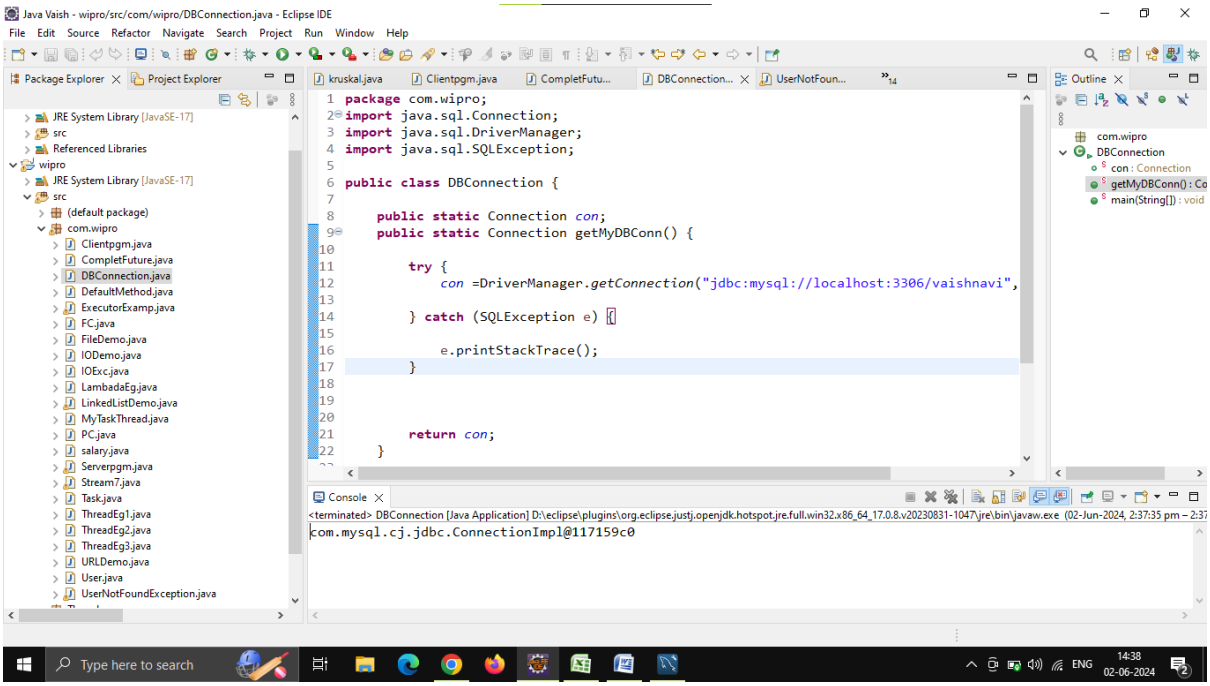
Day 21:

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

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

public class DBConnection {

	public static Connection con;
	public static Connection getMyDBConn() {

		try {
			con
=DriverManager.getConnection("jdbc:mysql://localhost:3306/vaishnavi"
, "root", "vaish");

		} catch (SQLException e) {

			e.printStackTrace();
		}



		return con;
	}

	public static void main(String[] args) {
		System.out.println(getMyDBConn());

	}

}
```



Task 2: SQL Queries using JDBC

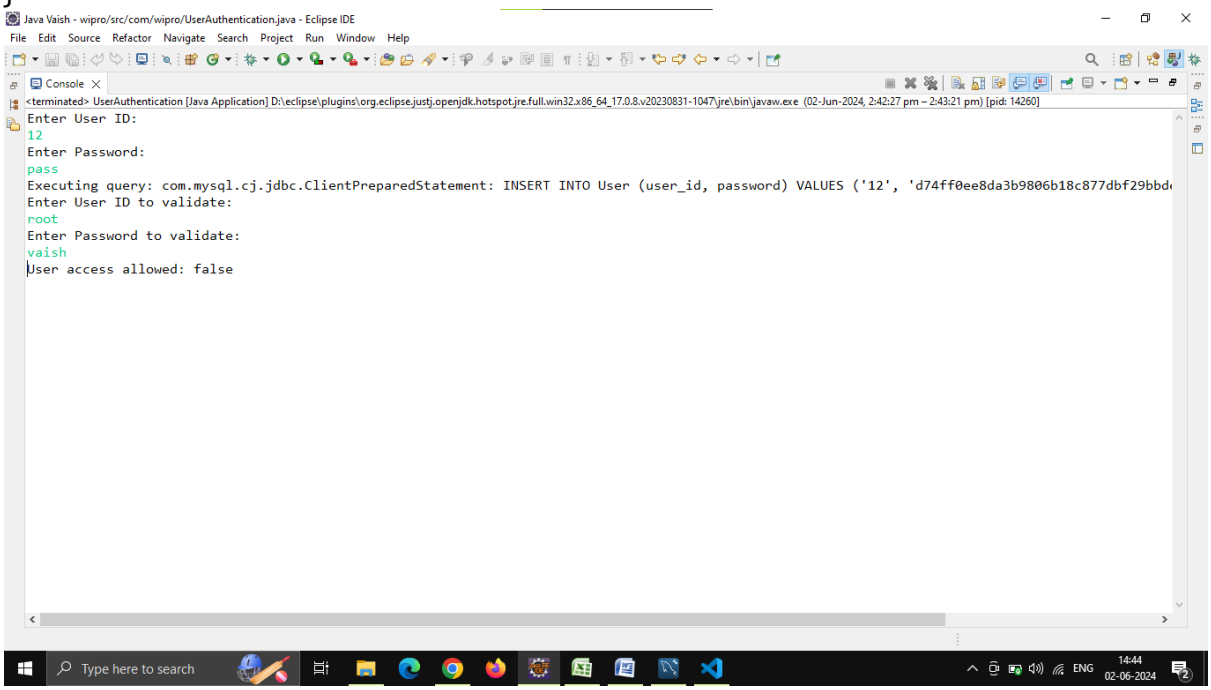
Create a table 'User with a following schema 'User ID' and 'Password stored as hash format (note you have researchion how to generate hash from a string), accepi "User ID" and "Password" as input and check in the table if they match to confirm whether user access is allowed not

```
package com.wipro;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Scanner;

public class UserAuthentication {

    private static final String JDBC_URL =
"jdbc:mysql://localhost:3306/Vaishnavi";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "vaish";

    public static void main(String[] args) {
```

```
        try {
            // Load MySQL JDBC Driver
            Class.forName("com.mysql.cj.jdbc.Driver");

            try (Connection connection =
DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD)) {
                createTable(connection);

                Scanner scanner = new Scanner(System.in);
                System.out.println("Enter User ID:");
                String userId = scanner.nextLine();
                System.out.println("Enter Password:");
                String password = scanner.nextLine();

                insertUser(connection, userId, password);

                System.out.println("Enter User ID to validate:");
                String userIdToValidate = scanner.nextLine();
                System.out.println("Enter Password to validate:");
                String passwordToValidate = scanner.nextLine();

                boolean isValid = validateUser(connection,
userIdToValidate, passwordToValidate);
                System.out.println("User access allowed: " +
isValid);

            } catch (SQLException e) {
                e.printStackTrace();
            }
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }

    private static void createTable(Connection connection) throws
SQLException {
        String createTableSQL = "CREATE TABLE IF NOT EXISTS User ("
                                + "user_id VARCHAR(255) PRIMARY KEY,
"
                                + "password VARCHAR(255) NOT NULL)";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(createTableSQL)) {
            preparedStatement.execute();
        }
    }

    private static String hashPassword(String password) {
        try {
            MessageDigest md = MessageDigest.getInstance("SHA-256");
```

```
            byte[] hashedPassword = md.digest(password.getBytes());
            StringBuilder sb = new StringBuilder();
            for (byte b : hashedPassword) {
                sb.append(String.format("%02x", b));
            }
            return sb.toString();
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }

    private static void insertUser(Connection connection, String
userId, String password) throws SQLException {
        String hashedPassword = hashPassword(password);
        String insertUserSQL = "INSERT INTO User (user_id, password)
VALUES (?, ?)";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(insertUserSQL)) {
            preparedStatement.setString(1, userId);
            preparedStatement.setString(2, hashedPassword);
            System.out.println("Executing query: " +
preparedStatement.toString());
            preparedStatement.executeUpdate();
        }
    }
    private static boolean validateUser(Connection connection,
String userId, String password) throws SQLException
    {
        String hashedPassword = hashPassword(password);
        String selectUserSQL = "SELECT password FROM User WHERE
user_id = ?";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(selectUserSQL))
        {
            preparedStatement.setString(1, userId);
            try (ResultSet resultSet =
preparedStatement.executeQuery())
            {
                if (resultSet.next())
                {
                    String storedHashedPassword =
resultSet.getString("password");
                    return
storedHashedPassword.equals(hashedPassword);
                } else {
                    return false;
                }
            }
        }
```

```
    }
}
```



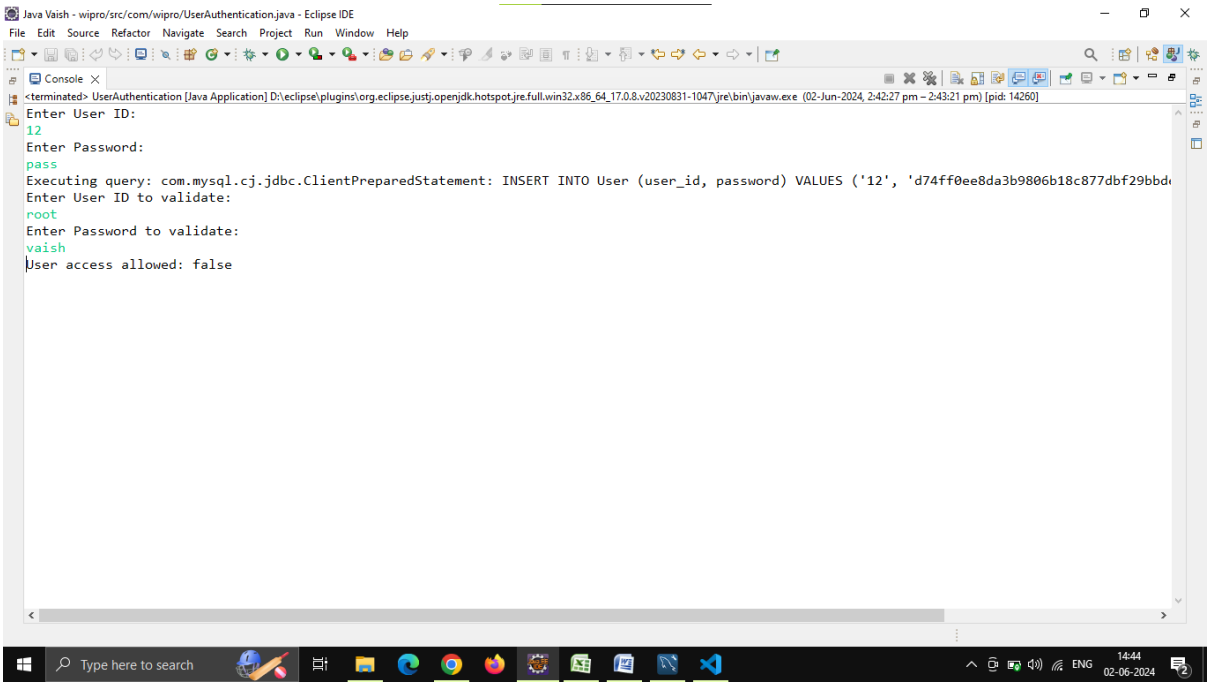
Task 3: PreparedStatement

Modify the SELECT query program o use PreparedStatement to parameterize the query and prevent SQL injection

```
package com.wipro;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.Scanner;

public class UserAuthentication {

    private static final String JDBC_URL =
"jdbc:mysql://localhost:3306/Vaishnavi";
    private static final String JDBC_USER = "root";
    private static final String JDBC_PASSWORD = "vaish";

    public static void main(String[] args) {
        try {
            // Load MySQL JDBC Driver
```

```
            Class.forName("com.mysql.cj.jdbc.Driver");

            try (Connection connection =
DriverManager.getConnection(JDBC_URL, JDBC_USER, JDBC_PASSWORD)) {
                createTable(connection);

                Scanner scanner = new Scanner(System.in);
                System.out.println("Enter User ID:");
                String userId = scanner.nextLine();
                System.out.println("Enter Password:");
                String password = scanner.nextLine();

                insertUser(connection, userId, password);

                System.out.println("Enter User ID to validate:");
                String userIdToValidate = scanner.nextLine();
                System.out.println("Enter Password to validate:");
                String passwordToValidate = scanner.nextLine();

                boolean isValid = validateUser(connection,
userIdToValidate, passwordToValidate);
                System.out.println("User access allowed: " +
isValid);

            } catch (SQLException e) {
                e.printStackTrace();
            }
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }

    private static void createTable(Connection connection) throws
SQLException {
        String createTableSQL = "CREATE TABLE IF NOT EXISTS User ("
                                + "user_id VARCHAR(255) PRIMARY KEY,
"
                                + "password VARCHAR(255) NOT NULL)";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(createTableSQL)) {
            preparedStatement.execute();
        }
    }

    private static String hashPassword(String password) {
        try {
            MessageDigest md = MessageDigest.getInstance("SHA-256");
            byte[] hashedPassword = md.digest(password.getBytes());
            StringBuilder sb = new StringBuilder();
```

```
            for (byte b : hashedPassword) {
                sb.append(String.format("%02x", b));
            }
            return sb.toString();
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException(e);
        }
    }

    private static void insertUser(Connection connection, String
userId, String password) throws SQLException {
        String hashedPassword = hashPassword(password);
        String insertUserSQL = "INSERT INTO User (user_id, password)
VALUES (?, ?)";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(insertUserSQL)) {
            preparedStatement.setString(1, userId);
            preparedStatement.setString(2, hashedPassword);
            System.out.println("Executing query: " +
preparedStatement.toString());
            preparedStatement.executeUpdate();
        }
    }
    private static boolean validateUser(Connection connection,
String userId, String password) throws SQLException
    {
        String hashedPassword = hashPassword(password);
        String selectUserSQL = "SELECT password FROM User WHERE
user_id = ?";
        try (PreparedStatement preparedStatement =
connection.prepareStatement(selectUserSQL))
        {
            preparedStatement.setString(1, userId);
            try (ResultSet resultSet =
preparedStatement.executeQuery())
            {
                if (resultSet.next())
                {
                    String storedHashedPassword =
resultSet.getString("password");
                    return
storedHashedPassword.equals(hashedPassword);
                } else {
                    return false;
                }
            }
        }
    }
}
```


Java Vaish - wipro/src/com/wipro/UserAuthentication.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Console
<terminated> UserAuthentication [Java Application] D:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\javaw.exe (02-Jun-2024, 2:42:27 pm – 2:43:21 pm) [pid: 14260]
Enter User ID:
12
Enter Password:
pass
Executing query: com.mysql.cj.jdbc.ClientPreparedStatement: INSERT INTO User (user_id, password) VALUES ('12', 'd74ff0ee8da3b9806b18c877dbf29bbd
Enter User ID to validate:
root
Enter Password to validate:
vaish
User access allowed: false
Type here to search
ENG
14:44
02-06-2024