Component Design Document

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

5/5/2019

1. **Overview**

This document details the internal design of the individual components of the Organization-Wide Silent Alert System.

2. **Components**

There are four main components of this system

- Rest-API
- Database
- Web-Based User Interface
- Java-based client program.

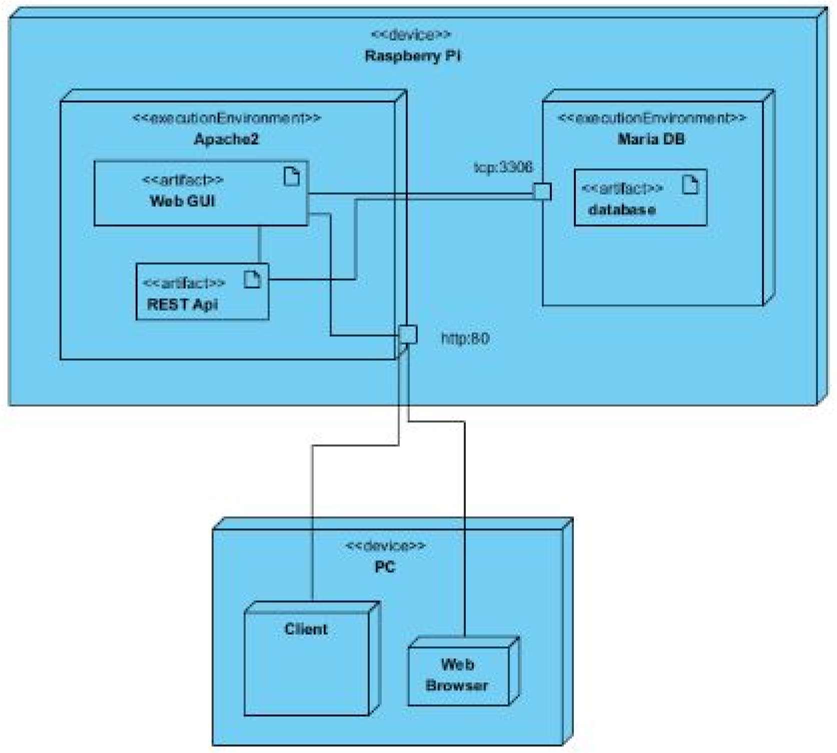


Fig 2.1 Deployment Diagram

The deployment diagram illustrates the relationships between the Web GUI, the Rest API, the Database, and the client program.

2.1. **Client Component Detail**

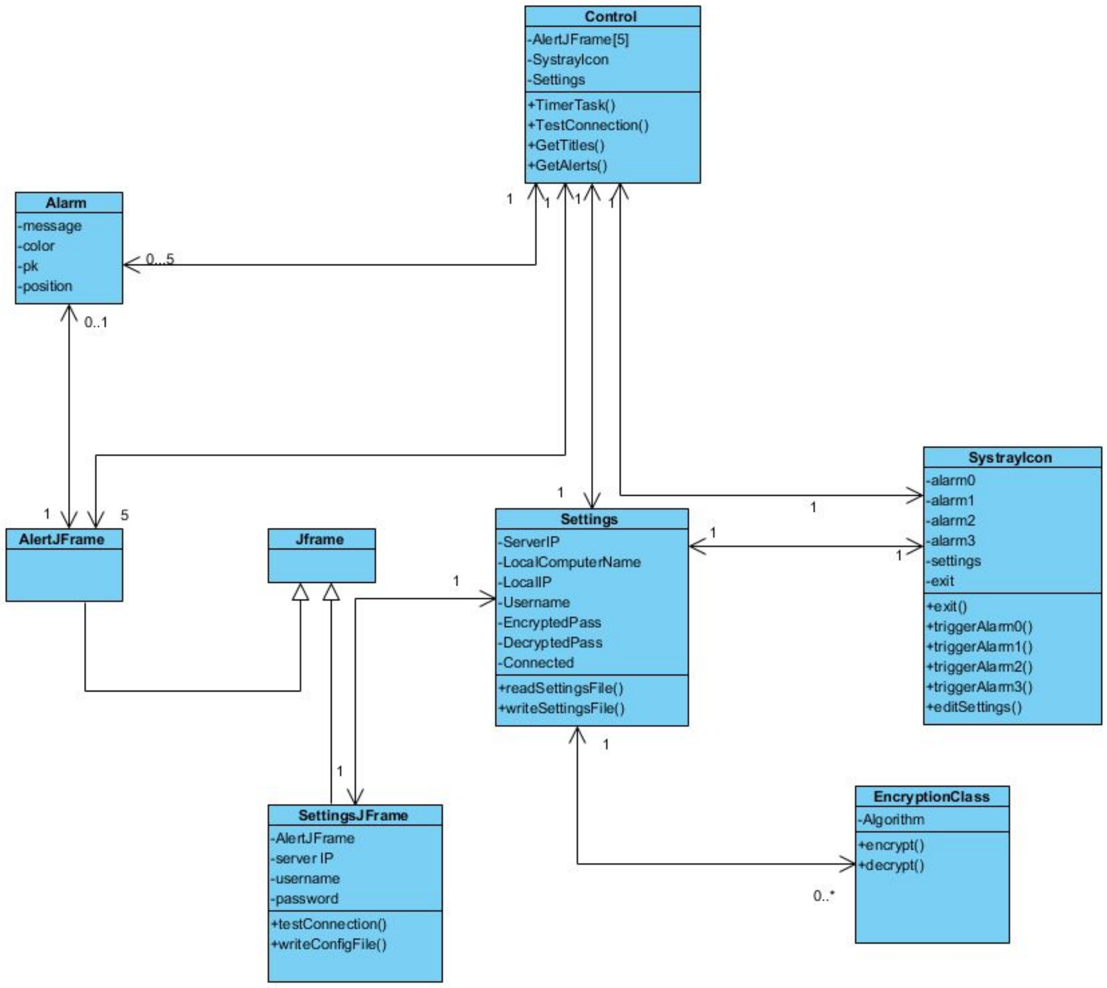
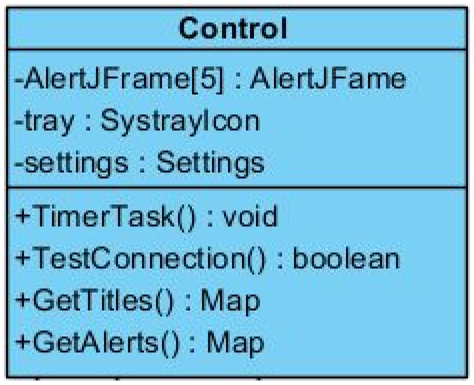


Fig 2.2 The Class Diagram for the Java Component

2.1.1. Control



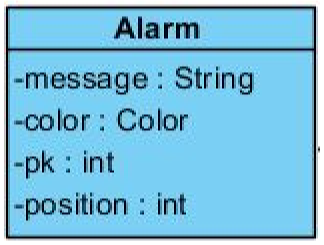
The control class is the main launching point for the program.

It instantiates the AlertJFrames, the system Settings, and the SystrayIcon and begins the TimerTask loop.

In the TimerTask loop, the Control Class makes two calls to the server's RestAPI. The first call retrieves a list of the Alarm Titles. The second call retrieves a list of active alarms. It parses the server's response and instantiates an Alarm class for each active alarm. It then updates the SystrayIcon with the most recent alarm titles.

This repeats every 8 seconds.
Attributes:

2.1.2. **Alarm**

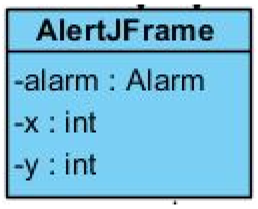


The Alarm class holds the properties of each active alarm. Each Alarm class is instantiated with:

- message - this is the title or message of the corresponding server alarm
- a color - this is the same color as the corresponding alarm on the server
- a pk or primary key - this is the way the alarm is identified by the database and the SystrayIcon class
- a position - a "y" coordinate offset that changes depending on the number of currently active alarms, so as to make the alarm windows appear "stacked" on one another.

  The Control can have a minimum of 0 and a maximum of 5 active alarms.

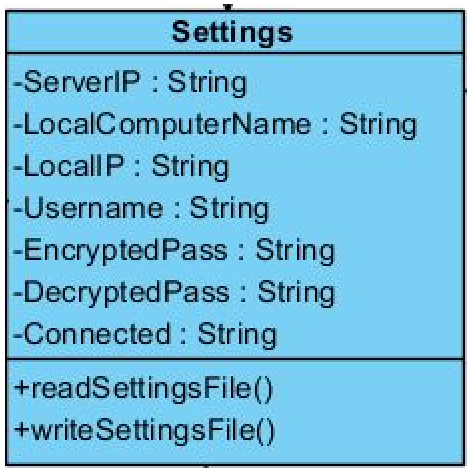
2.1.3. **AlertJFrame**



The AlertJFrame extends JFrame and is the visual windowed class that displays each alarm.

The color and location of the AlertJFrame varies depending on the properties of the Alarm it is displaying.
The y variable is passed from the Alarm. The x is set depending on the user's screen size.

2.1.4. Settings



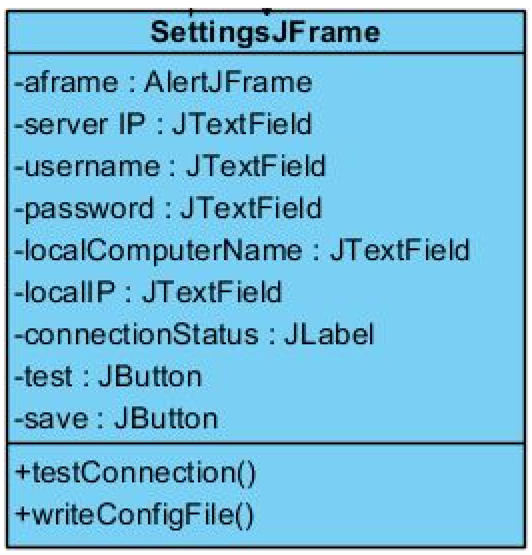
The Settings class holds settings used by the other classes.
These include

ServerIP - the IP address of the server
LocalComputerName - the hostname of the local computer
LocalIP - the IP address of the local computer
Username - the username used to authenticate to the server
EncryptedPass - the user's password in its encrypted form
DecryptedPass - the user's password in it decrypted form
Connected - a boolean variable that is set to true if the client successfully connects with the given settings.

The decrypted password is only used to display the password to the user on the Settings screen. Anytime the password is stored or transmitted the encrypted password is used.

The local IP address and hostname are queried from the system the first time the program is run, but the user can manually set them if no data is received.

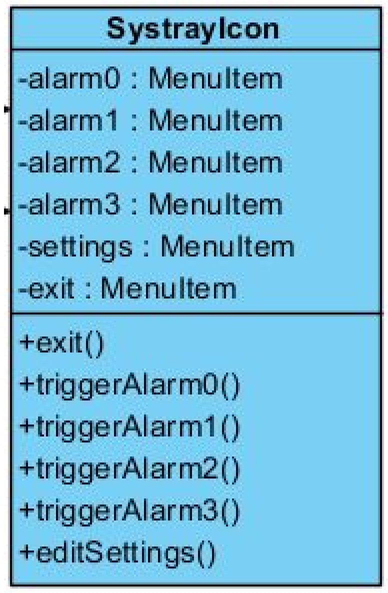
2.1.5. SettingsJFrame



The SettingsJFrame extends JFrame and is the visual window class for displaying and editing the settings.
The SettingsJFrame also has two buttons. The “test” button triggers testConnection() which sends the displayed settings to the server and checks the status of the response. The connectionStatus label updates to indicate the server response.

The “Save” button encrypts the displayed password and then writes all of the text in the various text fields to a configuration file.

2.1.6. SystrayIcon

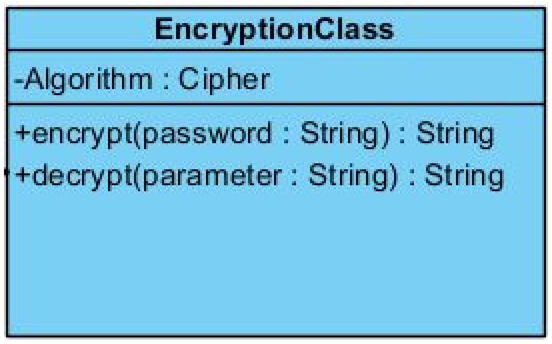


They SystrayIcon class puts a visual icon in the Windows Systray.

The menu items in this class include a link to trigger each of the 4 semi-permanent alarms, a link to open the Settings window, and a link to exit the program.

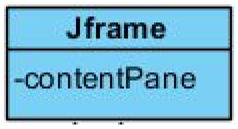
The operations of this class are actionListeners for their respective menu items.

2.1.7. **Encryption**


EncryptionClass
-Algorithm : Cipher
+encrypt(password : String) : String
+decrypt(parameter : String) : String

The Encryption class holds all of methods and algorithms to encrypt and decrypt the password. It uses the Javax crypto library. Each method takes a String as an input parameter and outputs a String.

2.1.8. **JFrame**



The JFrame class is a built-in class for creating GUI Windows. SettingsJFrame and AlertJFrame both extend the JFrame base class.

2.2. **Rest API**

The Rest API is a series of resources programed in the PHP language that allow authenticated users to interact with the database through GET requests. The syntax for GET requests and expected results are detailed in the Requirements Specification.

A detailed description of each component of the RestAPI follows:

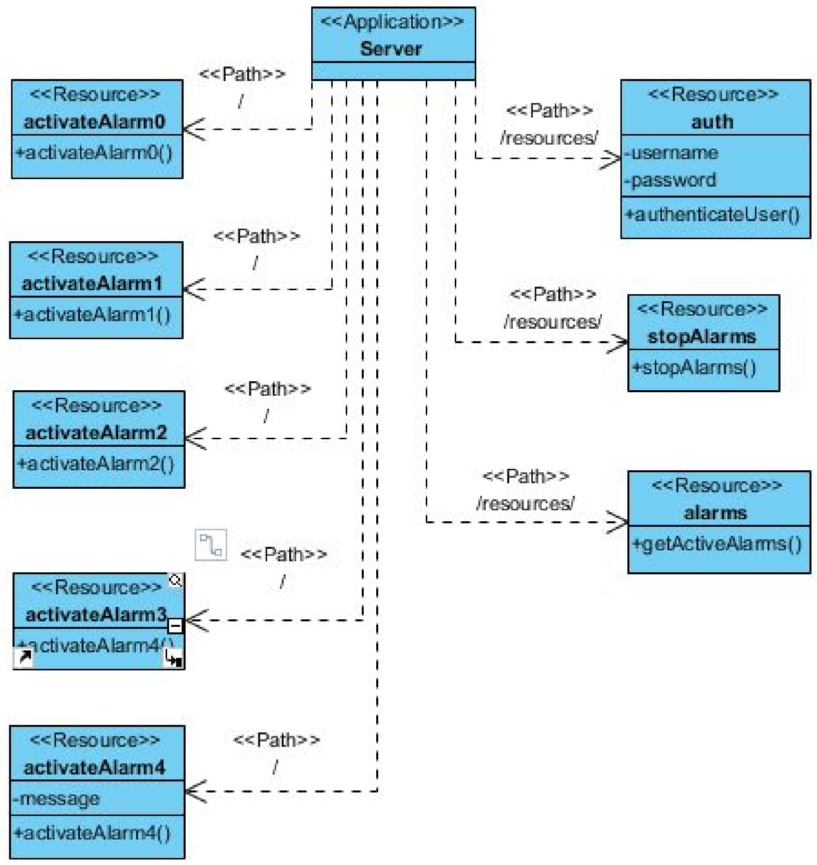


Figure 2.3 RestAPI

2.2.1. **Server**

The RestAPI runs on an Apache2 web server.

2.2.2. **Auth.php**

To access any of the server pages and many of the resources a user must be authenticated. The auth class is passed a username and an encrypted password in the header of the Get request. The auth class queries the database to determine if the password is correct. If the user is authenticating via a browser the auth class sets a Session variable indicating the user has successfully authenticated. If the user is authenticating from the Java client the auth class passes the credentials to requested resource. If the auth class is queried directly it returns a simple XML response indicating success or failure.

The path to the Auth resource is http://serverIP/resources/auth.php

2.2.3. **StopAlarms.php**

When StopAlarms.php receives a GET request it initiates the authentication process. If successful it sends an Update statement to the database to change the status of all alarms to inactive. It returns a success or failure notice to the requesting entity.

2.2.4. **Alarms.php**

When Alarms.php receives a GET request it initiates the authentication process. If successful it sends a query statement to the database to get the titles associated with each of the 4 semi-permanent alarms. It parses the results into an XML output document and returns that document to the requester.

2.2.5. **ActivateAlarm(0-4)**

When an activateAlarm resource receives a GET request, it initiates the authentication process. If successful, it sends an update statement to the database to change the status of the given alarm to active.

It then sends an Insert statement to the database to insert a record of the time, alarm triggered, computer and IP address of the requester, and the alarm's message at the time of triggering into the notification table to create a log of each alarm activity.

2.3. **Web GUI**

The WebGUI allows the user to access the functions of the server via a website using any standard browser. A detailed description of the pages and resources follows:

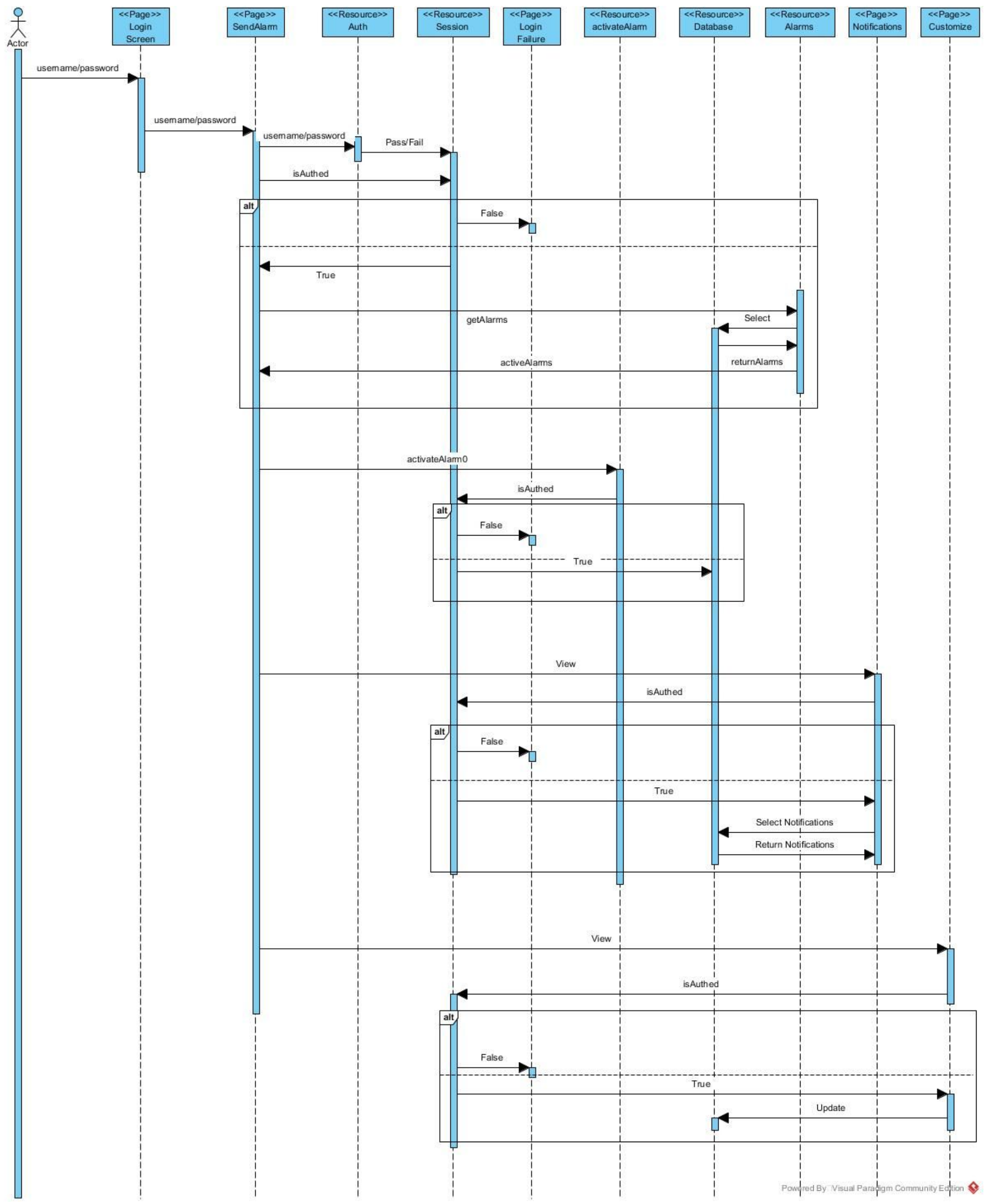


Fig 2.4 Sequence Diagram

This sequence diagram details how the pages interact with the resources available in the Web GUI

2.3.1. **Login Page**

The user first interacts with the login page where he types in his username and password.

2.3.2. **SendAlarm.php**

SendAlarm is the main page and the first page called by the login screen. If the user login process is successful, SendAlarm is displayed to the user. This page has 4 color-coded semi-permanent alarm buttons and 1 on-the-fly-message alarm button. An AJAX div calls the Rest resource named “alarms” to get a list of active alarms, and displays them at the top of the page.

2.3.3. Auth

Auth is a resource to authorize users. Auth determines whether a request came in as Rest request, or via a browser session by checking the headers of the request. If auth is accessed from the broswer, it compares the user’s encrypted credentials to those stored in the database and sets a session variable indicating whether the user has been authenticated.

All pages in the WebGUI check the Session variable to make sure a user is authenticated before displaying a page.

2.3.4. Session

Session is PHP’s built in Session array which stores information that is available to every page.

2.3.5. Login Failure

Login Failure is a page that displays telling the user their credentials did not work, or their Session has expired.

Any page that receives an Authenticated status of “False” will not display its contents and will redirect the user to the Login Failure page.

2.3.6. activateAlarmX

There are 5 activate alarm resources. Each one corresponds to a color coded alarm and each can be triggered by the Client or by the corresponding color-coded button on the SendAlarm page.

2.3.7. Database

Each page calls a resource which interacts with the database. The database design is detailed in the next section of this document.

2.3.8. Alarms

This is a Rest resource that returns a list of alarms in xml format. The SendAlarm page calls this resource, and then parses the result into an AJAX div that displays the active alarms.

2.3.9. Notifications

The notifications page queries the database for all alarm activity and displays it to the user in a table.

2.3.10. Customize

Customize is a page that allows the user to make changes to the server’s settings. It allows the user to change the alarm titles as well as add, edit, and delete users. The alarm title is also the message that is received by client machines when an alarm is triggered.

2.4. Database

The database stores the state of each alarm, the program data, and is updated by calling the resources of the Rest API

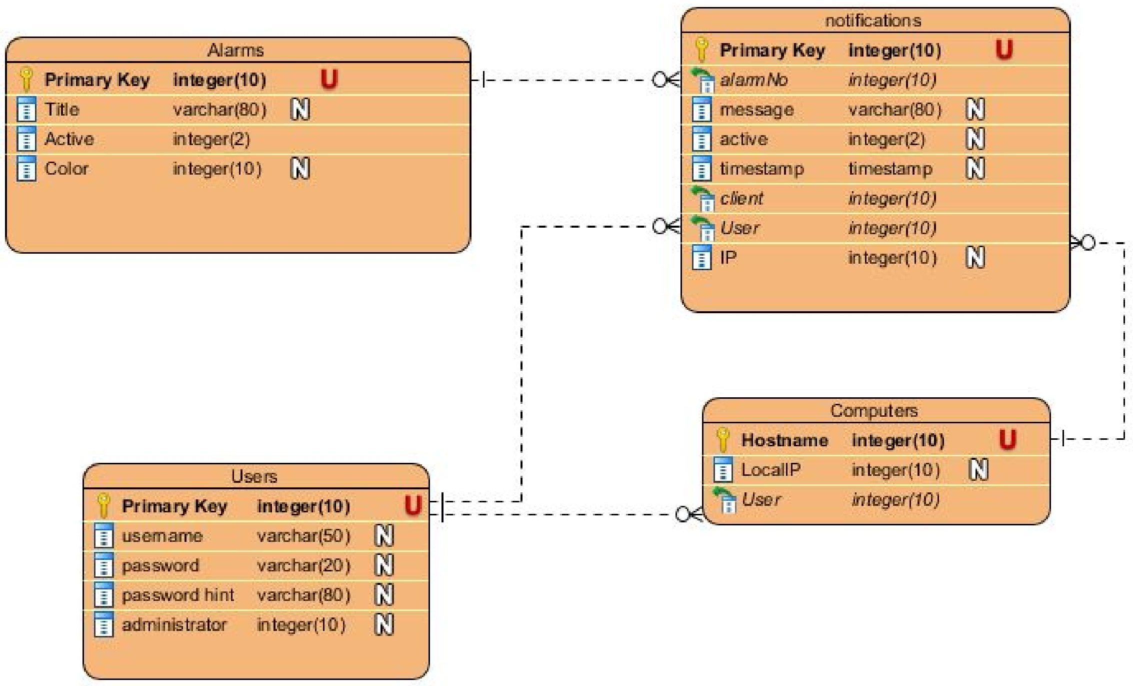


Fig 2.5 Object Relational Database Diagram

2.4.1. Alarms

The Alarms table stores the current state of each alarm. When an alarm is activated its Active field is set to 1. When it is deactivated it is set to 0. The resource alarms.php builds the list of active alarms by querying this table for all alarms whose Active field is 1.

The system can only support 5 alarms. Each alarm can be associated with many notifications.

The Titles field is updated by the Customization page of the WebGUI.

The Active field is updated by either the SendAlarm page of the WebGUI or the activateAlarmX resource of the RestAPI.

The Color field value is fixed.

The Primary Key of Alarms is an auto-incrementing integer. It is also the foreign key for the alarmNo field in Notifications.

2.4.2. Users

The Users table stores the username/password combinations that are used to authenticate to the system. Each computer has one user, but a single user can authenticate multiple computers.

A user can be associated with many notifications.

The Primary Key for the Users table is an auto-incrementing integer. It is the foreign key for the user field in notifications

2.4.3. Computers

Each computer has only one Hostname, Local IP address, and one user to authenticate.

A computer can appear in multiple notifications

The Primary Key for Computers is hostname. It is the foreign key for the client field in the notifications table.

2.4.4. Notifications

Each notification can have only one user, one computer, and one alarm.

The Primary Key for the notifications table is an auto-incrementing integer.

3. References

How to Model Relational Database Design with ERD. (2015, August 24). Retrieved from https://www.visual-paradigm.com/tutorials/how-to-model-relational-database-with-erd.jsp

Jef, J. (2011, July 22). An Example of Modeling Rest Web Services. Retrieved from https://firstinfinity.wordpress.com/modeling_rest_web_services/

Kuppa, K. (n.d.). Airline Reservation System [Scholarly project]. Retrieved from http://people.cs.ksu.edu/~kaavya/Vision Document_MSE_Phase I.pdf

Lastrapes, J. RCRA Enforcement and Compliance History (REACH) System (2018)