tecla enter. Primeiro o usuário tem que pressionar a tecla " 1 " do teclado para escolher fazer a autenticação e teclar enter como mostra a figura abaixo:

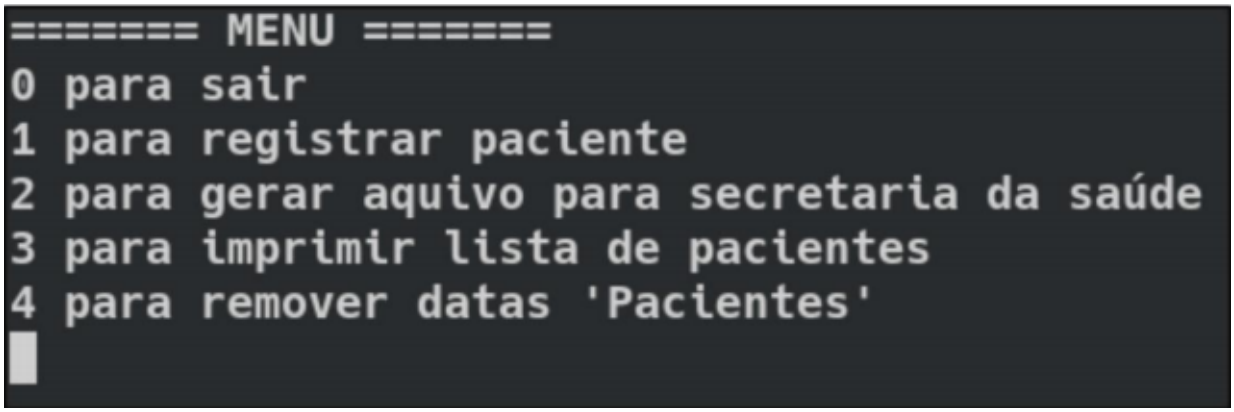
Figura 5 – Figura do Menu



Fonte: Os autores

Após a escolha da opção será redirecionado para o menu principal contendo as seguintes funcionalidades como mostra a figura abaixo, podendo escolher entre sair do programa, registrar novos pacientes, gerar arquivo para secretaria e imprimir lista de pacientes como na figura abaixo:

Figura 6 – Figura do Menu de opções



Fonte: Os autores

12.2 Registro de pacientes

Quando o usuário selecionar a opção número um para registrar um novo paciente na lista, será pedido para ele entrar com as informação de nome, cpf, telefone, endereço, data do diagnóstico, data de nascimento, email, cep e comorbidades, em seguida automaticamente o programa ia criar uma arquivo em texto puro contendo as informações do paciente junto a lista e caso algum paciente estiver no grupo de risco será criado o arquivo para ser enviado a secretaria da saúde

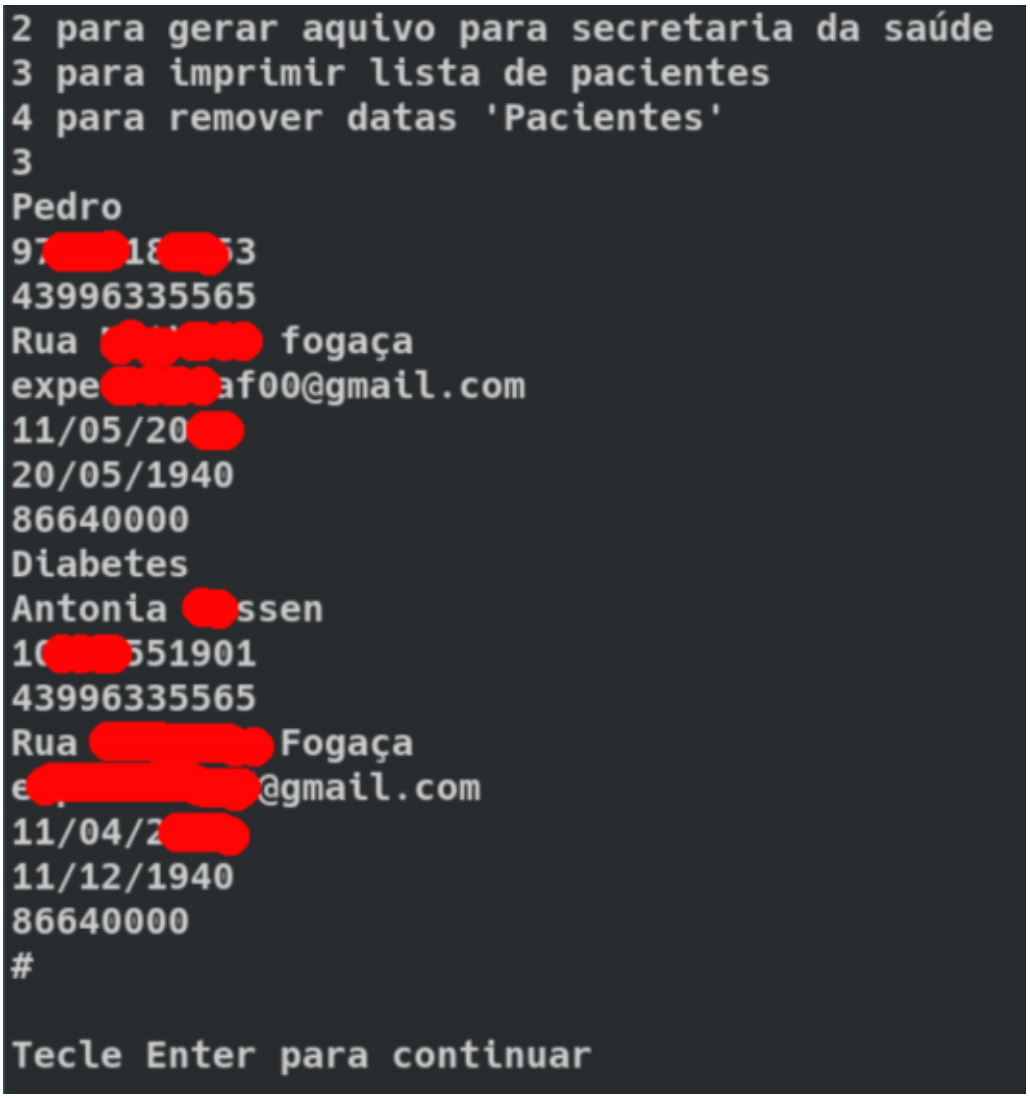
12.3 Imprimir lista de pacientes

Quando o usuário selecionar a opção número três será exposto na tela a lista de todos os pacientes já cadastrados. Na figura abaixo é mostrado um exemplo fictício de como ficaria.

12.4 Remover dados

Quando o usuário pressionar o tecla número quatro o programa vai pedir uma confirmação e caso o usuário digitar ‘S‘ maiúsculo o programa removerá os arquivos ‘data.txt‘ que possui informações referentes a todos os usuários já cadastrados, e o ‘secretaria.txt‘ que possui as informações de CEP e Idade dos pacientes do grupo de risco

Figura 7 – Figura da lista de pacientes cadastrados



Fonte: Os autores

13 CÓDIGO

O código foi desenvolvido objetivando princípios de simplicidade e funcionalidade única para manter um software simples, legível e fácil de manter e ao mesmo tempo compatível com diversas plataformas e podendo ser executado em quase todos os sistemas operacionais que possuem suporte a libc padrão. A fonte do software pode ser encontrado hospedado no github na qual é a maior plataforma de compartilhamento de código aberto do mundo, pode ser encontrado no endereço: <https://github.com/PedroExpedito/trabalho> ou baixar o arquivo em zip conforme as exigências da Unip em <https://codeload.github.com/PedroExpedito/trabalho/zip/main.>

14 CONCLUSÃO

O software desenvolvido neste projeto vai ajudar suprir a necessidade dos hospitais de cadastrar grande quantidade de dados de pacientes, realizando relatórios para enviar à secretaria da saúde municipal através de um arquivo de texto, além disso irá apoiar nas tomadas de decisões durante todo o processo de diagnóstico até o registro de casos, possibilitando a intensificação de ações para prevenção e combate ao Covid-19. O cadastro será feito de modo fácil e quase intuitivo, inclusive até pessoas com dificuldades para utilizarem recursos tecnológicos poderão efetuar o cadastro sem dificuldades.

Em parte técnica manteve um código legível de forma simplificada e compreensível, com poucos erros de fácil manutenção e sem dependências que dificultam a melhoria do software. Principal benefício de não possuir dependências de software não livre, ou seja, o programa está sobre os direitos do autor sem grandes interferências externas podendo ser alterado de maneira rápida.

A utilização do software ainda prevê otimizar processos longos de cadastros , reunindo apenas informações que sejam relevantes para melhor gerenciamento do números de casos , apontando pacientes que são do grupo de risco e se possuem patologias que elevam o grau de gravidade da doença , em quais regiões estes pacientes estão localizados ,fazendo com que seja possível mapear bairros , cidades e estados com maior número de infectados e possibilita também entender a capacidade mínima que os hospitais possuem para atendimento de pacientes com Covid-19.

O diferencial deste produto não é apenas por ser um software livre,simples, intuitivo , mas por ter sido construído com a finalidade de ser um facilitador e uma solução tecnológica fundamental para sistematizar um serviço imprescindível para a área da saúde, que em meio a todo esse caos vivenciado nos dias atuais , o uso de recursos e ferramentas inteligentes e seguras podem ser aliadas na gestão da saúde no brasil.

Referências

ABRAN, A.; MOORE, J. *Guide to the Software Engineering Body of Knowledge: Trial Version*. IEEE Computer Society, 2001. ISBN 9780769510002. Disponível em: <https://books.google.com.br/books?id=YqVQAAAAMAAJ>.

D'SOUZA, D. *Components and Frameworks with UML, The Catalysis Approach*. [S.l.: s.n.], 1999.

GANCARZ, M. *The UNIX Philosophy*. Elsevier Science, 1995. ISBN 9781555581237. Disponível em: <https://books.google.com.br/books?id=0Ktl1bB-vksC>.

JUANA-ESPINOSA, S. de. *Human Resource Management in the Digital Economy: Creating Synergy between Competency Models and Information: Creating Synergy between Competency Models and Information*. Information Science Reference, 2011. (Advances in Human Resources Management and Organizational Development (2327-3372)). ISBN 9781613502082. Disponível em: <https://books.google.com.br/books?id=wWYJx453Oy0C>.

KERZNER. *Fundamentos do gerenciamento de projetos*. Editora FGV, 2002. (FGV Management). ISBN 9788522515073. Disponível em: <https://books.google.com.br/books?id=_CmHCgAAQBAJ>.

PEDRICZ, P. *Engenharia De Software*. 1. ed. [S.l.]: Makron Books, 1995.

SOMMERVILLE, I. *Engenharia De Software*. PEARSON BRASIL, 2019. ISBN 9788543024974. Disponível em: <https://books.google.com.br/books?id=tfivwwEACAAJ>.

SUTHERLAND, K. S. e J. *Guia Do Scrum*. 2013. Scrumguides. Disponível em: <https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf>. Acesso em: 04 nov 2020.

SZYPERSKI, C. *omponent Software: Beyond Object-Oriented Programming*. 1. ed. [S.l.]: Makron Books, 2002.