main.c

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

#include "baseDeDados.h" // Possui as structs com os dados do hotel
#include "textos.h"      // Possui os textos base do sistema
#include "admin.h"       // Possui a pagina do administrador
#include "menuHotel.h"   // Possui a pagina principal do hotel
#include "login.h"       // Possui as verificacoes de login e registro do sistema

// O FUNCIONAMENTO DO CODIGO APENAS PARA WINDOWS !!


int main()
{
   int s_adm=0;//situacao adm
   while(1){
      int opcaoAcesso;
      int opcaoInvalida = 0;

      do{
         system("cls");

         CABECALHO();
         if(opcaoInvalida){
            printf("\nOpcao invalida, por favor digite novamente:\n");
         }
         if(s_adm){
            printf("\nAdministrador nao encontrado.\n");
         }
         if(!opcaoInvalida && !s_adm){
            printf("\nEscolha seu nivel de acesso:\n");
         }
         printf("\n0 - Sair");
         printf("\n1 - Cliente");
         printf("\n2 - Administrador\n");
         FIM_CABECALHO();
         scanf("%d", &opcaoAcesso);
         getchar();//limpar buffer

         s_adm = 0;
         opcaoInvalida = 1;
      }while(opcaoAcesso > 2 || opcaoAcesso < 0);

      switch(opcaoAcesso){ // Verifica a opcao de acesso do usuario e o direciona a proxima parte.
```

```
            case 0:
                printf("Sistema encerrado.\n\n");
                return 0;

            case 1:
                CLIENTE();
                break;

            case 2:
                s_adm = ADMIN();
                break;

            default:
                printf("\nErro Switch - Acesso_Inicial - main.c.\n");
                system("pause");
        }
    }
}
```

baseDeDados.h

```
#ifndef BASEDEDADOS_H_INCLUDED
#define BASEDEDADOS_H_INCLUDED

#define N_AD 5 //quantidade de admins

char nomeHotel[200] = "HOTEL";    // Nome do hotel

struct CLIENTES
{
   char nomeCliente[100];
   int cpfCliente;
}cliente;

typedef struct
{
   int dia;
   int mes;
   int ano;
}Data;

struct ADMIN
{
   char nomeAdmin[N_AD][100];
   int idAdmin[N_AD];
}adm;

//Admins
struct ADMIN adm = {
   .nomeAdmin = {
      "Aguinelo",
      "Gustavo",
      "Rayssa",
      "Matheus",
      "Octavio"
   },
   .idAdmin = {
      12345,
      23456,
      34567,
      45678,
      56789
   }
};

#endif // BASEDEDADOS_H_INCLUDED
```

textos.h

```
#ifndef TEXTOS_H_INCLUDED
#define TEXTOS_H_INCLUDED


void CABECALHO()
{
   printf("\n");
   printf("-----------------------------------------------");
   printf("\n\t BEM VINDO AO HOTEL %s\n", nomeHotel);
   printf("-----------------------------------------------");
   printf("\n");
}

void ADM_CABECALHO()
{
   printf("\n-----------------------------------------------\n");
   printf("\n\t  PAINEL DO ADMINISTRADOR\n");
   printf("\n-----------------------------------------------\n");
}

void SOB_CABECALHO()
{
   printf("\n");
   printf("-----------------------------------------------");
   printf("\n\t\tHOTEL %s\n", nomeHotel);
   printf("-----------------------------------------------");
   printf("\n");
}

void FIM_CABECALHO()
{
   printf("\n");
   printf("-----------------------------------------------");
   printf("\n");
}


#endif // TEXTOS_H_INCLUDED
```

admin.h

```
#ifndef ADMIN_H_INCLUDED
#define ADMIN_H_INCLUDED


void VERIFICA_CLIENTES()
{

   FILE *clientes;

   clientes = fopen("arquivos/clientes.txt","r");
   printf("\n\n\tCLINTES CADASTRADOS\n");

   int cpfVerifica;
   char nomeVerifca[100],sobrenomeVerifica[100];

   while(fscanf(clientes, "%s %s %d",nomeVerifca, sobrenomeVerifica, &cpfVerifica) != EOF){
      printf("\nNome: %s %s\tCPF: %d\n", nomeVerifca, sobrenomeVerifica, cpfVerifica);

   }

   printf("\n");
   system("pause");
}

int VERIFICA_RESERVAS()
{

   FILE *reservas;

   reservas = fopen("arquivos/reservas.txt","r");

   if(reservas == NULL){
      return 1;// Erro ao abrir o arquivo
   }

   char nomeVerifca[100],sobrenomeVerifica[100];
   int dataE[3];
   int dataS[3];
   char bed[20];
   int quartoEscolhido;

   rewind(reservas);
   while(fscanf(reservas, "%s %s %d %d %d %d %d %d %s %d\n",nomeVerifca, sobrenomeVerifica,
   &dataE[0],&dataE[1],&dataE[2],
   &dataS[0],&dataS[1],&dataS[2], bed, &quartoEscolhido) != EOF){
```

```
    printf("\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");
    printf("\n- Nome: %s %s.", nomeVerifca, sobrenomeVerifica);
    printf("\n- Data de entrada: %d/%d/%d.", dataE[0], dataE[1], dataE[2]);
    printf("\n- Data de saida: %d/%d/%d.", dataS[0], dataS[1], dataS[2]);
    printf("\n- Cama escolhida: %s.", bed);
    printf("\n- Numero do quarto: %d.", quartoEscolhido);
    printf("\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");
    }

    fclose(reservas);
    system("pause");

    return 0;
}

int PRECOS_QUARTOS()
{

    int cnt = 0, cama;
    char nomeCama[20];
    char camaEscolhida[20];
    float valorCama, newValue;
    char* camasOpcoes[3] = {"Solteiro","Casal","QueenSize"};
    float valores[3];


    FILE *camas;
    camas = fopen("arquivos/camasValor.txt","r");

    if(camas == NULL){
        return 1;
    }


    while (fscanf(camas,"%s %f",nomeCama, &valorCama) != EOF) {
        printf("%d - Quarto com Cama de %s R$: %.2f\n",cnt + 1, nomeCama, valorCama);
        valores[cnt] = valorCama;
        cnt++;
    }

    fclose(camas);

    //printf("\n\ALTERACAO DE VALOR DOS QUARTOS\n");

    int opcaoInvalida = 0;

    do{
```

```
    system("cls");

    ADM_CABECALHO();
    if(opcaoInvalida){
        printf("\nOpcao invalida, por favor digite novamente:\n");
    }
    if(!opcaoInvalida){
        printf("\nQual cama deseja alterar?\n");
    }
    printf("\n1 - Solteiro");
    printf("\n2 - Casal");
    printf("\n3 - Queen size\n");
    printf("\n0 - Voltar");
    FIM_CABECALHO();
    scanf("%d", &cama);
    getchar();

    opcaoInvalida = 1;
}while(cama > 3 || cama < 0);

switch(cama){
    case 0:
        return 0;

    case 1: // Cama de solteiro
        printf("\nDigite o novo valor do quarto com cama de Solteiro: ");
        scanf("%f", &newValue);
        getchar();

        strcpy(camaEscolhida, "Solteiro");
        break;

    case 2: // Cama de casal
        printf("\nDigite o novo valor do quarto com cama de Casal: ");
        scanf("%f", &newValue);
        getchar();

        strcpy(camaEscolhida, "Casal");
        break;

    case 3: // Cama queen size
        printf("\nDigite o novo valor do quarto com cama Queen size: ");
        scanf("%f", &newValue);
        getchar();

        strcpy(camaEscolhida, "QueenSize");
        break;
```

```
        default:
            printf("\nErro Switch - Precos_Quartos - admin.h");
            system("pause");

    }

    camas = fopen("arquivos/camasValor.txt", "w");

    if (camas == NULL) {
        printf("Erro ao abrir o arquivo para escrita.\n");
        return 1;
    }

    for(int i = 0; i < 3; i++){
        if (strcmp(camasOpcoes[i], camaEscolhida) == 0) {
            valores[i] = newValue;
        }

        fprintf(camas,"%s %.2f\n",camasOpcoes[i], valores[i]);
    }

    fclose(camas);

    //if(cama != 0)
    return 0;
}


int INFRACAO()
{

    int verific = 0;
    float multa;
    char nome[30], infracao[100];
    int cpf;

    FILE *clientes;

    FILE *clienteMultado;

    clientes = fopen("arquivos/clientes.txt", "r");

    if(clientes == NULL){
        return 1;
    }

    printf("Digite o cpf do infrator: ");
```

```
scanf("%d",&cpf);  //tirando o \n

int cpfVerifica;
char nomeVerifca[100],sobrenomeVerifica[100];

while (fscanf(clientes, "%s %s %d",nomeVerifca, sobrenomeVerifica, &cpfVerifica) != EOF) {
   char nomeCompleto[100];
   sprintf(nomeCompleto, "%s %s", nomeVerifca, sobrenomeVerifica);

   if(cpfVerifica == cpf) {
      strcpy(nome, nomeCompleto);
      verific = 1;

   }

}

clienteMultado = fopen("arquivos/clienteMultado.txt", "a");

if(clienteMultado == NULL){
   fclose(clientes);
   return 1;
}

if(verific == 1){

      system("cls");

      ADM_CABECALHO();

      printf("Digite qual a infracao do hospede (maximo de 100 caracteres): ");
      getchar();
      fgets(infracao, sizeof(infracao), stdin);
      infracao[strcspn(infracao, "\n")] = '\0';
      getchar();

      fprintf(clienteMultado, "CPF: %d\nNome: %s\nInfrao: %s\n\n", cpf, nome, infracao);
      printf("Dados do infrator registrados.\n");

   int tipo_multa;
   int opcaoInvalida = 0;



}else{
   printf("Cliente no encontrado, tente novamente.\n");
}
```

```
    fclose(clientes);
    fclose(clienteMultado);
    return 0;
}

int ALTERAR_INFOS()
{
    while(1){
        int opcaoAcesso;
        int opcaoInvalida = 0;

        do{
            system("cls");

            ADM_CABECALHO();
            if(opcaoInvalida){
                printf("\nOpcao invalida, por favor digite novamente:\n");
            }
            if(!opcaoInvalida){
                printf("\nQual informacao gostaria de alterar?\n");
            }
            printf("\n1 - Alterar nome do hotel");
            printf("\n2 - Alterar valores");
            printf("\n0 - Voltar\n");
            FIM_CABECALHO();
            scanf("%d", &opcaoAcesso);
            getchar();

            opcaoInvalida = 1;
        }while(opcaoAcesso > 2 || opcaoAcesso < 0);

        switch(opcaoAcesso){
            case 0:
                return 0; // Retorna 0 ao menu anterior

            case 1: //Altera o nome do hotel
                printf("\nQual o novo nome do hotel? ");

                getchar();
                fgets(nomeHotel, sizeof(nomeHotel), stdin);
                nomeHotel[strcspn(nomeHotel, "\n")] = '\0';
                getchar();

                printf("\nTroca bem sucedida!\n");
                system("pause");
                break;
```

```
    case 2: // Altera os valores do quartos, camas e servicos
        while(1){
            int opcPrecos;
            int opcInvalida = 0;
            int precos_quart = 0;

            do{
                system("cls");

                ADM_CABECALHO();
                if(opcInvalida){
                    printf("\nOpcao invalida, por favor digite novamente:\n");
                }
                if(precos_quart){
                    printf("\nErro ao abrir o arquivo.");
                }
                if(!opcInvalida){
                    printf("\nQual valor deseja alterar\n"); // TEM QUE ADICIONAR MAIS
                }
                printf("\n1 - Valores dos quartos por tipo de cama");
                printf("\n0 - Voltar\n");
                FIM_CABECALHO();
                scanf("%d", &opcPrecos);
                getchar();

                precos_quart = 0;
                opcInvalida = 1;
            }while(opcPrecos > 2 || opcPrecos < 0);

            switch (opcPrecos){
                case 0:
                    return 0; //Retorna 0 ao menu anterior

                case 1: //Altera os precos dos quartos
                    precos_quart = PRECOS_QUARTOS();
                    break;

                default:
                    printf("\nErro Switch2 - Altera_Infos - admin.h.\n");
                    system("pause");
            }
        }
        break;

    default:
        printf("\nErro Switch1 - Altera_Infos - admin.h.\n");
        system("pause");
}
```

```
    }
}

int VERIFICA_ADM(int id)
{
    for(int i=0; i<N_AD; i++){
        if(id == adm.idAdmin[i])
            return i;
    }
    return -1;
}

int PAINEL_ADM(int id_user)
{
    char nomeAdm[100] = "";

    strcpy(nomeAdm, adm.nomeAdmin[id_user]); // Pega o nome do admin logado

    while(1){
        int opcaoAcesso;
        int opcaoInvalida = 0;
        int verif_reserva = 0;
        int infr = 0;

        do{
            system("cls");

            ADM_CABECALHO();
            printf("\nSeja bem vindo %s.\n", nomeAdm);
            if(opcaoInvalida){
                printf("\nOpcao invalida, por favor digite novamente:\n");
            }
            if(verif_reserva || infr){
                printf("Erro ao abrir o arquivo.");
            }
            if(!opcaoInvalida && !verif_reserva && !infr){
                printf("\nSelecione a opcao desejada:\n");
            }
            printf("\n1 - Verificar clientes cadastrados");
            printf("\n2 - Verificar reservas");
            printf("\n3 - Informar infracao");
            printf("\n4 - Alterar informacoes do hotel");
            printf("\n0 - Voltar\n");
            FIM_CABECALHO();
            scanf("%d", &opcaoAcesso);
            getchar();

            infr = 0;
```

```
            verif_reserva = 0;
            opcaoInvalida = 1;
        }while(opcaoAcesso > 4 || opcaoAcesso < 0);

        switch(opcaoAcesso){
            case 0:
                return 0;// Retorna 0 ao menu anterior

            case 1:
                VERIFICA_CLIENTES(); // Exibe os clientes cadastrados e gera extrato
                break;

            case 2:
                verif_reserva = VERIFICA_RESERVAS(); // Exibe as reservas feitas e gera extrato
                // Falta fazer a funcao de alterar/cancelar as reservas
                break;

            case 3: // Falta fazer a funcao de multa
                infr = INFRACAO();
                break;

            case 4:
                ALTERAR_INFOS();  // Funcao para alterar os valores e informacoes do hotel
                break;

            default:
                printf("\nErro Switch - Painel_Adm - admin.h");
                system("pause");
        }
    }
}

#endif // ADMIN_H_INCLUDED
```

menuHotel.h

```
#ifndef MENUHOTEL_H_INCLUDED
#define MENUHOTEL_H_INCLUDED


Data CONVERTER_DATA( char *data)
{
   Data dataConvertida;

   sscanf(data, "%2d/%2d/%4d", &dataConvertida.dia, &dataConvertida.mes, &dataConvertida.ano);
   return dataConvertida;
}

int PRECO_CAMAS()
{
   FILE *camas;
   char nomeCama[50];
   float valorCama;
   int cnt = 0;

   camas = fopen("arquivos/camasValor.txt","r");

   if(camas == NULL){
      printf("Erro ao abrir o arquivo");
      return 0;
   }

   printf("\n");

   while (fscanf(camas,"%s %f",nomeCama, &valorCama) != EOF) {
      printf("%d - Quarto com Cama de %s R$: %.2f\n",cnt + 1, nomeCama, valorCama);
      cnt++;
   }

   fclose(camas);
   return 0;
}

int CONFERIR_QUARTOS(char camaOption[])
{

   FILE *quartos = fopen("arquivos/quartosDisponiveis.txt", "r");

   if (!quartos) {
      printf("\nErro no sistema. Por favor, tente novamente em outra hora.\n");
```

```
    return -1;
}

FILE *tempFile = fopen("arquivos/temp.txt", "w");
if (!tempFile) {
    fclose(quartos);
    printf("\nErro ao criar o arquivo temporario.\n");
    return -1;
}

char tipoQ[15];         // Inicializando a variavel
int restantes;
int encontrou = 0;
int quartoEscolhido = 0;

while (fscanf(quartos, "%s %d", tipoQ, &restantes) != EOF) {
    if (strcmp(camaOption, tipoQ) == 0) {
        encontrou = 1;

        if ((strcmp(tipoQ, "Solteiro") == 0 && restantes < 40) ||
            (strcmp(tipoQ, "Casal") == 0 && restantes < 80) ||
            (strcmp(tipoQ, "QueenSize") == 0 && restantes < 100)) {
            restantes++;

            quartoEscolhido = restantes;

            fprintf(tempFile, "%s %d\n", tipoQ, restantes);

        } else {
            printf("Erro: Nao ha mais quartos disponiveis para '%s'.\n", tipoQ);
            fclose(quartos);
            fclose(tempFile);
            remove("arquivos/temp.txt");
            return -1;
        }
    } else {
        fprintf(tempFile, "%s %d\n", tipoQ, restantes);
    }
}

fclose(quartos);
fclose(tempFile);

if (!encontrou) {
    printf("Tipo de cama nao encontrado.\n");
    return -1;
}
```

```
    remove("arquivos/quartosDisponiveis.txt");
    rename("arquivos/temp.txt", "arquivos/quartosDisponiveis.txt");

    return quartoEscolhido;
}

int FAZER_RESERVA(char nome[], char entrada[], char saida[], int cama)
{
    FILE *reservas;
    reservas = fopen("arquivos/reservas.txt", "r");

    char camaOption[50];

    if(cama == 1){
        strcpy(camaOption, "Solteiro");

    }else if(cama == 2){
        strcpy(camaOption, "Casal");

    }else if(cama == 3){
        strcpy(camaOption, "QueenSize");
    }

    int disponibilidade = CONFERIR_QUARTOS(camaOption);

    if(disponibilidade == -1){
        return -1;
    }


    Data dataEntrada = CONVERTER_DATA(entrada);
    Data dataSaida = CONVERTER_DATA(saida);



    if (reservas == NULL) {
        printf("\nErro ao abrir o arquivo.");
        return -1;
    }

    char nomeVerifca[100];
    char sobrenomeVerifica[100];

    int dataE[3];
    int dataS[3];
    int bed;
    int quartoEscolhido;
```

```
    while (fscanf(reservas, "%s %s %d %d %d %d %d %d %s %d",nomeVerifca, sobrenomeVerifica,
    &dataE[0],&dataE[1],&dataE[2],
    &dataS[0],&dataS[1],&dataS[2], &bed, &quartoEscolhido) != EOF) {

        char nomeCompleto[100];
        sprintf(nomeCompleto, "%s %s", nomeVerifca, sobrenomeVerifica);

        if(strcmp(cliente.nomeCliente, nomeCompleto) == 0) {
            printf("\nVoce ja tem uma reserva cadastrada, nao e possivel ter mais de uma.");
            return -1;
        }
    }

    fclose(reservas);

    if(dataEntrada.mes > 12 || dataSaida.mes > 12 ||
       dataEntrada.dia > 31 || dataSaida.dia > 31 ||
       dataEntrada.ano < 2024 || dataSaida.ano < 2024){
        printf("\nColoque uma data valida para o cadastro da reserva.");
        return -1;
    }

    reservas = fopen("arquivos/reservas.txt", "a");

    if (reservas == NULL) {
        printf("Erro ao abrir o arquivo.\n");
        return -1;
    }

    char nomeV[50], sobrenomeV[50];
    char *token = strtok(nome, " ");

    strcpy(nomeV, token);

    token = strtok(NULL, " ");
    strcpy(sobrenomeV, token);


    fprintf(reservas, "%s %s %d %d %d %d %d %d %s %d\n",nomeV, sobrenomeV,
    dataEntrada.dia,dataEntrada.mes,dataEntrada.ano,
    dataSaida.dia,dataSaida.mes,dataSaida.ano, camaOption,disponibilidade);


    fclose(reservas);

    return disponibilidade;
}
```

```
int CONSULTA_RESERVA(char nome[])
{
   FILE *reservas;

   reservas = fopen("arquivos/reservas.txt", "r");
   if(reservas == NULL){
      return 1;
   }

   char nomeVerifica[100], sobrenomeVerifica[100];
   int dataE[3], dataS[3];
   char bed[20];
   int quartoEscolhido;

   while(fscanf(reservas, "%s %s %d %d %d %d %d %d %s %d\n",
            nomeVerifica, sobrenomeVerifica,
            &dataE[0], &dataE[1], &dataE[2],
            &dataS[0], &dataS[1], &dataS[2],
            bed, &quartoEscolhido) != EOF)
   {
      char nomeCompleto[100];
      sprintf(nomeCompleto, "%s %s", nomeVerifica, sobrenomeVerifica);
      if(strcmp(cliente.nomeCliente, nomeCompleto) == 0) { // Se o nome for encontrado
         printf("\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");
         printf("\tINFORMACOES DE SUA RESERVA\n");
         printf("\n- Nome: %s %s", nomeVerifica, sobrenomeVerifica);
         printf("\n- Data de entrada: %d/%d/%d.", dataE[0], dataE[1], dataE[2]);
         printf("\n- Data de saida: %d/%d/%d.", dataS[0], dataS[1], dataS[2]);
         printf("\n- Cama escolhida: %s \n- Numero do quarto: %d\n", bed, quartoEscolhido);
         printf("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n");
         fclose(reservas);
         system("pause");
         return 0;
      }
   }

   fclose(reservas);
   // Se o nome não for encontrado após o término do loop
   return 2;
}


int CANCELAR_RESERVA()
{

   FILE *reservas = fopen("arquivos/reservas.txt", "r");
```

```
    FILE *temp = fopen("arquivos/temp_reservas.txt", "w");

    if (reservas == NULL || temp == NULL) {
        if (reservas) fclose(reservas);
        if (temp) fclose(temp);
        return 1;// erro arquivo
    }

    char nomeVerifica[50], sobrenomeVerifica[50];
    int dataE[3], dataS[3], bed, quartoEscolhido;
    int reservaEncontrada = 0;

    while (fscanf(reservas, "%s %s %d %d %d %d %d %d %s %d", nomeVerifica, sobrenomeVerifica,
           &dataE[0], &dataE[1], &dataE[2],
           &dataS[0], &dataS[1], &dataS[2],
           &bed, &quartoEscolhido) != EOF) {

        char nomeCompleto[100];
        sprintf(nomeCompleto, "%s %s", nomeVerifica, sobrenomeVerifica);

        if (strcmp(cliente.nomeCliente, nomeCompleto) == 0) {
            printf("Reserva de %s removida.\n", nomeCompleto);
            reservaEncontrada = 1;
        } else {
            fprintf(temp, "%s %s %d %d %d %d %d %d %s %d\n",
                    nomeVerifica, sobrenomeVerifica,
                    dataE[0], dataE[1], dataE[2],
                    dataS[0], dataS[1], dataS[2],
                    bed, quartoEscolhido);
        }
    }

    fclose(reservas);
    fclose(temp);

    if (!reservaEncontrada) {
        remove("arquivos/temp_reservas.txt");
        return 2; // reserva nao encontrada
    }

    remove("arquivos/reservas.txt");
    rename("arquivos/temp_reservas.txt", "arquivos/reservas.txt");

    return 0; // retorna ao menu
}


int MENU_HOTEL(char nome[])
```

```
{
   while (1){
      int opcaoAcesso;
      int opcaoInvalida = 0;
      int consult_reserva = 0;
      int can_reserva = 0;

      do{
         system("cls");

         SOB_CABECALHO();
         printf("\nSeja bem vindo %s.\n", nome);
         if(opcaoInvalida){
            printf("\nOpcao invalida, por favor digite novamente:\n");
         }
         if(consult_reserva == 1 || can_reserva == 1){
            printf("Erro ao abrir o arquivo.");
         }
         if(consult_reserva == 2){
            printf("\nVoce ainda nao fez nenhuma reserva!\n");
         }
         if(can_reserva == 1){
            printf("\nReserva cancelada com sucesso! \n");
         }
         if(!opcaoInvalida && !consult_reserva && !can_reserva){
            printf("\nMenu: \n");
         }
         printf("\n1 - Solicitar reserva");
         printf("\n2 - Consultar Reserva");
         printf("\n3 - Cancelar reserva");
         printf("\n0 - Sair\n");
         FIM_CABECALHO();
         scanf("%d", &opcaoAcesso);
         getchar();//limpar buffer

         can_reserva = 0;
         consult_reserva = 0;
         opcaoInvalida = 1;
      }while(opcaoAcesso > 3 || opcaoAcesso < 0);

      switch(opcaoAcesso){
         case 0:
            return 0;  //Retorna 0 ao menu anterior

         case 1:
            system("cls");

            int n=1;
```

```
    char dtEntrada[11], dtSaida[11];
    int tipoCama=0, numQuarto;

    do{
       system("cls");

       printf("\n--------------------------------------\n");
       printf("\n\tCADASTRO DE RESERVA\n");
       printf("\n--------------------------------------\n");

       if(n){
          printf("\nQual a data de entrada? (Formato: DD/MM/AA) ");
          fgets(dtEntrada, sizeof(dtEntrada), stdin);
          dtEntrada[strcspn(dtEntrada, "\n")] = '\0';
          getchar();

          printf("\nQual a data de saida? (Formato: DD/MM/AA) ");

          fgets(dtSaida, sizeof(dtSaida), stdin);
          dtSaida[strcspn(dtSaida, "\n")] = '\0';
          n = 0;
          getchar();
       }else{
          printf("\nQual o tipo de quarto desejado? ");
          PRECO_CAMAS();
          scanf("%d", &tipoCama);
          getchar();
       }

    }while(tipoCama > 3 || tipoCama < 1);

    printf("\nVerificando disponibilidade...\n");

    numQuarto = FAZER_RESERVA(nome, dtEntrada, dtSaida, tipoCama);

    if(numQuarto != -1){
       printf("\nReserva efetuada com sucesso!\nSeu quarto e o numero %d.", numQuarto);
       system("pause");
    }else{
       printf("\nNao foi possivel fazer a reserva. ");
       system("pause");
    }
    break;

case 2:
    consult_reserva = CONSULTA_RESERVA(nome); // Consulta a reserva
    break;
```

```
            case 3:
                can_reserva = CANCELAR_RESERVA(); // Cancela a reserva
                if(can_reserva == 1){printf("CANCELADO");}

                /*
                printf("\nDeseja emitir o extrato de cancelamento da reserva do quarto? (s/n) ");
                char emitirExtratoCancelamento;
                getchar();
                scanf("%c", &emitirExtratoCancelamento);

                if(emitirExtratoCancelamento == 's' || emitirExtratoCancelamento == 'S'){
                 //EXTRATO_CANCELAMENTO();     // Emite o extrato (arquivo) do cancelamento da
reserva quarto
                }
                */
                break;

            default:
                printf("\nErro Switch - Menu_Hotel - menuHotel.h.");
                system("pause");
        }
    }
}

#endif // MENUHOTEL_H_INCLUDED
```

login.h

```
#ifndef LOGIN_H_INCLUDED
#define LOGIN_H_INCLUDED

int ENTRAR(char nome[],int cpf)
{
   FILE *clientes;
   clientes = fopen("arquivos/clientes.txt","r");

   if(clientes == NULL){
      printf("Erro ao abrir o arquivo\n");
      system("pause");
      return 0;
   }

   int cpfVerifica;
   char nomeVerifca[100],sobrenomeVerifica[100];

   while (fscanf(clientes, "%s %s %d",nomeVerifca, sobrenomeVerifica, &cpfVerifica) != EOF) {
      char nomeCompleto[100];
      sprintf(nomeCompleto, "%s %s", nomeVerifca, sobrenomeVerifica);

      if(cpfVerifica == cpf && strcmp(nome, nomeCompleto) == 0) {
         cliente.cpfCliente = cpf;   // GUARDA OS DADOS DE ACESSO CASO PRECISE
FUTURAMENTE
         strcpy(cliente.nomeCliente, nome);
         fclose(clientes);
         return 2;
      }

   }

   fclose(clientes);

   return 1; // cpf nao encontrado
}

int CADASTRO(char nome[], int cpf)
{
   int contador = 0;
   char linha[200];
   FILE *clientes;

   clientes = fopen("arquivos/clientes.txt","r+");
```

```
   if(clientes == NULL){
      printf("ERRO INESPERADO POR PARTE DO SERVIDOR, POR FAVOR TENTE
NOVAMENTE UMA OUTRA HORA");
      return 1;
   }

   while (fgets(linha, sizeof(linha), clientes) != NULL) {
      contador++;
   }

   fclose(clientes);
   if(contador >= 100){
      printf("LIMITE DE USUARIOS CADASTRADOS ALCANCADO(100).");
      return 1;
   }else{
      if(strlen(nome) > 0){ // Verifica  se o nome nao esta vazio
         clientes = fopen("arquivos/clientes.txt","a");

         if(clientes == NULL){
            printf("ERRO INESPERADO POR PARTE DO SERVIDOR, POR FAVOR TENTE
NOVAMENTE UMA OUTRA HORA");
            return 1;
         }

         fprintf(clientes,"%s %d\n",nome,cpf);
         fclose(clientes);

         printf("\nCliente cadastrado com sucesso!\n");

         cliente.cpfCliente = cpf;   // GUARDA OS DADOS DE ACESSO CASO PRECISE
FUTURAMENTE
         strcpy(cliente.nomeCliente, nome);
         MENU_HOTEL(nome);   // Envia o cliente para a pagina principal

      }
   }

   fclose(clientes);
   return 0;
}

int CLIENTE()
{
   int s_cad = 0;// situacao cadastro
   int lg_cliente = 0;

   while (1){
```

```
int cpf;
char nome[100];
int opcaoAcesso;
int opcaoInvalida = 0;

do{
   system("cls");

   SOB_CABECALHO();
   printf("\nSeja bem vindo querido Cliente!\n");
   if(opcaoInvalida){
      printf("\nOpcao invalida, por favor digite novamente:");
   }
   if(lg_cliente == 1){
      printf("\nLogin invalido.\n");
   }
   if(s_cad){
      printf("\nErro ao executar cadastro.");
   }
   if(!opcaoInvalida && !s_cad && !lg_cliente){
      printf("\nPor favor, selecione sua opcao de entrada:");
   }
   printf("\n1 - Entrar");
   printf("\n2 - Cadastro");
   printf("\n0 - Voltar\n");
   FIM_CABECALHO();
   scanf("%d", &opcaoAcesso);
   getchar();//limpar buffer

   s_cad = 0;
   lg_cliente = 0;
   opcaoInvalida = 1;
}while(opcaoAcesso > 2 || opcaoAcesso < 0);

switch(opcaoAcesso){
   case 0:
      return 0; // retorna para a pagina inicial

   case 1:
      system("cls");

      printf("\n-----------------------------------------\n");
      printf("\t\tLOGIN DE CLIENTE");
      printf("\n-----------------------------------------\n");
      printf("\nDigite seu nome e sobrenome: ");

      fgets(nome, sizeof(nome), stdin);
      nome[strcspn(nome, "\n")] = '\0';
```

```
            printf("\nDigite seu cpf(Apenas numeros!): ");
            scanf("%d", &cpf);
            getchar();
            FIM_CABECALHO();

            if(ENTRAR(nome,cpf) == 2){
               printf("\nLogin bem sucedido!\n");
               system("pause");
               MENU_HOTEL(nome); // Envia o cliente para o menu principal
            }else{
               lg_cliente = 1; //login invalido
            }
            break;

         case 2:
            system("cls");

            printf("\n-----------------------------------------------\n");
            printf("\t\t\bCADASTRO DE CLIENTE");
            printf("\n-----------------------------------------------\n");
            printf("\nDigite seu nome e sobrenome: ");

            fgets(nome, sizeof(nome), stdin);
            nome[strcspn(nome, "\n")] = '\0';
            getchar();

            printf("\nDigite seu cpf(Apenas numeros!): ");
            scanf("%d", &cpf);
            getchar();
            FIM_CABECALHO();

            s_cad = CADASTRO(nome, cpf); // Faz a insercao na base de dados
            break;

         default:
            printf("\nErro swicth - Cliente - login.h.");
            system("pause");
      }
   }
}

int ADMIN()
{
   int id_user;

   printf("Por favor insira a sua senha de acesso: ");
   scanf("%d", &id_user);
```

```
    getchar();

    id_user = VERIFICA_ADM(id_user);// Verifica senha

    if(id_user != -1){
        PAINEL_ADM(id_user); // Envia o adm para sua pagina principal
    }else{
        return 1; // senha errada
    }
    return 0; // Retorna 0 ao menu anterior
}



#endif // LOGIN_H_INCLUDED
```

clientes.txt

Matheus Felipe 6421908
Aguinelo P 6421300
Felipe Souza 123123
TESTE TESTE 123456

quartosDisponiveis.txt

Solteiro 1
Casal 40
QueenSize 80

camasValor.txt

Solteiro 50.00
Casal 120.00
QueenSize 500.00