**Proiect la LP2**

Top of Form

Bottom of Form

**Profesor Coordonator Student**

**Olteanu Gabriela Negoita Alexandru**

**Fudulu Cosmin**

**Aplicatie software pentru stocuri si inventar**

**Profesor Coordonator Student**

**Olteanu Gabriela Negoita Alexandru**

**Fudulu Cosmin**

**CUPRINS**

**INTRODUCERE………………………………………………………………………………..4**

Motivarea alegerii temei…………………………………………………………………………..4

Obiectivele propuse in cadrul lucrarii……………………………………………………………..4

**TEHNOLOGII UTILIZATE……………………………………………………………………5**

**2.1 C++……………………………………………………………………………………………5**

**2.2 Qt Creator…………………………………………………………………………………….6**

**2.3 PROGRAMARE ORIENTATA PE OBIECT (OOP) …………………………………….6**

**2.4 CMAKE………………………………………………………………………………………6**

**3.STUDIU DE CAZ…………………………………………………………………………….6**

**3.1 PLANIFICAREA APLICATIEI………………………………………………………….7**

**3.2 Realizarea claselor………………………………………………………………………….7**

**3.3 Salvarea datelor cu ajutorul JSON………………………………………………………10**

**3.4 Interfata grafica a aplicatiei………………………………………………………………11**

**3.5 Declaratia clasei logice mainwindow……………………………………………………12**

**3.6 Implementarea functiei logice……………………………………………………………..13**

**4.Concluzie………………………………………………………………………………………15**

**4.1 Perspective de viitor………………………………………………………………………16**

**5.Bibliografie……………………………………………………………………………………17**

**INTRODUCERE**

Motivarea alegerii temei

In contextul actual al digitalizarii si automatizarii proceselor din aproape toate domeniile de activitate, gestionarea eficienta a stocurilor reprezinta un element fundamental pentru buna functionare a unui magazin, depozit sau orice alta unitate de tip comercial. O buna organizare a produselor, a cantitatilor disponibile si a preturilor nu doar ca optimizeaza activitatea interna, dar contribuie si la evitarea pierderilor materiale sau a erorilor operationale.

Am ales aceasta tema deoarece consider ca reflecta o problema practica reala, intalnita frecvent in mediile economice si comerciale. Multe afaceri mici si mijlocii nu dispun de un sistem digitalizat pentru urmarirea stocurilor, ceea ce duce la evidente scrise de mana sau in fisiere simple, greu de gestionat. Prin realizarea unei aplicatii software de gestiune a inventarului, am urmarit sa propun o solutie simpla, intuitiva si eficienta, care sa faciliteze procesul de urmarire a produselor aflate in stoc.

**Obiectivele propuse in cadrul lucrarii**

Scopul principal al acestei lucrari este realizarea unei aplicatii simple, dar utile, pentru gestionarea stocului unui magazin. Am ales sa dezvolt acest proiect pentru a pune in practica cunostintele acumulate pana acum legate de programare orientata pe obiect si interfete grafice, folosind limbajul C++ si platforma Qt.

Printre obiectivele concrete pe care mi le-am propus se numara:

Crearea unei aplicatii de tip desktop care sa poata fi folosita local, fara internet, si care sa fie usor de utilizat de oricine.

Implementarea unei interfete grafice (GUI) care sa permita introducerea de produse in stoc, stergerea acestora si afisarea lor intr-un mod organizat.

Organizarea logica a codului prin utilizarea claselor, in special pentru definirea produselor si a stocului (folosind clasele Product si Storage).

Integrarea functionalitatilor in interfata: butoanele trebuie sa reactioneze corect, iar lista de produse sa se actualizeze in timp real.

Asigurarea unei structuri clare a proiectului cu fisiere separate pentru codul logic si partea de interfata, astfel incat sa fie usor de inteles si modificat ulterior.

Testarea aplicatiei pentru a ma asigura ca functioneaza corect indiferent de numarul de produse adaugate sau sterse.

Prin acest proiect mi-am dorit sa invat cum se realizeaza o aplicatie completa, de la idee la produs functional, si sa inteleg mai bine cum se imbina partea de programare cu partea vizuala intr-un proiect real.

**2.TEHNOLOGII UTILIZATE IN REALIZAREA PROIECTULUI**

Pentru realizarea acestei aplicatii am folosit mai multe tehnologii si instrumente care m-au ajutat atat pe partea de programare, cat si pe partea de design si testare.

**2.1 C++**

C++ este un limbaj de programare puternic, de nivel inalt, dar care permite si control detaliat asupra resurselor sistemului. In acest proiect limbajul de programare C++ a fost folosit pentru a implementa logica aplicatiei (operatii cu produse, gestionarea datelor, etc..).

**2.2 Qt Creator**

Pentru partea de interfata grafica am folosit Qt, o biblioteca cross-platform foarte puternica si folosita pe scara larga in aplicatiile comerciale. Cu ajutorul Qt am putut crea ferestre, butoane, campuri de text si alte elemente vizuale.

Am folosit Qt Creator ca mediu de dezvoltare (IDE), deoarece este special conceput pentru proiecte Qt. Acesta vine cu un editor de design vizual care m-a ajutat sa creez rapid interfata fara sa scriu manual codul pentru fiecare element.

**2.3 PROGRAMARE ORIENTATA PE OBIECT (OOP)**

Intreaga aplicatie este structurata folosind conceptele de OOP (programare orientata pe obiect). Am creat clase precum Product, care reprezinta un produs individual, si Storage, care gestioneaza lista de produse. Aceasta abordare face ca aplicatia sa fie mai clara, mai organizata si mai usor de modificat sau extins.

**2.4 CMAKE**

Proiectul foloseste sistemul de build automatizat din Qt Creator, care se bazeaza pe CMake. Acesta se ocupa de compilare, linkare si generarea fisierelor executabile. Totul se face automat cand apas pe „Build” in IDE, ceea ce usureaza mult procesul.

**3.STUDIU DE CAZ**

Aplicatia pe care am realizat-o are ca scop gestionarea stocului de produse dintr-un magazin. Am pornit de la ideea ca, in practica, multe magazine mici folosesc tabele in Excel sau chiar hartie pentru a tine evidenta produselor. Acest lucru este incomod si predispus la greseli. Prin urmare, am decis sa creez o aplicatie desktop care sa permita adaugarea, stergerea si vizualizarea produselor intr-un mod automatizat.

**3.1 PLANIFICAREA APLICATIEI**

Inainte de a incepe scrierea efectiva a codului, am considerat important sa planific modul in care aplicatia va functiona si ce functionalitati trebuie sa includa. Aceasta etapa este esentiala deoarece ofera o directie clara in dezvoltarea aplicatiei si reduce riscul de a face modificari majore pe parcurs.

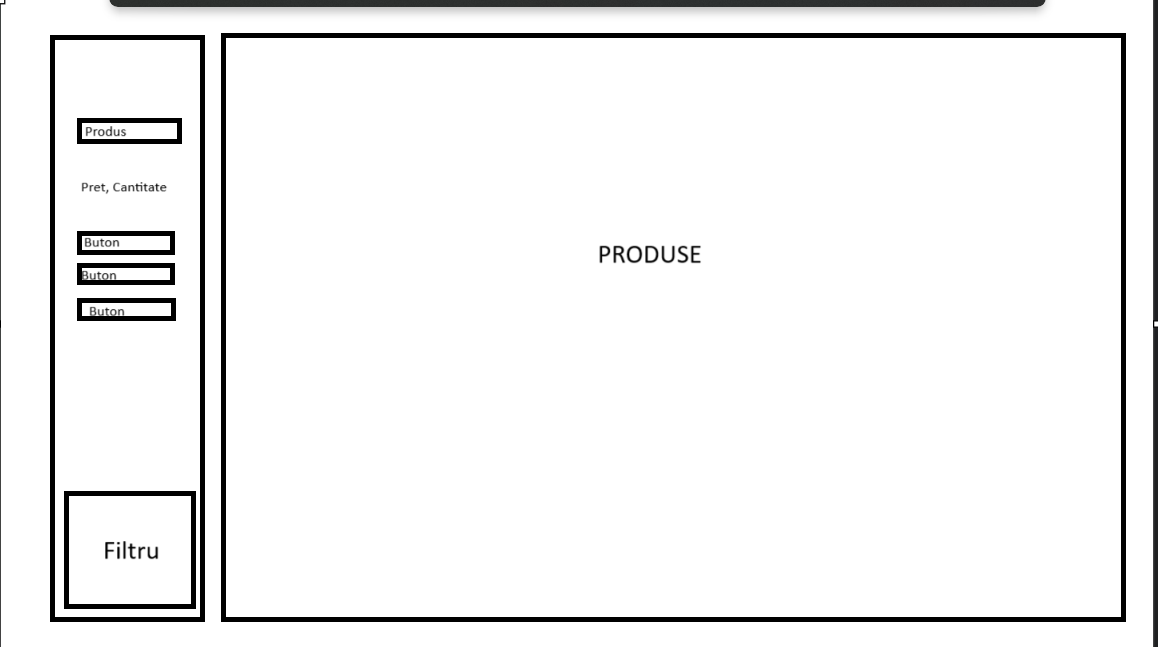


Fig 1 Asezarea aplicatiei

**3.2 Realizarea claselor**

Definirea claselor de baza

Am introdus 2 clase, clasa Product care reprezinta modelul de baza al aplicatiei. Aceasta clasa este responsabila cu definirea unui produs individual din stoc. Fiecare produs are un nume, un pret si o cantitate. Aceasta clasa e foarte importanta pentru ca reprezinta unitatea de baza in toata logica aplicatiei.

A screen shot of a computer program

AI-generated content may be incorrect.

Fig1.2 Header-ul clasei Product

Dupa realizarea fisierului product.h am implementat metodele in product.cpp, fisierul sursa:

A screen shot of a computer code

AI-generated content may be incorrect.

Fig1.3 Codul sursa product

Aceasta clasa ajuta la pastrarea informatiilor despre fiecare produs in mod clar si usor de folosit.

Cea de a doua clasa folosita este storage, actioneaza ca un fel de „baza de date” temporara in memorie. Aici sunt pastrate toate produsele introduse de utilizator.

A screen shot of a computer program

AI-generated content may be incorrect.

Fig1.3 codul header storage

**3.3 Salvarea datelor cu ajutorul JSON**

Pentru a face posibila salvarea datelor si dupa inchiderea aplicatiei si pornirea din nou a acesteia am utilizat JSON (JavaSript Object Notation). Pentru a pastra datele introduce de utilizator intre sesiunile aplicatiei, proiectul utilizeaza formatul JSON. Acesta este un format de text usor de citit si de procesat, atat de catre oameni cat si de calculatoare.

Produsele sunt salvate intr-unf isier numit products.json cu ajutorul metodei savetoFile() care se apeleaza automat dupa fiecare adaugare a unui produs sau a stergere unui produs.

Se creeaza un QJsonArray in care fiecare produs este convertit intr-un QJsonObject, fiecare obiect JSON contine: name( numele produsului), quantity(cantitatea) si price(pretul).

Incarcarea produselor este realizata cu metoda loadFromFile() care se apeleaza in clasa Storage la deschiderea aplicatiei. Aceasta verifica daca fisierul products.json exista, il citeste si il transforma intr-un QJsonDocument, parcurge fiecare QjsonObject si extrage atributele produsele iar apoi creaza obiecte si le adauga in lista products.

**3.4 Interfata grafica a aplicatiei**

Fisierul mainwindow.ui reprezinta fisierul in care desenezi interfata grafica a aplicatiei cu ajutorul Qt Designer. Acesta continue butoane, campuri de text, tabele, label-uri si alte componente vizuale

Acesta functioneaza pe baza faptului ca Qt genereaza automat un fisier C++ numit ui\_mainwindow.h la compilare pe baza .ui-ului, obiectele din UI spre exemplu butoanele vor putea fi accesate in codul sursa, mainwindow.cpp prin ui->.

Pe scurt, acesta reprezinta scheletul visual al aplicatiei mele.

A screenshot of a computer

AI-generated content may be incorrect.

Fig1.4 Qt Designer

**3.5 Declaratia clasei logice mainwindow**

Mainwindow.h reprezinta fisierul in care declari structura clasei MainWindow. Aici definesti ce functii va avea fereastra si ce variabile foloseste storage-ul sau product-ul.

A screen shot of a computer

AI-generated content may be incorrect.

Fig1.5 Codul Header MainWindow

**3.6 Implementarea functiei logice**

Fisierul mainwindow.cpp defineste comportamentul principal al aplicatiei grafice si gestioneaza logica de afisare si manipulare a produselor intr-un tabel. La initializarea ferestrei principale, este configurat un model de tip QStandardItemModel cu trei coloane: Nume, Cantitate si Pret. Acest model este asociat cu un QTableView, care afiseaza lista de produse intr-un mod tabelar, cu randuri selectabile si coloane redimensionabile. Tabelul este configurat pentru a fi doar in citire, iar stilul vizual este personalizat pentru a oferi o interfata moderna, cu culori prietenoase si evidentiere pentru elemente importante.

Constructorul clasei MainWindow stabileste conexiunile dintre semnalele butoanelor din interfata si functiile aferente. De exemplu, butoanele pentru adaugare, stergere, editare, filtrare si resetare a filtrului sunt legate de functiile proprii care se ocupa de logica fiecarei actiuni. In plus, este activata functia care permite completarea automata a campurilor de editare atunci cand utilizatorul face dublu click pe un rand din tabel.

Un comportament important al aplicatiei este preluarea automata a valorilor atunci cand utilizatorul face dublu click pe un rand din tabel. Functia on\_table\_doubleClicked extrage valorile randului respectiv si le introduce automat in campurile de input pentru o editare rapida.



Fig1.6 Exemplu de doubleClicked

Pentru filtrare, on\_filterButton\_clicked preia textul introdus de utilizator in campul de filtrare si apeleaza updateProductList, afisand doar produsele corespunzatoare. In cazul in care utilizatorul doreste sa stearga filtrul, functia on\_resetFilterButton\_clicked goleste campul de text si reincarca toata lista de produse.



Fig1.7 Filter

Functia on\_addButton\_clicked este responsabila de adaugarea unui produs in stoc. Verifica daca numele produsului este completat, apoi creeaza un obiect nou si il adauga in stoc. Dupa aceea, actualizeaza tabelul si bara de stare si goleste campurile de input. In mod similar, functia on\_removeButton\_clicked verifica daca exista o selectie in tabel, extrage numele produsului si il sterge din stoc, actualizand apoi interfata.

A screen shot of a computer program

AI-generated content may be incorrect.

Fig 1.8 Adauga produs/Sterge produs

A screenshot of a computer

AI-generated content may be incorrect.

Fig 1.9 Aplicatia finalizata

**3.7 Rularea aplicatiei**

Pentru a rula aplicatia de gestiune a produselor, este necesar ca mediul de dezvoltare Qt sa fie configurat corect (de exemplu, Qt Creator, sau Visual Studio Code cu suport pentru Qt si qmake). Dupa ce proiectul a fost deschis si compilat cu succes, aplicatia poate fi lansata prin rularea fisierului executabil generat.

La pornire, interfata principala ofera o masa de produse incarcata automat din fisierul local de stocare. Utilizatorul poate adauga produse noi completand campurile din formular si apasand butonul „Adauga”. De asemenea, produsele pot fi editate sau sterse, fie prin selectarea unui rand si utilizarea butoanelor dedicate, fie prin dublu-click pe un produs pentru a prelua valorile sale in formular.

Functionalitatea de filtrare permite cautarea rapida a produselor dupa nume, iar status bar-ul afiseaza in permanenta statistici utile: numarul total de produse, valoarea totala a stocului si numarul produselor cu stoc redus.

Dupa fiecare operatie (adaugare, editare, stergere), aplicatia salveaza automat datele. La inchiderea aplicatiei, toate informatiile sunt persistate local si vor fi disponibile la urmatoarea rulare.

**4.Concluzie**

La nivel arhitectural, aplicatia este bine organizata, folosind clase distincte pentru fiecare responsabilitate. Clasa Product modeleaza fiecare produs in mod individual, Storage gestioneaza intreaga colectie de produse si se ocupa de salvarea acestora, iar MainWindow coordoneaza interfata grafica si interactiunea cu utilizatorul. Aceasta separare clara face ca proiectul sa fie usor de intretinut, extins sau refactorizat pe viitor.

In concluzie, acest proiect demonstreaza o aplicare coerenta si eficienta a principiilor programarii orientate pe obiecte, lucrului cu interfete grafice si manipularii modelelor de date. Aplicatia poate servi atat ca instrument practic, cat si ca punct de plecare pentru dezvoltari software mai avansate, fiind un exemplu clar de echilibru intre simplitate, functionalitate si potential de scalare.

**4.1 Perspective de viitor**

Pe viitor, chiar daca in momentul actual aplicatia ofera functionalitati de baza necesare pentru gestionarea stocului, exista multiple directii de dezvoltare. Un prim pas de dezvoltare ar fi realizarea unei baze de date mai avansate cum ar fi MySQL. Un alt pas pentru dezvoltare este implementarea unui system de autentificare cu conturi de utilizatori si roluri diferite ( administrator, visitator, comparator) fiecare cu drepturi specific. Aceasta functionalitate ar putea da dezvolta alte modificari cumar fi implementarea unui alt ecran pentru a vinde si a cumpara online. In final, se poate avea in conectarea aplicatiei la internet pentru a oferi o varianta web sau portarea aplicatiei pe platformele ANDROID SI IOS.

**Bibliografie:**

1.<https://www.youtube.com/watch?v=R6zWLfHIYJw>

2.<https://www.youtube.com/watch?v=H2ud-ATLIdI&t=1s>

3.<https://doc.qt.io/qtcreator/creator-overview.html>

4.<https://www.w3schools.com/cpp/default.asp>

5. <https://www.youtube.com/playlist?list=PLS1QulWo1RIZiBcTr5urECberTITj7gjA>

6. <https://www.w3schools.com/cpp/cpp_oop.asp>

7. <https://cmake.org/documentation/>

8. <https://www.youtube.com/watch?v=y7Xj-gJW914>