Architecture Design

Organization-Wide Alert System

For

CS 895 MSE Project

Department of Computer Science

Kansas State University

Submitted to

Dr. Mitch Neilsen

Dr. Torben Amtoft

Dr. Scott Deloach

Submitted by

Angela Hall

2/12/2019

1. Overview

This document lays out the complete architectural design of the Organization-Wide Alert System

1.1. High Level Overview

1.1.1. Deployment Diagram

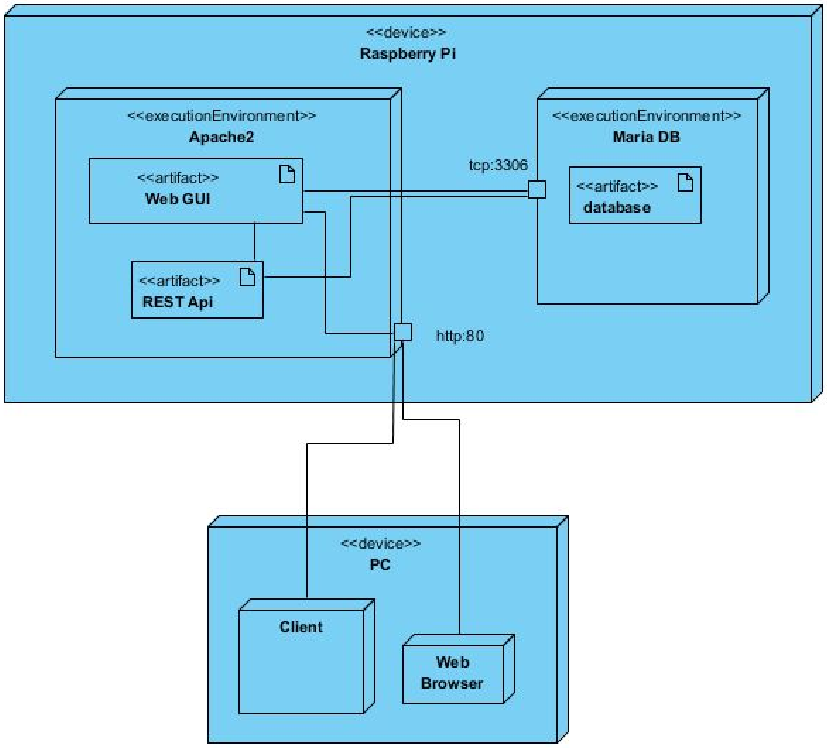


Figure 1.1

The system consists of a server running on a Raspberry Pi device and numerous clients running on local PCs throughout the organization. The server runs an Apache2 web server which contains a Web GUI application and a REST API. The server also runs a MariaDB database. Each PC runs a Java client and interacts with the GUI via a web browser.

2. Architecture

There are four main parts of the Organization-Wide silent alert system: a Web-based GUI that allows users to perform server operations, a REST API that delivers server resources, a database that holds the state of each alarm, and a client that runs on users' local machines and pops up alarm notifications.

2.1. Database

2.1.1. Database Tables

The database is a MariaDB database with three tables:

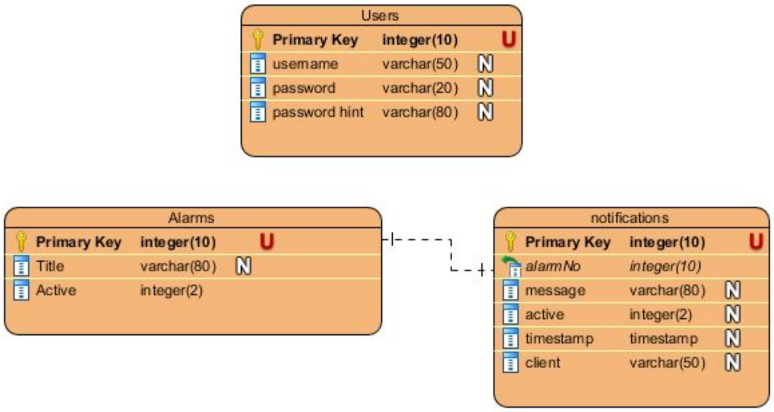
alarms

notifications
users

2.1.2. Entity Relationship Diagram

The users table associates users who are allowed to trigger alarms with their password. The Alarms table holds the current state of each alarm. The active column of the alarms table can hold either a 1 or 0 indicating that an alarm is either currently active or inactive.

The notifications table records a history of actions taken, including which alarms were involved and their state at the time.



Users

Primary Key	integer(10)	U
username	varchar(50)	N
password	varchar(20)	N
password hint	varchar(80)	N

Alarms

Primary Key	integer(10)	U
Title	varchar(80)	N
Active	integer(2)	

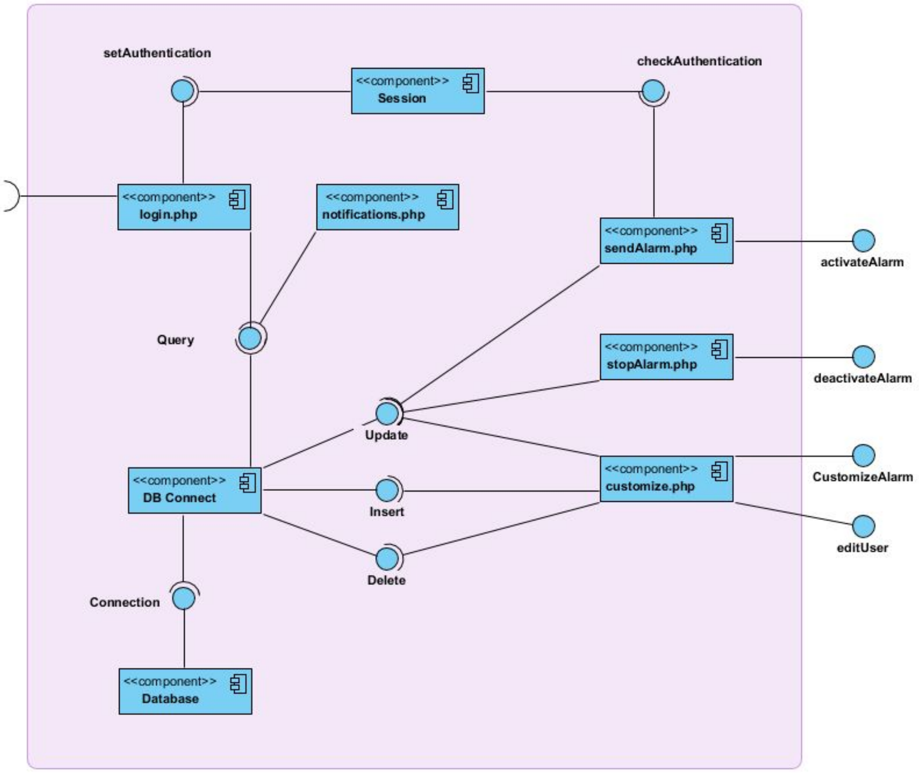
notifications

Primary Key	integer(10)	U
alarmNo	*integer(10)*	
message	varchar(80)	N
active	integer(2)	N
timestamp	timestamp	N
client	varchar(50)	N

2.2. Web based GUI

2.2.1. Web application design

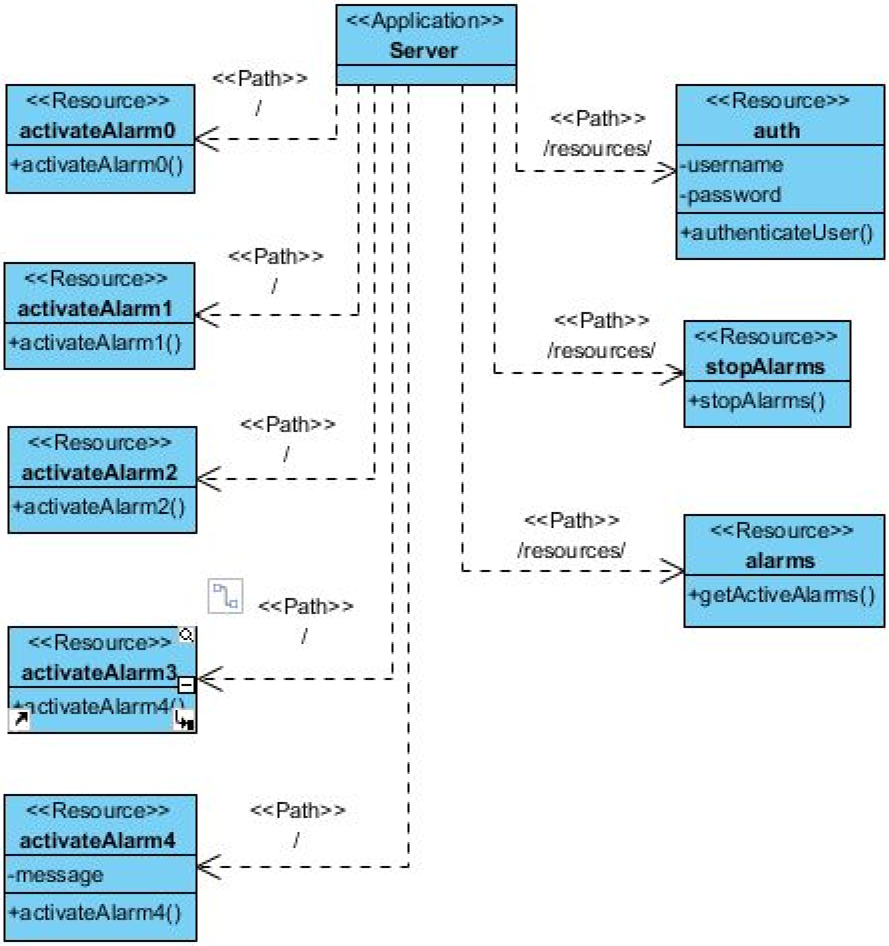
The web GUI provides an interface allowing the user to interact with the database. The following component diagram shows the web application’s components and the functions exposed to the user.



2.3. Rest API

The REST API is formally specified in the Formal Requirements Specification document using the OpenAPI specification language.

The following is a class diagram depicts the resources offered by the REST API and their given path.

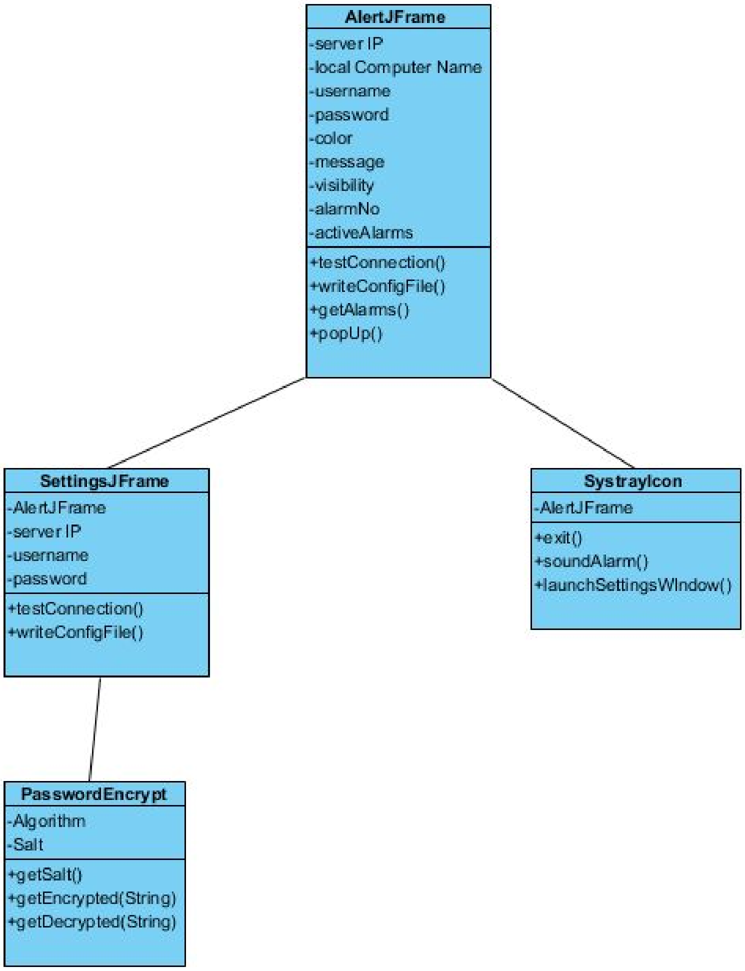


2.4. Client

The client runs on each users local PC. It sends a GET request to the server every 8 seconds. If the results indicate an active alarm, it displays a color-coded popup to the user indicating the alarm message.

The client also has a settings window for changing and recording user and network settings, and a system tray icon to access the client’s features.

The following class diagram represents the Client application.


AlertJFrame
-server IP
-local Computer Name
-username
-password
-color
-message
-visibility
-alarmNo
-activeAlarms
+testConnection()
+writeConfigFile()
+getAlarms()
+popUp()
SettingsJFrame
-AlertJFrame
-server IP
-username
-password
+testConnection()
+writeConfigFile()
SystrayIcon
-AlertJFrame
+exit()
+soundAlarm()
+launchSettingsWIndow()
PasswordEncrypt
-Algorithm
-Salt
+getSalt()
+getEncrypted(String)
+getDecrypted(String)