**Documento de Concepto de Operación**

Proyecto:

**Baliza de Integración Continua**

Materia:

**Diseño y Arquitectura de Sistemas de Computación**

Profesores:

**Joaquín Alejo Fernandez, Jorge Fossati.**

Alumnos:

**David Wasyluk, Franco Sgro**

**Introducción**

El proyecto "Baliza de Integración Continua" tiene como objetivo desarrollar un sistema que utilice un ESP32 para controlar un semáforo, el cual representará el estado de una integración continua alojada en un repositorio de GitHub. Se utilizará la API de GitHub para obtener el estado del CI, permitiendo una comunicación directa y segura entre el ESP32 y la plataforma de desarrollo.

Este documento proporciona una visión detallada del funcionamiento del sistema, sus componentes principales y los procedimientos necesarios para su configuración y operación efectiva.

**Contexto**

La práctica de Integración Continua requiere feedback oportuno al equipo de desarrollo. Este feedback toma normalmente la forma de una notificación por email o chat, y debería ser visible para todos los miembros del equipo.

**Concepto de Operación**

Este proyecto describe la necesidad y características esperadas para una Baliza de Integración Continua que haga visible en un espacio de trabajo el estado de Integración Continua de un producto específico configurado.

La baliza deberá mostrar una luz fija en color verde por su connotación positiva mientras el estado del build de Integración Continua monitoreado sea Exitoso, y alertar con un cambio de color y eventualmente otros comportamientos (ver detalle en la sección requerimientos) cuando el estado cambie a Fallido.

**Requerimientos**

Los atributos de calidad básicos de la baliza son el desempeño, dado que el aviso de cambio de estado generado por la baliza compite con otros mecanismos de notificación y debería ser el más visible dado que es un objeto físico. El otro atributo de calidad prioritario es la usabilidad, la selección de colores y otros comportamientos deben llamar la atención e indicar el estado pero sin molestar a los

usuarios ni consumir demasiado su atención, logrando un fino balance.

**Descripción del Sistema**

**Componentes Principales:**

* **ESP32**: Un microcontrolador WiFi y Bluetooth de bajo costo utilizado para controlar los LEDs del semáforo y establecer la comunicación con la red local y la API de GitHub.
* **LEDs** (Rojo, Amarillo, Verde): Representarán los estados de la integración continua. Cada LED indicará un estado específico.
* **Resistencias**: Se utilizarán para limitar la corriente a través de los LEDs y evitar daños.
* **Red Wi-Fi**: Se requerirá una red Wi-Fi para conectar el ESP32 a Internet y permitir la comunicación con la API de GitHub.

**Funcionamiento del Sistema:**

1. El ESP32 se conectará a la red Wi-Fi utilizando las credenciales proporcionadas.
2. El ESP32 enviará solicitudes GET a la API de GitHub para obtener información sobre el estado de la integración continua.
3. La API de GitHub responderá con un JSON que contiene el estado de la integración.
4. El ESP32 procesará la respuesta JSON para determinar el estado de la integración (éxito, fallo, etc.).
5. Según el estado obtenido, el ESP32 controlará los LEDs del semáforo para reflejar el estado de la integración.
6. El proceso se repetirá a intervalos regulares para mantener actualizado el estado del semáforo.

**Requerimientos del Sistema**

**Hardware:**

* ESP32
* LEDs (Rojo, Amarillo, Verde)
* Resistencias para los LEDs
* Fuente de alimentación para el ESP32 y LEDs
* Acceso a una red Wi-Fi

**Software**:

* IDE de Arduino con soporte para ESP32
* Bibliotecas de Arduino para Wi-Fi y HTTPClient
* Token de acceso personal de GitHub para la autenticación en la API

**Procedimiento de Configuración**

1. Conectar físicamente los LEDs al ESP32 y configurar las resistencias para cada LED.
2. Cargar el código en el ESP32 utilizando el IDE de Arduino, asegurándote de configurar correctamente las credenciales de red Wi-Fi y el token de acceso de GitHub.
3. Asegurarte de que el ESP32 esté conectado a la misma red que el dispositivo desde el cual deseas acceder al semáforo.
4. Iniciar el ESP32. Este se conectará a la red Wi-Fi y comenzará a enviar solicitudes a la API de GitHub.
5. La API de GitHub responderá con información sobre el estado de la integración.
6. El ESP32 procesará la respuesta y controlará los LEDs del semáforo según el estado obtenido.

**Pruebas**

Empleamos la biblioteca ArduinoUnit para llevar a cabo pruebas exhaustivas que permiten verificar múltiples aspectos cruciales de nuestro semáforo de integración continua. A través de este enfoque de pruebas, sometemos a rigurosos escrutinios la funcionalidad del dispositivo, abarcando desde la estabilidad de la conexión WiFi hasta la correcta interpretación y procesamiento del archivo JSON. Asimismo, evaluamos la capacidad de los LEDs para parpadear en los momentos y secuencias previstos.

Este enfoque meticuloso de pruebas nos ha proporcionado resultados coherentes con nuestras expectativas y nos otorga la confianza necesaria en el desempeño robusto y fiable de nuestro semáforo de integración continua en distintos escenarios y condiciones operativas. Cada aspecto probado, desde la conectividad WiFi hasta el rendimiento de los indicadores visuales, ha sido validado mediante esta metodología rigurosa, brindando una garantía adicional de la calidad y fiabilidad del dispositivo en el contexto de su implementación práctica.

**Estrategia de Polling**

El sistema implementa la estrategia de polling para obtener el estado de la integración continua desde la API de GitHub. Esto implica que el ESP32 envía solicitudes periódicas a la API para verificar si ha habido cambios en el estado. Esta estrategia se elige por su simplicidad y eficiencia en este contexto, ya que el sistema no requiere una comunicación constante y puede funcionar de manera autónoma.

**Lenguaje de Programación C y Entorno de Desarrollo**

El código del sistema está desarrollado en el lenguaje de programación C, compatible con el IDE de Arduino. Esta elección se basa en la amplia compatibilidad y soporte ofrecido por el entorno de desarrollo, lo que facilita el proceso de programación y depuración, además de aprovechar la vasta comunidad y recursos disponibles.

**Uso del Semáforo como Periférico**

El semáforo, una vez configurado y conectado, puede ser considerado como un periférico de visualización. Simplemente se conecta a través de un puerto USB y se coloca en el escritorio o en un lugar visible para que los miembros del equipo puedan monitorear el estado de la integración continua de manera intuitiva y rápida.

Esta configuración permite una integración efectiva con el entorno de trabajo, proporcionando un indicador visual claro y directo del estado de la integración continua, lo que facilita la toma de decisiones y la colaboración del equipo de desarrollo.

**Justificación del Diseño Físico**

El diseño físico del semáforo de integración continua se ha concebido cuidadosamente para optimizar su funcionalidad, durabilidad y estética. La elección de una caja de plástico negra, manufacturada mediante impresión 3D, se basa en diversas consideraciones fundamentales:

* **Robustez y Durabilidad:**

El plástico utilizado en la impresión 3D es conocido por su alta resistencia y durabilidad, lo que asegura que el semáforo pueda soportar condiciones adversas y su uso prolongado en entornos de desarrollo y laboratorios.

* **Peso Ligero y Portabilidad:**

La utilización de plástico impreso en 3D permite obtener un diseño ligero que facilita su transporte y manipulación, brindando flexibilidad para su ubicación en distintos espacios de trabajo.

* **Personalización y Adaptabilidad:**

La impresión 3D brinda la oportunidad de crear diseños a medida, permitiendo la incorporación de características específicas para adaptarse a las necesidades particulares del entorno en el cual será utilizado.

* **Estética Profesional y Discreta:**

El color negro seleccionado confiere un aspecto elegante y discreto, lo que permite una integración armoniosa en el entorno de trabajo. Esto evita distracciones innecesarias y se alinea con estándares de presentación profesional.

* **Tecnología de Fabricación Avanzada:**

La utilización de impresión 3D representa una técnica de fabricación avanzada que brinda versatilidad en la creación de formas y estructuras complejas, permitiendo una construcción precisa y detallada del semáforo.

* **Sostenibilidad y Eficiencia en la Producción:**

La impresión 3D reduce el desperdicio de material y energía en comparación con métodos de fabricación tradicionales, al tiempo que posibilita una producción más eficiente y ecoamigable.

**Consideraciones de Seguridad**

El token de acceso personal de GitHub debe mantenerse de forma segura y no debe compartirse públicamente.

Se recomienda utilizar una conexión segura (HTTPS) al comunicarse con la API de GitHub.

**Mantenimiento y Soporte**

Se recomienda realizar pruebas periódicas para asegurarse de que el sistema siga funcionando correctamente.

Si se realizan cambios en la red Wi-Fi o en el repositorio de GitHub, es posible que sea necesario actualizar la configuración en el código del ESP32.

**Checklist**

| **#** | **Título** | **Descripción** | **Razones** | **Atributo de calidad** |  |
| --- | --- | --- | --- | --- | --- |
| **1** | **Actualizar**  **estado de la**  **baliza** | **El estado de la baliza deberá**  **actualizarse ante cada cambio de**  **estado del build.** | **Si no muestra el estado**  **correcto será desestimada**  **por los usuarios** | **Desempeño/**  **Corrección** | ***⇃*** |
| **2** | **Color asociado**  **a estado** | **La baliza deberá mostra luz de**  **color de acuerdo al estado del**  **build:**  **Exitoso => Verde**  **Fallido => Rojo** | **Debe sugerir fácilmente el**  **significado maximizando la**  **correspondencia natural** | **Usabilidad** | ***⇃*** |
| **3** | **Aviso de**  **cambio de**  **estado** | **La baliza deberá destacar el**  **cambio de estado (Exitoso a**  **Fallido o Fallido a Exitoso) con un**  **alarma sonora corta y/o**  **parpadeo de la luz.** | **Debe captar la atención de**  **los usuarios para que se**  **perciba el cambio en forma**  **inmediata sin molestar** | **Usabilidad** | ***⇃*** |
| **4** | **Conexión a**  **través de WiFi** | **La baliza deberá conectarse al**  **servidor del sistema de**  **Integración Continua mediante**  **WiFi** | **Permite emplazarla**  **libremente y usa tecnología**  **muy difundida** | **Usabilidad** | ***⇃*** |
| **5** | **Aviso de**  **desconexión** | **La baliza deberá destacar su**  **desconexión con un color distinto**  **a los demás, por ejemplo,**  **Naranja** | **Si la baliza no puede**  **conectarse al servidor, no**  **puede mantenerse**  **actualizada, ver #1** | **Confiabilidad**  **Usabilidad** | ***⇃*** |
| **6** | **Configuración**  **de WiFi** | **La baliza deberá permitir**  **configurar la red y credenciales**  **para conectarse a WiFi** | **Permite que se use en**  **múltiples ubicaciones y que**  **se actualice la configuración**  **de WiFi** | **Flexibilidad** | **X** |
| **7** | **Configuración**  **de build** | **La baliza deberá permitir**  **configurar la url del build**  **monitoreado y las credenciales**  **de acceso al sistema de**  **Integración Continua.** | **Permite que se use para**  **distintos proyectos y que se**  **actualice la configuración**  **de acceso al sistema** | **Flexibilidad** | ***⇃*** |
| **8** | **Agregar**  **Herramienta**  **de Integración**  **Continua** | **La baliza deberá permitir agregar**  **conectores para nuevas**  **herramientas de Integración**  **Continua.** |  | **Extensibilidad** | **X** |

**Conclusiones**

Este documento proporciona una descripción detallada del sistema, los componentes involucrados y los pasos para su configuración y operación. Con este sistema, se logrará controlar un semáforo que representará el estado de la integración continua de un repositorio de GitHub de manera eficiente y segura.