

Lant de florarii

Student: Timis Iulia Georgeana

Grupa: 30236

An 3

1. Enuntul problemei

In cazul acestui proiect, trebuie sa dezvoltam o aplicatie desktop care poate fi utilizata intr-un lant de florarii. In cazul nostru, avem 2 tipuri de utilizatori: angajat si administrator, fiecare avand diferite functii.

Trebuie sa respectam arhitectura Model-View-Presenter.

**Model–view–presenter** ( **MVP** ) este o derivație a modelului [arhitectural model](https://en.wikipedia.org/wiki/Architectural_pattern)[–view–controller](https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller) (MVC) și este folosit mai ales pentru construirea de interfețe cu utilizatorul.

În MVP, *prezentatorul* își asumă funcționalitatea „omul de mijloc”. În MVP, toată logica de prezentare este împinsă către prezentator. [[1]](https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93presenter#cite_note-1)

Diagram

Description automatically generated

1. Instrumente utilizate

Prima etapa in realizarea acestei teme, am creat diagrama de use-case in aplicatia StarUML unde am pus actorii aplicatiei si ce pot face ei.

Diagram, schematic

Description automatically generated

Aici se afla Use-Case-ul problemei. Drept actori, avem un administrator si un angajat, acestia putand face diverse actiuni.

Avem doi actori care pot face diferite functii intre care exista diferite relatii.

De asemenea, tot in acest tool, am facut si diagrama de clase a aplicatiei.

Voi prezenta pachetele clasei pe rand. Acestea au un nume sugestiv: Model, View si Presenter. Fiecare pachet are mai multe clase implementate in aplicatie, acestea avand atributele si operatiile necesare functionarii aplicatiei.

Pachetul Model:

Diagram

Description automatically generated

Pachetul View:  
Diagram

Description automatically generated

Pachetul Presenter:

A picture containing diagram

Description automatically generated

1. Descrierea claselor folosite in aplicatie:

Text

Description automatically generated

Pachetul Model contine 5 clase si 2 interfete.

Prima clasa este “Floare” unde avem caracteristicile unei flori: pretul, culoarea, cantitatea, disponibilitatea si tipul acesteia.

In clasa “Florarie” avem numele unei florarii de unde apartine o floare si o lista de flori.

In clasa “Utilizator” avem datele necesare crearii unui utilizator pentru aplicatia noastra: un nume, o parola si un rol.

In clasa de “PersistentaUtilizator ” avem o lista de utilizatori si diferite metode de lucru cu fisierul XML unde avem stocati utilizatorii. Realizam operatiile CRUD: creare utilizator, actualizare utilizator, stergere utilizator. Utilizatorul va fi citit din fisierul Xml.

In clasa de “PersistentaFlorarie ” avem o lista de flori si una de florarii si diferite metode de lucru cu fisierul XML unde avem stocate florile si florariile. Realizam operatiile CRUD: creare floare, actualizare floare, stergere floare,adaugare floare. De asemenea realizam si operatiile de cautare si filtrare. Floriile si floraria de care apartin se citesc din fisier

Interfata”RaportGeneratorCsv” si “RaportGeneratorJSON” realizeaza salvarea rapoartelor in format .csv si .json. a florilor.

In format .csv fisierul generat arat astfel:

Text

Description automatically generated

In format .json fisierul creat arata astfel:

Text

Description automatically generated

Pachetul Presenter

Acesta contine 3 clase.

Clasa”PresenterAdmin” care realizeaza legatura dintre persistenta utilizatorului din model si interfata pentru administrator.

Clasa”PresenterAngajat” care realizeaza legatura dintre persistenta florariei din model si interfata pentru angajat.

Clasa”PresenterUser” care realizeaza legatura dintre persistenta utilizatorului din model si interfata pentru utilizator.

Pachetul View reprezinta o interfața pasivă care afisează date(modelul) și direcționează comenzile utilizatorului (evenimente ) către prezentator pentru a acționa asupra acestor date.

Cuprinde 3 clase si 3 interfete.

Interfata „InterfataLogin” cuprinde metode pentru pagina de login.

Interfata „InterfataAngajat” cuprinde metode pentru pagina angajatului.

Interfata „InterfataLogin” cuprinde metode pentru pagina administratorului.

Clasa ”ViewUserLogin” realizeaza conexiunea cu componenta de login din presenter si se proiecteaza interfata de log in prin numeroase atribute si metode atribuite butoanelor.

Graphical user interface, text, application

Description automatically generated

Clasa ”ViewAdministrator” realizeaza conexiunea cu componenta de administrator din presenter si se proiecteaza interfata administratorului prin numeroase atribute si metode atribuite butoanelor.

Chart

Description automatically generated with low confidence

Clasa ”ViewAngajat” realizeaza conexiunea cu componenta de angajat din presenter si se proiecteaza interfata angajatului prin numeroase atribute si metode atribuite butoanelor.

Graphical user interface

Description automatically generated

1. Descrierea aplicatiei

Pentru inceput trebuia sa dezvoltam (analiză, proiectare, implementare) o aplicație desktop care poate fi utilizată într-un lanț de florării.

Aplicația va avea două tipuri de utilizatori: angajat al lanțului de florării și administrator. Utilizatorii de tip angajat al lanțului de florării pot efectua următoarele operații după autentificare: vizualizarea tuturor florilor dintr-o florărie selectată, filtrarea florilor după următoarele criterii( disponibilitate, preț, culoare, cantitate) , căutarea unei flori după denumire, operații CRUD în ceea ce privește persistența florilor si salvarea rapoarte/liste cu flori în mai multe formate: csv, json.

Utilizatorii de tip administrator pot efectua următoarele operații după autentificare: operații CRUD pentru informațiile legate de utilizatori.

Codul implementat de mine este intr-un final simplu, dupa multe incercari. Am respectat arhitectura MVP si modelul SOLID pentru a putea schimba aplicatia fara a face modificari majore.

Aplicatia este conceputa pentru a fi utilizata de orice categorie de persoane, atat cei care stiu programare cat si cei care nu. Interfata este simpla si user friendly, din punctul meu de vedere este usor de utilizat. Pentru o posibila imbunatatire, pe viitor as dori sa fac interfata mai usor de utilizat si sa incerc sa restructrez codul mai usor de inteles de la inceput.