VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI 590018

Project Report on

"AI_GUARDIAN"

By

Anushree M (1BM23CS415) Anushri K (1BM23CS416)

Apurva P (1BM23CS417) Archana P (1BM23CS418)

Under the Guidance of

Bhavana

Monisha H M Professor, Department of CSE

BMS College of Engineering

Work carried out at

Department of Computer Science and Engineering

BMS College of Engineering

(Autonomous college under VTU)

P.O. Box No.: 1908, Bull Temple Road, Bangalore-560 019

2025-2026

BMS COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the OOPS with JAVA project titled "**AI GUARDIAN**" has been carried out by Anushree M (1BM25CS415), Anushri K (1BM25CS416), Apurva P(1BM25CS417), Archana P (1BM25CS418) during the academic year 2025- 2026.

Signature of the guide
Bhavana
Assistant Professor,
Department of Computer Science and Engineering
BMS College of Engineering, Bangalore

BMS COLLEGE OF ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We, Anushree M (1BM25CS415), Anushri K (1BM25CS416), Apurva P (1BM25CS417), Archana P (1BM25CS418) III rd Semester, B.E, Department of Computer Science and Engineering, BMS College of Engineering, Bangalore, hereby declare that, this project work entitled "AI-GUARDIAN" has been carried out by us under the guidance of Bhavana , Monisha Professor, Department of CSE, BMS College of Engineering, Bangalore during the academic semester Sep-Dec 2025. We also declare that to the best of our knowledge and belief, the project reported here is not from part of any other report by any other students.

Signature of the Candidates

Anushree M (1BM25CS415)

Anushri K (1BM25CS416)

Apurva P (1BM25CS417)

Archana P(1BM25CS418)

TABLE OF CONTENTS

1) Problem statement
2) Introduction
3) Overview of the project (Block diagram)
4) Tools Used
5) OOPs concept used and its explanation
6) Implementation/code
7) Result /snapshots
8) References

PROBLEM STATEMENT

With the rapid growth of digital communication platforms such as SMS, messag applications, and social media, users are increasingly exposed to online scams, phishi messages, and fraudulent links. These scams are often designed to appear legitimate, making it difficult for users—especially elderly people and non-technical users—to identify potenti threats. The lack of real-time detection and awareness leads many users to unknowingly fall victim to financial fraud, data theft, and privacy breaches. Existing security mechanisms are either too technical, reactive in nature, or fail to provide immediate guidance and support to users.

To address this issue, there is a need for an intelligent and user-friendly system that can analyze suspicious messages and assess their risk level before harm occurs. The system should no only classify messages as low, medium, or high risk but also provide clear safety guidance and alert trusted guardians in critical situations. Therefore, this project aims to develop a Guardian-based Smart Alert System that enhances digital safety by detecting scam messages, guiding users with preventive advice, and ensuring timely intervention through real-time notifications.

INTRODUCTION

In the modern digital era, communication through SMS, instant messaging applications, and online platforms has become an essential part of daily life. While these technologies provide convenience and speed, they have also increased the risk of online scams, phishing messages, and fraudulent activities. Many users receive suspicious messages that attempt to steal personal information or financial details. Identifying such threats is often difficult, especially for elderly users and individuals with limited technical knowledge, making them more vulnerable to cyber fraud.

Traditional security tools such as spam filters and antivirus software provide limited protection and may not detect newly emerging scam techniques in real time. Moreover, these tools usually do not offer clear guidance or immediate assistance to users once a suspicious message is detected. As a result, users may ignore warnings or fail to take appropriate action, leading to serious consequences such as monetary loss and data compromise. This highlights the need for a smarter and more user-friendly system that can assist users in understanding and responding to potential scam messages effectively.

This project proposes the development of a Guardian-based Smart Alert System using Java and object-oriented programming principles. The system analyzes suspicious messages provided by users, classifies them based on risk levels, and provides appropriate safety guidance. In high-risk situations, the system sends real-time alerts to registered guardians to ensure timely intervention. By combining intelligent analysis, user-friendly design, and secure notification mechanisms, the system aims to enhance digital safety and provide reliable protection against online scams.

OVERVIEW OF THE PROJECT

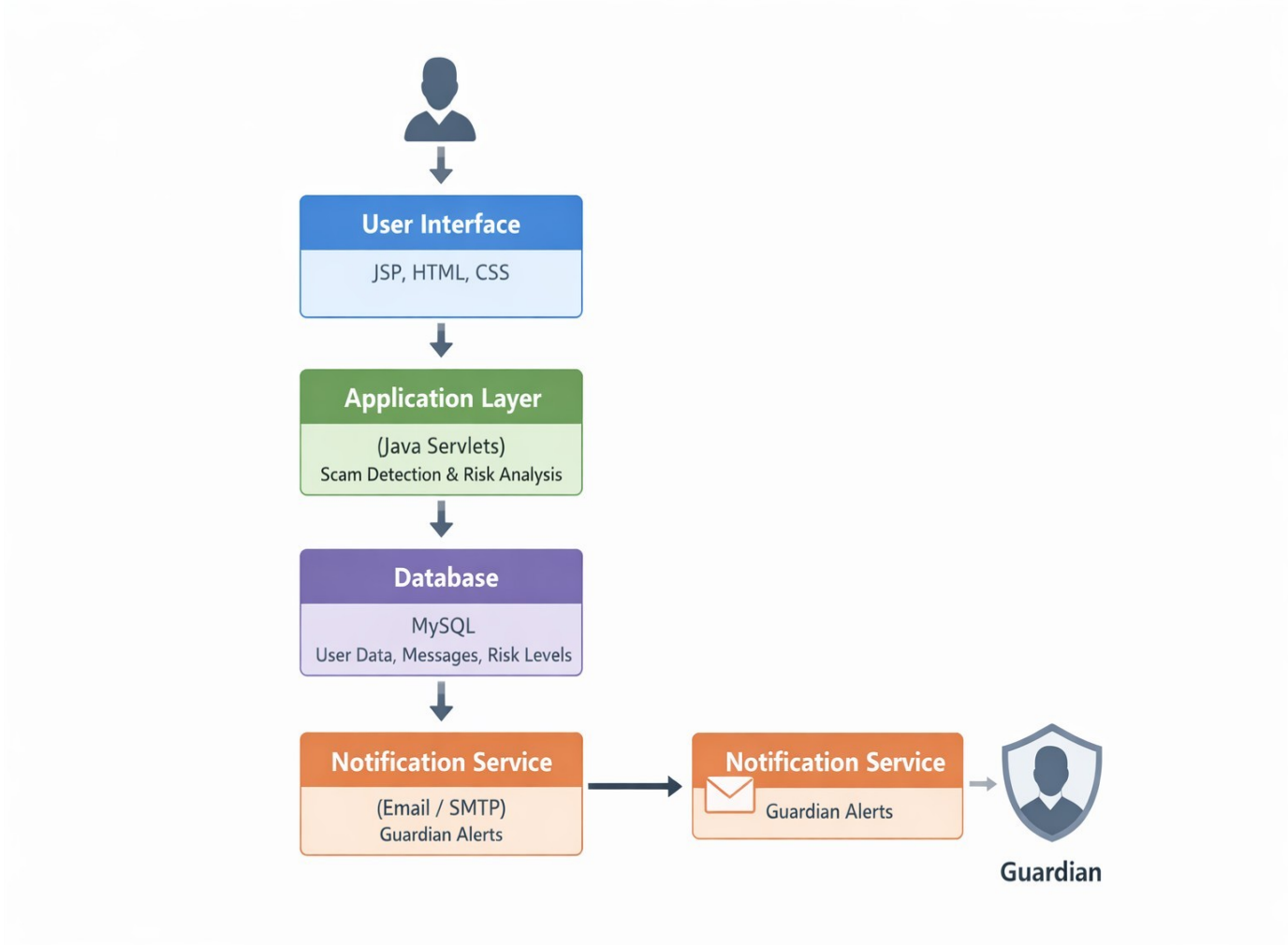
The increasing use of digital communication platforms such as SMS, WhatsApp, and other messaging services, users are frequently exposed to scam messages and fraudulent conten These messages are often designed to appear genuine and may trick users into sharing personal, financial, or sensitive information. Elderly users, students, and non-technical individuals are particularly vulnerable to such scams due to lack of awareness and real-time guidance. Existing security mechanisms provide limited protection and often fail to deliver immediate assistance or alerts in critical situations.

The Guardian-based Smart Alert System is designed to address this problem by providing a secure, intelligent, and user-friendly solution for scam detection and prevention. The system allows users to register and log in securely, after which they can submit suspicious messages received via SMS or messaging applications. These messages are analyzed by the system using rule-based logic to identify scam indicators and determine the level of risk associated with the content.

Once the risk level is identified, the system provides appropriate safety guidance to the user in simple and easy-to-understand language. This helps users make informed decisions and avoid falling victim to scams. In the case of high-risk messages, the system immediately sends an alert notification to a pre-registered guardian or emergency contact via email. This feature ensure timely intervention and adds an additional layer of protection, especially for vulnerable users.

The project also includes multi-language support to improve accessibility and usability. Users can interact with the system in English, Hindi, or Kannada, making it suitable for a wider audience. An admin module is provided to monitor detected messages, risk levels, and alert activities, ensuring effective system management and reliability. The entire system is developed using Java and object-oriented programming principles, deployed on an Apache Tomcat server, and uses MySQL for secure data storage. Overall, the project aims to enhance digital safety by detecting scam messages early, guiding users appropriately, and ensuring guardian support in critical situations.

OVERVIEW OF THE PROJECT (BLOCK DIAGRAM).



TOOLS USED

The Guardian-based Smart Alert System is developed using Java as the primary programming language due to its object-oriented features, platform independence, and strong security support. Java Servlets are used to implement the backend logic of the application, handling user authentication, message analysis, risk classification, and session management. The frontend of the system is designed using JSP, HTML, and CSS, which provide a simple and interactive user interface for users to submit suspicious messages and view risk results and safety guidance. This combination ensures smooth interaction between the user and the system.

For data storage and management, MySQL is used as the database management system. It stores user details, suspicious messages, detected risk levels, and guardian contact information securely. Database connectivity between the Java application and MySQL is established using JDBC, which enables efficient data retrieval and updates. The use of a relational database ensures data consistency, reliability, and easy maintenance of records required for monitoring analysis.

The application is deployed and executed on the Apache Tomcat server, which acts as the web container for running Java Servlets and JSP pages. Eclipse IDE is used as the development environment, providing tools for coding, debugging, and testing the application efficiently. For notification purposes, an Email service using SMTP is integrated to send real-time alerts to guardians in high-risk situations. These tools collectively support the development, deployment, and reliable operation of the system.

OOP's CONCEPTS USED

The Guardian-based Smart Alert System is developed using Java, which follows t Object-Oriented Programming (OOPS) paradigm. OOPS concepts help in structuring the application into reusable, secure, and maintainable components. In this project, concepts such as Class and Object, Encapsulation, Inheritance, Polymorphism, Abstraction, and Message Passing are effectively implemented to achieve modular design and efficient system functionality.

1. Class and Object

A class represents a blueprint for creating objects, while an object is an instance of a class. In this project, various classes are designed to represent real-world entities involved in scam detection and alert management. Examples include classes for user information, messa handling, risk analysis, notification services, and admin operations.

Objects of these classes are created dynamically during runtime when a user registers, logs in, submits a suspicious message, or when an alert is generated. This approach helps in modeling real-world scenarios within the system and ensures proper separation of responsibilities among different components.

2. Encapsulation

Encapsulation is the mechanism of wrapping data and methods together within a class and restricting direct access to data. This is achieved using access modifiers such as private, protected and public.

In the AIGuardian project, sensitive information such as user credentials, guardian email addresses, and risk classification results are declared as private variables. These variables are accessed only through public getter and setter methods. Encapsulation ensures data security, prevents unauthorized access, and maintains data integrity throughout the application.

3. Inheritance

Inheritance allows one class to acquire the properties and methods of another class promoting code reuse and reducing redundancy. In this project, inheritance is used to create specialized classes based on common functionality.

For example, an admin-related class can inherit common attributes and behaviors from a user-related base class while adding additional features such as monitoring alerts and managing syste data. This reduces duplication of code and makes the application easy to extend in the future.

4. Polymorphism

Polymorphism enables the same method to perform different operations depending on the context. In Java, this is achieved through method overloading and method overriding.

In the project, polymorphism is applied in areas such as notification handling and risk processing, where similar method names are used for different behaviors. For instance, a notification method may be overridden to send email alerts for high-risk messages, while another implementation ma handle user warnings. This provides flexibility and dynamic behavior within the system.

5. Abstraction

Abstraction focuses on exposing only essential features while hiding implementat details. In Java, abstraction is implemented using abstract classes and interfaces.

In this project, abstraction is used in defining services such as scam detection notification mechanisms. Interfaces define the required methods, while the actual implementation is handled in concrete classes. This allows changes in internal logic without affecting other modules, improving system scalability and maintainability.

6. Message Passing

Message passing refers to communication between objects using method calls. Objects interact with one another by invoking methods to perform specific tasks.

In the AIGuardian system, message passing occurs when user input is passed from the user interface layer to the backend analysis module. The risk analysis result is then passed to the notification module to trigger guardian alerts when required. This ensures smooth interaction between different modules and maintains logical flow within the system.

7. Modularity and Reusability

By applying OOPS principles, the project is divided into independent and reusable modules such as user management, scam detection, risk analysis, notification, and admin monitoring. Each module performs a specific function and can be modified or extended without affecting the entire system.

For example, the existing email notification service can be extended in the future to include SMS or push notifications. This modular design improves code maintainability and supports future enhancements.

IMPLEMENTATION/CODE

DBConnection.java

AIGuardian > src > main > java > util > DBConnection.java

```java
package util;

import java.sql.Connection;
import java.sql.DriverManager;


public class DBConnection {

    private static final String URL =
        "jdbc:mysql://localhost:3306/ai_guardian";
    private static final String USER = "root";      // change if needed
    private static final String PASSWORD = "apurva";  // change if needed

    public static Connection getConnection() {
        Connection con = null;
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            con = DriverManager.getConnection(URL, USER, PASSWORD);
            System.out.println("✅ Database connected successfully");
        } catch (Exception e) {
            System.out.println("❌ Database connection failed");
            e.printStackTrace();
        }
        return con;
    }
}
```

detect.jsp M

AIGuardian > src > main > webapp > detect.jsp > ?

You, 4 days ago | 1 author (You)

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import="java.util.*" %>
<%
    String lang = (String) session.getAttribute("lang");
    if (lang == null) lang = "en";

    ResourceBundle bundle =
        ResourceBundle.getBundle("i18n.messages", new Locale(lang));
%>

<%
    if (session.getAttribute("user") == null) {
        response.sendRedirect("login.jsp");
        return;
    }
%>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Detect Threat | AI Guardian</title>
</head>
<body>

<h2>AI Threat Detection</h2>

<form action="DetectThreatServlet" method="post">
    <textarea name="inputText" rows="6" cols="50"
        placeholder="Paste suspicious message / call description here..."
        required></textarea>
    <br><br>
    <button type="submit">Analyze</button>
</form>

<br>
<a href="dashboard.jsp">⬅ Back to Dashboard</a>

</body>
</html>
```

dashboard.jsp M

AIGuardian > src > main > webapp > dashboard.jsp > ?

You, 4 days ago | 1 author (You)

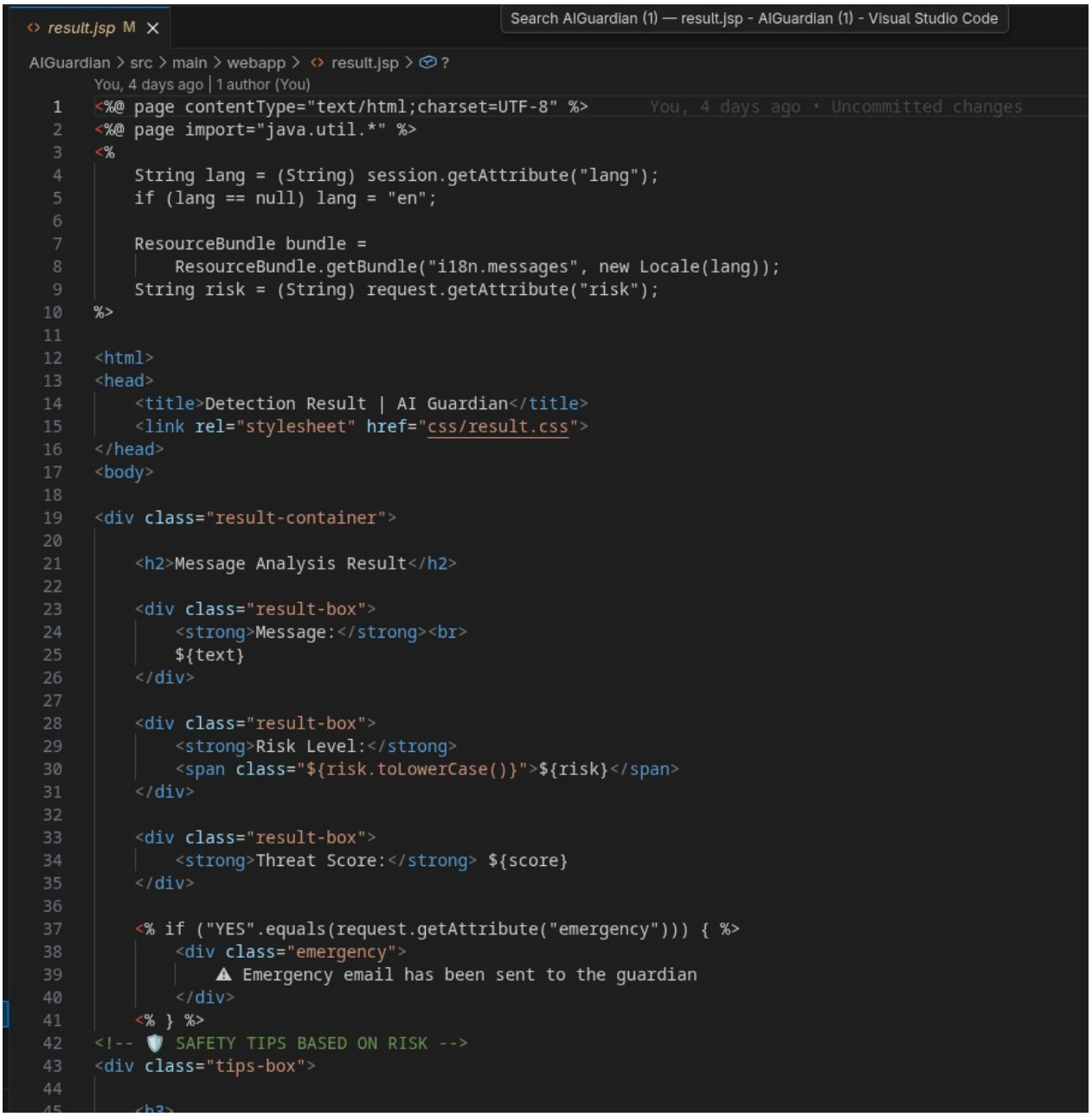
```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import="java.util.*" %>
<%
    String lang = (String) session.getAttribute("lang");
    if (lang == null) lang = "en";

    ResourceBundle bundle =
        ResourceBundle.getBundle("i18n.messages", new Locale(lang));
%>

<%
    Object user = session.getAttribute("user");
    if (user == null) {
        response.sendRedirect("login.jsp");
        return;
    }
%>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Dashboard</title>
<link rel="stylesheet" href="css/global.css">
<link rel="stylesheet" href="css/dashboard.css">
</head>
<body>
<div class="container">
<h2>Welcome to AI Guardian 🎉</h2>
<p>Login successful ✅</p>

<a href="detect.jsp">Detect Threat</a><br><br>
<a href="emergency.jsp">Emergency Contacts</a><br><br>
<a href="logout.jsp">Logout</a>
</div>
</body>
</html>
```

register.jsp M

AIGuardian > src > main > webapp > register.jsp > ? > ? > ? > html > body

You, 4 days ago | 1 author (You)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
    <%@ page import="java.util.*" %>
<%
    String lang = (String) session.getAttribute("lang");
    if (lang == null) lang = "en";

    ResourceBundle bundle =
        ResourceBundle.getBundle("i18n.messages", new Locale(lang));
%>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Register | AI Guardian</title>
<link rel="stylesheet" href="css/global.css">
<link rel="stylesheet" href="css/auth.css">
</head>
<body>

<div class="auth-box">
    <h2>User Registration</h2>

    <form action="RegisterServlet" method="post">
        <input type="text" name="name" placeholder="Full Name" required>
        <input type="email" name="email" placeholder="Email" required>
        <input type="password" name="password" placeholder="Password" required>
        <input type="text" name="phone" placeholder="Phone Number" required>
        <input type="Email" name="guardianEmail" placeholder="Email of trusted family member" requi
        <button type="submit">Register</button>
    </form>

    <p style="text-align:center;">
        Already have an account?
        <a href="login.jsp">Login</a>
    </p>
</div>
      You, 4 days ago • Uncommitted changes
</body>
</html>
```


result.jsp M
Search AIGuardian (1) — result.jsp - AIGuardian (1) - Visual Studio Code
AIGuardian > src > main > webapp > result.jsp > ?
You, 4 days ago | 1 author (You)
<%@ page contentType="text/html;charset=UTF-8" %> You, 4 days ago • Uncommitted changes
<%@ page import="java.util.*" %>
<%
String lang = (String) session.getAttribute("lang");
if (lang == null) lang = "en";
ResourceBundle bundle =
ResourceBundle.getBundle("i18n.messages", new Locale(lang));
String risk = (String) request.getAttribute("risk");
%>
<html>
<head>
<title>Detection Result | AI Guardian</title>
<link rel="stylesheet" href="css/result.css">
</head>
<body>

<h2>Message Analysis Result</h2>

Message:
${text}

Risk Level:
<span class="${risk.toLowerCase()}">${risk}

Threat Score: ${score}

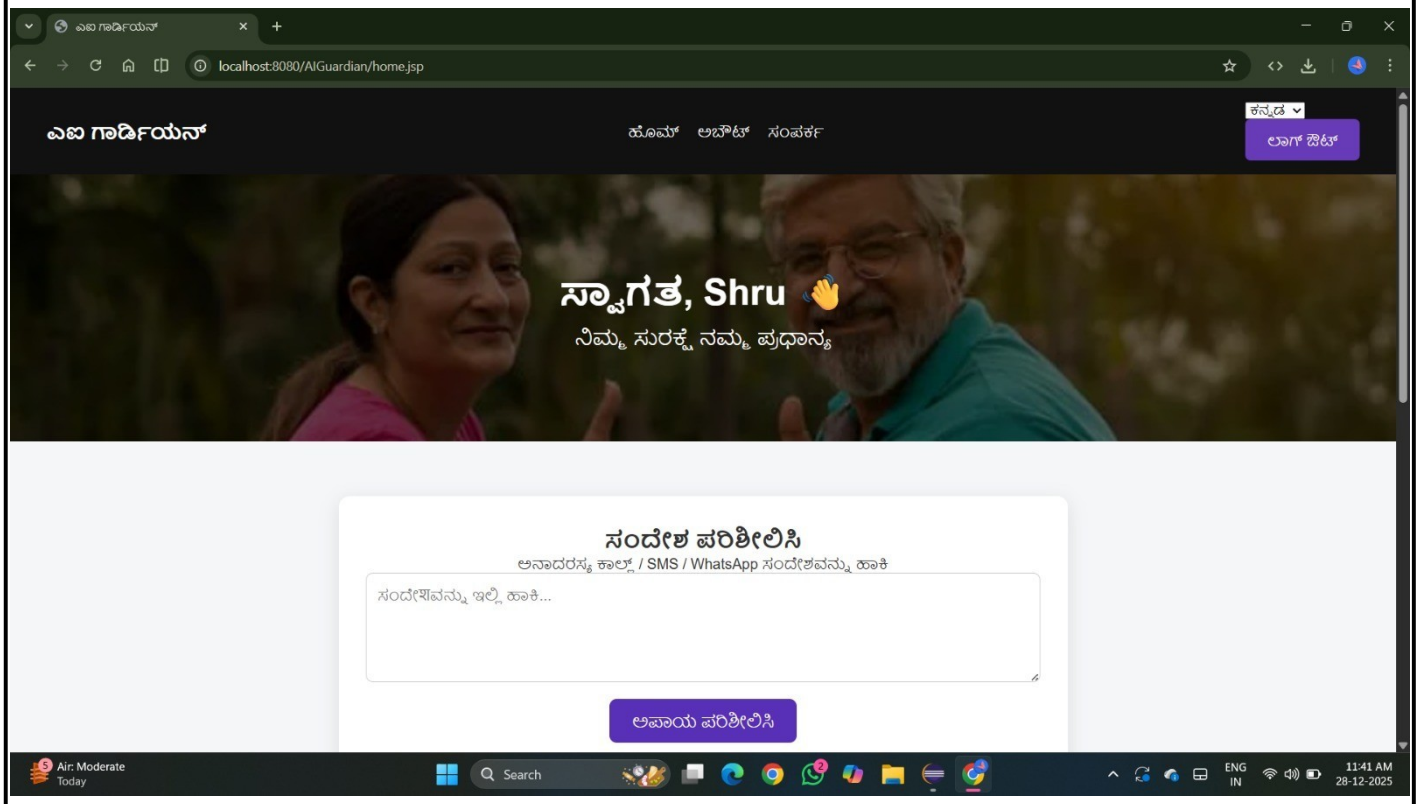
<% if ("YES".equals(request.getAttribute("emergency"))) { %>

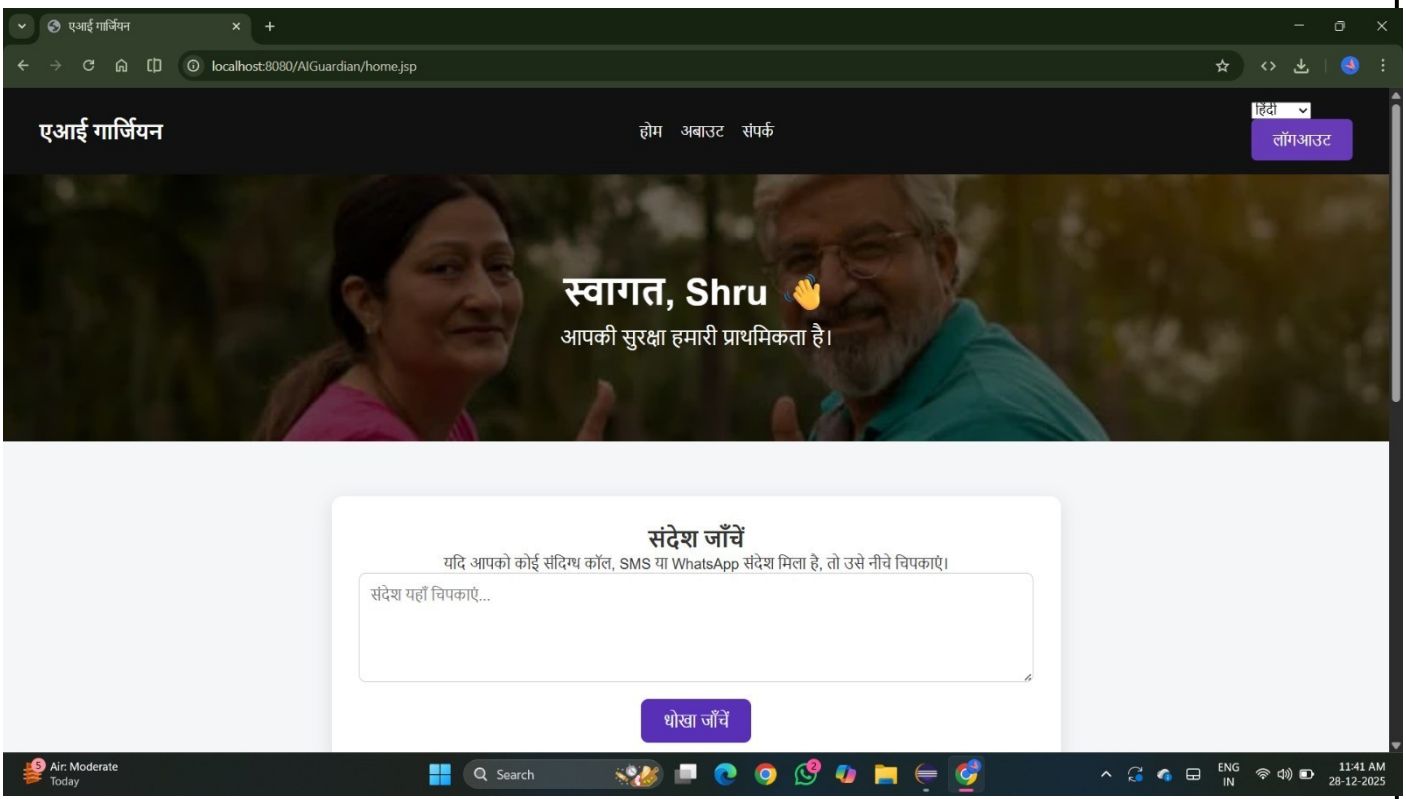
⚠ Emergency email has been sent to the guardian

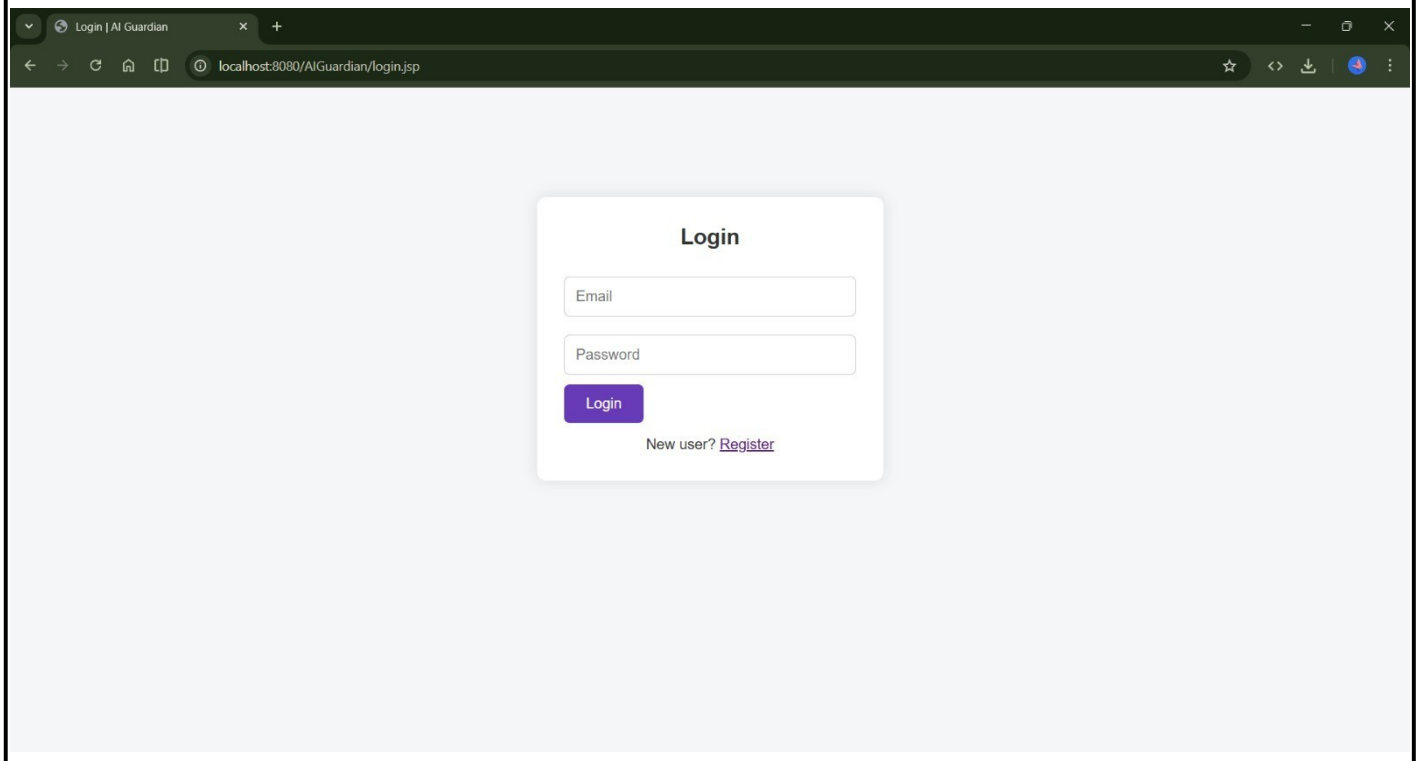
<% } %>
<!-- 🛡️ SAFETY TIPS BASED ON RISK -->

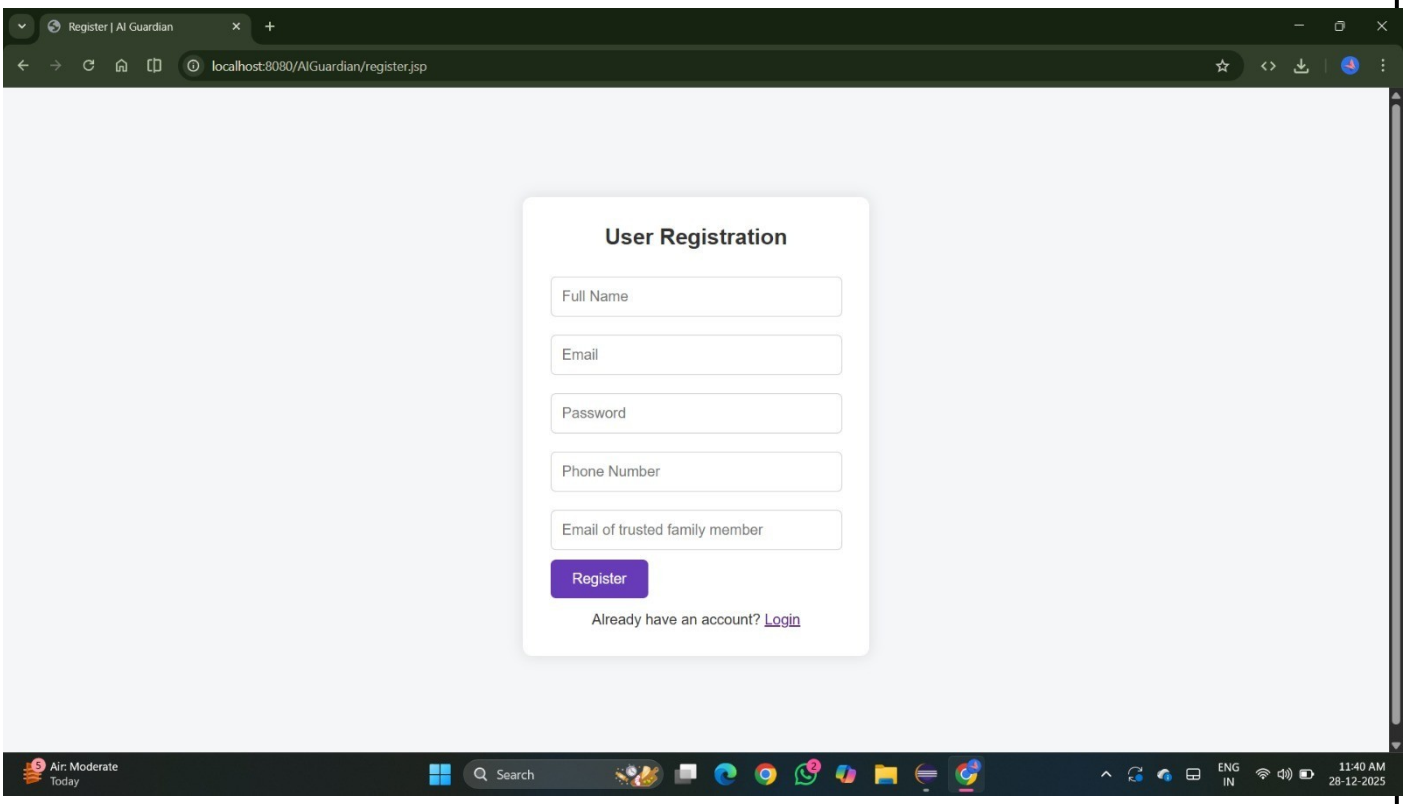
<h3>

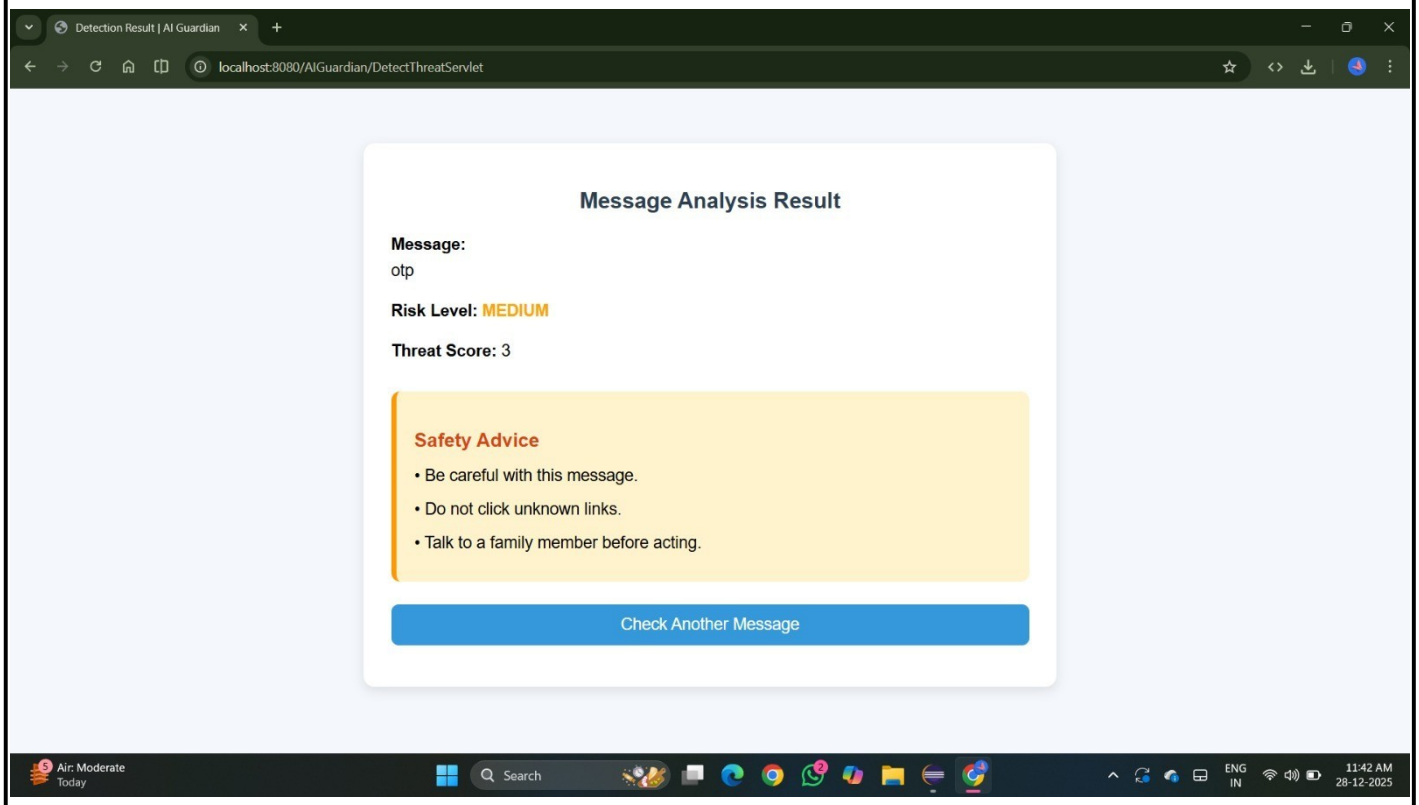
Snapshots

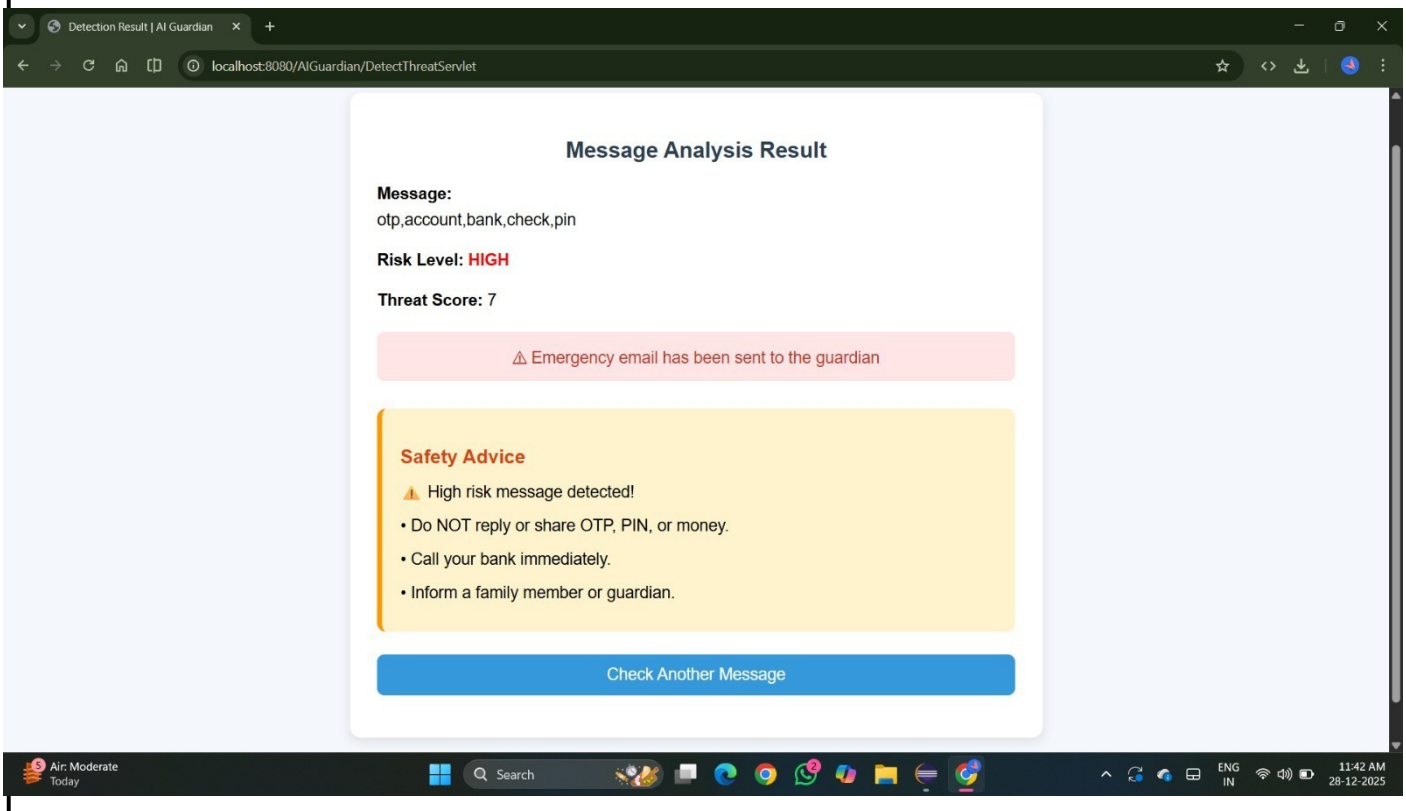

localhost:8080/AIGuardian/home.jsp
ಎಐ ಗಾರ್ಡಿಯನ್
ಹೊಮ್ ಅಬೌಟ್ ಸಂಪರ್ಕ
ಕನ್ನಡ
ಲಾಗ್ ಔಟ್
ಸ್ವಾಗತ, Shru
ನಿಮ್ಮ ಸುರಕ್ಷೆ ನಮ್ಮ ಪ್ರಧಾನ್ಯ
ಸಂದೇಶ ಪರಿಶೀಲಿಸಿ
ಅನಾದರಸ್ಯ ಕಾಲ್ಸ್ / SMS / WhatsApp ಸಂದೇಶವನ್ನು ಹಾಕಿ
ಸಂದೇಶವನ್ನು ಇಲ್ಲಿ ಹಾಕಿ...
ಅಪಾಯ ಪರಿಶೀಲಿಸಿ


localhost:8080/AIGuardian/home.jsp
एआई गार्जियन
होम अबाउट संपर्क
हिंदी
लॉगआउट
स्वागत, Shru
आपकी सुरक्षा हमारी प्राथमिकता है।
संदेश जाँचें
यदि आपको कोई संदिग्ध कॉल, SMS या WhatsApp संदेश मिला है, तो उसे नीचे चिपकाएं।
संदेश यहाँ चिपकाएं...
धोखा जाँचें


Login | AI Guardian
localhost:8080/AIGuardian/login.jsp
Login
Email
Password
Login
New user? Register


Register | AI Guardian
localhost:8080/AIGuardian/register.jsp
User Registration
Full Name
Email
Password
Phone Number
Email of trusted family member
Register
Already have an account? Login
Air: Moderate
Today
Search
ENG
IN
11:40 AM
28-12-2025


Detection Result | AI Guardian
localhost:8080/AIGuardian/DetectThreatServlet
Message Analysis Result
Message:
otp
Risk Level: MEDIUM
Threat Score: 3
Safety Advice
• Be careful with this message.
• Do not click unknown links.
• Talk to a family member before acting.
Check Another Message
Air: Moderate
Today
Search
ENG
IN
11:42 AM
28-12-2025


Detection Result | AI Guardian
localhost:8080/AIGuardian/DetectThreatServlet
Message Analysis Result
Message:
otp,account,bank,check,pin
Risk Level: HIGH
Threat Score: 7
⚠ Emergency email has been sent to the guardian
Safety Advice
⚠ High risk message detected!
• Do NOT reply or share OTP, PIN, or money.
• Call your bank immediately.
• Inform a family member or guardian.
Check Another Message
Air: Moderate
Today
Search
ENG
IN
11:42 AM
28-12-2025

References

Oracle Corporation, Java SE Documentation.
https://docs.oracle.com/javase/
Oracle Corporation, Java Servlet Technology Documentation.
https://docs.oracle.com/javaee/7/tutorial/servlets.htm
Apache Software Foundation, Apache Tomcat Documentation.
https://tomcat.apache.org/documentation.html
MySQL, MySQL Official Documentation.
https://dev.mysql.com/doc/
GeeksforGeeks, Object Oriented Programming (OOPS) Concepts in Java.
https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-java/
W3Schools, JSP and Servlet Tutorial.
https://www.w3schools.com/jsp/
JavaTpoint, Java Mail API Tutorial.
https://www.javatpoint.com/java-mail-api