



MSA UNIVERSITY  
جامعة أكتوبر للعلوم الحديثة والآداب



UNIVERSITY of  
GREENWICH



CSE Department – Faculty of Engineering - MSA

Spring 2025

GSE122 GSE122i COM265 PROGRAMMING 2

Course Project

Course Instructor: Dr. Ahmed El Anany

**Due Date 9/MAY/2025 11:59 PM on E-learning**

**Discussion inside lecture 18/May till 23/May inside lab as per lab slot**

Student Name	Amir Ashraf	Student ID	246161
Student Name	Yousef amr	Student ID	246217
Student Name	Mazen Mahmoud	Student ID	244005
Student Name	Zeyad Mohamed	Student ID	243945
Student Name	Mohamed Ehab	Student ID	235727
TA Name	Eng. Dina Magdy Eng. Gehad Ehab Eng. Mohamed Khaled Eng. Hussien Mostafa	Grade:	/

## Modern Periodic Table (MPT) Database Management System



## Table of Contents

<b>Project Overview</b>	<b>3</b>
Objectives	3
Roles and Responsibilities	4
Algorithm and external libraries	5
GUI and Database Usage	6
<b>Code explaining</b>	<b>7</b>
<b>Output and results</b>	<b>8</b>
<b>GitHub(optional)</b>	<b>9</b>
<b>References</b>	<b>10</b>



# Project Overview

## Objectives

The primary goal of this project is to develop a Java-based application that provides an interactive interface for users to explore and manage information about chemical elements and compounds. The application connects to a MySQL database to perform CRUD (Create, Read, Update, Delete) operations, offering functionalities such as:

- 1-Viewing detailed information about chemical elements.
- 2-Adding, modifying, and deleting elements and compounds.
- 3-Displaying a graphical representation of the periodic table.

## Tools and Technologies

- 1-Programming Language:** Java
- 2-Database:** MySQL
- 3-Database Connectivity:** JDBC (Java Database Connectivity)
- 4-Integrated Development Environment (IDE):** NetBeans
- 5-User Interface Library:** Swing (Javax)

## Responsibilities

- 1-Database Management:** Designing and maintaining the MySQL database schema.
- 2-Backend Development:** Implementing Java classes for database operations.
- 3-Frontend Development:** Designing and developing the GUI using Swing.
- 4-Testing and Validation:** Ensuring the application functions correctly and efficiently.

## Technical Concepts



**1-Object-Oriented Programming (OOP):** Utilizing classes and objects to structure the application.

**2-Event-Driven Programming:** Handling user interactions within the GUI.

**3-Database Connectivity:** Establishing and managing connections between the Java application and the MySQL database using JDBC.

## System Design / Architecture

### Database Schema (Simplified):

#### elements

atomic\_number: INT (Primary Key)

name: VARCHAR

symbol: VARCHAR

atomic\_mass: FLOAT

group\_number: INT

period: INT

category: VARCHAR

#### compounds

name: VARCHAR (Primary Key)

formula: VARCHAR

properties: TEXT

### Package: database.mpt

#### Classes:

DBConnection: Handles MySQL DB connection.

ElementDelete: Deletes an element by name.

ModifyElement: Updates an element's data.



CompoundDatabase: Insert & retrieve compound objects.

DatabaseMPT: Main method to list elements from DB.

ElementInfoWindow: GUI window showing element details.

## Objectives

The project aims to:

Create an interactive GUI representing the periodic table.

Enable users to view, add, modify, and delete information about chemical elements and compounds.

Store and retrieve data from a MySQL database securely and efficiently.

Provide a user-friendly interface for educational and research purposes.

## Roles and Responsibilities

Yousef &Mazen	Main,GUI,CompoundDatabase,DBconnection,DatabaseMPT,ElementInfoWindow
Amir	Report,ElementDelete,ModifyElement
Mohamed&zyad	Presentation,ModifyElement



## Algorithm and external libraries

### Algorithms

The application employs straightforward algorithms for database operations:

**Insertion Algorithm:** Collects user input and inserts new records into the database.

**Deletion Algorithm:** Removes records based on user-specified criteria.

**Update Algorithm:** Modifies existing records with new data provided by the user.

**Retrieval Algorithm:** Fetches and displays data from the database upon user request.

### External Libraries

**JDBC (Java Database Connectivity):** Facilitates communication between the Java application and the MySQL database.

**Swing (Javax):** Provides components for building the graphical user interface.

## GUI and Database Usage

### Graphical User Interface (GUI)

The GUI presents a visual representation of the periodic table, with each element represented as a clickable button. Key features include:

**Element Buttons:** Clicking on an element opens a window displaying detailed information.

**Search Functionality:** Users can search for elements by name or atomic number.

**Compound Management:** Users can add new compounds by entering their name, formula, and properties.

### Database Schema

The MySQL database consists of two primary tables:

#### elements

atomic\_number (INT, Primary Key)



name (VARCHAR)

symbol (VARCHAR)

atomic\_mass (FLOAT)

group\_number (INT)

period (INT)

category (VARCHAR)

### compounds

name (VARCHAR, Primary Key)

formula (VARCHAR)

properties (TEXT)

---

## Code explaining

### a. DBConnection.java

This class handles the connection between the Java application and the MySQL database. It defines the database URL, username, and password, and establishes a connection that is reused throughout the program.

**Purpose:** Acts as a gateway to access and manipulate the data stored in MySQL.

---

### b. ElementDelete.java

This class is responsible for deleting an element from the periodic table. The user enters the element name, and the class executes a SQL DELETE query to remove the corresponding row from the `elements` table.

**Purpose:** Deletes an element from the database when the user provides its name.

---

### c. ModifyElement.java



This class allows the user to update existing element data in the table. It supports changing all attributes of an element, including name, symbol, atomic number, mass, group, period, and category.

**Purpose:** Modifies any detail about an existing element in the database.

---

#### d. CompoundDatabase.java

This class serves as the database handler for compounds. When the user clicks on the "Compound" button and enters the compound's **name**, **formula**, and **properties**, this class saves the information in the `compounds` table in the database.

**Purpose:** Stores and retrieves compound-related information entered by the user.

---

#### e. DatabaseMPT.java

This class acts as the entry point for the application and handles general logic such as listing elements or triggering actions. It communicates with other classes to perform database interactions.

**Purpose:** Central management class for interacting with element data.

---

#### f. ElementInfoWindow.java

This class creates a pop-up window that displays the full details of an element when the user clicks its button on the GUI. The window shows:

Name

Symbol

Atomic Number

Atomic Mass

Group

Period

Category

**Purpose:** Provides a visual, user-friendly display of element data in a pop-up format.





## Output and results

### Main Interface:

Displays the periodic table with clickable buttons for each element.

### Element Information Window:

Shows detailed information about the selected element, including name, symbol, atomic number, atomic mass, group, period, and category.

### Compound Addition Form:

Allows users to add new compounds by entering the name, formula, and properties.

## Functionality Demonstrations

**Viewing Element Details:** Clicking on an element button opens a window with comprehensive information.

**Adding a New Element:** Users can input details for a new element, which is then added to the database.

**Modifying Element Information:** Existing elements can be updated with new data.

**Deleting an Element:** Elements can be removed from the database by specifying their name.

**Managing Compounds:** Users can add new compounds with relevant details.

## Code

```
package database.mpt;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

/**
 *
 * @author Yousef
 */
public class ElementDelete {

    public int deleteElement(String elementName) {
        try (Connection conn = DBConnection.getConnection());
```



```
PreparedStatement ps = conn.prepareStatement(
    "DELETE FROM elements WHERE name=?") {

    ps.setString(1, elementName);

    return ps.executeUpdate();

} catch (Exception e) {
    e.printStackTrace();
    return 0;
}
}

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
template
 */
package database.mpt;
import java.sql.*;

/**
 *
 * @author Yousef
 */
public class CompoundDatabase {

    public static void insertCompound(Compound compound) {
        try (Connection conn = DBConnection.getConnection();
            PreparedStatement ps = conn.prepareStatement("INSERT INTO compounds
VALUES (?, ?, ?))) {

            ps.setString(1, compound.getName());
            ps.setString(2, compound.getFormula());
            ps.setString(3, compound.getProperties());
            ps.executeUpdate();
        }
    }
}
```



```
} catch (Exception e) {
    e.printStackTrace();
}
}

public static Compound getCompoundByName(String name) {
    try (Connection conn = DBConnection.getConnection();
        PreparedStatement ps = conn.prepareStatement("SELECT * FROM compounds
WHERE name=?")) {

        ps.setString(1, name);
        ResultSet rs = ps.executeQuery();

        if (rs.next()) {
            return new Compound(
                rs.getString("name"),
                rs.getString("formula"),
                rs.getString("properties")
            );
        }

    } catch (Exception e) {
        e.printStackTrace();
    }
    return null;
}

}

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
template
 */
package database.mpt;

import java.sql.Connection;
import java.sql.DriverManager;
```



```
import java.sql.SQLException;
/**
 *
 * @author Yousef
 */
class DBConnection {
    private static final String URL = "jdbc:mysql://localhost:3306/modern_periodic_table";
    private static final String USER = "root";
    private static final String PASSWORD = "";

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL, USER, PASSWORD);
    }
}

package database.mpt;
import java.sql.*;

/**
 *
 * @author Yousef
 */
public class DatabaseMPT {
    static final String port = ":3306";
    static final String jdbcDriver = "com.mysql.jdbc.Driver";
    static final String dbURL = "jdbc:mysql://localhost" + port +
"/modern_periodic_table";

    public static void main(String[] args) throws Exception {
        Connection conn = null;
        Statement stmt = null;
        ResultSet rs = null;

        Class.forName(jdbcDriver);
        conn = DriverManager.getConnection(dbURL, "root", "");
        stmt = conn.createStatement();

        String sql;
```



```
sql = "SELECT * FROM elements";
rs = stmt.executeQuery(sql);

while (rs.next()) {
    int atomicNumber = rs.getInt("atomic_number");
    String symbol = rs.getString("symbol");
    String name = rs.getString("Name");
    float atomicMass = rs.getFloat("atomic_mass");
    int group = rs.getInt("group_number");
    int period = rs.getInt("period");
    String category = rs.getString("category");

    System.out.println(atomicNumber + " " + symbol + " " + name + " " +
        atomicMass + " " + group + " " + period + " " + category);
}

rs.close();
stmt.close();
conn.close();
}
}

package database.mpt;

import javax.swing.*;
import java.awt.*;
import java.sql.*;

/**
 *
 * @author Yousef
 */

public class ElementInfoWindow extends JFrame {
    public ElementInfoWindow(int atomicNumber) {
        setTitle("Element Info");
        setSize(300, 300);
        setLocationRelativeTo(null);
    }
}
```



```
setLayout(new BorderLayout());

JTextArea textArea = new JTextArea();
textArea.setEditable(false);
add(new JScrollPane(textArea), BorderLayout.CENTER);

try {
    Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/modern_periodic_table",
"root", "");
    String sql = "SELECT * FROM elements WHERE atomic_number = ?";
    PreparedStatement ps = conn.prepareStatement(sql);
    ps.setInt(1, atomicNumber);
    ResultSet rs = ps.executeQuery();

    if (rs.next()) {
        String info = "Name: " + rs.getString("name") + "\n" +
            "Symbol: " + rs.getString("symbol") + "\n" +
            "Atomic Number: " + rs.getInt("atomic_number") + "\n" +
            "Atomic Mass: " + rs.getFloat("atomic_mass") + "\n" +
            "Group: " + rs.getInt("group_number") + "\n" +
            "Period: " + rs.getInt("period") + "\n" +
            "Category: " + rs.getString("category");

        textArea.setText(info);
    }

    rs.close();
    ps.close();
    conn.close();
} catch (Exception e) {
    e.printStackTrace();
}

setVisible(true);
}
}

/*
```



\* Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license

\* Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template

```
*/  
package database.mpt;  
import java.sql.Connection;  
import java.sql.PreparedStatement;  
/**  
 *  
 * @author Yousef  
 */  
public class ModifyElement {  
  
    public void modifyElement(String currentName, String newName, String newSymbol,  
                              int newAtomicNumber, double newAtomicMass, int newGroupNumber,  
                              int newPeriod, String newCategory) throws Exception {  
        String sql = "UPDATE elements SET name=?, symbol=?, atomic_number=?,  
atomic_mass=?, group_number=?, period=?, category=? WHERE name=?";  
        try (Connection conn = DBConnection.getConnection();  
             PreparedStatement ps = conn.prepareStatement(sql)) {  
  
            ps.setString(1, newName);  
            ps.setString(2, newSymbol);  
            ps.setInt(3, newAtomicNumber);  
            ps.setDouble(4, newAtomicMass);  
            ps.setInt(5, newGroupNumber);  
            ps.setInt(6, newPeriod);  
            ps.setString(7, newCategory);  
            ps.setString(8, currentName);  
  
            int rowsUpdated = ps.executeUpdate();  
            if (rowsUpdated == 0) {  
                throw new Exception("Element with name '" + currentName + "' not found.");  
            }  
        }  
    }  
}
```



## Output screen shots

**Modern Periodic Table**

Enter An Atomic Number  Search Modify Compound

Enter The Element's Name  Delete

40°C  
صباح

11:41 AM 1/10/2024

**Modern Periodic Table**

**Element Info**

Name: Scandium  
Symbol: Sc  
Atomic Number: 21  
Atomic Mass: 44.956  
Group: 3  
Period: 4  
Category: Transition Metal

Enter An Atomic Number  Search Modify Compound

Enter The Element's Name  Delete

صباح  
طقس حار

11:41 AM 1/10/2024





## Modern Periodic Table

**Element Info**

Name: Magnesium  
 Symbol: Mg  
 Atomic Number: 12  
 Atomic Mass: 24.305  
 Group: 2  
 Period: 3  
 Category: Alkaline Earth Metal

Enter An Atomic Number:

Enter The Element's Name:

## Periodic Table

**New Element Form:**

Current Name:

New Name:

New Symbol:

New Atomic Number:

New Atomic Mass:

New Group Number:

New Period:

New Category:

Enter An Atomic Number:

Enter The Element's Name:



Periodic Table

Name

Formula  Add Compound

Properties

Enter An Atomic Number  Search Modify Compound

Enter The Element's Name  Delete

Windows taskbar: 10:15 AM 11/16/2019

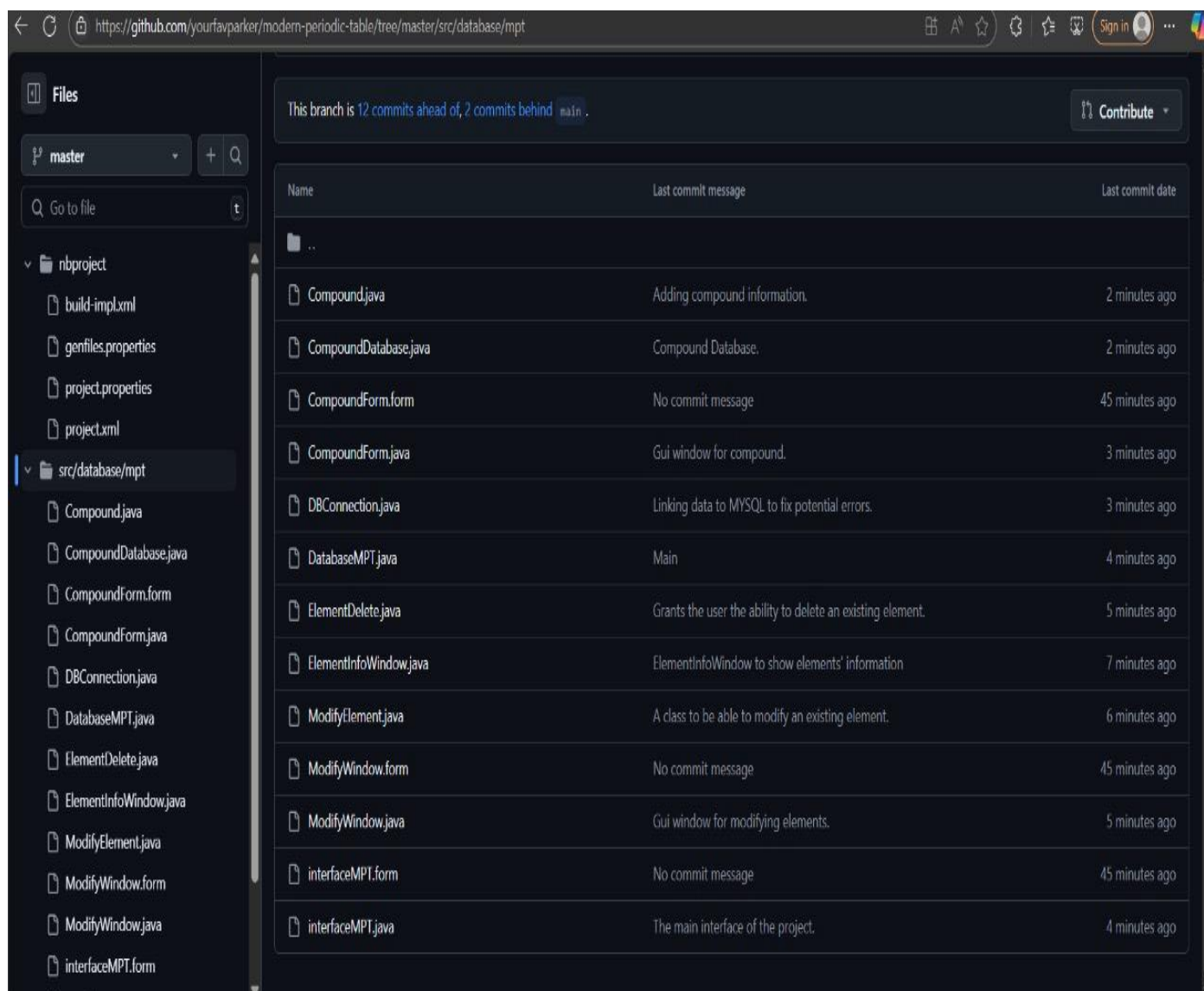


# GitHub

The complete source code and project report are available on GitHub:

<https://github.com/yourfavparker/modern-periodic-table>

## Repository Screenshot



The screenshot shows the GitHub web interface for the repository 'yourfavparker/modern-periodic-table'. The browser address bar displays the URL 'https://github.com/yourfavparker/modern-periodic-table/tree/master/src/database/mpt'. The page indicates that the current branch is 12 commits ahead of and 2 commits behind the 'main' branch. A 'Contribute' button is visible in the top right corner.

The left sidebar shows the file explorer with the following structure:

- nbproject
  - build-impl.xml
  - genfiles.properties
  - project.properties
  - project.xml
  - src/database/mpt
    - Compound.java
    - CompoundDatabase.java
    - CompoundForm.form
    - CompoundForm.java
    - DBConnection.java
    - DatabaseMPT.java
    - ElementDelete.java
    - ElementInfoWindow.java
    - ModifyElement.java
    - ModifyWindow.form
    - ModifyWindow.java
    - interfaceMPT.form
    - interfaceMPT.java

The main content area displays a table of files in the 'src/database/mpt' directory, including their names, last commit messages, and last commit dates.

Name	Last commit message	Last commit date
..		
Compound.java	Adding compound information.	2 minutes ago
CompoundDatabase.java	Compound Database.	2 minutes ago
CompoundForm.form	No commit message	45 minutes ago
CompoundForm.java	Gui window for compound.	3 minutes ago
DBConnection.java	Linking data to MYSQL to fix potential errors.	3 minutes ago
DatabaseMPT.java	Main	4 minutes ago
ElementDelete.java	Grants the user the ability to delete an existing element.	5 minutes ago
ElementInfoWindow.java	ElementInfoWindow to show elements' information	7 minutes ago
ModifyElement.java	A class to be able to modify an existing element.	6 minutes ago
ModifyWindow.form	No commit message	45 minutes ago
ModifyWindow.java	Gui window for modifying elements.	5 minutes ago
interfaceMPT.form	No commit message	45 minutes ago
interfaceMPT.java	The main interface of the project.	4 minutes ago



MSA UNIVERSITY  
جامعة أكتوبر للعلوم الحديثة والآداب



UNIVERSITY of  
GREENWICH



yourfavparker / modern-periodic-table

Type [7] to search

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

modern-periodic-table Public

Pin Unwatch 1 Fork 0 Star 0

master had recent pushes 3 minutes ago

Compare & pull request

main 2 Branches 0 Tags

Go to file Add file Code

yourfavparker created ElementInfoWindow 1d25a10 · 17 minutes ago 2 Commits

README.md Initial commit 41 minutes ago

commitsData created ElementInfoWindow 17 minutes ago

README

### modern-periodic-table

A modern periodic table project using java programming language, it aims to be user-friendly allowing the user to browse through the periodic table while being able to modify,delete, or even add a new element.

About

A modern periodic table project using java programming language, it aims to be user-friendly allowing the user to browse through the periodic table while being able to modify,delete, or even add a new element.

Readme

Activity

0 stars

1 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Files

master + Q

Go to file t

nbproject

src/database/mpt

Compound.java

CompoundDatabase.java

CompoundForm.form

CompoundForm.java

DBConnection.java

DatabaseMPT.java

ElementDelete.java

ElementInfoWindow.java

ModifyElement.java

ModifyWindow.form

ModifyWindow.java

interfaceMPT.form

interfaceMPT.java

build.xml

manifest.mf

modern-periodic-table / src / database / mpt / ElementInfoWindow.java

yourfavparker ElementInfoWindow to show elements' information 54b21eb · 8 minutes ago History

Code Blame 51 lines (42 loc) · 1.52 KB Code 55% faster with GitHub Copilot

Raw Copy Download Edit

```
1 package database.mpt;
2
3 import javax.swing.*;
4 import java.awt.*;
5 import java.sql.*;
6
7 /**
8  *
9  * @author Yousef
10  */
11
12 public class ElementInfoWindow extends JFrame {
13     public ElementInfoWindow(int atomicNumber) {
14         setTitle("Element Info");
15         setSize(300, 300);
16         setLocationRelativeTo(null);
17         setLayout(new BorderLayout());
18
19         JTextArea textArea = new JTextArea();
20         textArea.setEditable(false);
21         add(new JScrollPane(textArea), BorderLayout.CENTER);
22
23         try {
24             Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/modern_periodic_table", "root", "");
25             String sql = "SELECT * FROM elements WHERE atomic_number = ?";
26             PreparedStatement ps = conn.prepareStatement(sql);
27             ps.setInt(1, atomicNumber);
28             ResultSet rs = ps.executeQuery();
```



**MSA UNIVERSITY**  
جامعة أكتوبر للعلوم الحديثة والآداب



## References

**Java JDBC Documentation:** <https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/>

**MySQL Reference Manual:** <https://dev.mysql.com/doc/>

**Java Swing Tutorial:** <https://docs.oracle.com/javase/tutorial/uiswing/>

**GeeksforGeeks - Java Database Connectivity with MySQL:** <https://www.geeksforgeeks.org/java-database-connectivity-with-mysql/>