

### **Certificate**

This is to certify that Lovepreet Kaur, a student of Master of Computer Applications (MCA) – , has successfully completed the Minor Project titled “ Notice Board" under the esteemed guidance of Ms. Tanya in University Institute of Computing (UIC), Chandigarh University.

This project was undertaken as a part of the academic curriculum and is submitted in partial fulfilment of the requirements for the MCA program. The work presented in this project is a result of independent research, diligent effort, and dedication, demonstrating the student’s ability to apply theoretical knowledge to practical problem-solving.

I hereby confirm that this project is an original work carried out by the student and has not been submitted elsewhere for the award of any other degree, diploma, or certification.

Project Guide

Ms. Tanya

University Institute of Computing

Chandigarh University

## Acknowledgement

I would like to express my deep gratitude to **Ms. Tanya** at Chandigarh University, for their invaluable guidance, encouragement, and support throughout the duration of this project. Their insightful feedback and expertise helped me to successfully complete this project on "**Notice Board**".

I also extend my thanks to the **Department of MCA ,UIC**, for providing the necessary resources and a conducive environment to carry out this project. Special thanks to who contributed their time and knowledge to assist me during the project work.

**Name : Lovepreet Kaur**

**UID : 24MCA20199**

## Abstract

This project report presents the design and development of a Digital Notice Board System that enables the administrator to add, update, and delete notices dynamically. Built using Java for backend logic, deployed via Apache Tomcat Server, and managed with MySQL via XAMPP, this web-based system ensures quick and efficient dissemination of information. The system enhances communication efficiency within an organization or institution by digitizing traditional notice board functionalities.

The Digital Notice Board System is a web-based application designed to facilitate efficient and paperless communication within an institution. In conventional settings, notices are typically printed and physically posted on notice boards, which not only incurs recurring costs but also delays the dissemination of time-sensitive information. The objective of this project is to digitalize this process by developing an interactive and centralized notice board system that can be accessed via the internet or an intranet.

This system is implemented using Java for backend logic, Apache Tomcat as the web server, and XAMPP (which includes MySQL) for managing the database. The application allows an authenticated administrator to create, edit, and delete notices dynamically. The use of server-side technologies ensures data integrity and scalability, while the frontend interface ensures ease of use and accessibility.

By automating the notice publication process, the system enhances communication efficiency within organizations like schools, colleges, or offices. It is especially useful in environments where information needs to be updated frequently and accessed by a wide audience in real-time.

## Introduction

In today's fast-paced digital world, the need for efficient communication within institutions and organizations is vital. Traditional notice boards have served this purpose for years but come with several limitations such as limited accessibility, manual updates, and lack of remote control. The digital notice board aims to overcome these shortcomings by providing a centralized, web-based system for managing and displaying notices in real time. This report outlines the complete development process of such a system using Java, Tomcat, and MySQL (XAMPP).

In the era of rapid technological advancement, traditional methods of communication are being replaced by digital alternatives. Educational institutions and corporate organizations often rely on notice boards to convey information to students, staff, or employees. However, manually managing notices involves significant time, effort, and resources. There is also a lack of real-time updates and limited reach, especially for users who are not physically present on-site.

The Digital Notice Board System aims to solve these challenges by providing a centralized online platform for managing and viewing notices. Built with a combination of Java, Apache Tomcat, and XAMPP (MySQL), this system enables an administrator to log in securely and manage notices efficiently. Notices can be displayed instantly to end-users with no delay and can be updated or deleted as required.

This project demonstrates how web technologies can streamline routine administrative functions and make institutional communication more effective and environmentally friendly. The system's user-friendly design ensures that even users with limited technical skills can interact with it smoothly.

## Objective

The main objective of the Digital Notice Board System is to simplify and digitize the process of creating, editing, and deleting notices. The goal of this project is to design and implement a Digital Notice Board System that replaces the conventional physical notice board with a more efficient and accessible online solution. Specific objectives include:

- To implement secure admin login functionality to ensure only authorized users can manage notices.
- To enable the admin to add new notices with relevant information, including titles, descriptions, dates, and expiry.
- To provide functionalities for editing and deleting existing notices with minimal effort.
- To design a responsive and intuitive frontend interface for easy viewing of notices by users.
- To store and manage all data persistently using a relational database (MySQL).
- To ensure the system is scalable and adaptable for use in educational institutions, corporate offices, or public service departments.
- To develop a user-friendly web interface for notice management.
- To enable administrators to manage notices remotely via a secure login system.
- To ensure notices are displayed in an organized and timely manner.
- To utilize open-source technologies for better cost-effectiveness and scalability.

## System Requirements

### Hardware Requirements:

- Processor: Intel Core i3 or higher
- RAM: Minimum 4 GB
- Hard Disk: At least 10 GB free space

### Software Requirements:

- Operating System: Windows 10/Linux (Ubuntu preferred)
- Java Development Kit (JDK) 1.8 or above
- Apache Tomcat Server 9 or later
- XAMPP (includes Apache Server and MySQL database)
- Web Browser (Google Chrome, Mozilla Firefox)
- Integrated Development Environment (IDE) like NetBeans or Eclipse

## System Design

### Architecture

The system follows a client-server model where:

- The client interacts through a web browser.
- The server processes the request using Java Servlets.
- The MySQL database stores the notice data.

### UML Diagrams

- **Use Case Diagram:** Illustrates user interactions, showing how the admin logs in and performs CRUD operations on notices.
- **Class Diagram:** Represents the structure of the classes, including Admin, Notice, and Database Connector.
- **Sequence Diagram:** Details the interaction between the client, server, and database when adding or retrieving notices.

## Modules Description

**Admin Module:** This module is responsible for authenticating the administrator and granting access to the dashboard. Features include:

- Secure login using credentials stored in the database.
- Dashboard view with options to manage notices.

**Notice Management Module:** This is the core module of the system. It includes:

- **Add Notice:** Allows admin to input title and content.
- **Edit Notice:** Enables modifications to existing notices.
- **Delete Notice:** Removes outdated or incorrect notices.
- **View All Notices:** Displays all notices sorted by date.

## Technologies Used

**Programming Language:** Java is used for server-side logic and servlet development.

### Frontend Technologies:

- HTML for structure
- Java Server Pages (JSP) for dynamic content rendering.
- Xampp for styling.

### Backend Technologies:

- Apache Tomcat as the servlet container and web server.

### Database:

- MySQL for storing user credentials and notice information.
- Managed via phpMyAdmin interface in XAMPP.

### Development Tools:

- NetBeans or Eclipse IDE
- MySQL Workbench for database schema design

## Testing and Validation

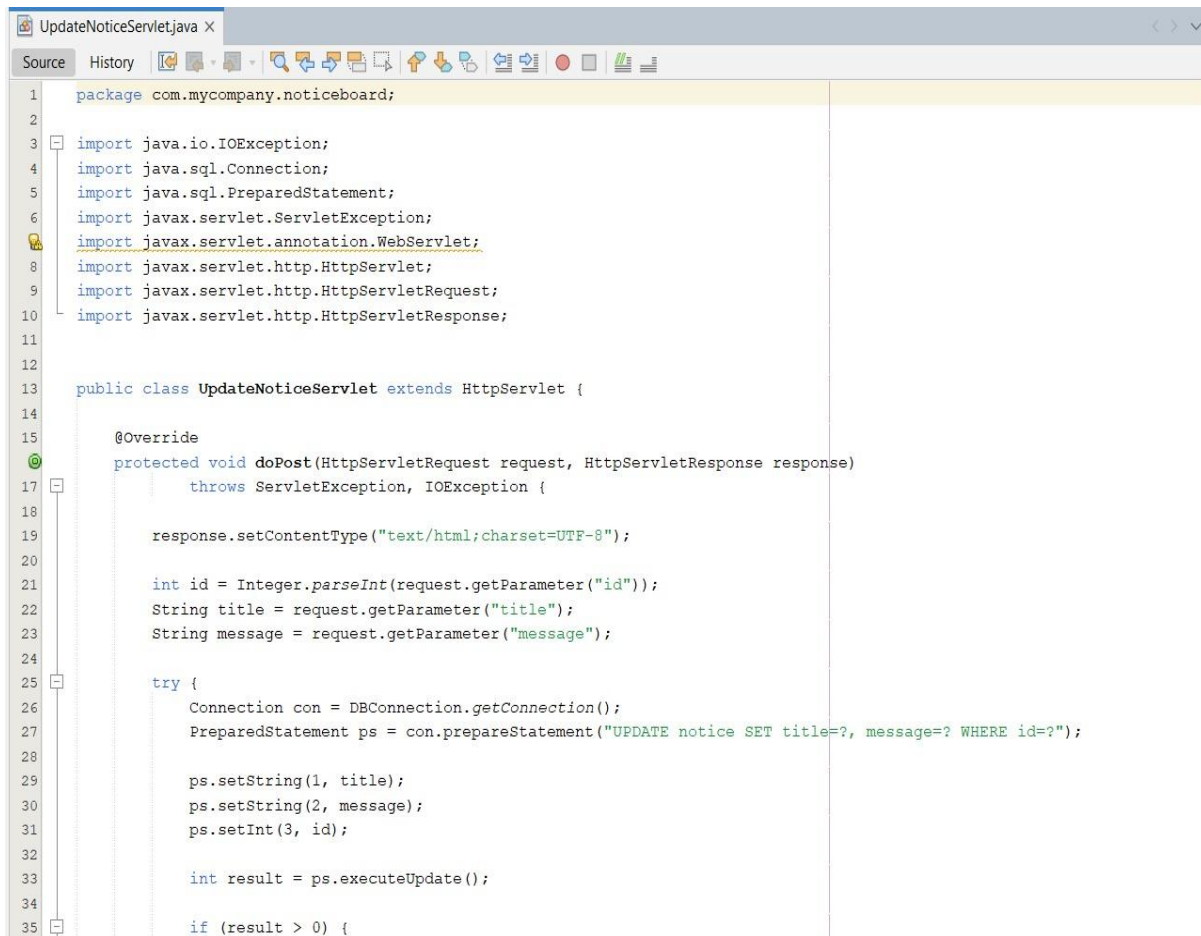
### Test Cases Include:

- Login with correct credentials.
- Adding a new notice and verifying its appearance on the homepage.
- Deleting a notice and confirming removal from the database.

### Testing Tools:

- Manual testing through browser
- Console debugging and log tracing

## Code



```
UpdateNoticeServlet.java X
Source History
1 package com.mycompany.noticeboard;
2
3 import java.io.IOException;
4 import java.sql.Connection;
5 import java.sql.PreparedStatement;
6 import javax.servlet.ServletException;
7 import javax.servlet.annotation.WebServlet;
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11
12
13 public class UpdateNoticeServlet extends HttpServlet {
14
15     @Override
16     protected void doPost(HttpServletRequest request, HttpServletResponse response)
17         throws ServletException, IOException {
18
19         response.setContentType("text/html;charset=UTF-8");
20
21         int id = Integer.parseInt(request.getParameter("id"));
22         String title = request.getParameter("title");
23         String message = request.getParameter("message");
24
25         try {
26             Connection con = DBConnection.getConnection();
27             PreparedStatement ps = con.prepareStatement("UPDATE notice SET title=?, message=? WHERE id=?");
28
29             ps.setString(1, title);
30             ps.setString(2, message);
31             ps.setInt(3, id);
32
33             int result = ps.executeUpdate();
34
35             if (result > 0) {
```



```
<%@page contentType="text/html;charset=UTF-8"%>
<%@page import="java.sql.*"%>
<%@page import="javax.servlet.http.HttpSession"%>
<%@page import="com.mycompany.noticeboard.DBConnection"%>
```

```
<%
    HttpSession session1 = request.getSession(false);
    if(session1 == null || session1.getAttribute("username") == null){
        response.sendRedirect("login.jsp");
        return;
    }
%>
```

```
<!DOCTYPE html>
<html>
<head>
    <title>All Notices</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background: #f4f6f7;
            margin: 0;
            padding: 20px;
        }

        h2 {
            text-align: center;
            color: #2c3e50;
        }

        .top-links {
            text-align: center;
            margin-bottom: 20px;
        }
    </style>
</head>
```

```
LogoutServlet.java X
Source History
1 package com.mycompany.noticeboard;
2
3 import java.io.IOException;
4 import javax.servlet.ServletException;
5 import javax.servlet.annotation.WebServlet;
6 import javax.servlet.http.HttpServlet;
7 import javax.servlet.http.HttpServletRequest;
8 import javax.servlet.http.HttpServletResponse;
9 import javax.servlet.http.HttpSession;
10
11
12 public class LogoutServlet extends HttpServlet {
13
14     @Override
15     protected void doGet(HttpServletRequest request, HttpServletResponse response)
16         throws ServletException, IOException {
17
18         HttpSession session = request.getSession(false);
19         if (session != null) {
20             session.invalidate();
21         }
22         response.sendRedirect("login.jsp");
23     }
24 }
25
```

```

index.jsp x
Source History
1 <%@ page import="java.sql.*" %>
2 <%@ page import="com.mycompany.noticeboard.DBConnection" %>
3
4 <h2>Online Notice Board</h2>
5
6 <table border="1" cellpadding="10" cellspacing="0">
7     <tr>
8         <th>Title</th>
9         <th>Description</th>
10        <th>Posted On</th>
11    </tr>
12
13    <%
14    Connection con = null;
15    PreparedStatement ps = null;
16    ResultSet rs = null;
17
18    try {
19        con = DBConnection.getConnection();
20        ps = con.prepareStatement("SELECT * FROM notice ORDER BY posted_on DESC");
21        rs = ps.executeQuery();
22
23        while(rs.next()) {
24
25            <tr>
26                <td><%= rs.getString("title") %></td>
27                <td><%= rs.getString("description") %></td>
28                <td><%= rs.getString("posted_on") %></td>
29            </tr>
30
31        }
32    } catch(Exception e) {
33        out.println("<tr><td colspan='3' style='color:red;'>Error loading notices: " + e.getMessage() + "</td></tr>");
34        e.printStackTrace();
35    } finally {
36

```

```

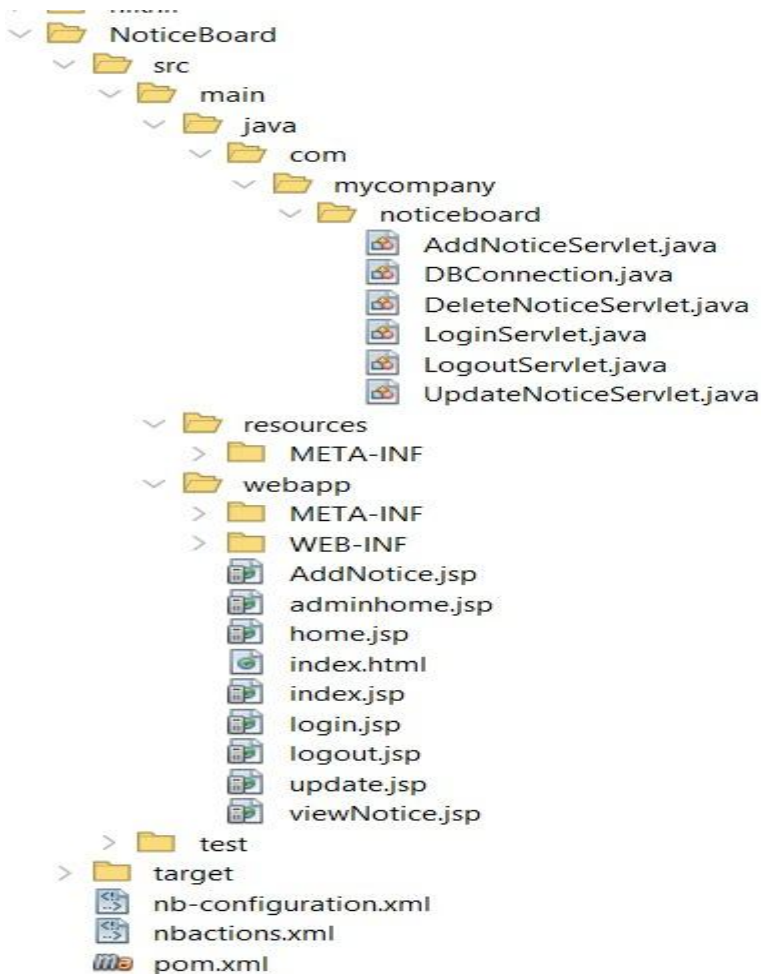
login.jsp x
Source History
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <!DOCTYPE html>
3 <html>
4 <head>
5     <meta charset="UTF-8">
6     <title>Login - NoticeBoard</title>
7 </head>
8 <body bgcolor="#f0f0ff" style="font-family:Arial, sans-serif;">
9
10    <table width="100%" height="100%" align="center">
11        <tr>
12            <td align="center" valign="middle">
13                <table cellpadding="20" cellspacing="0" border="1" bordercolor="#4B9CD3" bgcolor="#ffffff" style="border
14
15                    <tr>
16                        <td align="center" colspan="2">
17                            <h2 style="color: #4B9CD3;">Welcome to NoticeBoard Login</h2>
18                            <p style="font-size: 14px; color: #555;">Please enter your credentials to access your account
19                        </td>
20                    </tr>
21
22                    <% String error = request.getParameter("error"); if (error != null) { %>
23                    <tr>
24                        <td colspan="2" align="center">
25                            <font color="red"><b><%= error %></b></font>
26                        </td>
27                    </tr>
28                    <% } %>
29
30                    <form action="LoginServlet" method="post">
31                        <tr>
32                            <td align="right"><label for="username">Username:</label></td>
33                            <td><input type="text" id="username" name="username" placeholder="e.g. admin123" title="Enter
34                        </tr>
35                        <tr>
36                            <td align="right"><label for="password">Password:</label></td>
37                            <td><input type="password" id="password" name="password" placeholder="Enter your password" title="Enter

```

```

1 package com.myccompany.noticeboard;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 public class DBConnection {
8
9     private static final String URL = "jdbc:mysql://localhost:3306/notice_board?useSSL=false&allowPublicKeyRetrieval=true";
10    private static final String USER = "admin";
11    private static final String PASSWORD = "admin123";
12
13    public static Connection getConnection() {
14        Connection con = null;
15        try {
16            // Load driver (optional with newer versions but safe to include)
17            Class.forName("com.mysql.cj.jdbc.Driver");
18            con = DriverManager.getConnection(URL, USER, PASSWORD);
19        } catch (ClassNotFoundException e) {
20            System.out.println("MySQL JDBC Driver not found.");
21            e.printStackTrace();
22        } catch (SQLException e) {
23            System.out.println("Connection failed.");
24            e.printStackTrace();
25        }
26        return con;
27    }
28 }
29

```



```
adminhome.jsp x
Source History
<%%@page contentType="text/html" pageEncoding="UTF-8"%>
<%%@page import="javax.servlet.http.HttpSession"%>
<%
    HttpSession session1 = request.getSession(false);
    if(session1 == null || session1.getAttribute("username") == null){
        response.sendRedirect("login.jsp");
    }
%>

<!DOCTYPE html>
<html>
<head>
    <title>Admin Home</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f2f2f2;
            padding: 30px;
            text-align: center;
        }

        h2 {
            color: #333;
        }

        .container {
            background-color: white;
            border-radius: 10px;
            display: inline-block;
            padding: 30px 50px;
            box-shadow: 0px 4px 15px rgba(0, 0, 0, 0.1);
        }

        .btn {
            display: block;
            margin: 20px auto;
        }
    </style>
</head>
<body>
    <h2>Admin Home</h2>
    <div class="container">
        <div class="btn">
            <a href="#">Admin Home</a>
        </div>
    </div>
</body>
</html>
```

```
AddNotice.jsp x
Source History
<%%@page contentType="text/html" pageEncoding="UTF-8"%>
<%%@page import="javax.servlet.http.HttpSession"%>
<%
    HttpSession session1 = request.getSession(false);
    if(session1 == null || session1.getAttribute("username") == null){
        response.sendRedirect("login.jsp");
        return;
    }
%>

<!DOCTYPE html>
<html>
<head>
    <title>Add Notice</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background: #f1f1f1;
            padding: 30px;
        }

        .container {
            background-color: white;
            max-width: 600px;
            margin: auto;
            padding: 25px 40px;
            border-radius: 10px;
            box-shadow: 0px 0px 12px rgba(0,0,0,0.2);
        }

        h2 {
            text-align: center;
            color: #2c3e50;
        }

        label {
            display: block;
            margin-bottom: 10px;
        }
    </style>
</head>
<body>
    <h2>Add Notice</h2>
    <div class="container">
        <div class="form">
            <input type="text" value="Notice Title" />
            <input type="text" value="Notice Content" />
            <input type="button" value="Add Notice" />
        </div>
    </div>
</body>
</html>
```



```

update.jsp x
Source History
1 <%@page contentType="text/html" pageEncoding="UTF-8"%>
2 <%@page import="java.sql.*"%>
3 <%@page import="com.mycompany.noticeboard.DBConnection"%>
4
5
6 <%
7     int id = Integer.parseInt(request.getParameter("id"));
8     Connection con = DBConnection.getConnection();
9     PreparedStatement ps = con.prepareStatement("SELECT * FROM notice WHERE id=?");
10    ps.setInt(1, id);
11    ResultSet rs = ps.executeQuery();
12
13    String title = "";
14    String message = "";
15
16    if(rs.next()){
17        title = rs.getString("title");
18        message = rs.getString("message");
19    }
20 >%
21
22 <!DOCTYPE html>
23 <html>
24 <head>
25 <title>Update Notice</title>
26 <style>
27     body {
28         font-family: Arial, sans-serif;
29         background-color: #f4f6f7;
30         padding: 20px;
31     }
32
33     h2 {
34         text-align: center;
35         color: #2c3e50;
36     }
37 </style>
38 </head>
39 <body>
40
41 </body>
42 </html>

```

```

AddNoticeServlet.java x
Source History
1 package com.mycompany.noticeboard;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.sql.Connection;
6 import java.sql.PreparedStatement;
7 import javax.servlet.ServletException;
8 import javax.servlet.http.HttpServlet;
9 import javax.servlet.http.HttpServletRequest;
10 import javax.servlet.http.HttpServletResponse;
11
12 public class AddNoticeServlet extends HttpServlet {
13
14     @Override
15     protected void doPost(HttpServletRequest request, HttpServletResponse response)
16         throws ServletException, IOException {
17
18         response.setContentType("text/html;charset=UTF-8");
19
20         String title = request.getParameter("title");
21         String message = request.getParameter("message");
22
23         try (PrintWriter out = response.getWriter()) {
24             try (Connection con = DBConnection.getConnection();
25                 PreparedStatement ps = con.prepareStatement("INSERT INTO notice(title, message) VALUES(?, ?)")) {
26
27                 ps.setString(1, title);
28                 ps.setString(2, message);
29
30                 int result = ps.executeUpdate();
31
32                 if (result > 0) {
33                     // Redirect with context path to avoid 404 errors
34                     response.sendRedirect(request.getContextPath() + "/viewNotice.jsp");
35                 } else {
36
37                 }
38             }
39         }
40     }
41 }

```

Output

All Notices

[Dashboard](#) [Logout](#)

ID	Title	Message	Action
4	Again Checking...	use to delete sucess	<a href="#">Update</a> <a href="#">Delete</a>
9	again checking	ok ok ok ok	<a href="#">Update</a> <a href="#">Delete</a>
10	hlo hlo	bye bye	<a href="#">Update</a> <a href="#">Delete</a>
11	aip	done	<a href="#">Update</a> <a href="#">Delete</a>
12	Tomorrow is holiday.	Enjoy guyzz	<a href="#">Update</a> <a href="#">Delete</a>

Admin Home

[Welcome, admin 🧑‍💻](#)  
[Add Notice](#)  
[View Notices](#)  
[Logout](#)

All Notices

localhost:8080/NoticeBoard/viewNotice.jsp


All Notices

[Dashboard](#) [Logout](#)

ID	Title	Message	Action
4	Again Checking...	use to delete sucess	<a href="#">Update</a> <a href="#">Delete</a>
11	aip	done	<a href="#">Update</a> <a href="#">Delete</a>

26°C Mostly cloudy

Search



ENG IN

21:10 10-04-2025

Login - NoticeBoard

localhost:8080/NoticeBoard/login.jsp

Welcome to NoticeBoard Login

Please enter your credentials to access your account..

Username:

admin

Password:

\*\*\*\*\*


Login

New here? [Create an account](#)

[Forgot Password?](#)

26°C Mostly cloudy

Search



ENG IN

21:06 10-04-2025

All Notices

localhost:8080/NoticeBoard/viewNotice.jsp

localhost:8080 says  
Are you sure to delete this notice?

OK

Cancel

ID	Title	Message	Action
4	Again Checking...	use to delete sucess	Update  Delete
10	hlo hlo	bye bye	Update  Delete
11	aip	done	Update  Delete

Add Notice

localhost:8080/NoticeBoard/AddNotice.jsp

Post a New Notice

Title:

Tomorrow is holiday

Message:

Enjoy guys

Add Notice

Home

Logout



## Limitations

While the Digital Notice Board System addresses several inefficiencies of traditional notice boards, it also comes with certain limitations that can be improved upon in future iterations:

1. **Single Admin Role:**

The current implementation only supports a single admin account. There is no provision for multiple user roles such as editor, viewer, or moderator. This limits collaborative management and decentralization of tasks.

2. **No User Authentication for Viewers:**

The system does not require users to log in to view notices, which may pose a problem in environments where sensitive or internal communication is shared. There is no distinction between public and private notices.

3. **Basic UI/UX:**

The frontend interface is functional but lacks modern design features such as mobile responsiveness, rich animations, or accessibility features for users with disabilities.

4. **No Notification System:**

The system currently does not support push notifications, emails, or SMS alerts when a new notice is posted. Users need to manually check the platform for updates.

5. **No Expiry or Archival Mechanism:**

Notices remain on the platform unless manually deleted by the admin. There is no automated system to archive or hide outdated notices based on their expiry date.

6. **Lack of Analytics or Reporting:**

The system doesn't provide any statistics or insights like how many people viewed a notice, which can be valuable in understanding reach and engagement.

7. **Limited Security Measures:**

Security measures such as encryption of passwords, input validation, and protection against SQL injection are basic or minimal, which may pose a risk in larger deployments.

## Future Enhancements

Several future enhancements can be made to improve the functionality, scalability, and user experience of the Digital Notice Board System:

### 1. Multi-role Access Control:

Introduce various user roles such as Admin, Editor, Moderator, and Viewer, each with specific access rights. This would allow multiple users to collaboratively manage notices while maintaining accountability.

### 2. User Authentication System:

Implement a login system for users who wish to view personalized or restricted notices, enabling user-specific features and improved data protection.

### 3. Mobile Responsive Design:

Redesign the frontend using responsive frameworks like Bootstrap or Material UI to make the system fully functional across different screen sizes and devices.

### 4. Notification & Subscription System:

Allow users to subscribe to specific notice categories and receive notifications via email, SMS, or in-app alerts whenever new relevant notices are posted.

### 5. Archival and Search Features:

Add automatic expiry, archiving of old notices, and robust search filters to help users find relevant past communications quickly.

### 6. Integration with Calendar and Events:

Link the notice board with a calendar system so that notices related to events automatically appear on a shared institutional calendar.

### 7. Enhanced Security Features:

Implement encryption for sensitive data, CAPTCHA for login forms, session management, and protection against common vulnerabilities like SQL injection and CSRF.

## Conclusion

The Digital Notice Board System is a significant step toward modernizing internal communication within educational institutions, offices, and other organizations. It eliminates the dependency on physical notices, which are often time-consuming to update, environmentally wasteful, and difficult to manage in real-time. Through the use of Java for backend development, Apache Tomcat as a servlet container, and MySQL (via XAMPP) as the database, this system provides a complete and functional web-based solution that is both robust and easy to maintain.

The project has successfully achieved its core objectives—providing an admin interface for managing notices, offering a platform where information can be posted and updated instantly, and ensuring that end-users have access to relevant updates without delays. The modular design of the system also ensures that future improvements and integrations can be carried out smoothly.

This system can serve as a foundation for larger, more complex communication platforms. With additional features like role-based access, automated notifications, and user analytics, the system can be scaled to meet the needs of larger institutions or even public communication networks.

In summary, this project not only demonstrates how web technologies can enhance traditional systems but also highlights the potential for further innovation in how organizations share and manage information. The experience gained through this project contributes to a deeper understanding of full-stack web development, user interface design, and secure information management.

## References

1. **Oracle Java Documentation**  
<https://docs.oracle.com/javase/>  
– Official documentation for Java SE development, including libraries and APIs used for backend logic.
2. **Apache Tomcat Documentation**  
<https://tomcat.apache.org/tomcat-9.0-doc/>  
– Detailed information on configuring and deploying Java web applications on the Tomcat server.
3. **XAMPP and MySQL Documentation**  
<https://www.apachefriends.org/>  
– For setup and management of MySQL databases in a local development environment.
4. **W3Schools - HTML, CSS, and JavaScript Tutorials**  
<https://www.w3schools.com/>  
– Useful frontend web development resources for designing the user interface.
5. **JSP (JavaServer Pages) Tutorials**  
<https://www.javatpoint.com/jsp-tutorial>  
– A comprehensive guide to understanding and using JSP in Java web projects.