Project 2 – Open Source Incident Management System

Introduction

Incident Management is a key process in IT operations and software development, ensuring that infrastructure and application issues are properly tracked, logged, and resolved. Effective incident management prevents downtime and improves system reliability.
This project delivers an open-source Incident Management System built using Python Flask, SQLite, Bootstrap, and Docker. It allows users to log new incidents, track existing ones, and resolve them in a simple web portal. Notifications are sent by email to keep stakeholders updated in real time. The project showcases core DevOps concepts such as automation, containerization, and safe handling of sensitive data.

Abstract

The Incident Management System is designed to provide a lightweight solution for tracking infrastructure and application incidents.

Users can:

- Log new incidents with a title and description.
- View a list of existing incidents with their status (Open or Resolved).
- Mark incidents as resolved with a single click.
- Whenever an incident is created or resolved, an automatic email is sent to notify the responsible parties.
- The system stores all incidents persistently in an SQLite database, ensuring no data is lost between restarts.
- Docker is used to containerize the app for easy deployment, and all secrets are managed via .env files to prevent accidental exposure of sensitive information.

Tools Used

- **Python (Flask Framework):** For building the backend web application and REST APIs.
- **SQLite:** Lightweight relational database to store incident data.
- **Bootstrap:** Frontend framework to create a simple and responsive UI.
- **Flask-Mail:** To send automated email notifications via SMTP (Gmail).
- **Docker:** To package the application in a container for portability and deployment ease.
- **Git & GitHub:** For version control and source code management.

Steps Involved in Building the Project

1. **Project Setup:** Created a Flask application with routes for adding, listing, and resolving incidents.
2. **Database Integration:** Implemented SQLite to store incidents with fields: ID, Title, Description, and Status.
3. **Email Notification:** Configured Flask-Mail with Gmail SMTP and handled sending emails when incidents are created or resolved.
4. **Frontend Development:** Built simple HTML pages using Bootstrap for listing incidents, adding new ones, and resolving them via buttons.
5. **Environment Management:** Used .env files to store sensitive data (MAIL_USERNAME, MAIL_PASSWORD), and ensured they are excluded from GitHub using .gitignore.
6. **Containerization:** Wrote a Dockerfile to containerize the application and tested it locally using Docker commands.
7. **Version Control & Deployment:** Pushed the clean, secure project code (without secrets) to GitHub for version control.

Conclusion

This Incident Management System demonstrates key principles of modern software development and DevOps practices. It simplifies tracking and resolving incidents while ensuring real-time notifications to stakeholders. Containerization via Docker enables consistent deployment environments, and secrets management with .env ensures security best practices.