



Tema 1 - Sablonul arhitectural MVP

Proiectare Software

Student: Mirisan Octavian

Grupa: 30231

FACULTATEA DE AUTOMATICA
SI CALCULATOARE

Martie 2025

Cuprins

1	Enuntul Problemei	2
2	Instrumente utilizate	2
3	Justificarea limbajului de programare	2
4	Descriere diagrame UML	3
4.1	Diagrama de Use Case	3
4.2	Diagramele de activitate	4
4.3	Diagrama de pachete	7
4.4	Diagrama de clase	8
4.5	Diagrama entitate-relatie	9
5	Descriere aplicatie	10
5.1	Interfata artistilor	10
5.2	Interfata operelor de arta	11

1 Enuntul Problemei

Problema 5

Dezvoltați o aplicație soft care poate fi utilizată într-o galerie de artă pentru gestiunea operelor de artă expuse. Softul va permite:

- ❖ Adăugarea, ștergerea și actualizarea artiștilor care au expuse opere de artă în galeria de artă;
- ❖ Vizualizarea listei tuturor artiștilor care au expuse opere de artă în galeria de artă (pentru fiecare artist se va afișa numele, data nașterii, locul nașterii, naționalitatea, o fotografie și lista tuturor operelor de artă realizate de artist și expuse în acest muzeu);
- ❖ Căutarea unui artist după nume;
- ❖ Adăugarea, ștergerea și actualizarea operelor de artă expuse în galeria de artă;
- ❖ Vizualizarea listei tuturor operelor de artă expuse în galeria de artă sortate după artist (vizualizarea include și redarea unor imagini cu operele de artă; între 1 și 3 imagini pentru fiecare operă de artă);
- ❖ Căutarea unei opere de artă după titlu;
- ❖ Filtrarea listei operelor de artă după următoarele criterii: artist, preț sau tipul operei de artă;
- ❖ Salvarea listei cu operele de artă expuse în fișiere de tip csv și doc.

Figura 1: Enuntul problemei

2 Instrumente utilizate

Instrumentele utilizate în procesul de dezvoltare a aplicatiei software includ:

- **JavaFX:** framework-ul JavaFX a fost utilizat pentru crearea interfeței grafice a aplicatiei, fiind ideal pentru aplicații desktop interactive
- **Scene Builder:** a fost folosit pentru proiectarea interfeței grafice în format FXML. Acest instrument permite crearea rapidă și eficientă a layout-urilor pentru interfețele dedicate artiștilor și operelor de artă.
- **IntelliJ IDEA:** un IDE specializat pe limbajul Java, utilizat pentru scrierea, depanarea și testarea codului aplicatiei. Funcționalitățile sale avansate au îmbunatatit productivitatea în timpul dezvoltării.
- **SQLite:** a fost ales ca sistem de gestiune a bazelor de date relationale datorită simplității și portabilității sale. Baza de date a fost utilizată pentru persistența artiștilor, operelor de artă, precum și imaginilor asociate operelor.

3 Justificarea limbajului de programare

Limbajul de programare ales pentru implementarea aplicatiei este **Java**, iar pentru interfața grafică s-a utilizat **JavaFX**.

Motivele pentru care am ales aceste limbi sunt:

- **Suport pentru programare OOP:** structura aplicatiei, bazată pe pachete precum **Model, Repository, View, Presenter**, se aliniază în mod natural cu paradigma object oriented și cu tot ceea ce presupune ea (mostenire, implementare, etc.)
- **Integrare cu JavaFX:** JavaFX, ca parte a ecosistemului Java, a fost alegerea naturală pentru crearea interfeței grafice. Aceasta oferă suport pentru componente vizuale complexe, cum ar fi tabelele (pentru afișarea artiștilor și a operelor), ImageView-urile (pentru afișarea imaginilor în fereastra), TextField-uri, Label-uri, etc.
- **Gestionarea bazelor de date:** Java oferă suport nativ pentru lucrul cu baze de date, prin **JDBC** (Java Database Connectivity). În acest proiect, JDBC a fost folosit pentru a interacționa cu SQLite, permitând astfel operații CRUD.

4 Descriere diagrame UML

4.1 Diagrama de Use Case

Diagrama de use case descrie interactiunile utilizatorului cu aplicatia de gestionare a unei galerii de arta. Utilizatorul, reprezentat ca actor, poate efectua actiuni precum adaugarea, stergerea si actualizarea artistilor si operelor de arta, precum si vizualizarea listelor acestora. De asemenea, utilizatorul are posibilitatea de a cauta opere de arta, de a filtra liste si de a salva liste de opere pentru export. Sistemul, denumit "Aplicatie", incapsuleaza toate aceste functionalitati, oferind o interfata centralizata pentru gestionarea datelor galeriei.

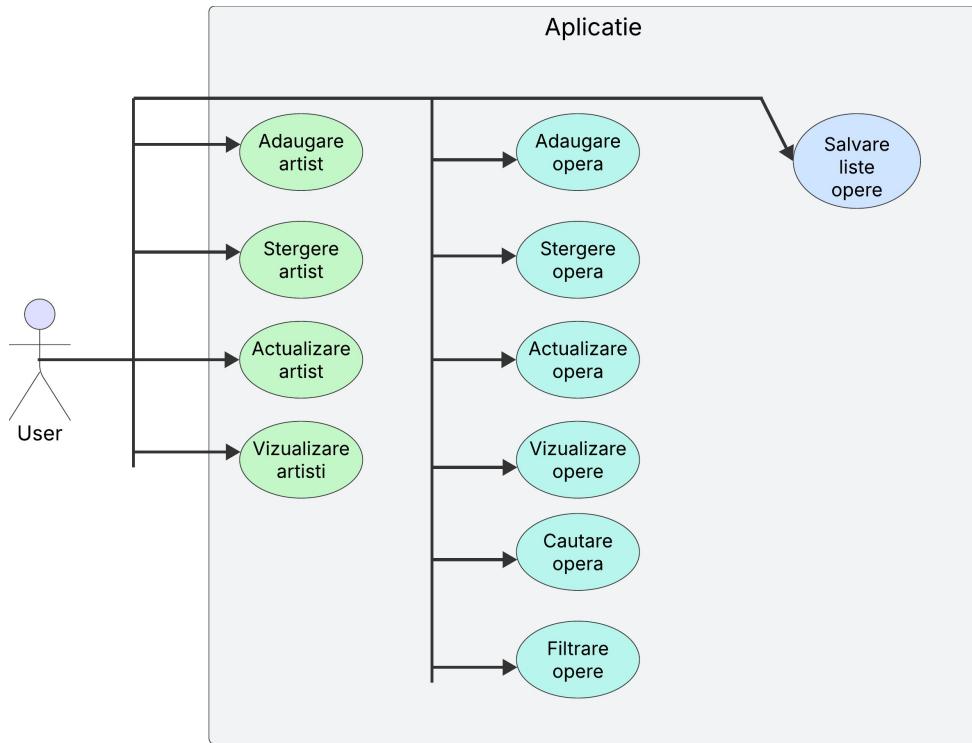


Figura 2: Diagrama de Use Case

4.2 Diagramele de activitate

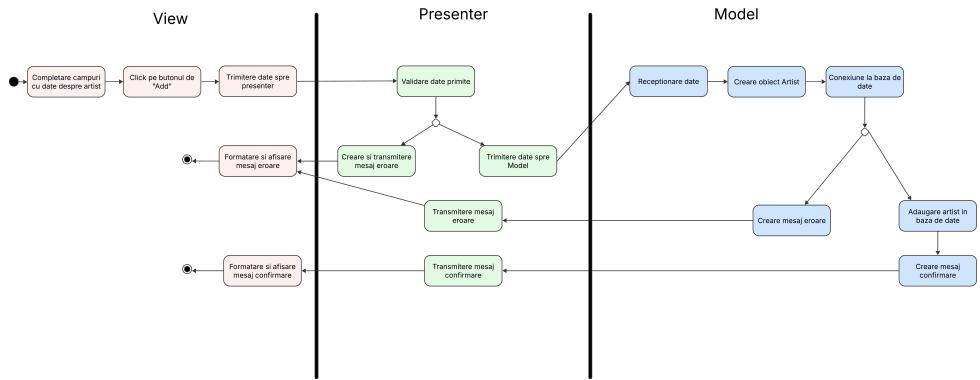


Figura 3: Adaugare artist in baza de date.

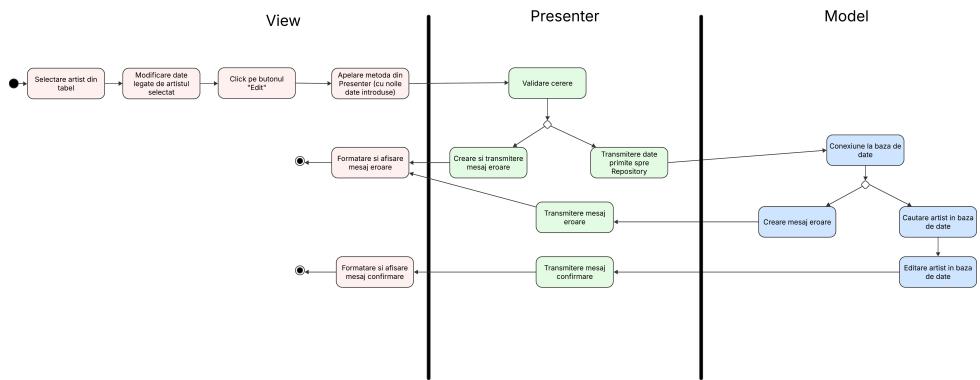


Figura 4: Editare artist in baza de date.

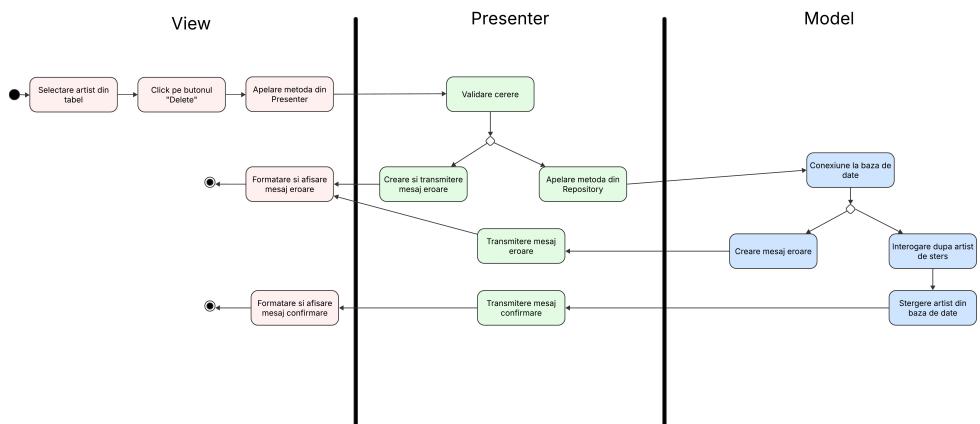


Figura 5: Stergere artist din baza de date.

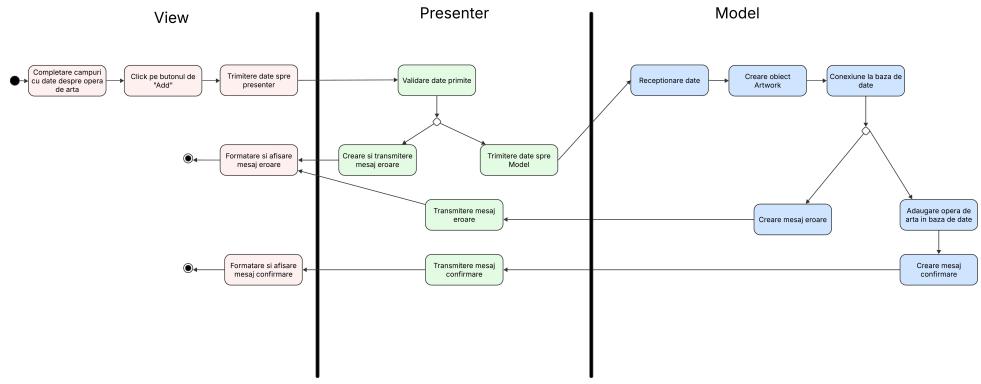


Figura 6: Adaugare opera de artă în baza de date.

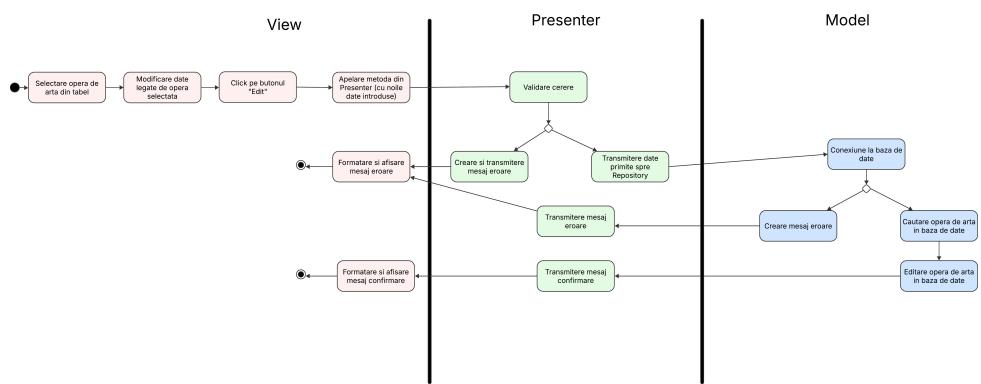


Figura 7: Editare opera de artă în baza de date.

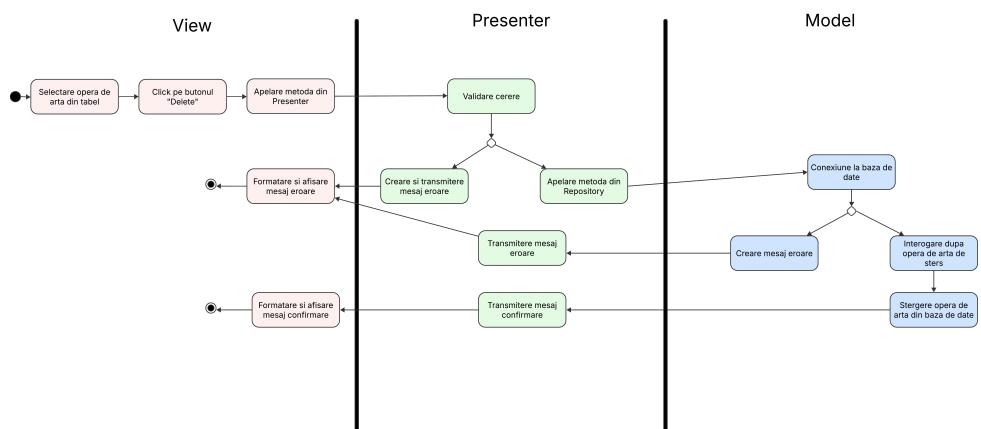


Figura 8: Stergere opera de artă din baza de date.

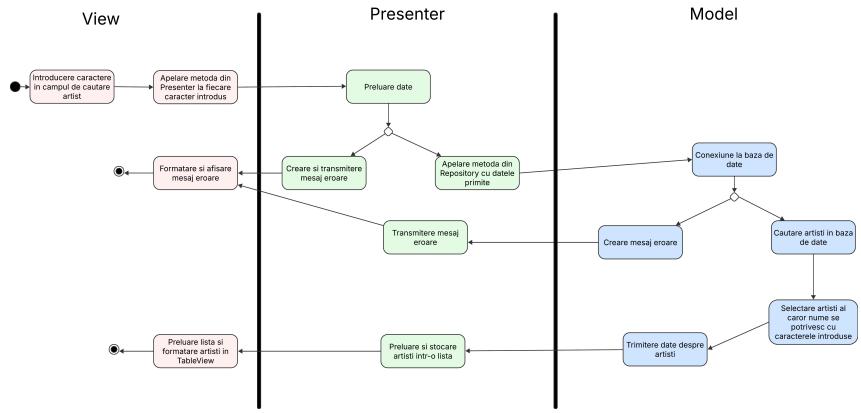


Figura 9: Cautare artist dupa nume.

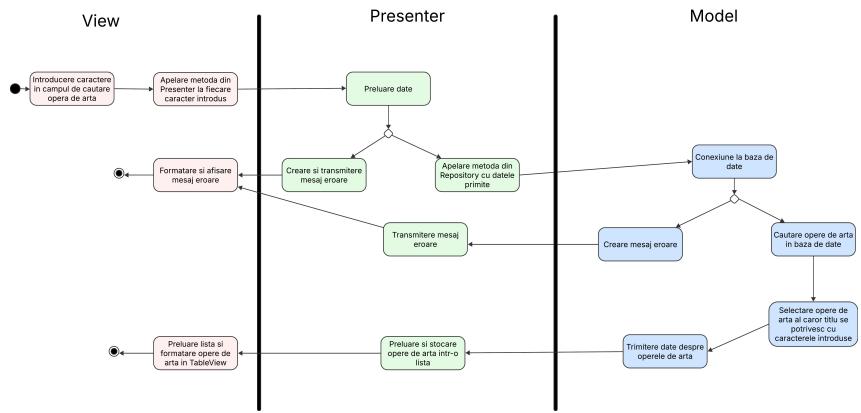


Figura 10: Cautare opera de arta dupa titlu.

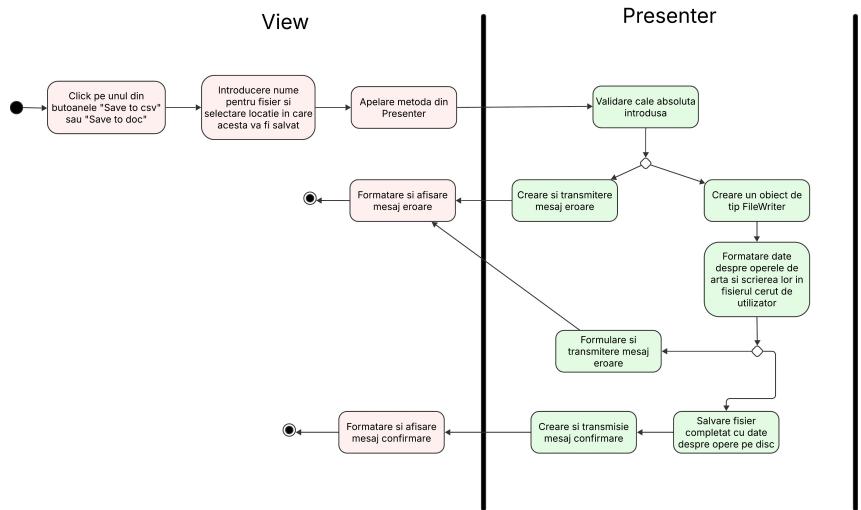


Figura 11: Export opere de arta in format .csv sau .doc

4.3 Diagrama de pachete

Diagrama de pachete ilustreaza organizarea structurala a aplicatiei de gestionare a galeriei de arta, evidentiind dependentele dintre pachete. Pachetul Model contine clasele de baza, cum ar fi Artist, Artwork si ArtworkImage, si este dependent de pachetul Repository, care gestioneaza accesul la baza de date prin clase precum ArtistRepo, ArtworkRepo si ArtworkImageRepo. Pachetul Presenter (denumit Prezenter in diagramă) actioneaza ca un intermediar intre Model si View, coordonand logica aplicatiei prin clasa GalleryPresenter. Pachetul View este responsabil pentru interfata grafica, utilizand JavaFX pentru a afisa si interactiona cu utilizatorul, si depinde de Presenter pentru a primi datele procesate.

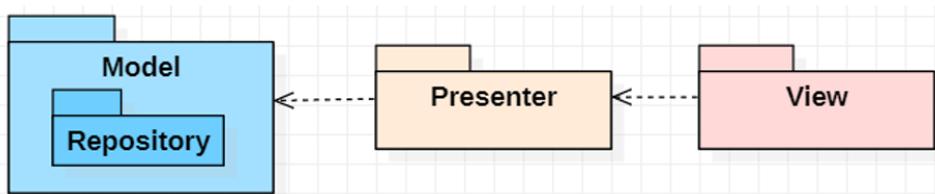


Figura 12: Diagrama de pachete a aplicatiei

4.4 Diagrama de clase

Explicatii:

- **Pachetul Model:** este responsabil de modelarea si definirea entitatilor principale ale aplicatiei: Artist, Artwork, si ArtworkImage
- **Pachetul Repository:** este utilizat pentru persistenta datelor. Clasele din acest pachet au acces direct la baza de date si permit efectuarea de operatii CRUD
- **Pachetul View:** are responsabilitatea de a asigura functionarea interfetei grafice a utilizatorului. Clasa GalleryPresenter reprezinta functionarea pachetului View, iar fisierul main-window.fxml reprezinta fisierul corespunzator interfetei grafice modelate in soft-ul SceneBuilder.
- **Pachetul Presenter:** are rolul de intermediar intre interfata utilizatorului (definita in View) si modelul de date (definit in Model). Clasa GalleryPresenter contine logica de business a aplicatiei.

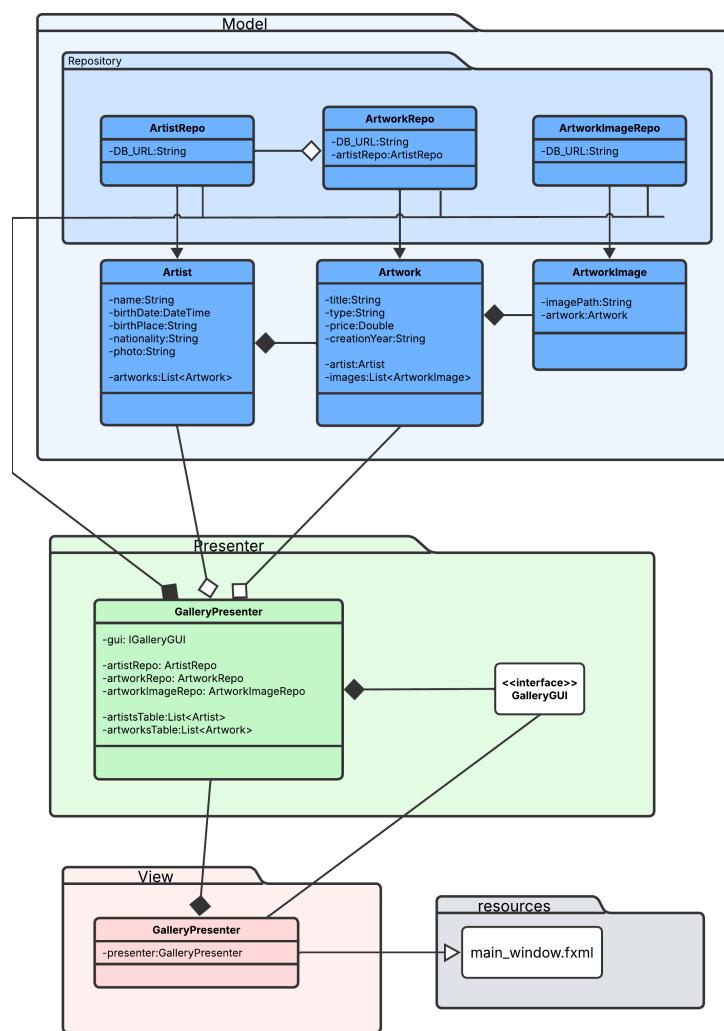


Figura 13: Diagrama de clase

4.5 Diagrama entitate-relatie

Relatiile intre entitati sunt in felul urmator:

- Un artist poate crea **una sau mai multe** opere de arta.
- O opera de arta apartine **cel mult** unui artist.
- O opera de arta contine **una sau mai multe** imagini.
- O imagine este asociata **unei singure** opere de arta.

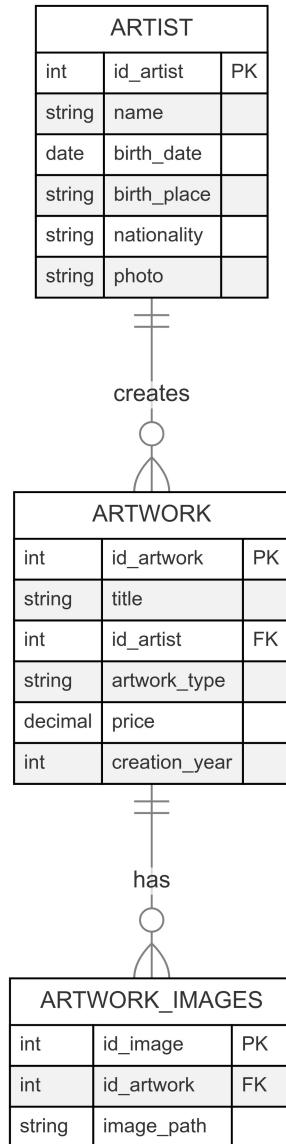


Figura 14: Diagrama entitate-relatie

5 Descriere aplicatie

5.1 Interfata artistilor

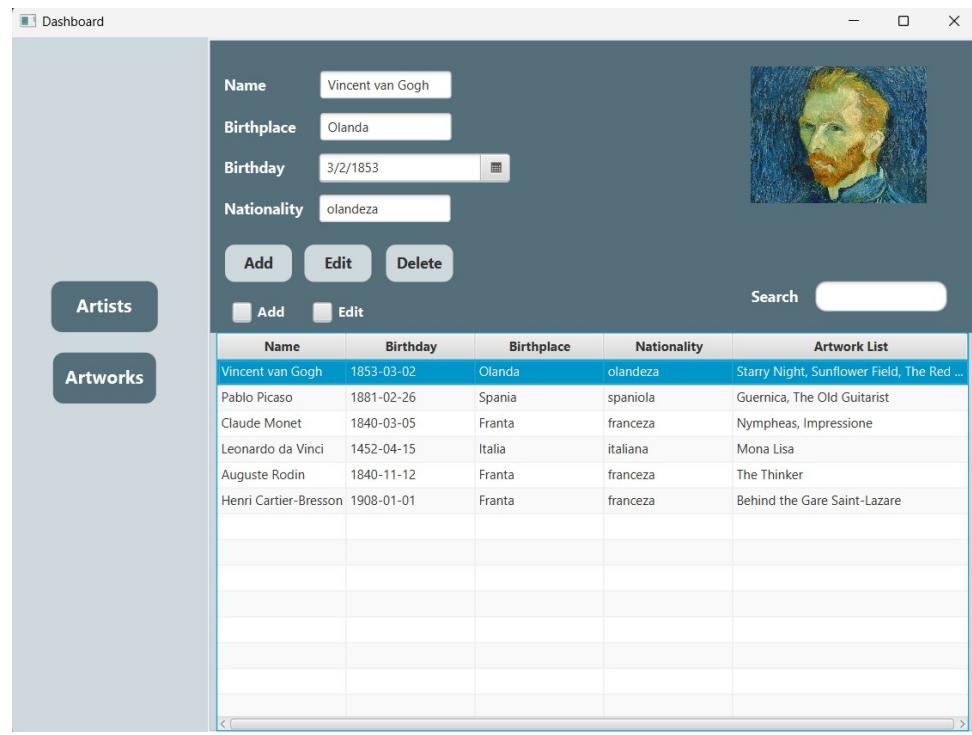


Figura 15: Interfata grafica pentru artiști

5.2 Interfata operelor de arta

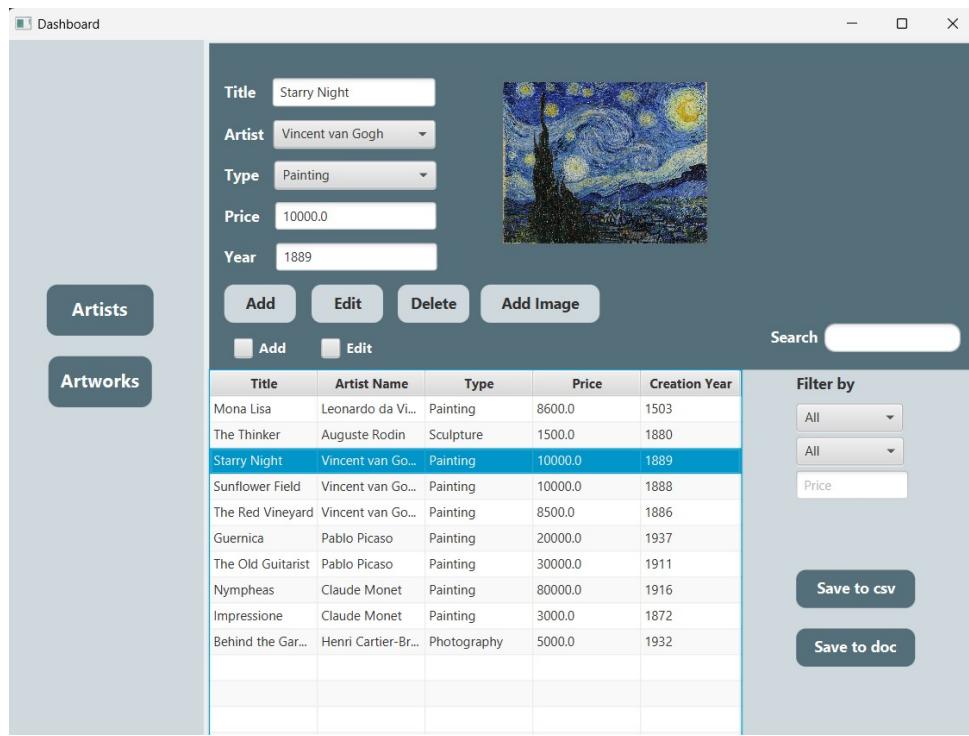


Figura 16: Interfata grafica pentru operele de arta