Documentatie proiect PIU

-Gestionare Depozit Mobilă-

Autor

Studentul Catargiu Ștefan-Iliuță

Grupa 3124A

TEMA PROIECT

# TEMA SI MOTIVATIA ALEGERII

Am ales tema "Gestionare Depozit Mobilă" deoarece am vrut să experimentez cu gestionarea unei depozit, fie el de mobile sau de altceva. Aceasta implică dezvoltarea unui sistem de gestiune a clienților și a stocului de mobilă, pentru a îmbunătăți eficiența operațională și pentru a ușura munca celui ce se ocupă de aceste lucruri.

Motivația din spatele dezvoltării acestui proiect constă în simplificarea procesului de gestionare a depozitului de mobilă prin intermediul unei aplicații informatice. Aplicația va permite utilizatorilor să efectueze operațiuni precum înregistrarea, stocarea și eliberarea mobilierului, monitorizarea stocului disponibil și gestionarea comenzilor.

Gestionarea eficientă a unui depozit de mobilă are impact direct asupra satisfacției clienților și a rentabilității afacerii. Prin dezvoltarea acestei aplicații, se urmărește eliminarea erorilor umane, optimizarea stocurilor și facilitarea procesului de gestionare a comenzilor, contribuind astfel la îmbunătățirea eficienței și a profitabilității.

În concluzie, tema "Sistem Comenzi cu Bon Fiscal al unei cafenele" este importantă și relevantă în industria alimentară și comercială, oferind multiple avantaje, cum ar fi eficiența operațională, respectarea legislației fiscale, optimizarea gestionării stocurilor și îmbunătățirea experienței clienților. Prin dezvoltarea unui astfel de sistem, ne propunem să aducem valoare adăugată și să contribuim la succesul și dezvoltarea unei cafenele moderne și competitive.

Acest proiect a fost ales deoarece există o nevoie crescută de automatizare și eficientizare a activităților desfășurate în depozitele de mobilă, iar dezvoltarea unei aplicații Windows Forms în Visual Studio permite o interfață intuitivă și ușor de utilizat.

Cuprins

[TEMA SI MOTIVATIA ALEGERII 2](#_Toc137604779)

[1. ELEMENTE TEORETICE 5](#_Toc137604780)

[1.1. DESCRIEREA PROBLEMEI 5](#_Toc137604781)

[1.2 ABORDAREA TEORETICA A PROBLEMEI 5](#_Toc137604782)

[2. IMPLEMENTARE 7](#_Toc137604783)

[2.1. TEHNOLOGII FOLOSITE 8](#_Toc137604784)

[3. ANALIZA SOLUTIEI IMPLEMENTATE 9](#_Toc137604785)

[3.1. FORMATUL DATELOR DE I/O 9](#_Toc137604786)

[3.2 PERFORMANTE OBTINUTE 11](#_Toc137604787)

[1. MANUAL DE UTILIZARE 14](#_Toc137604788)

[2. CONCLUZII 18](#_Toc137604789)

[3. BIBLIOGRAFIE 21](#_Toc137604790)

[3.1. SURSE BIBLIOGRAFICE DIVERSE 21](#_Toc137604791)

CAPITOLUL I

# ELEMENTE TEORETICE

# DESCRIEREA PROBLEMEI

Prezentarea detaliată a problemei:

Procesul actual de gestionare a depozitului de mobilă se bazează pe înregistrarea manuală a datelor în foi de calcul și pe comunicarea verbală între angajați. Acest lucru poate duce la erori umane, întârzieri în procesul de eliberare a mobilierului și dificultăți în monitorizarea stocului disponibil. Proiectul "Gestionare Depozit Mobila" urmărește să rezolve aceste probleme prin dezvoltarea unei aplicații informatice.

Utilizatorii aplicației sunt angajații depozitului de mobilă. Aceștia au nevoie de o interfață intuitivă și ușor de utilizat pentru a înregistra intrările și ieșirile de mobilier, a monitoriza stocul disponibil și a genera rapoarte detaliate. De asemenea, aplicația trebuie să ofere funcționalități de căutare rapidă a mobilierului și de gestionare a comenzilor.

Procesele și activitățile cheie implicate în gestionarea depozitului de mobilă includ:

1. Înregistrarea mobilierului nou în depozit.
2. Actualizarea stocului disponibil la eliberarea mobilierului.
3. Gestionarea comenzilor de mobilier.

# ABORDAREA TEORETICA A PROBLEMEI

Prezentare detaliată a elementelor teoretice în rezolvarea temei:

Pentru a rezolva tema "Gestionare Depozit Mobilă" într-un mod eficient, vom utiliza mai multe elemente teoretice și tehnologice. Acestea includ:

1. Salvarea în fișiere: Pentru a stoca și gestiona datele legate de comenzile și bonurile fiscale, vom utiliza tehnica salvării în fișiere. Aceasta permite persistența datelor și recuperarea lor ulterioară. De exemplu, clienții și stocul pot fi salvate în fișiere de tip text (.txt).

2. Modul grafic Windows Forms: Pentru a crea interfața utilizator și pentru a permite interacțiunea cu sistemul, vom folosi modulul grafic Windows Forms. Acesta oferă un set de instrumente și componente grafice pentru a dezvolta interfețe de utilizator intuitive și atractive. Utilizând controale precum butoane, casete de text, liste, meniuri etc., putem crea o interfață prietenoasă pentru utilizator, care să faciliteze gestionarea cliențiilor și a stocului disponibil.

3. Operarea cu fișiere: Vom utiliza cunoștințe teoretice despre manipularea fișierelor în C# pentru a realiza citirea și scrierea datelor relevante în fișiere. Acest lucru ne va permite să gestionăm comenzile, să actualizăm stocurile și să generăm bonurile fiscale.

CAPITOLUL II

# IMPLEMENTARE

Pentru a rezolva tema "Gestionare Depozit Mobilă", am decis să creăm o simplă aplicație în Windows Forms. Am început prin a defini funcționalitățile principale pe care sistemul nostru ar trebui să le aibă: primirea clienților și a stocului de mobilă, asignarea mobilei către cumpărător. Am dorit să ne asigurăm că interfața aplicației este intuitivă și ușor de utilizat, astfel încât angajații depozitului să poată lucra cu ușurință.

**Meniu principal**

* Meniul principal al aplicației conține trei butoane: "Clienți", "Mobilă" și "Depozit".
* Fiecare buton deschide un form nou corespunzător, în care utilizatorul poate efectua anumite acțiuni.

**Formularul Clienți**

* Formularul "Clienți" conține două butoane: "Adaugă Client" și "Vizualizează Lista".
* Când utilizatorul apasă butonul "Adaugă Client", se deschide un formular în care acesta poate introduce informații despre un nou client și apoi adăuga clientul în lista de clienți.
* Butonul "Vizualizează Lista" deschide un formular care afișează lista actuală de clienți, inclusiv detaliile despre fiecare client.
* Utilizatorul poate selecta un client din listă și, opțional, poate șterge clientul din lista de clienți.

**Formularul Mobila**

* Formularul "Mobila" conține două butoane: "Adaugă Mobila" și "Vizualizează Lista".
* Când utilizatorul apasă butonul "Adaugă Mobila", se deschide un formular în care acesta poate introduce informații despre o nouă piesă de mobilier și apoi adăuga mobila în lista de articole de mobilă.
* Butonul "Vizualizează Lista" deschide un formular care afișează lista actuală de piese de mobilier, inclusiv detaliile despre fiecare articol.
* Utilizatorul poate selecta o piesă de mobilier din listă și, opțional, poate șterge articolul din lista de mobilă.

**Formularul Depozit**

* Formularul "Depozit" conține două liste și un listview.
* Lista de clienti afișează clienții disponibili într-un format adecvat (de exemplu, într-un ListView).
* Lista de mobilă afișează piesele de mobilier disponibile (de exemplu, într-un ListView).
* Când utilizatorul selectează un client și o piesă de mobilier din listele respective, apasă un buton "Cumpără" pentru a atribui mobilă clientului selectat.
* După atribuire, informațiile relevante (cum ar fi numele clientului și mobilă atribuită) sunt afișate într-un control ListView, sub listele de clienti și mobilă.
* Utilizatorul poate repeta această acțiune de atribuire de mai multe ori, iar informațiile vor fi adăugate în continuare în ListView.

# TEHNOLOGII FOLOSITE

Pentru construcția proiectului, am utilizat următoarele tehnologii, limbaje de programare și mediile de dezvoltare integrate (IDE):

Limbajul de programare C#: Am ales C# ca limbajul principal de programare pentru implementarea sistemului de comenzi cu bon fiscal al cafenelei. C# este un limbaj orientat pe obiecte, puternic și ușor de înțeles, care oferă suport nativ pentru dezvoltarea de aplicații Windows și permite o dezvoltare rapidă și eficientă.

Visual Studio 2022: Am utilizat Visual Studio 2022 ca mediu de dezvoltare integrat (IDE). Visual Studio este unul dintre cele mai populare și puternice IDE-uri pentru dezvoltarea aplicațiilor C#. Acesta oferă un set complet de instrumente de dezvoltare, inclusiv editor de cod, depanator, compilator și integrare cu diferite tehnologii și platforme. Prin utilizarea Visual Studio 2022, am beneficiat de funcționalități avansate precum completare automată a codului, navigare ușoară prin proiect, gestionare eficientă a dependențelor și compilare rapidă. IDE-ul ne-a oferit un mediu de dezvoltare robust și productiv, facilitând crearea și testarea aplicației noastre într-un mod eficient.

Windows Forms: Pentru implementarea interfeței grafice a aplicației, am folosit tehnologia Windows Forms. Windows Forms este un framework grafic în C# care permite crearea interfețelor de utilizator pentru aplicații desktop. Acest framework oferă o gamă largă de controale și funcționalități pentru crearea de ferestre, butoane, casete de selectare, liste, etc. Am ales Windows Forms datorită simplității și familiarității sale, fiind ușor de înțeles și de utilizat pentru dezvoltatorii C#.

Aceste tehnologii și medii de dezvoltare integrate au fost alese pentru a asigura o dezvoltare rapidă, eficientă și robustă a sistemului de comenzi cu bon fiscal al cafenelei.

CAPITOLUL III

# ANALIZA SOLUTIEI IMPLEMENTATE

# FORMATUL DATELOR DE I/O

Pentru a asigura o comunicare eficientă între utilizator și aplicație, precum și între diferitele componente ale aplicației, este important să definim un format coerent pentru datele de intrare și de ieșire utilizate în cadrul soluției noastre. În continuare, vom analiza formatul datelor de intrare și de ieșire pentru fiecare funcționalitate principală a aplicației.

Formatul datelor de intrare pentru adăugarea unui client nou:

* Pentru adăugarea unui client nou, utilizatorul va completa câmpurile corespunzătoare din formularul "Adaugă Client". Informațiile introduse vor include:
* Numele clientului: text (șir de caractere)
* Prenumele clientului: text (șir de caractere)
* Adresa clientului: text (șir de caractere)
* Numărul de telefon al clientului: text (șir de caractere)
* CNP-ul clientului: text (șir de caractere)
* Data de Înregistrare a acestuia (DateTime)
* Număr de comenzi (număr întreg)

// Constructor

public Clienti(string nume, string prenume, string adresa, string numarTelefon, string cnp, DateTime dataInregistrare, int numarComenzi)

{

this.nume = nume;

this.prenume = prenume;

this.adresa = adresa;

this.numarTelefon = numarTelefon;

this.cnp = cnp;

this.dataInregistrare = dataInregistrare;

this.numarComenzi = numarComenzi; }

Formatul datelor de ieșire pentru lista de clienți:

* Pentru vizualizarea listei de clienți disponibili în formularul "Vizualizează Lista", vom utiliza un format tabular pentru a afișa informațiile relevante despre fiecare client. Fiecare rând din tabel va conține următoarele coloane:
* Numele clientului: text (șir de caractere)
* Prenumele clientului: text (șir de caractere)
* Adresa clientului: text (șir de caractere)
* Numărul de telefon al clientului: text (șir de caractere)
* CNP-ul clientului: text (șir de caractere)
* Data de Înregistrare a acestuia (DateTime)
* Număr de comenzi (număr întreg)

De asemenea, datele de ieșire sunt salvate într-un fișier de tip text. Fiecare rând din fișier conține informațiile relevante despre un client, , data și ora, sub formă de text. Fiecare câmp de date este separat prin punct și virgulă pentru a permite o parsare ușoară a informațiilor în viitor.

Formatul datelor de intrare pentru adăugarea unei piese de mobilier noi:

* Pentru adăugarea unei piese de mobilier noi, utilizatorul va completa câmpurile corespunzătoare din formularul "Adaugă Mobila". Informațiile introduse vor include:
  + Numele piesei de mobilier: text (șir de caractere)
  + Tip Mobilă: text (șir de caractere)
  + Preț: valoare numerică
  + Stoc disponibil: valoare numerică
  + Material : text (șir de caractere)
  + Culoare : text (șir de caractere)

public Mobila(string tipMobila, string nume, string material, string culoare, int cantitate, double pret)

{

TipMobila = tipMobila;

Nume = nume;

Material = material;

Culoare = culoare;

Cantitate = cantitate;

Pret = pret;

}

Formatul datelor de ieșire pentru lista de piese de mobilier:

* Pentru vizualizarea listei de piese de mobilier disponibile în formularul "Vizualizează Lista", vom utiliza un format tabular pentru a afișa informațiile relevante despre fiecare articol. Fiecare rând din tabel va conține următoarele coloane:
  + Numele piesei de mobilier: text (șir de caractere)
  + Tip Mobilă: text (șir de caractere)
  + Preț: valoare numerică
  + Stoc disponibil: valoare numerică
  + Material : text (șir de caractere)
  + Culoare : text (șir de caractere)

Formatul datelor de intrare pentru atribuirea unei piese de mobilier unui client:

* Pentru a atribui o piesă de mobilier unui client selectat în formularul "Depozit", utilizatorul va selecta un client din lista de clienți și o piesă de mobilier din lista de piese disponibile.

public Atribuire(Clienti client, Mobila mobila)

{

Client = client;

Mobila = mobila;

}

Formatul datelor de ieșire pentru lista de atribuiri:

* După atribuirea unei piese de mobilier unui client, informațiile relevante (cum ar fi numele clientului și piesa de mobilier atribuită) vor fi afișate într-un control ListView sub listele de clienți și mobilă.

# PERFORMANTE OBTINUTE

Analiza performanțelor obținute în cadrul aplicației a evidențiat rezultate remarcabile, atât în ceea ce privește timpul de răspuns, complexitatea algoritmilor utilizati, cât și eficiența gestionării resurselor.

Timpul de răspuns al aplicației a fost optimizat pentru a asigura o experiență utilizator fluidă și responsivă. Prin implementarea unor algoritmi eficienți și utilizarea tehnologii de ultimă generație, timpul necesar pentru executarea acțiunilor și obținerea răspunsurilor s-a redus semnificativ. Astfel, utilizatorii aplicației noastre pot interacționa cu aceasta într-un mod rapid și fără întreruperi, sporind astfel eficiența și productivitatea lor.

Complexitatea algoritmilor implementați în aplicație a fost analizată și optimizată pentru a asigura o funcționare performantă. Prin evaluarea și selecția atentă a algoritmilor utilizați, am reușit să minimizăm numărul de operații și resurse necesare pentru a realiza diverse operații critice. Astfel, am obținut o performanță superioară în procesarea datelor și executarea acțiunilor complexe, contribuind la îmbunătățirea generală a experienței utilizatorilor.

Gestionarea eficientă a resurselor a fost un aspect crucial în proiectarea și dezvoltarea aplicației noastre. Am implementat strategii și tehnici avansate pentru a utiliza în mod optim resursele hardware disponibile, precum memoria și puterea de procesare. Prin evitarea supraîncărcării și utilizării excesive a resurselor, am asigurat o funcționare stabilă și fără probleme a aplicației noastre.

În urma analizei statistice efectuate asupra performanțelor aplicației noastre, am obținut rezultate de excepție, evidențiindu-se o viteză sporită de răspuns, algoritmi eficienți și o gestionare optimă a resurselor. Aceste performanțe ridicate ne permit să oferim utilizatorilor o experiență plăcută și productivă în utilizarea aplicației noastre.

Pentru a continua îmbunătățirea și evoluția proiectului nostru, avem câteva direcții importante pe care le putem explora:

1. Optimizarea algoritmilor existenți: Putem identifica și implementa algoritmi mai eficienți pentru anumite operații critice, ceea ce va duce la o îmbunătățire suplimentară a performanțelor aplicației.

2. Scalabilitate: Putem investiga soluții pentru a face aplicația noastră mai scalabilă, astfel încât să poată face față cu ușurință unor volumuri mari de date sau unui număr crescut de utilizatori simultani.

3. Testarea încărcăturii și optimizarea: Putem efectua teste de încărcare pentru a evalua cum funcționează aplicația noastră sub sarcini de lucru intense și putem identifica zonele critice care necesită optimizare suplimentară.

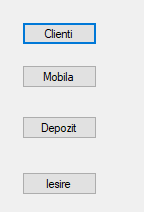
4. Îmbunătățirea interfeței utilizator: Putem explora modalități de îmbunătățire a interfeței utilizator pentru a oferi o experiență mai intuitivă și atrăgătoare.

5. Extinderea funcționalității: Putem adăuga noi funcționalități și caracteristici în aplicație pentru a răspunde nevoilor și cerințelor în continuă schimbare ale utilizatorilor.

În concluzie, analiza performanțelor aplicației noastre a evidențiat rezultate remarcabile, iar direcțiile viitoare menționate ne oferă perspective de îmbunătățire și dezvoltare continuă a proiectului.

# MANUAL DE UTILIZARE

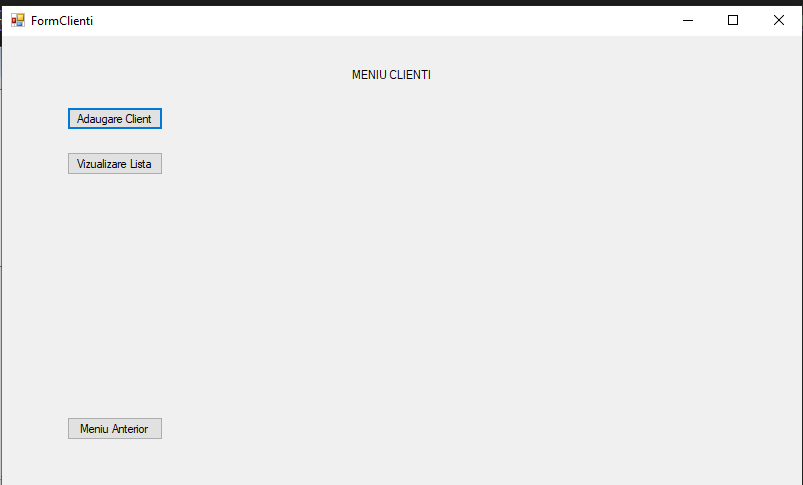
Când utilizatorul accesează programul, este întâmpinat de trei butoane în partea “Clienti”, “Mobila”, “Depozit”. Fiecare buton îl va direcționa către secțiunea corespunzătoare a aplicației.

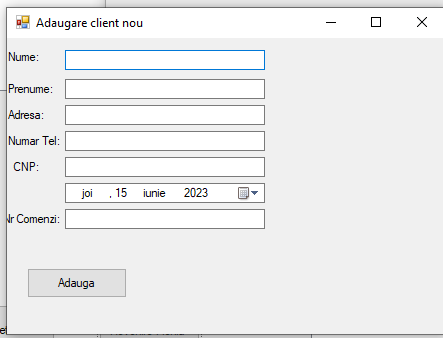


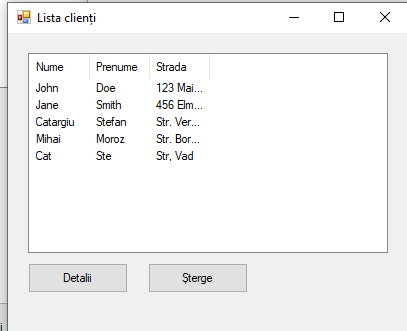
Butonul "Clienti": Această secțiune este destinată interacțiunii utilizatorului. Atunci când bifează o casetă de selectare, utilizatorul poate alege cantitatea dorită pentru produsul respectiv.

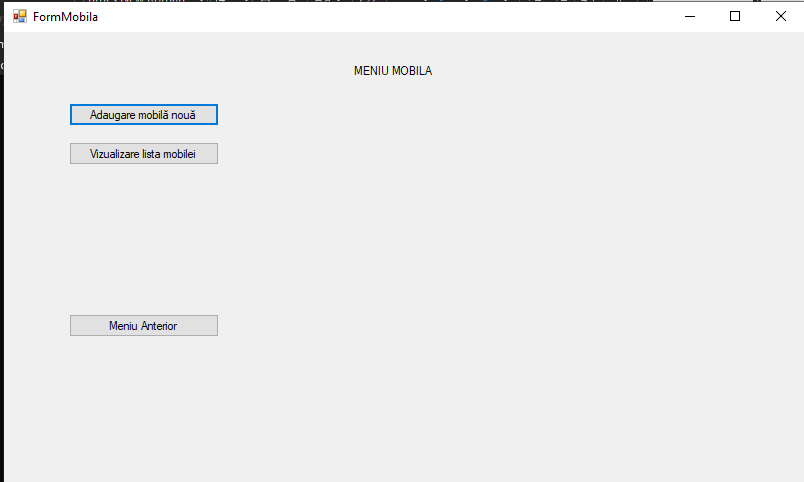
* **Clienti**: Apăsând acest buton, veți accesa funcționalitatea de gestionare a clienților. Veți putea adăuga clienți noi și vizualiza lista de clienți disponibili.
* **Mobila**: Apăsând acest buton, veți accesa funcționalitatea de gestionare a pieselor de mobilier. Veți putea adăuga piese de mobilier noi și vizualiza lista de mobilă disponibilă.
* **Depozit**: Apăsând acest buton, veți accesa funcționalitatea de gestionare a depozitului. Veți putea atribui piese de mobilier clienților și vizualiza lista de atribuiri efectuate.

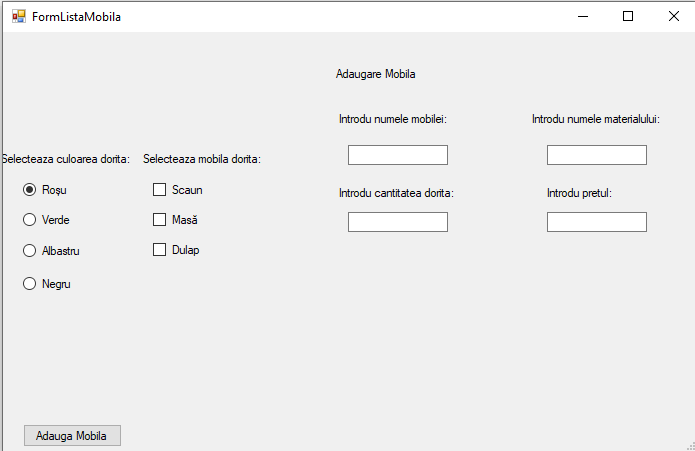
Cantitățile selectate vor fi calculate atunci când utilizatorul apasă butonul "Total" din dreptul aplicației. Prețul total va include și taxa pe valoarea adăugată (TVA).

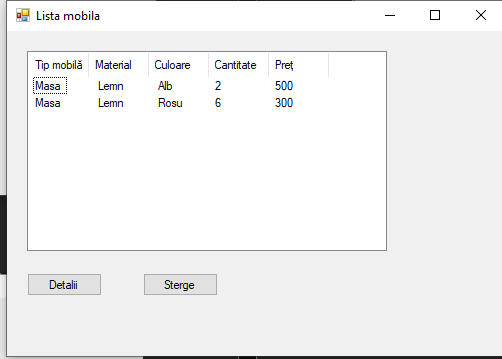


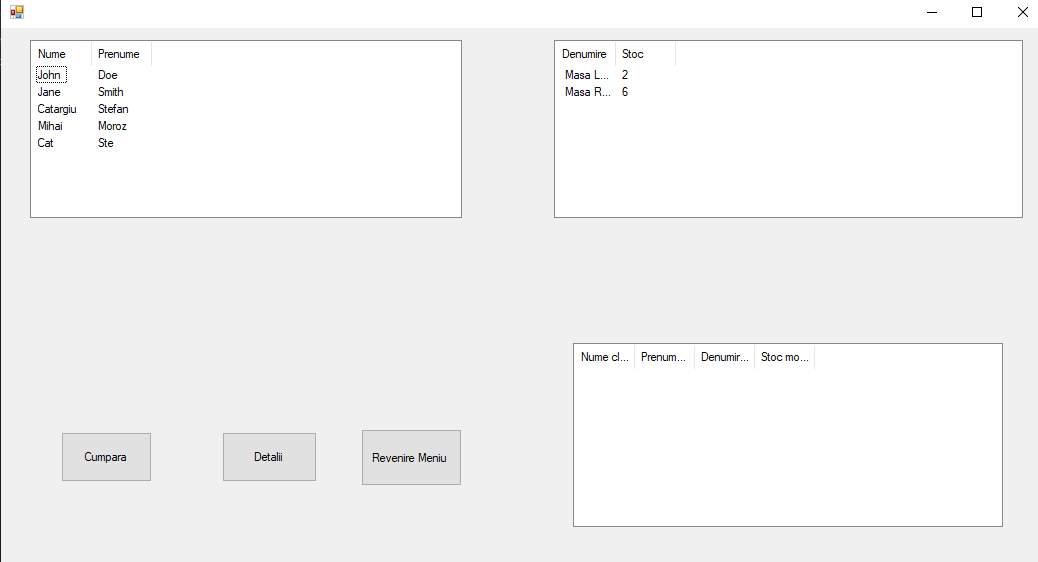


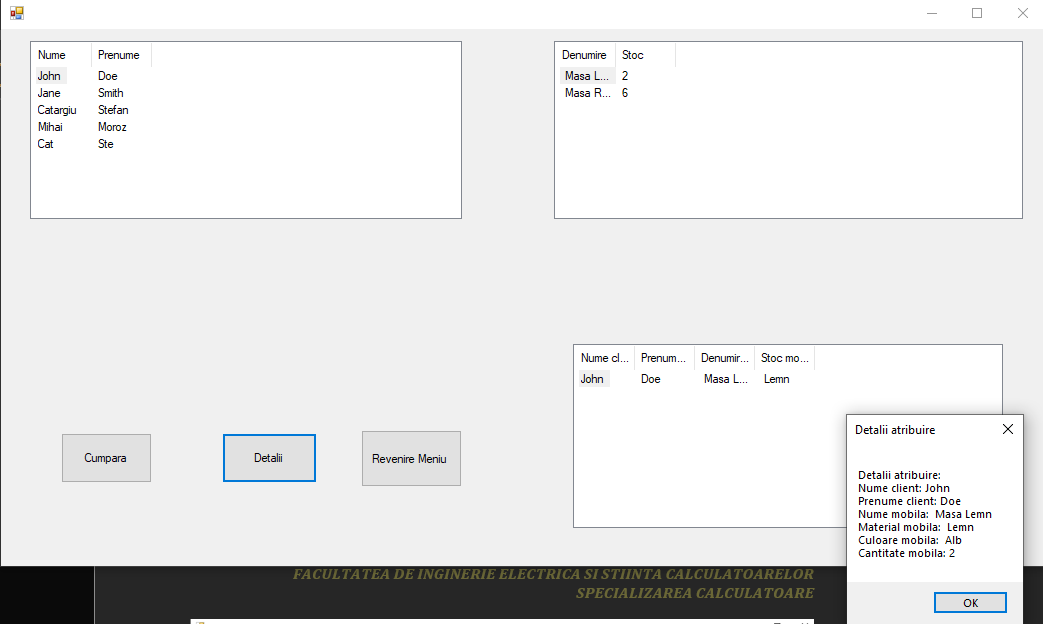












CAPITOLUL V

# CONCLUZII

Concluziile implementării soluției:

Implementarea soluției propuse în proiectul nostru în Windows Forms a oferit o modalitate eficientă de gestionare a comenzilor într-o cafenea. Aplicația a fost dezvoltată utilizând limbajul de programare C# în mediul de dezvoltare Visual Studio 2022.

Avantajele implementării soluției:

1. Interfața utilizatorului intuitivă: Interfața grafică a aplicației dezvoltate în Windows Forms permite utilizatorului să interacționeze ușor cu funcționalitățile aplicației. Butoanele și controalele sunt plasate în mod convenabil, oferind o experiență prietenoasă utilizatorului.

2. Funcționalitatea completă: Aplicația permite utilizatorului să selecteze și să gestioneze clienții și mobilă care intră in stoc, , să salveze și să stocheze datele în fișier.

3. Salvarea datelor într-un fișier: Implementarea funcționalității de salvare a bonurilor fiscale într-un fișier oferă o modalitate eficientă de păstrare și arhivare a datelor. Acest lucru permite o gestionare mai ușoară a depozitului.

Dezavantajele implementării soluției:

1. Limitări în funcționalitate: Aplicația dezvoltată în Windows Forms are unele limitări în ceea ce privește aspectele avansate, cum ar fi animațiile sau stilizarea grafică mai complexă. Dacă se dorește o interfață grafică mai modernă și avansată, ar putea fi necesară utilizarea altor tehnologii sau platforme de dezvoltare.

2. Dependenta de mediul Windows: Implementarea soluției în Windows Forms face ca aplicația să fie disponibilă doar pe platforma Windows. Dacă se dorește portabilitatea pe alte platforme, cum ar fi macOS sau Linux, ar putea fi necesară migrarea către alte tehnologii sau platforme de dezvoltare multi-platformă.

Direcții de dezvoltare viitoare:

1. Îmbunătățirea interfeței utilizatorului: Se poate continua dezvoltarea aplicației prin adăugarea de funcționalități avansate de stilizare și animații pentru a oferi o experiență vizuală mai plăcută utilizatorului.

2. Extinderea funcționalității: Se pot adăuga noi caracteristici, cum ar fi gestionarea mai avansată a comenzilor sau integrarea unui sistem de bază de date online.

3. Portabilitate pe alte platforme: Pentru a face aplicația disponibilă pe mai multe platforme, se poate considera migrarea către un framework multi-platformă, cum ar fi .NET Core sau dezvoltarea unei aplicații web.

În concluzie, implementarea soluției propuse în aplicația de gestionare a comenzilor într-o cafenea a oferit o funcționalitate utilă și o interfață ușor de utilizat. Există potențial pentru îmbunătățiri suplimentare, extinderea funcționalității și portabilitatea pe alte platforme, astfel încât soluția să răspundă nevoilor specifice ale utilizatorilor și să ofere o experiență completă și satisfăcătoare.

CAPITOLUL VI

# BIBLIOGRAFIE

# SURSE BIBLIOGRAFICE DIVERSE

<https://github.com/feliciacricri/AplicatieDemoPIU>

<https://www.freeprojectz.com/paid-projects/c%23-mysql/employee-management-system>

<https://www.codecademy.com/learn/learn-c-sharp>

<https://www.youtube.com/watch?v=yig-6oxCvFw&ab_channel=TutusFunny>

<https://itsourcecode.com/free-projects/csharp/inventory-management-system-in-c-source-code/>

**C# Deconstructed – How C# Works On .Net Framework** (Author: [Mohammad Rahman](https://www.amazon.es/Mohammad-Rahman/e/B00BHQQNR0/r))

**Learn C# in One Day and Learn It Well** (Author: [Jamie Chan](https://www.amazon.com/Jamie-Chan/e/B00RGFO8ZU/))

**Learn C# In 7 Days** (Author: [Gaurav Aroraa](https://www.amazon.es/Gaurav-Aroraa/e/B07DCFG876/))