Uffizi Gallery Museum

Man Alexandra GRUPA 10LF332

Cuprins

1. Argumentul lucrării 2
2. Descrierea aplicației 2
2.1 Baza de date 2
2.2 Clasele din Java 4
2.3 Interfețe Java 6
2.4 Interfață grafică JavaFX 7
3 Detalii tehnice de implementare 10
4 Cerințe Hard și Soft 10
5 Webgrafie 10

1. Argumentul lucrării

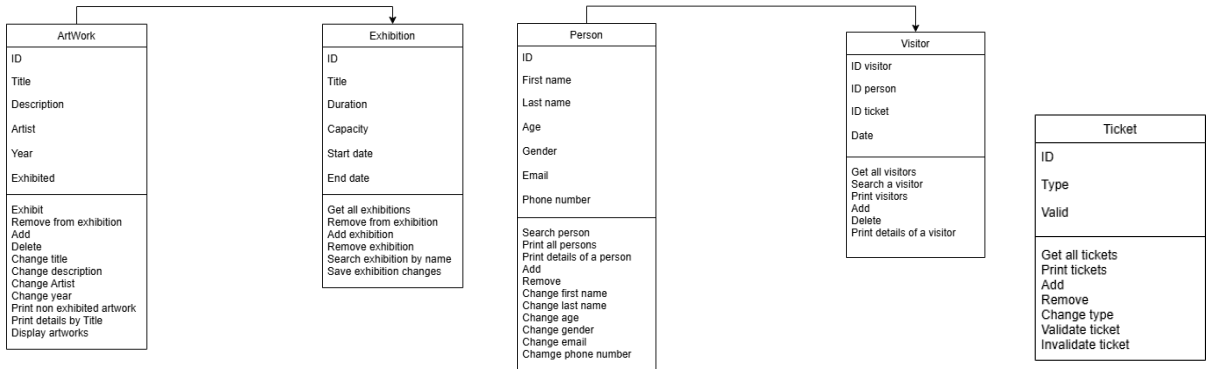
Bazele de date sunt coloana vertebrală a tehnologiei moderne. Fară să ne dăm seama acestea sunt prezente peste tot in viața noastră de zi cu zi, de la Instragram la Booking si E-learning. O bază de date stochează o mulțime vastă de informații și ușurează navigarea și organizarea acestora. Din aceste motive mi-am ales ca tema de proiect gestionarea unui muzeu de artă.

2. Descrierea aplicației

Pentru a realiza proiectul eficient și într-un mod cât mai clar l-am structurat în trei categorii: realizarea bazei de date a muzeului, implementarea funcționalităților pe baza de date în limbaj Java și crearea unei interfețe grafice în limbaj JavaFX.

2.1 Baza de date

În mod evident, în prima fază a proiectului am creat baza de date aferentă unui muzeu de artă. Am lucrat în pgAdmin și am folosit limbajul SQL. Pentru a avea o bază de date cât mai realista am adugat următoarele tabele:



(1)

Acestea au fost create cu ajutorul următorului cod:

```
-- Tabelul pentru ArtWork
CREATE TABLE ArtWork (
    ID INT PRIMARY KEY,
    Title VARCHAR(255) NOT NULL,
    Artist VARCHAR(255),
    Year INT,
    ExhibitionID INT,
    FOREIGN KEY (ExhibitionID) REFERENCES
Exhibition(ID));
```

```
-- Tabelul pentru Exhibition
CREATE TABLE Exhibition (
    ID INT PRIMARY KEY,
    Name VARCHAR(255) NOT NULL,
    StartDate DATE,
    EndDate DATE
);
```

```sql
-- Tabelul pentru Person
CREATE TABLE Person (
    ID INT PRIMARY KEY,
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL
);

-- Tabelul pentru Visitor, care este o extensie a Person
CREATE TABLE Visitor (
    ID INT PRIMARY KEY,
    FOREIGN KEY (ID) REFERENCES Person(ID)
);

-- Tabelul pentru Ticket
CREATE TABLE Ticket (
    ID INT PRIMARY KEY,
    VisitorID INT,
    ExhibitionID INT,
    PurchaseDate DATE,
    Price DECIMAL(10, 2),
    FOREIGN KEY (VisitorID) REFERENCES Visitor(ID),
    FOREIGN KEY (ExhibitionID) REFERENCES Exhibition(ID)
);
```

Pe urmă am inserat date în tabelele create pentru a avea pe ce date lucra. Inserarea se face astfel:

```sql
INSERT INTO ArtWork (ID, Title, Description) VALUES
(2, 'The Persistence of Memory', 'A surreal artwork with melting clocks'),
(3, 'The Sirens and Ulysses', 'The sirens were famous for the beauty of their singing, which would lure sailors to their deaths. Ulysses wanted to hear their song, so had his crew lash him to the ship mast under strict orders not to untie him, after which they blocked their ears until they were safely out of range of the island.'),
(4, 'The Garden of Earthly Delights', 'Regarded by scholars as his most puzzling work, The Garden of Earthly Delights provides a visual representation which expresses the fears that dominated life in the Middle Ages-the insatiable weakness of man for not resisting sinful physical temptation, and eternal damnation in hell as just punishment of lustful human.'),
(5, 'The Creation of Adam', 'A fresco painting by Michelangelo on the ceiling of the Sistine Chapel.'),
(6, 'Girl with a Pearl Earring', 'Going by various names over the centuries, it became known by its present title towards the end of the 20th century because of the earring worn by the girl portrayed there.'),
(7, 'MONA LISA', 'description');
```

Asemănător se fac inserțiile și în celelalte tabele.

2.2 Clasele din Java

După ce baza de date a fost creată am făcut un proiect de tip Maven în Java și am început să construiesc clasele ajutătoare după schema (1) prezentată mai sus. Am avut grijă ca tipul de date si membrii claselor să fie aceleași ca în baza de date.

```
public class Artwork implements ArtworkActions {  5 usages  1 inheritor



    // Attributes
    private int idExhibition;  3 usages
    private String title;  3 usages
    private String description;  2 usages
    private String artist;  3 usages
    private int year;  2 usages
    private boolean exhibited;  2 usages



    // Constructors
    public Artwork(int idExhibition,String title, String description, String artist, int year, boolean exhibited) {
        this.title = title;
        this.description = description;
        this.artist = artist;
        this.year = year;
        this.exhibited = exhibited;
        this.idExhibition = idExhibition;

    }
    public Artwork() {}  10 usages
```

Următorul pas a fost conectarea la bază de date din proiectul meu Maven, care a fost facuta prin adaugarea în fișierul *prom.xml* a dependențelor postgres:

```
<groupId>com.example</groupId>
<artifactId>museum-database</artifactId>
<version>1.0-SNAPSHOT</version>

<properties>
    <javafx.platform>win</javafx.platform>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
</properties>

<dependencies>
    <dependency>
        <groupId>org.postgresql</groupId>
        <artifactId>postgresql</artifactId>
        <version>42.5.0</version>
    </dependency>
```

Mai apoi am creat o clasă ce face conexiunea dintre baza mea de date și proiectul meu local:

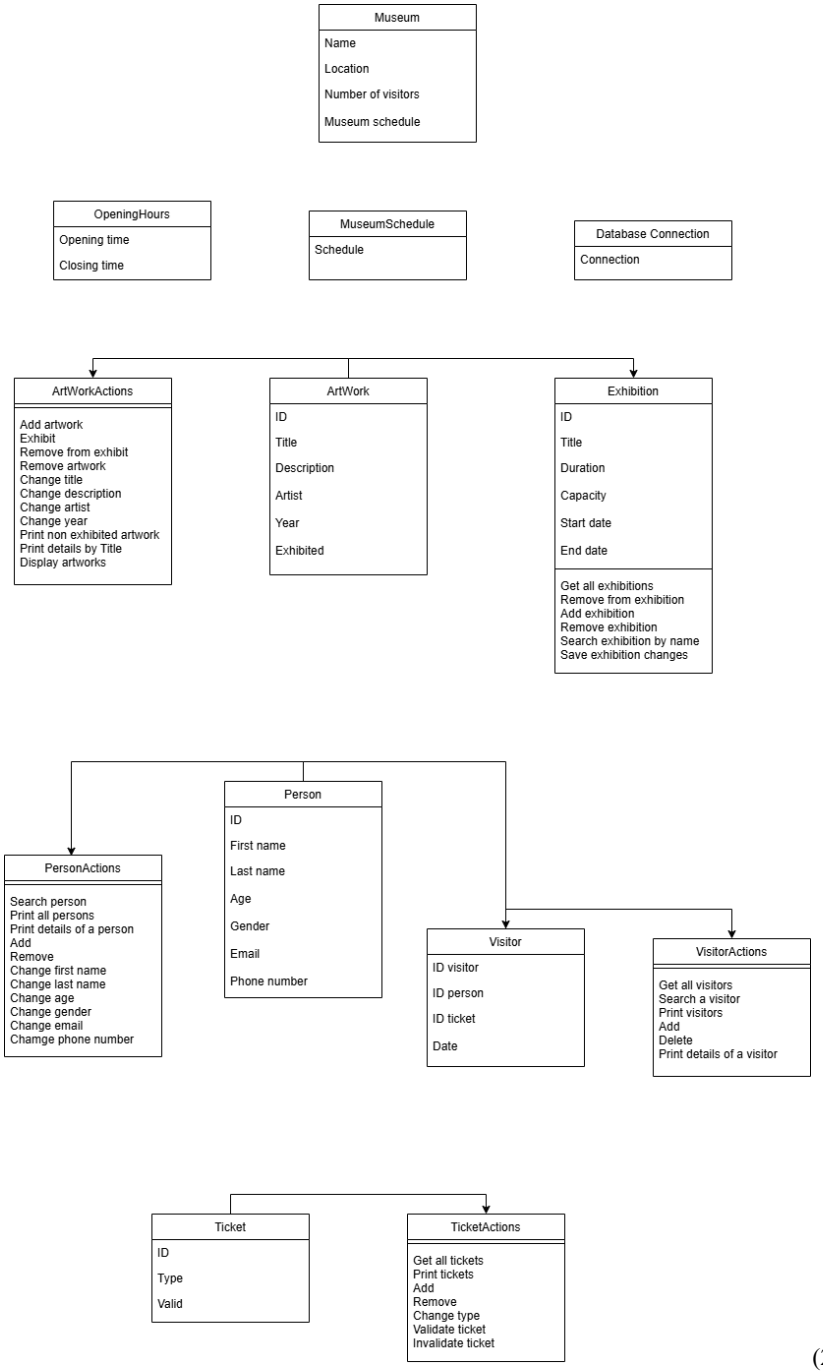
```java
package models;
import ...

public class DatabaseConnection {
    public static void main(String[] args) {
        // Connect to PostgreSQL
        String url = "jdbc:postgresql://localhost:5433/Museum"; // Name of database
        String user = "postgres";  // My PostgreSQL user
        String password = "1q2w3e";  // PostgreSQL password

        try (Connection connection = DriverManager.getConnection(url, user, password)) {
            System.out.println("Connected successfully to database!");

        } catch (SQLException e) {
            System.out.println("Error connecting to database : " + e.getMessage());
        }
    }
    public static Connection connect() {  43 usages
        // Detaliile conexiunii la baza de date
        String url = "jdbc:postgresql://localhost:5433/Museum";
        String username = "postgres";
        String password = "1q2w3e";

        try {
            Class.forName( className: "org.postgresql.Driver");
            return DriverManager.getConnection(url, username, password);
        } catch (ClassNotFoundException | SQLException e) {
            System.out.println("Error connecting: " + e.getMessage());
            return null;
        }
    }
}
```

După ce m-am conectat la baza de date am început să implementez diferite metode în clase pentru a manipula cu ușuritate datele. Codul este scris în limbajul de programare Java, însă conține și interogări SQL. Mai jos este o metodă ce schimbă statutul unei opere de artă:

```java
@Override  1 usage
public void exhibit(String title) {
    System.out.println("Attempting to expose the artwork '" + title + "'.");

    Connection connection = DatabaseConnection.connect();
    if (connection != null) {
        try {
            String updateQuery = "UPDATE Artwork SET exhibited = TRUE WHERE title = ? AND exhibited = FALSE";
            PreparedStatement preparedStatement = connection.prepareStatement(updateQuery);
            preparedStatement.setString( parameterIndex: 1, title);

            int rowsAffected = preparedStatement.executeUpdate();
            if (rowsAffected > 0)
                System.out.println("The artwork '" + title + "' has been exposed and updated in the database.");
             else
                System.out.println("No matching unexposed artwork found for '" + title + "'.");

            preparedStatement.close();
        } catch (SQLException e) {
            System.out.println("Error updating the database: " + e.getMessage());
        } finally {
            try {
                connection.close();
            } catch (SQLException e) {
                System.out.println("Error closing the connection: " + e.getMessage());
            }
        }
    } else {
        System.out.println("Failed to connect to the database.");
    }
}
```

2.3 Interfețe Java

Pentru o organizare cât mai bună am implementat patru interfețe Java care se ocupă de apelarea metodelor: *ArtWorkActions, PersonActions, TicketActions și VisitorActions*.



(2)

Structura unei interfețe Java arată astfel:

```java
package interfaces;
import java.sql.Connection;

public interface ArtworkActions {  2 usages  2 implementations
    void addArtworkToDatabase(Connection connection);  1 usage  1 implementation
    void exhibit(String title);  1 usage  1 implementation
    void removeFromExhibit(String title);  1 usage  1 implementation
    void removeArtwork(String title);  1 usage  1 implementation
    void changeTitle(String oldTitle, String newTitle);  1 usage  1 implementation
    void changeArtist(String title, String newArtist);  1 usage  1 implementation
    void changeYear(String title, int newYear);  1 usage  1 implementation
    void changeDescription(String title, String newDescription);  1 usage  1 implementation
    void printNonExhibitedArtworks();  1 usage  1 implementation
    void printArtworkDetailsByTitle(String titleToSearch);  1 usage  1 implementation
    void displayArtworks();  1 usage  1 implementation
    void ListArtworks();  1 usage  1 implementation
}
```

2.4 Interfața grafică JavaFX

Proiectul meu are funcționalitați atât în consolă, cât și în interfață grafică. În consolă se afișează un meniu cu opțiunile ce pot fi folosite de utilizator:

```java
if (option.equals("1")) {
    do {
        System.out.println("               MENU for Artworks");
        System.out.println("_____________________________________________________");
        System.out.println("1. Display All Artworks in the database");
        System.out.println("2. Display All Non Exhibited artworks in the database");
        System.out.println("3. Display details of an artwork in the database");
        System.out.println("4. Add a new artwork");
        System.out.println("5. Delete an artwork");
        System.out.println("6. Change the title of an artwork");
        System.out.println("7. Change the description of an artwork");
        System.out.println("8. Change the year of an artwork");
        System.out.println("9. Change the artist of an artwork");
        System.out.println("10. Exhibit an artwork");
        System.out.println("11. Remove an artwork from exhibition");
        System.out.println("12. Display All Artworks in ArtworkList.txt");
        System.out.println("E. Exit");
        System.out.println("Please enter the option: ");

        option = scanner.nextLine();

        switch (option) {
            case "1":
                artwork.displayArtworks();
                break;
            case "2":
                artwork.printNonExhibitedArtworks();
                break;
```

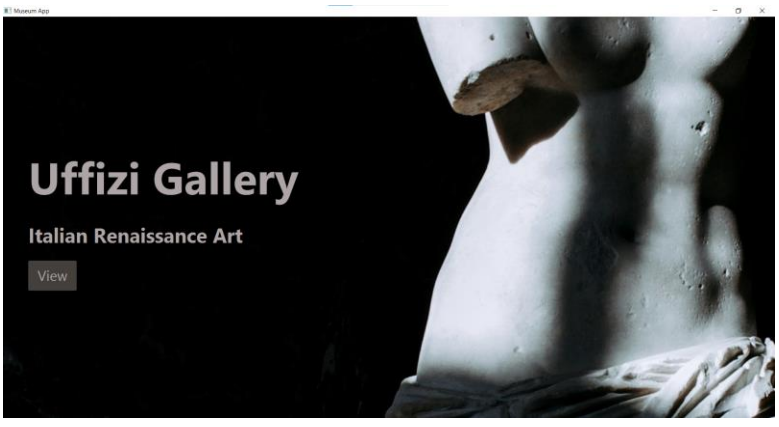
Interfața grafică a fost creată cu ajutorul limbajului de programare JavaFX. Am adaugat in fișierul *prom.xml* dependențele JavaFX și apoi am construit fereastra în care se vor afișa elementele construite cu JavaFX.

```
        <!-- JavaFX Controls -->
        <dependency>
            <groupId>org.openjfx</groupId>
            <artifactId>javafx-controls</artifactId>
            <version>17</version>
        </dependency>

        <dependency>
            <groupId>org.openjfx</groupId>
            <artifactId>javafx-fxml</artifactId>
            <version>17</version>
        </dependency>

    </dependencies>
```

```
public class MuseumApp extends Application {

    private Scene mainScene;  4 usages
    private double windowWidth = 600;  10 usages
    private double windowHeight = 400;  10 usages
    private Museum museum;  3 usages
    private Stage primaryStage;  14 usages

    @Override
    public void start(Stage primaryStage) {
        this.primaryStage = primaryStage;

        // Setting the museum
        museum = new Museum( nameOfMuseum: "Uffizi Gallery",  location: "Florence, Italy",  numberOfVisitors: 1000000);

        // Creating primary scene
        mainScene = createMainScene();

        primaryStage.setTitle("Museum App");
        primaryStage.setScene(mainScene);
        primaryStage.setWidth(windowWidth);
        primaryStage.setHeight(windowHeight);
        primaryStage.show();
    }
```

Interfața grafică se va axa mai mult pe clasa *Exhibition,* fiind singura clasa ce nu are o interfață Java.

Utilizatorul va fi întâmpinat de pagina de pornire:

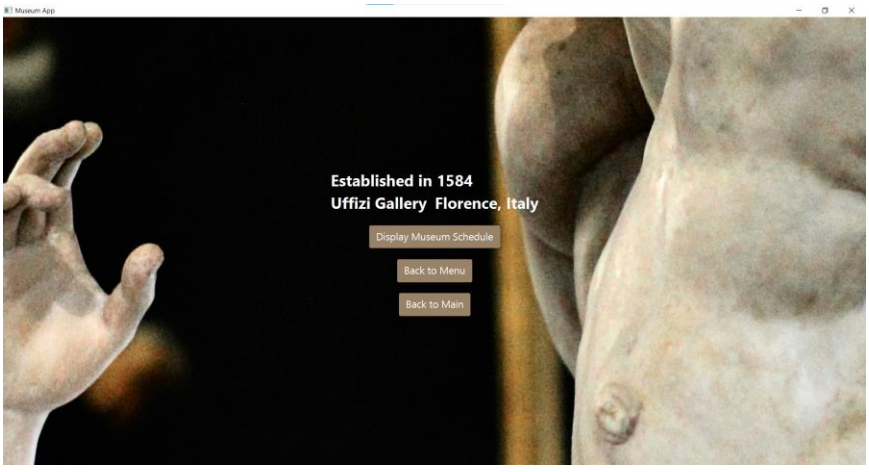


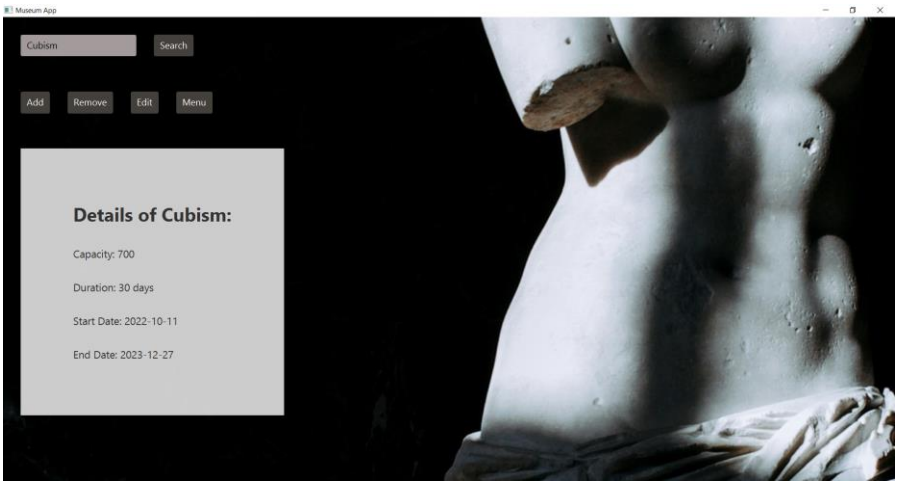
Codul aferent acestei scene:

```
private Scene createMainScene() {  2 usages
    // Background
    String imageUrl = "file:G:/Jubii/FACULTATE/ANU II SEM 1/MIP/Ufizzi Gallery Museum/src/main/resources/images/9.1.jpg";
    Image image = new Image(imageUrl);
    BackgroundImage backgroundImage = new BackgroundImage(image, BackgroundRepeat.NO_REPEAT,
            BackgroundRepeat.NO_REPEAT, BackgroundPosition.CENTER, BackgroundSize.DEFAULT);
    Background background = new Background(backgroundImage);


    // Labels
    Label messageLabel1 = new Label( s: "Uffizi Gallery");
    messageLabel1.setStyle("-fx-font-size: 88px; -fx-font-weight: bold; -fx-text-fill:#aea6a6; -fx-alignment: center;");
    messageLabel1.setAlignment(Pos.CENTER);

    Label messageLabel2 = new Label( s: "Italian Renaissance Art");
    messageLabel2.setStyle("-fx-font-size: 40px; -fx-font-weight: bold; -fx-text-fill: #aea6a6; -fx-alignment: center;");
    messageLabel2.setAlignment(Pos.CENTER);


    // Button
    Button menuButton = new Button( s: "View");
    menuButton.setStyle("-fx-font-size: 28px; -fx-background-color: #44403c; -fx-text-fill: #aea6a6;");
    menuButton.setOnAction(event -> {
        saveWindowDimensions();
        primaryStage.setScene(createMenuScene());
    });
```

```
        // Layout
        VBox mainLayout = new VBox( v: 20);
        mainLayout.setAlignment(Pos.CENTER_LEFT);
        mainLayout.setPadding(new Insets( v: 0,  v1: 0,  v2: 0,  v3: 50));
        mainLayout.setBackground(background);
        mainLayout.getChildren().addAll(messageLabel1, messageLabel2, menuButton);

        return new Scene(mainLayout, windowWidth, windowHeight);
    }
```

Interfața face trecerea de la o scenă la alta prin intermediul unor butoane, astfel incat utilizatorul să poată naviga cu ușurință. Câteva scene:





3 Detalii tehnice de implementare

Proiectul a fost creată cu ajutorul aplicației IntelliJ. IntelliJ IDEA este un mediu de dezvoltare integrat pentru aplicații Java de la compania JetBrains. Este poziționat ca cel mai inteligent și mai convenabil mediu de dezvoltare Java, cu suport pentru toate cele mai recente tehnologii și cadre. IntelliJ IDEA este unul dintre primele trei cele mai populare IDE-uri Java împreună cu Eclipse IDE și NetBeans IDE. Mai mult, la crearea bazei de data am folosit pgAdmin. PgAdmin este un instrument open-source de administrare și gestionare pentru baze de date PostgreSQL. Este utilizat pentru a interacționa cu baze de date PostgreSQL printr-o interfață grafică prietenoasă, în loc de linia de comandă. Pentru interfața grafică am folosit limbajul JavaFX. JavaFX este un framework modern pentru dezvoltarea de aplicații grafice de utilizator (GUI) în Java. Este o alternativă la biblioteca mai veche Swing și este inclusă în Java Development Kit (JDK). În mică masură sunt integrate si elemente din limbajul CSS pentru stilizare.

4 Cerințe Hard și Soft

Pentru a rula programul este necesara instalarea aplicatiei IntelliJ, versiunea 2024.2.3. Orice procesor echivalent cu cel de tip Intel Pentium IV sau versiuni mai recente sunt acceptate. Sistemele de operare recomandate sunt cele de minim Windows 10. În rest, nu trebuie îndeplinite condiții specifice.

5 Webgrafie

https://dev.to/bshadmehr/understanding-databases-their-importance-in-the-digital-age-bfi#:~:text=Data%20Organization%20and%20Retrieval%3A%20Databases,handle%20large%20amounts%20of%20data.

https://www.redswitches.com/blog/examples-of-databases/#:~:text=They%20are%20commonly%20used%20in,order%20history%2C%20and%20customer%20profiles.

https://www.pgadmin.org/

https://en.wikipedia.org/wiki/IntelliJ_IDEA

https://www.javatpoint.com/javafx-tutorial

https://www.geeksforgeeks.org/javafx-tutorial/

https://www.jetbrains.com/datagrip/features/?source=google&medium=cpc&campaign=EMEA_en_EAST_DataGrip_Search_RLSA&term=database%20gui&content=555125163774&gad_source=1&gclid=Cj0KCQiAvvO7BhC-ARIsAGFyToV-I9l2NLPyFaehxyWcK_sp7bDAUZfiOjUrBIy1dbXkhlOq5SfPsIwaAtRuEALw_wcB