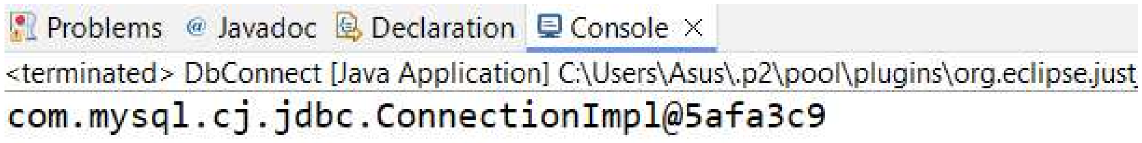
Day 21

Task 1: Establishing Database Connections
Write a Java program that connects to a SQLite database and prints out the connection object to confirm successful connection.

```
    package com.assignment.sql;
    import java.sql.Connection;
    import java.sql.DriverManager;
    import java.sql.SQLException;

    public class DbConnect {
        public static Connection con;
        public static Connection getMyDBConn() {
                try {

con=DriverManager.getConnection("jdbc:mysql://localhost:3306/SQLite", "root",
"Sonal@007");
                } catch (SQLException e) {
                        e.printStackTrace();
                }
                return con;
        }
        public static void main(String[] args) {
                System.out.println(getMyDBConn());
        }
}
```

Output:

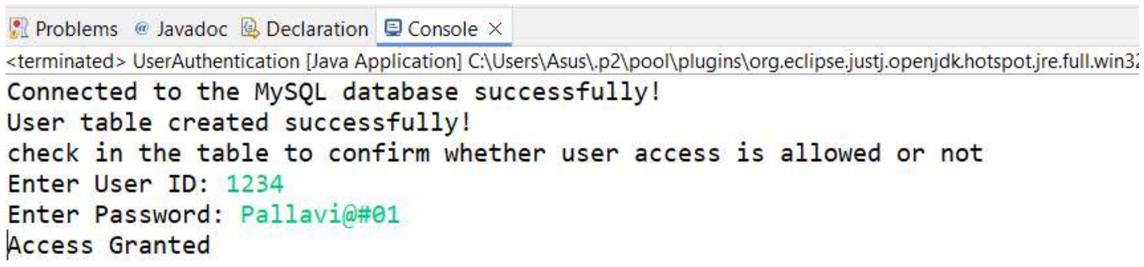


```
<terminated> DbConnect [Java Application] C:\Users\Asus\.p2\pool\plugins\org.eclipse.just
com.mysql.cj.jdbc.ConnectionImpl@5afa3c9
```

Task 2: SQL Queries using JDBC
Create a table 'User' with a following schema 'User ID' and 'Password' stored as hash format (note you have research on how to generate hash from a string), accept "User ID" and "Password" as input and check in the table if they match to confirm whether user access is allowed or not.

```
package com.assignment.sql;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Scanner;

public class UserAuthentication {

  public static void main(String[] args) {
      try (Scanner scanner = new Scanner(System.in)) {

          Connection connection = DbConnect.getMyDBConn();
          System.out.println("Connected to the MySQL database successfully!");
          createUserTable(connection);
          // commenting this line to avoid inserting the user on subsequent runs.
//            insertUser(connection, "1234", "Pallavi@#01");
          System.out.println("check in the table to confirm whether user access is
allowed or not");
          System.out.print("Enter User ID: ");
          String userID = scanner.nextLine();
          System.out.print("Enter Password: ");
          String password = scanner.nextLine();
          String passwordHash = hashPassword(password);

          if (authenticateUser(connection, userID, passwordHash)) {
              System.out.println("Access Granted");
          } else {
              System.out.println("Access Denied");
          }
          connection.close();
      } catch (SQLException | NoSuchAlgorithmException e) {
          e.printStackTrace();
      }
  }
  private static void createUserTable(Connection connection) throws SQLException {
      String createTableSQL = "CREATE TABLE IF NOT EXISTS User (" +
                              "user_id VARCHAR(255) PRIMARY KEY, " +
                              "password_hash TEXT NOT NULL)";
      try (Statement statement = connection.createStatement()) {
          statement.executeUpdate(createTableSQL);
          System.out.println("User table created successfully!");
      }
  }
  private static String hashPassword(String password) throws
NoSuchAlgorithmException {
      MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
      byte[] hashBytes = messageDigest.digest(password.getBytes());
      StringBuilder hashString = new StringBuilder();
      for (byte b : hashBytes) {
         hashString.append(String.format("%02x", b));
      }
      return hashString.toString();
  }
   private static void insertUser(Connection connection, String userID, String
password) throws SQLException, NoSuchAlgorithmException {
      String passwordHash = hashPassword(password);
      String insertSQL = "INSERT INTO User (user_id, password_hash) VALUES ('" +
userID + "', '" + passwordHash + "')";
      try (Statement statement = connection.createStatement()) {
         statement.executeUpdate(insertSQL);
         System.out.println("User added successfully!");
      }
  }
  // Method to authenticate user
  private static boolean authenticateUser(Connection connection, String userID,
String passwordHash) throws SQLException {
      String selectSQL = "SELECT * FROM User WHERE user_id = '" + userID + "'
AND password_hash = '" + passwordHash + "'";
      try (Statement statement = connection.createStatement()) {
         try (ResultSet resultSet = statement.executeQuery(selectSQL)) {
            return resultSet.next();
         }
      }
  }
}
```

Output:

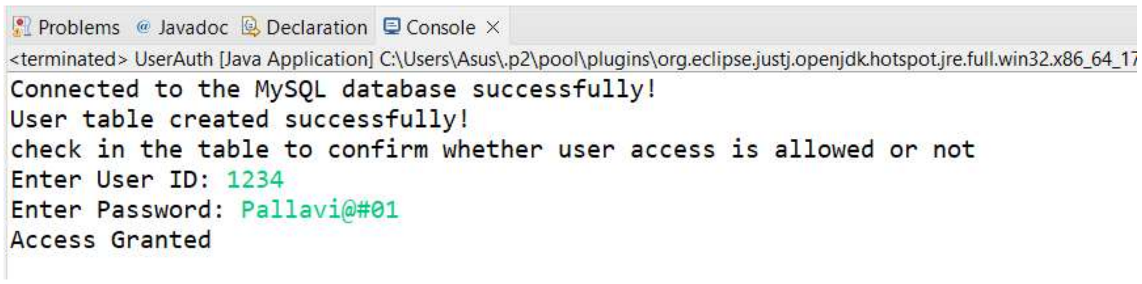

Problems Javadoc Declaration Console

```
<terminated> UserAuthentication [Java Application] C:\Users\Asus\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32
Connected to the MySQL database successfully!
User table created successfully!
check in the table to confirm whether user access is allowed or not
Enter User ID: 1234
Enter Password: Pallavi@#01
Access Granted
```

Task 3: PreparedStatement
Modify the SELECT query program to use PreparedStatement to parameterize the query and prevent SQL injection.

```
package com.assignment.sql;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Scanner;
public class UserAuth {
  public static void main(String[] args) {
     try (Scanner scanner = new Scanner(System.in)) {

        Connection connection = DbConnect.getMyDBConn();
        System.out.println("Connected to the MySQL database successfully!");

        createUserTable(connection);

//          commenting this line to avoid inserting the user on subsequent runs.
//          insertUser(connection, "1234", "Pallavi@#01");

        System.out.println("check in the table to confirm whether user access is
allowed or not");
        System.out.print("Enter User ID: ");
        String userID = scanner.nextLine();
        System.out.print("Enter Password: ");
        String password = scanner.nextLine();
```

```
            String passwordHash = hashPassword(password);

            if (authenticateUser(connection, userID, passwordHash)) {
                System.out.println("Access Granted");
            } else {
                System.out.println("Access Denied");
            }
            connection.close();
        } catch (SQLException | NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
    }

    private static void createUserTable(Connection connection) throws
SQLException {
        String createTableSQL = "CREATE TABLE IF NOT EXISTS User (" +
                                "user_id VARCHAR(255) PRIMARY KEY, " +
                                "password_hash TEXT NOT NULL)";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(createTableSQL)) {
            preparedStatement.executeUpdate();
            System.out.println("User table created successfully!");
        }
    }
    private static String hashPassword(String password) throws
NoSuchAlgorithmException {
        MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
        byte[] hashBytes = messageDigest.digest(password.getBytes());
        StringBuilder hashString = new StringBuilder();
        for (byte b : hashBytes) {
            hashString.append(String.format("%02x", b));
        }
        return hashString.toString();
    }
     private static void insertUser(Connection connection, String userID, String
password) throws SQLException, NoSuchAlgorithmException {
        String passwordHash = hashPassword(password);
        String insertSQL = "INSERT INTO User (user_id, password_hash) VALUES (?,
?)";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(insertSQL)) {
            preparedStatement.setString(1, userID);
            preparedStatement.setString(2, passwordHash);
            preparedStatement.executeUpdate();
            System.out.println("User added successfully!");
        }
    }
    // Method to authenticate user
   private static boolean authenticateUser(Connection connection, String userID,
String passwordHash) throws SQLException {
        String selectSQL = "SELECT * FROM User WHERE user_id = ? AND
password_hash = ?";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(selectSQL)) {
            preparedStatement.setString(1, userID);
            preparedStatement.setString(2, passwordHash);
            ResultSet resultSet = preparedStatement.executeQuery();
            return resultSet.next();

        }
    }

}
```
Output:



```
<terminated> UserAuth [Java Application] C:\Users\Asus\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17
Connected to the MySQL database successfully!
User table created successfully!
check in the table to confirm whether user access is allowed or not
Enter User ID: 1234
Enter Password: Pallavi@#01
Access Granted
```