# Open-Source Incident Management System (IMS)

## Introduction

The **Open-Source Incident Management System (IMS)** is designed to streamline the process of logging, tracking, and resolving infrastructure or application-related issues within an organization. This system provides a centralized platform where users can report incidents and administrators can efficiently manage and resolve them. With a user-friendly interface and role-based access control, the system ensures secure and organized handling of incidents, improving operational efficiency and communication among teams.

---

## Abstract

The **Incident Management System (IMS)** is a Flask-based web application that offers a simple yet efficient approach to handle incidents. It leverages **RESTful APIs** to create, update, assign, and resolve incidents, while integrating **SQLite** for data persistence. The project is **containerized using Docker** for easy deployment and scalability, ensuring consistent behavior across environments. Additionally, **SMTP-based email notifications** are used to alert users about updates in incident status.

This project serves as a foundational tool for understanding **DevOps workflows**, **API integration**, and **containerization**, demonstrating how automation and modular development can streamline IT service management.

---

## Tools Used

1. **Python (Flask):** Used to develop REST APIs and handle backend logic.
2. **SQLite:** Lightweight relational database for storing incidents and user details.
3. **Docker:** Containerization tool that ensures portability and environment consistency.
4. **Git:** Version control system for managing, tracking, and collaborating on code changes.
5. **SMTP:** Simple Mail Transfer Protocol used for sending automated email notifications about incident updates.

---

## Steps Involved in Building the Project

1. **Project Setup:** Initialize the Flask project structure and configure the SQLite database schema.
2. **API Development:** Build RESTful APIs for incident creation, updates, assignment, and resolution.
3. **Email Integration:** Configure SMTP settings to send real-time notifications to users about incident changes.
4. **Frontend Design:** Develop responsive HTML templates for better user experience.
5. **Dockerization:** Write a Dockerfile to containerize the application, simplifying deployment and scalability.
6. **Version Control:** Push the project to a Git repository to maintain version history and enable collaboration.
7. **Testing & Deployment:** Test all components locally and deploy the Docker container in a controlled environment.

---

## Conclusion

The Open-Source Incident Management System demonstrates the integration of DevOps principles in a practical project by combining web development, database management, and containerization. Using Flask, Docker, and SQLite, the project achieves modularity, simplicity, and scalability. The inclusion of role-based access and email notifications enhances usability and communication.

Overall, this project serves as a strong foundation for building more advanced, production-ready Incident Management or IT Service Management (ITSM) solutions that align with real-world organizational needs.