Monday, December 21, 2015

Abstract

Lab Request Application

SUMMARY

Introduction

The goal of this project is to create an application to allow engineers and technicians to collaborate efficiently by allowing engineers to communicate their testing needs to technicians quickly and easily, as well as allow technicians to manage tasks and deliver results in a timely manner.

The system will incorporate a website, accessible from a world wide web browser, than can be utilized by its users to add tasks that can then be viewed and managed by other team members.

Why A Web Site?

As stated above, the application will incorporate a web interface, accessible by team members through a web browser and an internet connection.

The advantages of having the system accessible through the web is that it can be accessed and manipulated by users anywhere where they have access to the internet.

This can increase their productivity by allowing employees to participate at anytime - whether they are present in their workplace, they are at home, or travelling abroad.

Thus, engineers and technicians can respond to urgent tasks where they would otherwise be limited from doing so.

Security

Because the Lab Request Application is intended to communicate information that may be confidential, security is a top priority in the design of the system.

Interception or modification of information being exchanged in the system, which is possible on unencrypted or public networks, or by a dedicated party attempting to penetrate it, can be circumvented by incorporating strong encryption practices, such as SSL (TLS), public-private encryption schemes and cryptographic signing practices to prevent unauthorized access to information.

Speed and Efficiency

A further goal of the Lab Request project is to create a method of communicating tasks and collaborations between members of the teams, including engineers and technicians, that is swift, to allow users to complete this step quickly, and allow them to focus on their own responsibilities.

In addition, for technicians, viewing and managing tasks will be optimized, so that they can prioritize lab work over administration of the lab request system.

USER INTERFACE

A user will initially be greeted with a login dialogue, where their credentials will be sent to the Node.js server and be authenticated with the OpenProject instance.

After authentication, the user will be presented with a dashboard, where they can view, create and manage tasks.

When users enter the dashboard, they will be able to choose between different tasks organized by their respective projects.

Tasks will also be arranged in the order of their due dates, with tasks with earlier due dates higher in the display.

In addition, users will have the ability of adding tasks that will be added to the list of tasks.

When viewing a task, users will be able to ‘tag’ members to notify them, or assign them the task to complete. They can also post comments, where they can attach uploaded files to discuss the task.

When creating a task, user will be able to ‘tag’ members to notify them, or assign them to the task. They can attach uploaded files to their submissions, and create a table of parameters they require for their test. They can also specify due dates, to communicate when they require the results of a test.

A task will be labelled with a status, to indicate its completion.

Users and technicians can follow tasks, to be notified when the task has been updated, and can opt into receiving email updates with notifications as well.
Users can also opt to receive notifications for tasks with a near due date.
The interface will also display the option to print the details of a task for reference.

TECHNICAL DETAILS

The application will follow an MVC-model (Model-View-Controller), where an Nginx or Apache web server will be used to provide a web frontend for users to interact with and a Node.js backend to manage API services, such as data management and email systems.
The system will be based on OpenProject, using its API (Application Programming Interface) to interface with it. An OpenProject instance hosted on the computing server will manage users, projects and task within the application.
The web interface will be composed of HTML documents, and will use JavaScript AJAX and WebSocket to request information from the Node.js API system.
The connection between the Nginx or Apache web server and Node.js API Service and the user will be secured through TLS, and connection will follow HTTPS protocol standards. TLS will be powered by an OpenSSL module active on the web server.
The website will be served with HTTP/2 connection to provide negligible loading times, and will utilize web technologies, such as AJAX and WebSockets to create a dynamic interface.
If the user’s browser is detected to lack support for the website, they will be redirected to a separate basic version of the website that is made of static HTML pages maintained by the Node.js instance, optimized for compatibility and speed.

CONCLUSION

The Lab Request Application will be a system for engineers and technicians to efficiently assign and complete testing task needed by the engineers.
The interface will incorporate a website for universal access.

The goals of the interface are:

- Security
  - Implementation of encryption
- Speed and Efficiency
  - Streamlined interface
  - Optimized technical stack

The user interface will be contain an organized system for the display of tasks posted by engineers, and the creation and administration of individual tasks.

The application will be powered by a OpenProject instance which will be integrated into a Node.js API service through OpenProject's API.

The frontend of the system will be managed by a web server (Nginx or Apache), with all connections between the servers and client encrypted through TLS to prevent interception from unauthorized penetrators.