IBM Innovation Center for Education

PHEMESOFT
Software That Matters
TRUSTED DELIVERY PARTNER for IBM ICE

RABBI ZIDNI 'ILMA
YENEPOYA
(DEEMED TO BE UNIVERSITY)

THE YENEPOYA INSTITUTE OF ARTS SCIENCE
COMMERCE AND MANAGEMENT
(a constituent unit of Yenepoya Deemed to be University)

# CLOUD SECURITY THREAT MONITORING SYSTEM
# PROJECT SYNOPSIS

MASTER OF COMPUTER APPLICATIONS

| | | |
|---|---|---|
| SUBMITTED BY: | | GUIDED BY: |
| Nandhana P | 24MCA217 | MR. SHASHANK |
| Nandana M V | 24MCA215 | |
| Dasari Tulasi | 24MCA206 | |

# TITLE PAGE

1. Name of the student: Dasari Tulasi

2. Class Roll No. 24MCA206

3. Campus ID: 35563

4. Present official Address: YIASCM Balmatta, Mangalore 575002

5. Email: 35563@yenepoya.edu.in

6. Phone No. +91 7396493543

7. Branch: Computer Science

8. Batch: 2024-2026

9. Proposed Topic: Cloud Security Threat Monitoring System

# TABLE OF CONTENTS

## 1.1 Introduction

This project aims to develop a Cloud Security Threat Monitoring System that monitors cloud-based environments for suspicious activities, policy violations, and unauthorized access. As cloud infrastructure becomes increasingly critical to businesses, real-time threat visibility and response capabilities have become essential.

The system utilizes cloud-native logging and monitoring services along with third-party threat intelligence to detect threats early and issue alerts to administrators. It aggregates logs from multiple sources like AWS CloudTrail, Azure Monitor, and others to create a unified threat analysis view, enhancing situational awareness and reducing response times.

## 1.2 Key Features

✓ **Cloud Log Aggregation:** Collects real-time logs from cloud services such as AWS, Azure, and GCP.

✓ **Threat Detection:** Identifies unusual activity, failed logins, open ports, and misconfigurations.

✓ **Threat Intelligence Integration:** Correlates data with external threat feeds for contextual analysis.

✓ **Real-Time Alerting:** Sends alerts via email and dashboard on detecting critical threats.

✓ **Custom Detection Rules:** Allows administrators to define detection patterns as per their cloud policy.

✓ **Reporting and Visualization:** Generates comprehensive reports with severity scores and remediation steps.

## 1.3 Technology Stack

**Frontend:**

✓ **ReactJS:** For building a responsive dashboard for threat visibility.
✓ **Chart.js / D3.js:** For rendering real-time analytics and visualizations.

**Backend:**

✓ **Python (Flask):** For handling logic and APIs.
✓ **AWS/Azure SDK:** To interface with cloud services.
✓ **Elasticsearch:** For indexing logs.
✓ **MongoDB:** For data storage and rule configuration.
✓ **Kibana:** For advanced visualization and threat hunting.

## 1.4 Specialized Field:

**Cloud Security**

This project falls under the field of **Cloud Security**, particularly focusing on **Security Information and Event Management (SIEM)** within cloud infrastructures. As businesses migrate to the cloud, monitoring and responding to threats in real-time becomes essential. This system provides a centralized solution for threat detection, mitigation, and reporting in cloud ecosystems, helping organizations secure sensitive data and comply with industry standards.

## 2.1 Methodology

The development of the Cloud Security Threat Monitoring System will follow a modular and agile approach to ensure continuous integration and testing. The system will be developed in phases, including cloud log ingestion, threat detection logic, alert generation, and dashboard development.

## 2.2 Requirement Analysis and Tool Selection

✓ Identify cloud services to be monitored: AWS, Azure, GCP.
✓ Select appropriate APIs and SDKs for log access and control.
✓ Choose open-source tools like ELK Stack, Suricata, and cloud-specific libraries.
✓ Define scope of detection and type of threats to monitor.

## 2.3 System Architecture and Design

✓ Design a modular architecture with log ingestion, processing, storage, and alerting units.
✓ Define flow between data collectors, backend processing units, and the frontend dashboard.
✓ Map interactions with external threat intelligence sources.

## 2.4 Frontend Development (Tkinter)

✓ Create a dynamic dashboard using ReactJS.
✓ Design components for viewing alerts, filtering by service/severity, and exporting reports.
✓ Implement user authentication and access control.

## 2.5 Backend Integration

✓ Implement Flask APIs to process log data and apply detection rules.
✓ Integrate with cloud provider APIs to fetch logs and security events.
✓ Connect with Elasticsearch for real-time indexing and Kibana for visual analysis.
✓ Automate alert workflows through email or messaging services.

## 2.6 Final Testing and Documentation

✓ Perform functional, performance, and security testing.
✓ Test on real cloud environments (using free-tier or trial accounts).
✓ Document system architecture, configurations, and user manual.

### 3.1 Facilities required for proposed work

The development of this project requires cloud service accounts, open-source monitoring tools, and a development environment with Python and JavaScript support.

### 3.2 Development Environment

✓ Python 3.12

✓ ReactJS

✓ Visual Studio Code

✓ Git and GitHub for version control

✓ Postman for testing APIs

### 3.3 Monitoring Tools

✓ AWS CloudWatch and CloudTrail

✓ Azure Monitor and Log Analytics

✓ Elasticsearch for log indexing

✓ Suricata or Snort (optional) for packet-level threat detection

✓ Kibana for visualizing indexed logs and alerts

### 3.4 Testing and Deployment

✓ AWS / Azure Free Tier accounts for simulation

✓ Docker for containerization

✓ Jenkins / GitHub Actions for CI/CD

✓ Kali Linux (for simulation of benign and malicious activities)

✓ Multi-platform testing (Windows/Linux)

### 3.5 Reporting Tools

✓ ReportLab / WeasyPrint for PDF report generation

✓ HTML report export feature in dashboard

✓ Scheduled email reports using Python’s SMTP libraries or cloud functions