

• Advanve_Java Task 1 :

1. Code :

```
package com.connection;
import java.sql.Connection;
import java.sql.DriverManager;

public class DBconnection {
    public static void main(String[] args) {
        String url="jdbc:oracle:thin:@localhost:1521:XE";
        String user="system";
        String pass="Vaishnavi";

        try {
            Class.forName("oracle.jdbc.OracleDriver");
            Connection con= DriverManager.getConnection(url,user,pass);
            if(con!=null){
                System.out.println("Connection Successful....");
            }
            else{
                System.out.println("Connection Failed.....");
            }
            con.close();
        } catch (ClassNotFoundException e){
            e.printStackTrace();
        } catch (Exception e){
            e.printStackTrace();
        }
    }
}
```

2. Step by step Explanation :

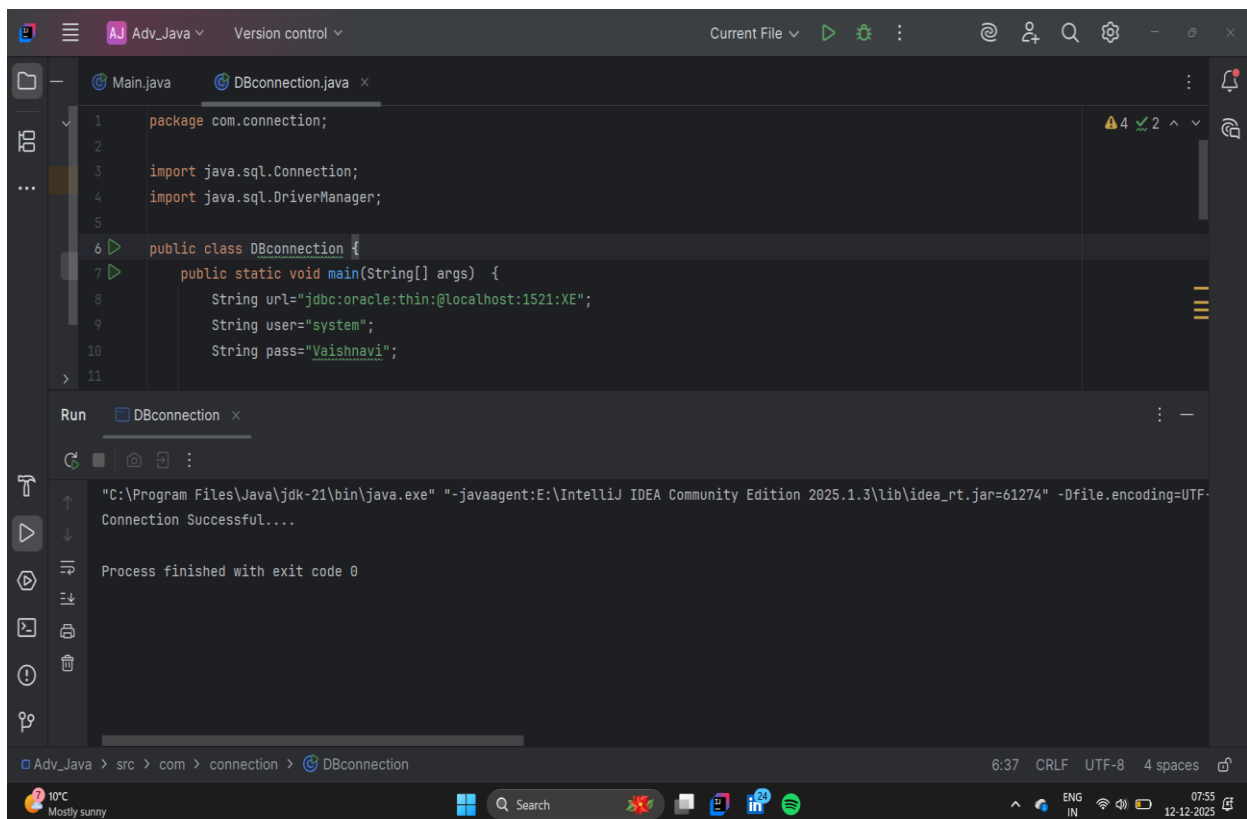
1. package com.connection;
Puts this class in the com.connection folder/namespace.
2. import java.sql.Connection; import java.sql.DriverManager;
Brings in JDBC classes needed to open a DB connection.
3. public class DBconnection { public static void main(String[] args) { ... } }
Standard Java program entry point. The main method runs when you start the program.
4. String url="jdbc:oracle:thin:@localhost:1521:XE";
Database URL: tells JDBC how to reach the Oracle DB (host localhost, port 1521, service/SID XE).
5. String user="system"; String pass="Vaishnavi";
Database username and password used to log in.

6. `Class.forName("oracle.jdbc.OracleDriver");`
Loads the Oracle JDBC driver class so Java can talk to Oracle. (Requires the Oracle JDBC JAR on your classpath.)
7. `Connection con = DriverManager.getConnection(url, user, pass);`
Attempts to connect to the database using the URL, username and password. Returns a Connection object if successful.
8. `if(con != null) { System.out.println("Connection Successful...."); } else { System.out.println("Connection Failed....."); }`
Simple check: if con is not null, print success; otherwise print failure.
9. `con.close();`
Closes the database connection to free resources (should be done when finished).
10. `try { ... } catch (ClassNotFoundException e) { e.printStackTrace(); } catch (Exception e) { e.printStackTrace(); }`
Wraps the connection code in a try/catch so the program handles errors (missing driver, wrong URL, bad credentials, DB down). `printStackTrace()` prints the error details to help debugging.

Note:

- Make sure the Oracle JDBC driver (e.g. `ojdbc8.jar`) is added to your project's classpath.
- If you get `ClassNotFoundException` the driver JAR is missing.
- If you get `SQLException: No suitable driver` check the URL format and driver jar.

3. Ouput Screenshot :



The screenshot displays an IDE window with a project named 'Adv_Java'. The 'DBconnection.java' file is open, showing the following code:

```
1 package com.connection;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5
6 public class DBconnection {
7     public static void main(String[] args) {
8         String url="jdbc:oracle:thin:@localhost:1521:XE";
9         String user="system";
10        String pass="Vaishnavi";
11    }
12 }
```

The 'Run' tab at the bottom shows the execution output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:E:\IntelliJ IDEA Community Edition 2025.1.3\lib\idea_rt.jar=61274" -Dfile.encoding=UTF-8
Connection Successful....
Process finished with exit code 0
```

The status bar at the bottom indicates the file is at 'Adv_Java > src > com > connection > DBconnection', with a cursor at line 6, column 17. The encoding is UTF-8, and the file uses 4 spaces for indentation.