# Trifan Elena-Iuliana

An II, ULBS, specializarea Informatică

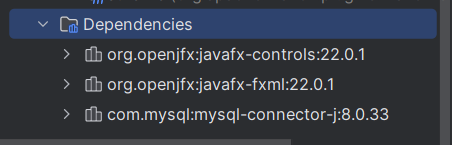
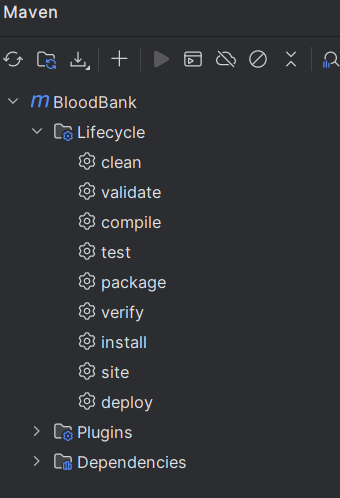
## Blood Blank

Aplicația "Blood Bank" are ca scop crearea unei interfețe de utilizator prin care utilizatorii pot interacționa cu baza de date a unei bănci de sânge/ centru de transfuzii. Această aplicație poate include funcționalități cum ar fi gestionarea donatorilor, verificarea stocului de sânge, și administrarea cererilor de sânge, însă în stadiul actual proiectul poate să afișeze lista cu utlizatorii centrului de transfuzii, să adauge sau să șteargă dintre aceștia.

Tehnologii Utilizate

* JavaFX: Folosit pentru crearea interfeței grafice a aplicației desktop, utilizând fișiere FXML și CSS pentru stilizare.
* JDK 22 (Oracle): Platforma Java Development Kit folosită pentru dezvoltarea și rularea aplicației.
* FXML: utilizat pentru a defini structura interfeței grafice pentru o separare clară a logicii de prezentare și a codului.

- Maven: un instrument de gestionare a proiectului și a dependențelo care gestionează bibliotecile externe (JavaFX) Aceasta oferă o structură standard cu directoare și resurse.



- My SQL : Aplicația se conectează la baza de date folosind JDBC mysql-connector-java (Java Database Connectivity) care permite aplicației să execute interogări SQL și să actualizeze baza de date=**Funcționalitățile CRUD (Create, Read, Update, Delete):**

* **Adăugarea unui Donator**: Utilizatorul introduce datele donatorului, care sunt apoi inserate în baza de date printr-o procedură stocată.
* **Vizualizarea Donatorilor**: Aplicația interoghează baza de date și afișează lista donatorilor.
* **Actualizarea Stocului de Sânge**: Utilizatorul introduce noile date de stoc, iar aplicația actualizează baza de date folosind procedura stocată corespunzătoare.
* **Vizualizarea Stocului de Sânge**: Aplicația interoghează și afișează informațiile despre stocul de sânge din baza de date.

### Concepte Java Utilizate

### - Programarea Orientată pe Obiecte (OOP): Clase și Obiecte: Proiectul folosește clase pentru a modela entități precum Main

### - JavaFX:Folosim fișiere FXML pentru a defini structura și layout-ul interfeței grafice. În exemplul nostru, main\_view.fxml este utilizat pentru a descrie un layout simplu cu un VBox care conține un Label și un Button.

### - Event Handling: Gestionarea evenimentelor precum click-uri de buton, utilizând handleri definiți în Java

### - Gestionarea Dependențelor cu Maven:( JavaFX, MySQL Connector) . De asemenea, realizează automat procesul de build.

### - Gestionarea Excepțiilor- pentru erorile care pot apărea în timpul execuției programului.De exemplu, a fost procesată încărcarea fișierelor FXML, care ar putea eșua dacă fișierul nu este găsit sau conține erori.

### Structura Proiectului

### src/main/java: Conține fișierele sursă Java.

### com.bloodbank.Main: Clasa principală care lansează aplicația.

### src/main/resources: Conține resursele proiectului, cum ar fi fișierele FXML și CSS.

### main\_view.fxml: Fișierul FXML care definește interfața grafică a aplicației.

### pom.xml: Fișierul de configurare Maven, unde sunt definite dependențele și configurațiile de build

CLASELE CREATE

Src/main/java

 **Main.java**: clasa principală care va inițializa aplicația. Metoda start() este punctul de intrare pentru aplicația JavaFX și este responsabilă pentru inițializarea interfeței grafice și afișarea ferestrei principale. com.bloodbank.

 **AuthenticationWindow.java**: fereastra de autentificare pentru utilizatori. solicită utilizatorului să introducă un nume de utilizator și o parolă, verifică dacă acestea sunt corecte și, în caz favorabil, deschide o noua fereastră pentru operațiuni CRUD com.bloodbank

 **CRUDWindow.java**: fereastra pentru operațiuni CRUD (Create, Read, Update, Delete). com.bloodbank

 **DatabaseConnector.java**: gestionează conexiunea la baza de date. com.bloodbank.db

 **UtilizatorDAO.java - DAO (Data Access Object)** clasa UtilizatorDAO, care este responsabilă pentru operațiunile de bază de acces la date (Data Access Object - DAO) legate de obiectele de tip Utilizato. Această clasă folosește SQL pentru a interacționa cu baza de date și efectuează operații CRUD (Create, Read, Update, Delete) pe tabelul utilizatori. Permite autentificarea utilizatorilor (getUtilizatorByUsernameAndPassword), adăugarea de noi utilizatori (addUtilizator), actualizarea informațiilor unui utilizator existent (updateUtilizator) și ștergerea unui utilizator (deleteUtilizator). Fiecare metodă gestionează eventualele erori SQL prin blocuri try-catch

com.bloodbank.dao

 **Utilizator.java** Clasa model care reprezintă un utilizator com.bloodbank.model

 **UtilizatorService.java face legătura între prezentare (UI) și baza de date.** com.bloodbank.service

Structura proiectului

