Template para Definição de Requisitos Funcionais

Projeto: DONATION MANAGEMENT SYSTEM (DMS)

Data: 16/08/2024
Autor: Igor Dias Modesto

1. Introdução

1.1 Descrição Geral:

Um projeto que pensou em uma dificuldade social de armazenamento e administração de estoques de doações, utilizando um sistema que possa guardar as informaçoes de forma organizada e demonstrativas, ajudando a organizaçoes amelhorar o armazenamento de estoque.

1.2 Objetivos do Projeto:

Busca melhorar a forma de administrar estoque de doação, para que nao haja desvio e perca de informação, sendo utilizado de forma simplificada para uso geral de usuarios.

2. Lista de Stakeholders

Stakeholder 1: Alex Nogueira Nanni
Stakeholder 2: Francisco Meneguini

Stakeholder 3: Igor Dias Modesto
Stakeholder 4:Maria Luiza Melo

Stakeholder 5: Vitória Afonso

3. Requisitos Funcionais

3.1 Requisitos Funcionais Gerais: Eventos

ID	Eventos	Descrição	Prioridade
RF-01	"Produção do código principal em C"	Criação de um codigo que ajudara na organização e administração de um estoque.	Alta
RF-02	"Termo de abertura do projeto"	Criação de um documento que explica o projeto e demonstrar a sua função e de seus	Alta

		colaboradores.	
RF-03	"Produção do Fluxograma"	Produção de um fluxograma que mostracomo o codigo/sistema funciona.	Média
RF-04	"Estrutura Analítica do Projeto e Cronograma do Projeto"	Realização de uma EAP e de um cronograma para demonstrar o tempo de projeto e algumas prioridade.	Média

3.2 Requisitos Funcionais por Módulo:

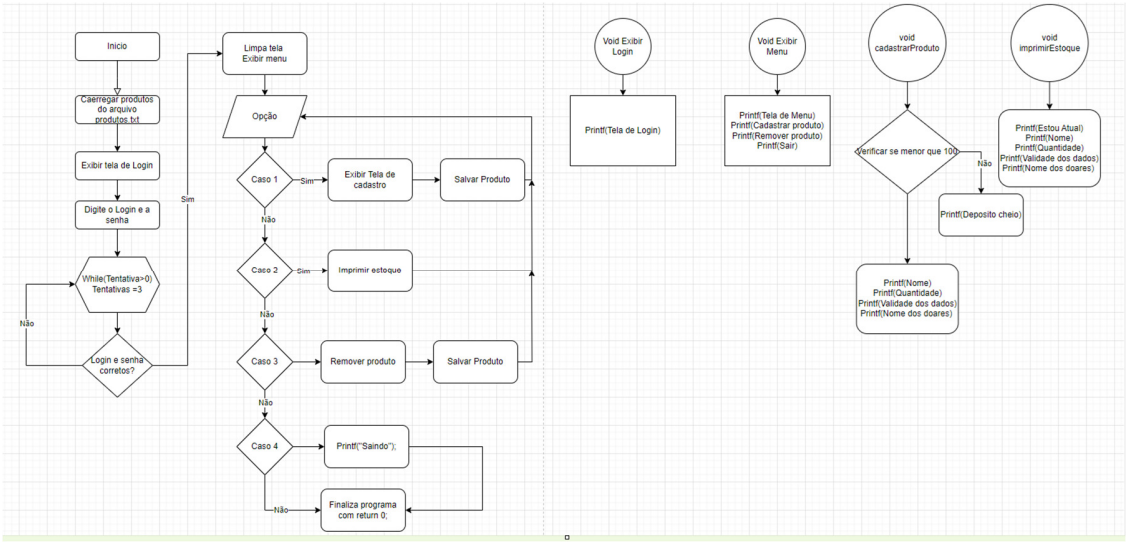
- **Módulo 1: Riscos

ID	Requisito FuncionalInsatisfação dos Usuários	Descrição	Prioridade
BF-01	Insatisfação dos Usuários	O usuário não aceitou o uso do programa e desistiu do sistema.	Baixa
BF-02	Falhas de Comunicação na Equipe	Problemas de comunicação podem causar atrasos e falhas.	Alta

4. Critérios de Aceitação

ID	Critérios de Aceitação
DF-01	Seguir o cronograma estabelecido, entregando conforme o cronograma e relatórios de progresso.
DF-02	Sistema funcional e intuitivo

5. Anexos




```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <locale.h>

// Estrutura para representar um produto
typedef struct {
    char nome[50];
    int quantidade;
    char dataValidade[11]; // DD/MM/AAAA
    char doador[50];
} Produto;

// Funções
void cadastrarProduto(Produto produtos[], int *numProdutos);
void imprimirEstoque(Produto produtos[], int numProdutos);
void salvarProdutos(Produto produtos[], int numProdutos);
void carregarProdutos(Produto produtos[], int *numProdutos);
void removerProduto(Produto produtos[], int *numProdutos);

void limparTela() {
    #ifdef _WIN32
        system("cls");
    #else
        system("clear");
    #endif
}

void exibirLogin() {
    printf("=======================================\n");
    printf("               TELA DE LOGIN           \n");
    printf("=======================================\n");
}

void exibirMenu() {
    printf("\n=======================================\n");
    printf("                MENU PRINCIPAL         \n");
    printf("=======================================\n");
    printf("1. Cadastrar produto\n");
    printf("2. Imprimir estoque\n");
    printf("3. Remover produto\n");
    printf("4. Sair\n");
    printf("Escolha uma opcao: ");
}
```

```c
int main() {
    Produto produtos[100]; // Array para armazenar os produtos
    int numProdutos = 0;
    int opcao;
    char usuario[20], senha[20];
    const char usuarioValido[] = "admin";
    const char senhaValida[] = "12345";
    int tentativas = 3;

    setlocale(LC_ALL, "Portuguese");

    // Carregar produtos do arquivo
    carregarProdutos(produtos, &numProdutos);

    while (tentativas > 0) {
        limparTela();
        exibirLogin();

        printf("Digite o nome de usuario: ");
        scanf("%19s", usuario); // Limita o tamanho da entrada
        printf("Digite a senha: ");
        scanf("%19s", senha); // Limita o tamanho da entrada

        if (strcmp(usuario, usuarioValido) == 0 && strcmp(senha, senhaValida) == 0) {
            printf("\nLogin bem-sucedido!\n");
            break;
        } else {
            tentativas--;
            if (tentativas > 0) {
                printf("\nERRO! A Senha/Login invalido.\n");
                printf("Voce tem %d tentativas restantes.\n", tentativas);
                #ifdef _WIN32
                    system("pause");
                #else
                    printf("Pressione Enter para continuar...");
                    getchar(); // Limpa o buffer do teclado
                    getchar(); // Aguarda a entrada do usuário
                #endif
            } else {
                printf("\nLimite de tentativas excedido.\n");
                #ifdef _WIN32
                    system("pause");
                #else
                    printf("Pressione Enter para continuar...");
                    getchar(); // Limpa o buffer do teclado
                    getchar(); // Aguarda a entrada do usuário
                #endif
            }
```

```
    }
}

if (tentativas == 0) {
    return 1; // Encerra o programa com erro
}

do {
    limparTela();
    exibirMenu();
    scanf("%d", &opcao);

    switch (opcao) {
        case 1:
            cadastrarProduto(produtos, &numProdutos);
            salvarProdutos(produtos, numProdutos); // Salvar após o cadastro
            #ifdef _WIN32
                system("pause");
            #else
                printf("Pressione Enter para continuar...");
                getchar(); // Limpa o buffer do teclado
                getchar(); // Aguarda a entrada do usuário
            #endif
            break;
        case 2:
            imprimirEstoque(produtos, numProdutos);
            #ifdef _WIN32
                system("pause");
            #else
                printf("Pressione Enter para continuar...");
                getchar(); // Limpa o buffer do teclado
                getchar(); // Aguarda a entrada do usuário
            #endif
            break;
        case 3:
            removerProduto(produtos, &numProdutos);
            salvarProdutos(produtos, numProdutos); // Salvar após a remoção
            #ifdef _WIN32
                system("pause");
            #else
                printf("Pressione Enter para continuar...");
                getchar(); // Limpa o buffer do teclado
                getchar(); // Aguarda a entrada do usuário
            #endif
            break;
```

```c
    do {
        switch (opcao) {
            case 4:
                break;
            default:
                printf("ERRO! Codigo inserido invalido.\n");
                #ifdef _WIN32
                    system("pause");
                #else
                    printf("Pressione Enter para continuar...");
                    getchar(); // Limpa o buffer do teclado
                    getchar(); // Aguarda a entrada do usuário
                #endif
                break;
        }
    } while (opcao != 4);

    return 0;
}

// Função para cadastrar um produto
void cadastrarProduto(Produto produtos[], int *numProdutos) {
    if (*numProdutos >= 100) {
        printf("Estoque cheio. Nao e possivel cadastrar mais produtos.\n");
        return;
    }

    Produto p;
    printf("Digite o nome do produto: ");
    scanf("%49s", p.nome);
    printf("Digite a quantidade: ");
    scanf("%d", &p.quantidade);
    printf("Digite a data de validade (DD/MM/AAAA): ");
    scanf("%10s", p.dataValidade);
    printf("Digite o nome do doador: ");
    scanf("%49s", p.doador);

    produtos[*numProdutos] = p;
    (*numProdutos)++;
}

// Função para imprimir o estoque
void imprimirEstoque(Produto produtos[], int numProdutos) {
    printf("=======================================\n");
    printf("              ESTOQUE ATUAL            \n");
    printf("=======================================\n");
    for (int i = 0; i < numProdutos; i++) {
        printf("Produto %d:\n", i + 1);
```

```c
// Função para salvar produtos em um arquivo
void salvarProdutos(Produto produtos[], int numProdutos) {
    FILE *file = fopen("produtos.txt", "w");
    if (file == NULL) {
        printf("Erro ao abrir o arquivo para escrita.\n");
        return;
    }

    for (int i = 0; i < numProdutos; i++) {
        fprintf(file, "%s,%d,%s,%s\n", produtos[i].nome, produtos[i].quantidade, produtos[i].dataValidade, produtos[i].doador);
    }

    fclose(file);
}

// Função para carregar produtos de um arquivo
void carregarProdutos(Produto produtos[], int *numProdutos) {
    FILE *file = fopen("produtos.txt", "r");
    if (file == NULL) {
        printf("Nenhum arquivo de produtos encontrado. Iniciando com estoque vazio.\n");
        return;
    }

    while (fscanf(file, "%49[^,],%d,%10[^,],%49[^\n]\n", produtos[*numProdutos].nome, &produtos[*numProdutos].quantidade, produtos[*numProdutos].dataValidade, produtos[*numProdutos].do
        (*numProdutos)++;
    }

    fclose(file);
}

// Função para remover uma quantidade específica de um produto
void removerProduto(Produto produtos[], int *numProdutos) {
    char nome[50];
```

```c
// Função para remover uma quantidade específica de um produto
void removerProduto(Produto produtos[], int *numProdutos) {
    char nome[50];
    int quantidadeRemover;
    int i;

    // Solicita o nome do produto a ser removido e a quantidade
    printf("Digite o nome do produto: ");
    scanf("%49s", nome);
    printf("Digite a quantidade a ser removida: ");
    scanf("%d", &quantidadeRemover);

    // Procura o produto pelo nome
    for (i = 0; i < *numProdutos; i++) {
        if (strcmp(produtos[i].nome, nome) == 0) {
            // Verifica se a quantidade a ser removida é maior do que a disponível
            if (quantidadeRemover > produtos[i].quantidade) {
                printf("Quantidade a ser removida é maior do que a disponível.\n");
                return;
            } else if (quantidadeRemover == produtos[i].quantidade) {
                // Remove o produto se a quantidade a ser removida for igual à disponível
                for (int j = i; j < *numProdutos - 1; j++) {
                    produtos[j] = produtos[j + 1];
                }
                (*numProdutos)--;  // Reduz o número de produtos
                printf("Produto removido com sucesso.\n");
                return;
            } else {
                // Reduz a quantidade do produto
                produtos[i].quantidade -= quantidadeRemover;
                printf("Quantidade removida com sucesso. Nova quantidade: %d\n", produtos[i].quantidade);
                return;
            }
        }
    }

    // Se o produto não for encontrado
    printf("Produto não encontrado.\n");
}
```