

Name-Ravi Kishan

Roll number-

2400291530154

Github-link--

[https://github.com/RA](https://github.com/RAVI-KISHAN-ui/Product_management_system)

VI-KISHAN-

ui/Product_management_system

S

Intelligent





Teacher Dashboard - Mark Attendance

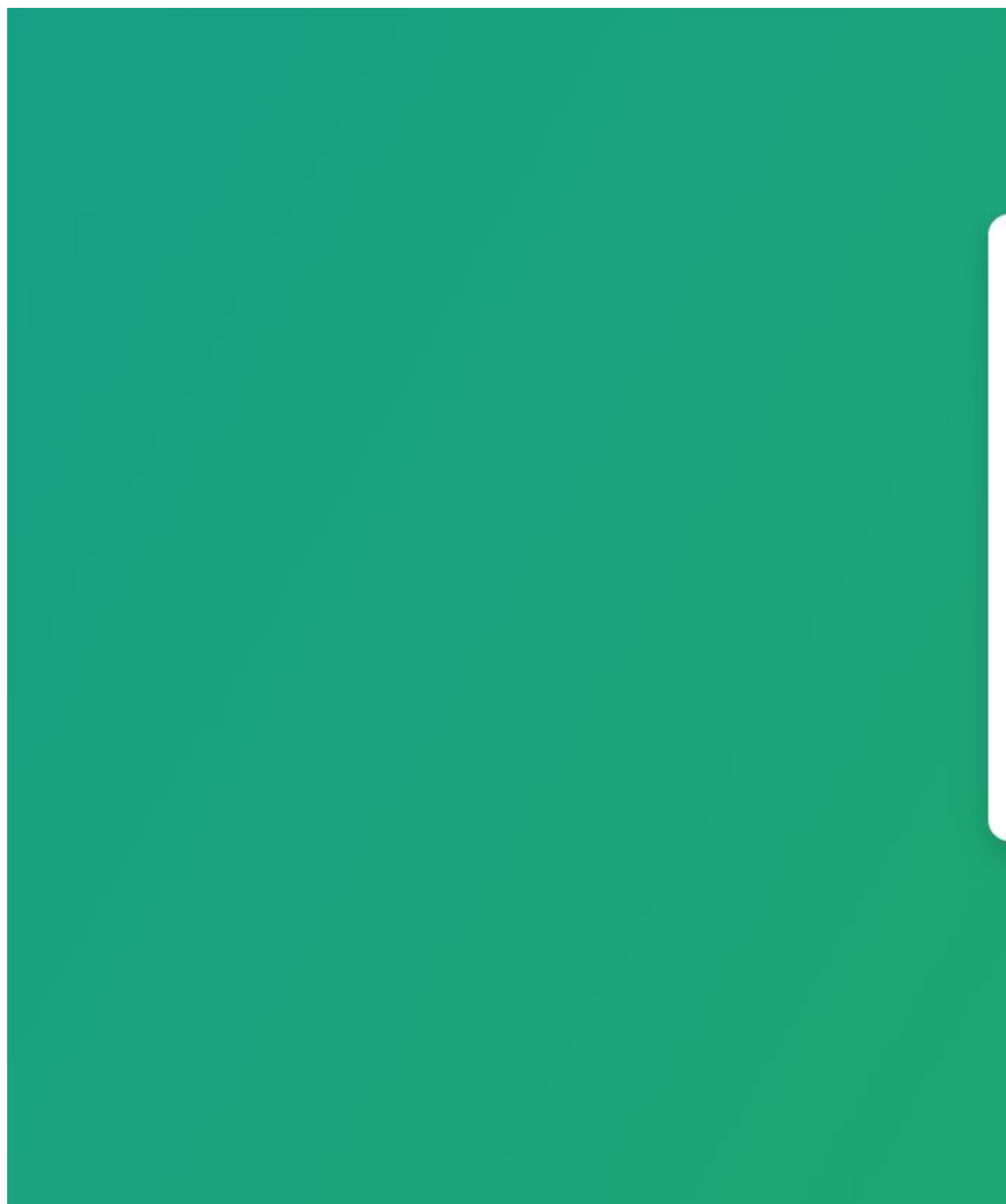
Mark Attendance

Date

dd-mm-yyyy

Roll Number
101
102
110

Submit Attendance



CODE:

```
package  
com.example.demo.co  
ntroller;
```

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



```
import
java.sql.DriverManager
;
import
java.sql.PreparedStatement;
import
java.sql.ResultSet;
import java.util.*;
```

```
import  
org.springframework.s  
tereotype.Controller;
```

```
import  
org.springframework.u  
i.Model;
```

```
import  
org.springframework.  
web.bind.annotation.*  
;
```

@Controller

public class

HomeController {

private final String

URL =

"jdbc:mysql://localhost:3306/attendance?us

```
eSSL=false&serverTimezone=UTC";
```

```
    private final String  
    USER = "root";
```

```
    private final String  
    PASS = "Ravi@1947";
```

```
    @GetMapping("/")  
    public String  
    home(Model model) {
```

```
        model.addAttribute  
e("message",  
"Welcome!");  
        return "index"; //  
index.jsp  
    }
```

```
@GetMapping("/stu  
dentLogin")
```

```
public String  
studentLogin() {  
    return  
    "studentLogin"; // JSP  
page  
}
```

```
@GetMapping("/stu  
dent")
```

```
public String  
studentDashboard(  
    @RequestParam  
    m(required = false)  
    Integer studentId,  
    Model model) {
```

```
    List<Map<String,  
Object>> requests =  
new ArrayList<>();
```

```
        if (studentId !=  
null) {  
            String sql =  
"SELECT date, reason,  
status " +  
            "FROM  
attendance_applicatio  
n " +
```



```
        "WHERE  
student_id = ? " +  
        "ORDER BY  
date DESC";
```

```
        try (Connection  
conn =  
DriverManager.getCon  
nection(URL, USER,  
PASS);
```

```
PreparedStatement ps =  
conn.prepareStatement(  
sql)) {
```

```
    ps.setInt(1,  
studentId);
```

```
    ResultSet rs =  
ps.executeQuery();
```

```
        while
(rs.next()) {
            Map<String
, Object> row = new
HashMap<>();
            row.put("da
te",
rs.getDate("date"));
```

```
        row.put("reason",  
rs.getString("reason"));  
        row.put("status",  
rs.getString("status"));  
        requests.add(row);  
    }
```

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

```
        model.addAttribute  
e("requests",  
requests);
```

```
        return "student";  
// student.jsp  
    }
```

```
    @PostMapping("/st  
udentLogin")
```

```
    public String  
handleStudentLogin(  
        @RequestPara  
m String rollNo,
```

```
        @RequestParam  
        String password,  
        Model model) {
```

```
        String url =  
        "jdbc:mysql://localhost:  
        3306/attendance";
```

```
        String user =  
        "root";
```

```
String pass =  
"Ravi@1947";
```

```
String sql =  
"SELECT * FROM  
students WHERE  
roll_number = ? AND  
password = ?";
```

```
try (Connection  
conn =
```



```
DriverManager.getConnection(url, user,  
pass);
```

```
        PreparedStatement  
ement ps =  
conn.prepareStatement(  
t(sql)) {
```

```
            ps.setString(1,  
rollNo);
```

```
ps.setString(2,  
password);
```

```
ResultSet rs =  
ps.executeQuery();  
if (rs.next()) {  
    // Student  
exists, login successful  
    return  
"redirect:/student"; //
```

forward to student
dashboard JSP

}

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

```
        model.addAttribute("error", "Database  
connection error");  
        return  
        "studentLogin";  
    }
```

```
        // If no record  
        found
```

```
        model.addAttribute("error", "Invalid roll  
number or password");  
        return  
        "studentLogin";  
    }
```

```
@GetMapping("/tea  
cher")
```

```
public String  
teacherDashboard(Mo  
del model) {
```

```
    List<Map<String,  
Object>> allRequests =  
new ArrayList<>();
```

```
    String sql =  
"SELECT ar.id, s.name  
AS studentName,
```

```
ar.date, ar.reason,  
ar.status " +  
        "FROM  
attendance_requests  
ar " +  
        "JOIN  
students s ON  
ar.student_id = s.id " +  
        "ORDER BY  
ar.date DESC";
```

```
try (Connection  
conn =  
DriverManager.getCon  
nection(URL, USER,  
PASS);  
        PreparedStat  
ement ps =  
conn.prepareStatement  
t(sql);
```



```
        ResultSet rs =  
ps.executeQuery()) {  
  
        while (rs.next())  
{  
  
                Map<String,  
Object> map = new  
HashMap<>();  
  
                map.put("id",  
rs.getInt("id"));
```

```
        map.put("studentName",  
rs.getString("studentName"));
```

```
        map.put("date",  
rs.getString("date"));
```

```
        map.put("reason",  
rs.getString("reason"));
```

```
        map.put("status",  
rs.getString("status"));  
        allRequests.add(map);  
    }  
}
```

```
    } catch (Exception  
e) {
```

```
        e.printStackTrace()  
e();  
    }
```

```
        model.addAttribute  
e("pending",  
allRequests); // this  
will be used in  
teacher.jsp
```

```
        return "teacher";  
// teacher.jsp  
    }
```

```
    @GetMapping("/teacherLogin")  
    public String  
teacherLoginPage() {
```

```
        return  
        "teacherLogin"; // JSP  
        page  
    }
```

```
    @PostMapping("/te  
acherLogin")
```

```
    public String  
    handleTeacherLogin(
```

```
        @RequestParam  
        String email,  
        @RequestParam  
        String password,  
        Model model) {
```

```
        String url =  
        "jdbc:mysql://localhost:  
        3306/attendance";
```

```
String user =  
"root";
```

```
String pass =  
"Ravi@1947";
```

```
String sql =  
"SELECT * FROM  
teachers WHERE email  
= ? AND password =  
?";
```



```
try (Connection  
conn =  
DriverManager.getCon  
nection(url, user,  
pass);  
        PreparedStat  
ement ps =  
conn.prepareStatement  
t(sql)) {
```

```
        ps.setString(1,  
email);
```

```
        ps.setString(2,  
password);
```

```
        ResultSet rs =  
ps.executeQuery();  
        if (rs.next()) {  
            // Teacher  
exists, login successful
```

```
        return  
        "redirect:/teacher"; //  
        redirect to teacher  
        dashboard  
    }
```

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

```
        model.addAttribute("error", "Database  
connection error");  
  
        return  
        "teacherLogin";  
    }
```

```
        model.addAttribute("error", "Invalid  
email or password");
```

```
        return  
        "teacherLogin";  
    }
```

```
    @GetMapping("/markAttendance")  
    public String  
    markAttendancePage()  
    {
```

```
        return  
        "markAttendance"; //  
JSP page  
    }  
  
    // Fetch students as  
JSON  
  
    @GetMapping("/stu  
dents")  
  
    @ResponseBody
```

```
public
List<Map<String,
Object>> getStudents()
{
    List<Map<String,
Object>> students =
new ArrayList<>();

    String sql =
"SELECT id, name,
```

```
roll_number FROM  
students";
```

```
try {  
    Class.forName("  
com.mysql.cj.jdbc.Driver"  
er");  
    try (Connection  
conn =  
DriverManager.getCon
```



```
nection(URL, USER,  
PASS);
```

```
        PreparedStatement  
statement ps =  
conn.prepareStatement  
t(sql);
```

```
        ResultSet rs  
= ps.executeQuery()) {
```

```
        while
(rs.next()) {
            Map<String
, Object> s = new
HashMap<>();
            s.put("id",
rs.getInt("id"));
            s.put("nam
e",
rs.getString("name"));
```

```
        s.put("roll_
number",
rs.getString("roll_num
ber"));

        students.ad
d(s);

    }

}

} catch (Exception
e) {
```

```
        e.printStackTrace()  
    e());  
    }
```

```
    return students;  
}
```

```
// Save attendance  
@PostMapping("/submitAttendance")
```

```
public String  
submitAttendance(@R  
equestParam  
Map<String, String>  
allParams, Model  
model) {  
    String date =  
allParams.get("date");  
    // get the date
```

```
        if (date == null ||
date.isEmpty()) {
            model.addAttribute("error", "Date is
required!");
            return "error";
// default error JSP
page
    }
```

// Replace "-" with
"_" for table name
(MySQL table names
can't have '-')

String tableName
= "attendance_" +
date.replace("-", "_");

String url =
"jdbc:mysql://localhost

t:3306/attendance?useSSL=false&serverTimezone=UTC";

String user =
"root";

String pass =
"Ravi@1947";

boolean c = false;

try {


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

```
try (Connection  
conn =  
DriverManager.getConnection(url, user,  
pass)) {
```

```
// 1. Create  
table if not exists  
  
String  
createTableSQL =  
"CREATE TABLE IF NOT  
EXISTS " + tableName  
+ " (" +  
  
        "id INT  
        AUTO_INCREMENT  
        PRIMARY KEY," +
```

```
        "roll_number VARCHAR(50)," +  
        "name VARCHAR(100)," +  
        "status VARCHAR(20)" +  
        ");";
```

```
        conn.setAuto  
Commit(true);
```

```
try
(PreparedStatement ps
=
conn.prepareStatement
t(createTableSQL)) {
    ps.execute(
);
}
```

// 2. Insert
each student's
attendance

String

```
insertSQL = "INSERT  
INTO " + tableName +  
" (roll_number, name,  
status) VALUES (?, ?,  
?);"
```

```
try
(PreparedStatement ps
=
conn.prepareStatement
(insertSQL)) {
    for
    (Map.Entry<String,
String> entry :
allParams.entrySet()) {
```

```
String key  
= entry.getKey();  
if  
(key.startsWith("status  
_")) {
```

```
String  
studentId =  
key.split("_")[1];
```

String

status =

entry.getValue();

//

Fetch student info

from "students" table

String

studentSQL = "SELECT

roll_number, name

FROM students

WHERE id = ?";

try

(PreparedStatement

psStudent =

conn.prepareStatement

t(studentSQL)) {

psSt

udent.setString(1,

studentId);

```
try
(ResultSet rs =
psStudent.executeQuery()
ry()) {
if
(rs.next()) {
String roll =
rs.getString("roll_num
ber");
```

```
String name =  
rs.getString("name");
```

```
ps.setString(1, roll);
```

```
ps.setString(2, name);
```

```
ps.setString(3, status);
```

```
ps.addBatch();
```

```
}
```

```
}
```

```
}
```

```
}
```

```
}
```

```
ps.executeB
```

```
atch();
```

```
}
```

```
        System.out.pr  
intln("Attendance  
submitted for " + date  
+ " " + c);  
  
    }  
    } catch (Exception  
e) {
```

```
        e.printStackTrace()  
    e());  
  
        model.addAttribute("error", "Database  
error occurred!");  
  
        return "error";  
  
    // default error JSP  
    page  
    }
```

```
        return  
        "redirect:/markAttend  
ance";  
    }
```

```
    @PostMapping("/ap  
plyAttendance")
```

```
    @ResponseBody  
    public String  
    applyAttendance(
```

```
        @RequestParam  
        Integer studentId, //  
        this is the student  
        submitting
```

```
        @RequestParam  
        String date,
```

```
        @RequestParam  
        String reason) {
```



```
String url =  
"jdbc:mysql://localhost:3306/attendance?useSSL=false&serverTimezone=UTC";
```

```
String user =  
"root";
```

```
String pass =  
"Ravi@1947";
```

```
try {  
    Class.forName("  
com.mysql.cj.jdbc.Driver");  
  
    try (Connection  
conn =  
DriverManager.getConnection(url, user,  
pass)) {
```

String

```
insertSQL = "INSERT  
INTO  
attendance_requests  
(student_id, date,  
reason, status) VALUES  
(?, ?, ?, 'PENDING')";
```

try

```
(PreparedStatement ps  
=
```

```
conn.prepareStatement  
(insertSQL)) {  
    ps.setInt(1,  
studentId);  
    ps.setString  
(2, date);  
    ps.setString  
(3, reason);  
    ps.executeUpdate  
Update();
```

```
        }  
    }  
} catch (Exception  
e) {  
    e.printStackTrace()  
    return  
"Database error  
occurred!";  
}
```

```
        return  
        "Attendance  
application  
submitted!";  
    }
```

```
    // -----  
    HELPER: GET STUDENT  
    REQUESTS -----
```

```
private
List<Map<String,
Object>>
getStudentRequests(in
t studentId) {

    List<Map<String,
Object>> requests =
new ArrayList<>();

    String sql =
"SELECT date, reason,
```

```
status FROM  
attendance_requests  
WHERE student_id = ?  
ORDER BY date DESC";
```

```
try (Connection  
conn =  
DriverManager.getCon  
nection(URL, USER,  
PASS);
```



```
        PreparedStatement  
statement ps =  
conn.prepareStatement(  
sql)) {
```

```
        ps.setInt(1,  
studentId);
```

```
        ResultSet rs =  
ps.executeQuery();
```

```
while (rs.next())  
{  
    Map<String,  
Object> map = new  
HashMap<>();  
    map.put("date",  
rs.getString("date"));
```

```
        map.put("reason",  
rs.getString("reason"));  
        map.put("status",  
rs.getString("status"));  
        requests.add(  
map);  
    }
```

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

```
    return requests;
```

```
}
```

```
// -----
```

APPROVE / REJECT

REQUESTS -----

-

```
@GetMapping("/ap  
prove")
```

```
public String  
approveRequest(@Req  
uestParam int id) {
```

```
    try (Connection  
conn =
```

```
DriverManager.getConnection(URL, USER,  
PASS)) {
```

```
    // 1. Get the  
    request details
```

```
        String selectSQL  
= "SELECT student_id,  
date FROM
```

attendance_requests

WHERE id = ?";

int studentId =
0;

String date =
null;

try
(PreparedStatement ps
=

```
conn.prepareStatement(selectSQL) {  
    ps.setInt(1,  
id);  
    try (ResultSet  
rs =  
ps.executeQuery()) {  
        if (rs.next())  
{
```



```
        studentId
=
rs.getInt("student_id")
;

        date =
rs.getString("date");
    } else {
        return
"redirect:/teacher"; //
request not found
```

```
}  
}  
}
```

// 2. Update
attendance table for
that date

String
tableName =

```
"attendance_" +  
date.replace("-", "_");
```

String

```
updateSQL = "UPDATE  
" + tableName + " a " +  
"JOIN  
students s ON s.id = ? "  
+
```

```
        "SET
a.status = 'PRESENT' "
+
        "WHERE
a.roll_number =
s.roll_number";
    try
        (PreparedStatement ps
=
```

```
conn.prepareStatement  
t(updateSQL)) {  
    ps.setInt(1,  
studentId);  
    ps.executeUpdate  
date());  
}
```

```
// 3. Delete the  
request from  
attendance_requests  
  
String  
deleteSQL = "DELETE  
FROM  
attendance_requests  
WHERE id = ?";  
  
try  
(PreparedStatement ps
```

=

```
conn.prepareStatement(deleteSQL) {  
    ps.setInt(1,  
id);  
    ps.executeUpdateUp  
date());  
}
```

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

```
        return  
"redirect:/teacher"; //  
redirect to teacher  
dashboard
```



```
}
```

```
@GetMapping("/reject")
```

```
public String  
rejectRequest(@RequestParam int id) {
```

```
    try (Connection  
conn =
```

```
DriverManager.getCon
```

```
nection(URL, USER,  
PASS)) {
```

```
    String
```

```
    deleteSQL = "DELETE  
FROM
```

```
attendance_requests  
WHERE id = ?";
```

```
    try
```

```
(PreparedStatement ps  
=
```

```
conn.prepareStatement(deleteSQL) {  
    ps.setInt(1,  
id);  
    ps.executeUpdate()  
date();  
}  
} catch (Exception  
e) {
```

```
        e.printStackTrace()
    }

    return
    "redirect:/teacher"; //
    redirect to teacher
    dashboard
}

}
```

