Architecture and Code Analysis of the Social Notifier System

This document explains the internal working of the Social Notifier project developed during Hackathon 2017. The intention of this document is to describe the architecture, technologies used, database structure, and overall logic of the system in a simple and realistic manner. The explanations are written from a developer’s point of view rather than as system-generated content.

1. Architecture Overview

The Social Notifier system is built using an event-driven architecture. Instead of relying on user interactions, the system reacts automatically to monitoring events generated by the IBM Domino server. These events are captured and forwarded to external collaboration platforms through integration services.

In simple terms, whenever an important system event occurs, the notifier plugin picks it up and sends a message to Watson Workspace. From there, the message can be delivered to tools such as Slack, allowing teams to respond quickly.

2. Technologies Used

The core implementation of the project is written in Java and runs inside the IBM Domino environment. The project uses the OSGi framework to package the notifier as a modular plugin. Communication between Domino and Watson Workspace is handled through RESTful APIs using JSON-formatted messages. All data transmission is secured using HTTPS.

3. Database Structure

This project does not use a traditional relational database such as MySQL or PostgreSQL. Instead, it relies on IBM Domino’s built-in document-oriented database system. Data is stored as documents that are created and updated automatically by the system.
These documents store information such as event details, timestamps, alert messages, and configuration data required for connecting to Watson Workspace. Forms define the structure of these documents, while views are used to display and filter them for administrative purposes.

4. Code Logic and Workflow

The logic of the system begins with Domino Domain Monitoring, which continuously checks the health and status of the Domino server. When a monitoring condition is triggered, the Social Notifier plugin intercepts the event. The event information is then transformed into a readable message format.

After formatting the message, the plugin authenticates itself using the configured credentials and sends the notification to Watson Workspace through an HTTP request. This entire process happens automatically without manual intervention.

5. Conclusion

The Social Notifier project demonstrates how enterprise monitoring systems can be integrated with modern collaboration platforms. By using an event-driven approach, the system ensures that important alerts are delivered in real time. This document was prepared to clearly explain the system behavior and can be used as a reference for further analysis or AI-based question answering.