**Documentație**

**Sistem de gestiune pentru filme și seriale**

Aplicația implementată constituie un sistem de gestionare pentru filme și seriale, dând posibilitatea utilizatorului să caute, adauge, afișeze și filtreze filmele, respectiv serialele pe care le dorește, prin intermediul unui meniu interactiv.

**Laborator 1 - Introducere în Java (output, tipuri de valori, funcții)**

Proiectul începe cu utilizarea metodelor fundamentale din Java, precum afișarea informațiilor către utilizator folosind System.out.println. În plus, funcționalitatea aplicației este modularizată prin intermediul mai multor metode, fiecare realizând operații specifice, cum ar fi adăugarea unui film sau listarea colecției. Tipurile de date de bază, precum String, int și double, sunt folosite pentru a defini atributele obiectelor.

**Laborator 2 - Introducere în Java (input, for, while, switch, if)**

Aplicația permite introducerea de date de la utilizator utilizând clasa Scanner. De exemplu, utilizatorul poate introduce detalii despre un film sau serial. Structurile de control precum if-else, switch-case, și buclele for și while sunt folosite extensiv pentru validarea datelor și gestionarea meniurilor. Un exemplu este utilizarea unui while pentru navigarea printr-un meniu interactiv.

**Laborator 3 - Colecții Java (Array, List, Map)**

Am utilizat o colecție de tip ArrayList pentru a gestiona lista de filme și seriale, implementând metode precum sortarea și filtrarea. De exemplu, metoda filterByGenre(String genre) returnează o listă de filme care aparțin unui anumit gen.

**Laborator 4 - Clase Java (clasă cu atribute și metode)**

Am creat clasele Film și Serial, fiecare având atribute specifice și metode care definesc comportamentul acestora. Clasa Film include atribute precum titlu, gen, an, și rating, și metode pentru filtrarea filmelor după gen și rating. Clasa Serial conține atribute precum titlu și sezoane, și o metodă de filtrare a serialelor după numărul de sezoane. Aceste clase respectă principiile de bază ale programării orientate pe obiect, oferind o structură clară și ușor de extins.

**Laborator 5 - Moștenire în Java, clase abstracte (să evidențiați relația dintre clase**

Am implementat moștenirea prin clasa abstractă Media, din care clasele Film și Serial moștenesc comportamentele și atributele comune, cum ar fi titlu și getDetalii(). Această abordare promovează reutilizarea codului și face ca structura aplicației să fie mai flexibilă. Clasa Media este abstractă deoarece nu poate fi instanțiată direct, iar metodele sale sunt implementate diferit în funcție de tipul de media (film sau serial).

**Laborator 6 - Interfețe în Java**

Am folosit interfața Filtrabil, care definește metoda filtreaza(), pentru a asigura că atât clasele Film, cât și Serial pot implementa o logică de filtrare. Această abordare permite extinderea aplicației prin adăugarea de noi tipuri de media care pot implementa această interfață și astfel beneficia de funcționalitățile de filtrare deja existente. Interfețele sunt utile pentru definirea comportamentului fără a impune o ierarhie de clase strictă.

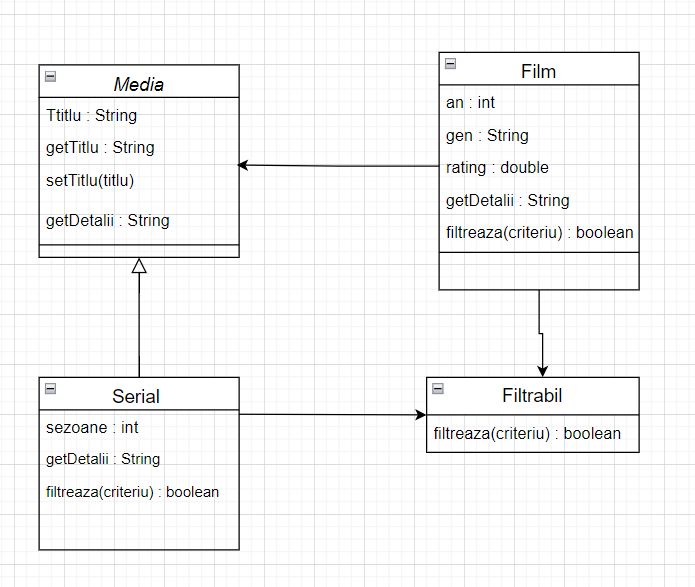
**Laborator 7 - Teste pentru fiecare metodă**

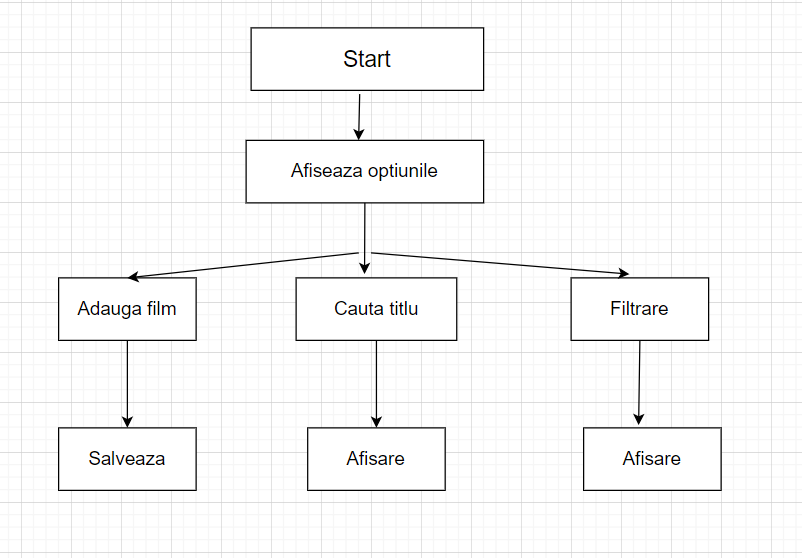
Pentru a asigura funcționarea corectă a aplicației, am implementat teste unitare folosind framework-ul JUnit. Testele pentru clasele Film și Serial verifică dacă metodele de filtrare funcționează corect și dacă metodele de obținere a detaliilor produc rezultatele așteptate. De asemenea, am testat comportamentele aplicației în funcție de diverse condiții de intrare, ceea ce contribuie la creșterea fiabilității și calității codului.

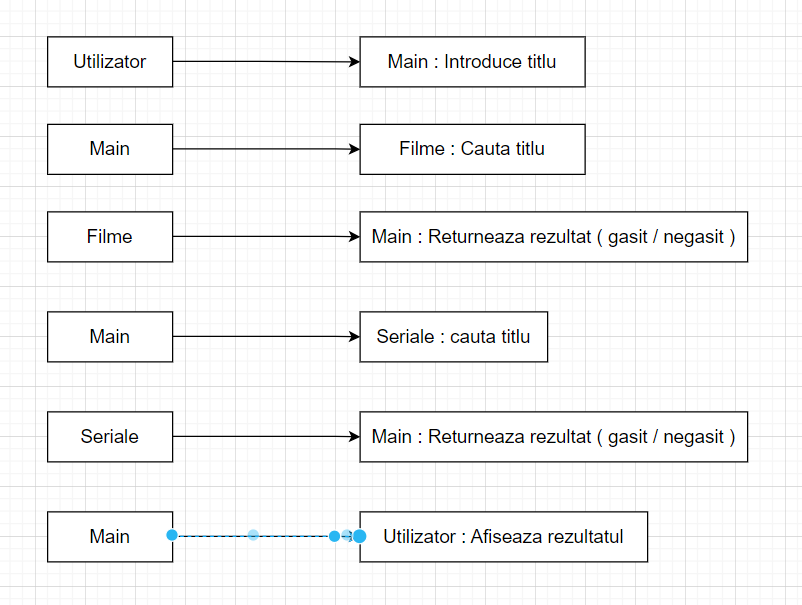
**Laborator 8 - Persistența datelor (salvare în fișiere .txt, .json sau alt format)**

Pentru a permite utilizatorului să salveze și să încarce datele, am implementat un mecanism de persistare a datelor folosind fișiere text. Clasa PersistentaTxt este responsabilă pentru citirea și scrierea datelor în fișiere .txt, astfel încât la repornirea aplicației datele anterioare să fie încărcate automat. Acest lucru este esențial pentru a păstra starea aplicației între sesiuni de utilizare.

**Laborator 9 – Realizați 3 diagrame UML**

****

****

****