“Main.c”

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>
#include "functions\prima_pagina.c"

int main(){
    firstPage();
    return 0;
}
```

Functions/prima_pagina.c

```c
#include "functii_angajati.c"
#include "functii_firma.c"
#include "functii_clienti.c"
#include "functii_imobile.c"
#include "inchidere.c"

void firstPage();

void secondPage(int option){
    switch (option)
    {
    case 1:
        option1();
        break;
    case 2:
        option2();
        break;
    case 3:
        option3();
        break;
    case 4:
        option4();
        break;
    case 0:
        option5();
        break;
    default:
        break;
    }

}
void menu(){
    printf("1 Detalii firma \n2 Detalii angajati \n3 Detalii clienti\n4 Imobile disponibile\n0
Iesire\n");
}
void firstPage(){
    system("cls");
    menu();
```

```
    int option;
    scanf("%d", &option);
    while(option < 0 || option > 4){
        menu();
        scanf("%d", &option);
    }
    secondPage(option);
}
```

Functions/functii_firma.c

```
typedef struct firma{
    char nume[211], strada[310], numar[300], telefon[101];
    long long int cod;
}firma;

void printFirma(firma df){
    fflush(stdin);
    printf("Nume firma: %s\n", df.nume);
    printf("Cod fiscal: %lli\n\n", df.cod);
    printf("Adresa: %s, %s\n", df.strada, df.numar);
    printf("Telefon: %s\n\n", df.telefon);
}
void option1(){
    system("cls");
    FILE *det_firma = fopen("data/detalii_firma.csv", "r");
    if(det_firma == NULL){
        perror("Unable to open the file");
        exit(1);
    }
    char line[1024];
    firma df;
    int row = 1;
    //parcurge fisierul si stocheaza in struct

    while(fgets(line, sizeof(line), det_firma)){
        char *token;
        int field = 0;
        token = strtok(line, ",");
        while(token != NULL){
            if(row == 1 && field == 1)
                strcpy(df.nume, token);

            else if(row == 2 && field == 1){
               df.cod = atoi(token); // Using atoi()
            }

            else if(row == 3 && field == 1)
                strcpy(df.strada, token);

            else if(row == 3 && field == 2)
                strcpy(df.numar, token);
```

```
            else if(row == 4 && field == 1)
                strcpy(df.telefon, token);

            field++;
            token = strtok(NULL, ",");
        }
        row++;
    }
    printFirma(df);

    fclose(det_firma);
    printf("Apasa orice pentru reintoarcere\n");
    getch();
    firstPage();
}
```

Functions/functii_angajati.c

```
void firstPage();

typedef struct angajat{
    char nume[251], sediu[151];
}angajat;

void print_angajati(angajat angajati[], int row){
    for(int i = 0; i < row; i++){
        printf("%d. Nume: %s | Sediu: %s\n", i + 1, angajati[i].nume, angajati[i].sediu);
    }
}

void adaugare_angajat(angajat date_angajat[], angajat a, int len){
    date_angajat[len + 1] = a;
    FILE *temp = fopen("data/temp.csv", "w");

    fprintf(temp, "%s,%s", date_angajat[0].nume, date_angajat[0].sediu);

    for(int i = 1; i <= len + 1; i++){
        fprintf(temp,"%s,%s", date_angajat[i].nume, date_angajat[i].sediu);
    }
    fclose(temp);
    remove("data/angajati_sedii.csv");
    rename("data/temp.csv", "data/angajati_sedii.csv");
}

void sterge_angajat(angajat s[], int len, int start_pos){
    for(start_pos; start_pos < len; start_pos++){
        s[start_pos] = s[start_pos + 1];
    }
    FILE *tmp = fopen("data/temp.csv", "w");
    if(len == 0){
        fclose(tmp);
        remove("data/angajati_sedii.csv");
        rename("data/temp.csv", "data/angajati_sedii.csv");
```

```
        return;
    }
    fprintf(tmp, "%s,%s", s[0].nume, s[0].sediu);
    for(int i = 1; i < len; i++){
        fprintf(tmp,"%s,%s", s[i].nume, s[i].sediu);
    }
    fclose(tmp);
    remove("data/angajati_sedii.csv");
    rename("data/temp.csv", "data/angajati_sedii.csv");
}

void afisare_oferte_angajat(angajat ang, int i){
    system("cls");
    printf("Angajatul %s de la sediul %s are repartizate urmatoarele imobile: \n\n", ang.nume,
ang.sediu);

    FILE *imobile = fopen("data/imobile.csv", "r");
    if(imobile == NULL){
        perror("Eroare");
        exit(1);
    }
    char line[1024];
    int contor = 0;
    while(fgets(line, sizeof(line), imobile)){
        char *token = strtok(line, ",");
        int field = -1;
        char date_imobil[200], oras[200], judet[200];

        while(token != NULL){
            int si = 0;
            if(field == -1){
                si = atoi(token);
                field = 0;
                if(si == i){
                    token = strtok(NULL, ",");
                    contor++;
                    continue;
                }
                break;
            }

            if(field == 0){
                strcpy(judet, token);
            }
            else if(field == 1){
                strcpy(oras, token);
            }
            else if(field == 2){
                strcpy(date_imobil, token);
                printf("\t%s de", date_imobil);
            }
            else if(field == 3){
                strcpy(date_imobil, token);
                printf(" %s la pretul ", date_imobil);
```

```
            }
            else if(field == 4){
                si = atoi(token);
                printf("%d | ", si);
            }
            else if(field == 5){
                si = atoi(token);
                printf("%d m^2 si ", si);
            }
            else{
                si = atoi(token);
                printf("%d camere | Judet: %s Oras: %s\n\n", si, judet, oras);
            }
            field++;
            token = strtok(NULL, ",");
        }
    }
    fclose(imobile);
    if(contor == 0){
        printf("\nAngajatul selectat nu are nici o oferta repartizata!\n");
    }
}

void option2(){
    system("cls");
    FILE *ang_sed = fopen("data/angajati_sedii.csv", "r");
    if(ang_sed == NULL){
        perror("Unable to open the file");
        exit(1);
    }
    angajat date_angajat[100];
    char line[1024];
    int row = 0;
    while(fgets(line, sizeof(line), ang_sed)){
        char *token;
        int field = 0;
        token = strtok(line, ",");

        while(token != NULL){
            if(field == 0){
                strcpy(date_angajat[row].nume, token);
            }
            else
                strcpy(date_angajat[row].sediu, token);
            field++;
            token = strtok(NULL, ",");
        }
        row++;
    }
    fclose(ang_sed);
    if(strcmp(date_angajat[0].nume, "") == 0){
        printf("\nNu exista angajati!\n");
    }
    else{
```

```
        print_angajati(date_angajat, row);
    }

    int option;
    printf("\n1 Adaugare angajat\n2 Stergere angajat\n3 Oferte repartizate\n4 Inapoi\n");
    scanf("%d", &option);
    switch (option)
    {
    case 1:
        system("cls");
        angajat a;
        printf("Introduceti numele: ");
        fflush(stdin);
        gets(a.nume);
        printf("\nIntroduceti sediul: ");
        fflush(stdin);
        gets(a.sediu);
        strcat(a.sediu, "\n");
        adaugare_angajat(date_angajat, a, row);
        printf("\n");
        option2();
        break;
    case 2:
        system("cls");
        print_angajati(date_angajat, row);
        printf("Introduceti indexul angajatului pe care doriti sa-l stergeti (0 - cancel)\nIn
cazul in care introduceti un index mai mare decat indexul ultimului angajat, se va sterge
ultimul\nla fel si la primul angajat");
        int i;
        scanf("%d", &i);
        if(i == 0){
            option2();
            break;
        }
        sterge_angajat(date_angajat, row - 1, i - 1);
        printf("\n");
        option2();
        break;
    case 3:
        system("cls");
        print_angajati(date_angajat, row);
        printf("\nIntroduceti indexul angajatului: ");
        int ind;
        scanf("%d", &ind);
        if(ind > row || ind < 1){
            printf("\nAngajatul nu exista!\n");
        }
        else
            afisare_oferte_angajat(date_angajat[ind-1], ind);
        printf("Pentru a te reintoarce la meniul de angajati apasa orice ");
        getch();
        option2();
        break;
```

```
    case 4:
        firstPage();
        break;
    default:
        break;
    }
}
```

Functions/functii_clienti.c

```
typedef struct client{
    char nume[251], tip[50], perioada[100];
}client;

void printClienti(client date_clienti[], int row){
    for(int i = 0; i < row; i++){
        printf("%d Nume: %s | Tip: %s | ", i + 1, date_clienti[i].nume, date_clienti[i].tip);
        if(!strcmp(date_clienti[i].perioada, "nu\n")){
            printf("Nu are contract\n\n");
        }
        else{
            printf("Perioada: %s\n", date_clienti[i].perioada);
        }
    }
}

void adaugare_client(client date_client[], client c, int len){
    int i, added = 0;
    client tmp;
    if(strcmp(c.nume, date_client[len - 1].nume) > 0){
        date_client[len] = c;
    }
    else{
        for(i = 0; i <= len + 1; i++){
            if(added){
                c = date_client[i];
                date_client[i] = tmp;
                tmp = c;
            }
            else if(strcmp(c.nume, date_client[i].nume) < 0){
                tmp = date_client[i];
                date_client[i] = c;
                added = 1;
            }
        }
    }
    FILE *temp = fopen("data/temp.csv", "w");

    fprintf(temp, "%s,%s,", date_client[0].nume, date_client[0].tip);
    if(!strcmp(date_client[0].perioada, "nu are contract\n"))
        fprintf(temp, "nu\n");
    else{
        fprintf(temp, "%s", date_client[0].perioada);
```

```
    }
    for(i = 1; i <= len; i++){
        fprintf(temp,"%s,%s,", date_client[i].nume, date_client[i].tip);
        if(!strcmp(date_client[i].perioada, "nu are contract\n"))
            fprintf(temp, "nu\n");
        else
            fprintf(temp, "%s", date_client[i].perioada);
    }
    fclose(temp);
    remove("data/clienti.csv");
    rename("data/temp.csv", "data/clienti.csv");
}

void sterge_client(client date_client[], int len, int start_pos){
    for(start_pos; start_pos < len; start_pos++){
        date_client[start_pos] = date_client[start_pos + 1];
    }
    FILE *tmp = fopen("data/temp.csv", "w");
    if(len == 0){
        fprintf(tmp, "");
        fclose(tmp);
        remove("data/clienti.csv");
        rename("data/temp.csv", "data/clienti.csv");
        return;
    }
    fprintf(tmp, "%s,%s,%s", date_client[0].nume, date_client[0].tip,
date_client[0].perioada);
    for(int i = 1; i < len; i++){
        fprintf(tmp,"%s,%s,%s", date_client[i].nume, date_client[i].tip,
date_client[i].perioada);
    }
    fclose(tmp);
    remove("data/clienti.csv");
    rename("data/temp.csv", "data/clienti.csv");
}

void option3(){
    system("cls");
    FILE *clienti = fopen("data/clienti.csv", "r");
    if(clienti == NULL){
        perror("Unable to open the file");
        exit(1);
    }
    client date_clienti[100];
    char line[1024];
    int row = 0;
    while(fgets(line, sizeof(line), clienti)){
        char *token;
        int field = 0;

        token = strtok(line, ",");

        while(token != NULL){
            if(field == 0){
```

```
                strcpy(date_clienti[row].nume, token);
            }
            else if(field == 1){
                strcpy(date_clienti[row].tip, token);
            }
            else{
                strcpy(date_clienti[row].perioada, token);
            }

            field++;
            token = strtok(NULL, ",");
        }
        row++;
    }
    fclose(clienti);

    printClienti(date_clienti, row);

    int opt;
    printf("1 Adaugare client\n2 Stergere client\n3 Inapoi\n");
    scanf("%d", &opt);
    switch (opt)
    {
    case 1:
        system("cls");
        client c;
        printf("Introduceti numele clientului: ");
        fflush(stdin);
        gets(c.nume);
        printf("Introduceti tipul clientului (fizic/firma): ");
        fflush(stdin);
        gets(c.tip);
        fflush(stdin);
        printf("Introduceti perioada contractului (nedeterminat/perioada/nu are contract): ");
        gets(c.perioada);
        strcat(c.perioada, "\n");
        fflush(stdin);
        adaugare_client(date_clienti, c, row);
        option3();
        break;

    case 2:
        system("cls");
        printClienti(date_clienti, row);
        printf("Introduceti indexul clientului pe care doriti sa-l stergeti (0 - cancel)\nIn
cazul in care introduceti un index mai mare decat indexul ultimului client, se va sterge
ultimul\nla fel si cu primul client\n");
        int i;
        scanf("%d", &i);
        if(i == 0){
            option3();
            break;
        }
        i--;
```

```
        sterge_client(date_clienti, row - 1, i);
        option3();
        break;

    case 3:
        firstPage();
        break;
    default:
        break;
    }
}
```

Functions/functii_imobile.c

```
typedef struct imobil{
    char judet[100], oras[100], tip[100], vanz_inch[100];
    int asign, pret, suprafata, nr_camere;
} imobil;

void print_imobile(imobil date_imobil[], int len){
    for(int i = 0; i < len; i++){
        printf("%d %s de %s la pretul %d | %d m^2 si %d camere | Judet: %s  Oras: %s\n", i+1,
date_imobil[i].tip, date_imobil[i].vanz_inch, date_imobil[i].pret, date_imobil[i].suprafata,
date_imobil[i].nr_camere, date_imobil[i].judet, date_imobil[i].oras);
    }
}

void adaugare_imobil(imobil date_imobil[], imobil im, int len){
    int i, added = 0;
    imobil tmp;
    if(strcmp(im.judet, date_imobil[len - 1].judet) > 0 || (strcmp(im.judet, date_imobil[len -
1].judet) == 0 && strcmp(im.oras, date_imobil[len - 1].oras) >= 0)){
        date_imobil[len] = im;
    }
    else{
        for(i = 0; i <= len; i++){
            if(added){
                im = date_imobil[i];
                date_imobil[i] = tmp;
                tmp = im;
            }
            else if(strcmp(im.judet, date_imobil[i].judet) < 0){
                tmp = date_imobil[i];
                date_imobil[i] = im;
                added = 1;
            }
            else if(strcmp(im.judet, date_imobil[i].judet) == 0 && strcmp(im.oras,
date_imobil[i].oras) <= 0){
                tmp = date_imobil[i];
                date_imobil[i] = im;
                added = 1;
            }
        }
```

```
    }
    FILE *temp = fopen("data/temp.csv", "w");
    for(i = 0; i <= len; i++){
        fprintf(temp, "%d,%s,%s,%s,%s,%d,%d,%d\n", date_imobil[i].asign, date_imobil[i].judet,
date_imobil[i].oras, date_imobil[i].tip, date_imobil[i].vanz_inch, date_imobil[i].pret,
date_imobil[i].suprafata, date_imobil[i].nr_camere);
    }
    fclose(temp);
    remove("data/imobile.csv");
    rename("data/temp.csv", "data/imobile.csv");
}

void stergere_imobil(imobil date_imobil[], int len, int start_pos){
    for(start_pos; start_pos < len; start_pos++){
        date_imobil[start_pos] = date_imobil[start_pos + 1];
    }
    FILE *tmp = fopen("data/temp.csv", "w");
    if(len == 0){
        fclose(tmp);
        remove("data/imobile.csv");
        rename("data/temp.csv", "data/imobile.csv");
        return;
    }
    for(int i = 0; i < len; i++){
        fprintf(tmp, "%d,%s,%s,%s,%s,%d,%d,%d\n", date_imobil[i].asign, date_imobil[i].judet,
date_imobil[i].oras, date_imobil[i].tip, date_imobil[i].vanz_inch, date_imobil[i].pret,
date_imobil[i].suprafata, date_imobil[i].nr_camere);
    }
    fclose(tmp);
    remove("data/imobile.csv");
    rename("data/temp.csv", "data/imobile.csv");
}

void afisare_judet(imobil date_imobil[], char jud[], int len){
    int gasit = 0;
    for(int i = 0; i < len; i++){
        if(strcmp(date_imobil[i].judet, jud) == 0){
            gasit++;
            printf("%d %s de %s la pretul %d | %d m^2 si %d camere | Judet: %s  Oras: %s\n",
gasit, date_imobil[i].tip, date_imobil[i].vanz_inch, date_imobil[i].pret,
date_imobil[i].suprafata, date_imobil[i].nr_camere, date_imobil[i].judet,
date_imobil[i].oras);
        }
    }
    if(!gasit){
        printf("Nu s-au gasit imobile in acel judet.\n");
    }
    printf("\nApasa orice pentru a te reintoarce. ");
    getch();
}

void afisare_oras(imobil date_imobil[], char oras[], int len){
    int gasit = 0;
    for(int i = 0; i < len; i++){
```

```
        if(strcmp(date_imobil[i].oras, oras) == 0){
            gasit++;
            printf("%d %s de %s la pretul %d | %d m^2 si %d camere | Judet: %s  Oras: %s\n",
gasit, date_imobil[i].tip, date_imobil[i].vanz_inch, date_imobil[i].pret,
date_imobil[i].suprafata, date_imobil[i].nr_camere, date_imobil[i].judet,
date_imobil[i].oras);
        }
    }
    if(!gasit){
        printf("Nu s-au gasit imobile in acel oras.\n");
    }
    printf("\nApasa orice pentru a te reintoarce. ");
    getch();
}

int citire_afisare_angajati(){
    FILE *angajati = fopen("data/angajati_sedii.csv", "r");
    if(angajati == NULL){
        perror("Unable to open the file");
        exit(1);
    }
    char line[1024];
    int row = 0;
    while(fgets(line, sizeof(line), angajati)){
        char *token;
        int field = 0;
        token = strtok(line, ",");
        while(token != NULL){
            if(field == 0){
                printf("%d. Nume: %s", row + 1, token);
            }
            else
                printf(" | Sediu: %s\n", token);
            field++;
            token = strtok(NULL, ",");
        }
        row++;
    }
    fclose(angajati);
    return row;
}

void option4(){
    system("cls");
    FILE *imobile = fopen("data/imobile.csv", "r");
    if(imobile == NULL){
        perror("Unable to open the file!!");
        exit(1);
    }
    imobil date_imobil[100];
    char line[1024];
    int row = 0;
    while(fgets(line, sizeof(line), imobile)){
        char *token = strtok(line, ",");
```

```
        int field = -1;
        while(token != NULL){
            if(field == -1){
                date_imobil[row].asign = atoi(token);
            }
            else if (field == 0){
                strcpy(date_imobil[row].judet, token);
            }
            else if(field == 1){
                strcpy(date_imobil[row].oras, token);
            }
            else if(field == 2){
                strcpy(date_imobil[row].tip, token);
            }
            else if(field == 3){
                strcpy(date_imobil[row].vanz_inch, token);
            }
            else if(field == 4){
                date_imobil[row].pret = atoi(token);
            }
            else if(field == 5){
                date_imobil[row].suprafata = atoi(token);
            }
            else{
                date_imobil[row].nr_camere = atoi(token);
            }
            field++;
            token = strtok(NULL, ",");
        }
        row++;
    }
    fclose(imobile);
    print_imobile(date_imobil, row);
    int opt;
    printf("\n1 Adaugare imobil\n2 Stergere imobil\n3 Afisare dupa judet\n4 Afisare dupa
oras\n5 Inapoi\n");
    scanf("%d", &opt);
    switch(opt){
        case 1:
            system("cls");
            imobil im;
            int angajat_asignat;
            printf("Introduceti timpul imobilului: ");
            fflush(stdin);
            gets(im.tip);
            printf("Imobilul este de [vanzare/inchiriere]: ");
            fflush(stdin);
            gets(im.vanz_inch);
            printf("Pretul imobilului este: ");
            fflush(stdin);
            scanf("%d", &im.pret);
            printf("Suprafata imobilului este: ");
            scanf("%d", &im.suprafata);
            printf("Numarul de camere al imobilului este: ");
```

```
            scanf("%d", &im.nr_camere);
            fflush(stdin);
            printf("Judetul in care se afla imobilul este: ");
            gets(im.judet);
            fflush(stdin);
            printf("Orasul in care se afla imobilul este: ");
            gets(im.oras);
            fflush(stdin);
            int lungime = citire_afisare_angajati();
            printf("Cui doresti sa-i asignezi acest contract:\n");
            scanf("%d", &im.asign);
            while(im.asign < 1 || im.asign > lungime){
                printf("Trebuie sa introduceti un index in intervalul %d-%d.\n", 1, lungime);
                printf("Introduceti din nou indexul: ");
                scanf("%d", &im.asign);
            }
            adaugare_imobil(date_imobil, im, row);
            option4();
            break;
        case 2:
            system("cls");
            print_imobile(date_imobil, row);
            printf("\nIntroduceti indexul imobilului pe care doriti sa-l stergeti:\n");
            int index;
            scanf("%d", &index);
            stergere_imobil(date_imobil, row - 1, index - 1);
            option4();
            break;
        case 3:
            system("cls");
            char jud[100];
            printf("Introduceti judetul de unde vreti sa vedeti oferte: ");
            fflush(stdin);
            gets(jud);
            printf("\n");
            afisare_judet(date_imobil, jud, row);
            option4();
            break;
        case 4:
            system("cls");
            char oras[100];
            printf("Introduceti orasul de unde vreti sa vedeti ofertele: ");
            fflush(stdin);
            gets(oras);
            printf("\n");
            afisare_oras(date_imobil, oras, row);
            option4();
            break;
        case 5:
            firstPage();
            break;
        default:
            break;
    }
```

```
}
```

Functions/inchidere.c

```
void option5(){
    system("cls");
    char yesno[1];
    printf("Esti sigur ca vrei sa iesi? [y/n] ");
    scanf("%s", yesno);
    if(!strcmp(yesno, "n")){
        firstPage();
    }
    else
        system("cls");
}
```