

Taller de Proyecto II

2021

Informe Final

ESP32 OTA WEB

Grupo de Desarrollo

- Kenneth Martin Lindner - 00785/1
- Markos Alonso Moscoso Ocampo - 01594/0

13/12/21

- Índice

1.- Proyecto	3
2.- Funcionalidad del Proyecto	4
2.1- Interacción con usuario	4
2.2- Interacción con hardware	6
2.3- Interacción con software externo	7
3.- Esquema Gráfico del Proyecto	8
3.1- Diagrama de Flujo	9
4.- Documentación del Software del Proyecto	10
4.1- platfomio.ini	10
4.2- pre_extra_script.py	10
4.3- OTAWEB.cpp	11
4.4- app.py	12
5.- Documentación en Formato Gráfico y Video	13
5.1- Imágenes	13
5.2 - Video	15
5.3 - Bitácora	15

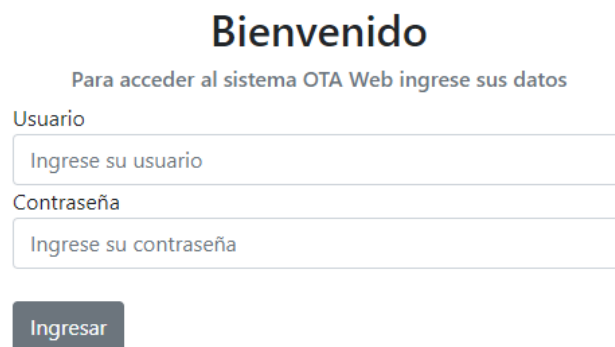
1.- Proyecto

El proyecto tiene como objetivo implementar la programación **OTA Web** en el microcontrolador ESP32 para poder realizar la consulta de la versión y actualización del firmware del dispositivo sin la necesidad de tener una conexión serie. Para acceder a las funcionalidades descritas anteriormente, se desarrolla un sistema web con interfaz y autenticación que registra en una base de datos la información detallada del usuario logueado en el sistema, el archivo, su versión y la fecha y hora en que fue realizada la actualización.

2.- Funcionalidad del Proyecto

2.1- Interacción con usuario

Para poder acceder a las funcionalidades implementadas, el usuario debe ingresar a la URL donde está hospedada la web (localhost:5000 o 192.168.4.2:5000 en nuestro caso particular) y loguearse en el sistema ingresando su nombre de usuario y contraseña en los respectivos campos como se indica en la imagen a continuación.



Bienvenido

Para acceder al sistema OTA Web ingrese sus datos

Usuario

Contraseña

Ingresar

Imagen 2.1.A: Interfaz de login

Si es la primera vez que ingresa al sistema y no posee usuario debe registrarse, para esto el usuario debe ingresar al endpoint '/add-new-user' (El cual no es accesible desde la interfaz por cuestiones de seguridad) y completar los campos como sea requerido, la misma se puede observar en la siguiente imagen.



Registrar Usuario

Usuario

Contraseña

Repita Contraseña

Registrar

Imagen 2.1.B: Interfaz de registro

Una vez el usuario ingrese al sistema se encontrará con dos botones disponibles, uno para actualizar el firmware del dispositivo y otro para consultar la versión actual que está corriendo en el microcontrolador (**Imagen 2.1.C**). También tendrá acceso a la pestaña de datos donde podrá visualizar las actualizaciones y la información detallada del usuario, version y fecha en que fue realizada (**Imagen 2.1.D**).

ESP32 OTA WEB

Actualizar

Version

OTA Web App

Imagen 2.1.C: Menú principal

Datos

Fecha	Usuario	Archivo	Version
Dec 05 2021 15:13:10	Felipe	test.bin	v1.0.0
Dec 05 2021 15:15:34	Julian	ota.bin	v1.0.1
Dec 05 2021 19:00:15	Kenneth	ota-1.0.3.bin	v1.0.3
Dec 05 2021 19:06:29	Kenneth	ota-1.0.4.bin	v1.0.4

Imagen 2.1.D: Tabla registro de información

Si el usuario le da click al botón de actualizar se redirige a otra pantalla que le solicita la carga del archivo, el cual sólo puede ser de extensión bin, y si luego le da click en actualizar se dispara la acción `/update` y se actualiza el firmware del dispositivo (**Imagen 2.1.E**).

Actualizar Firmware

Seleccionar archivo

Ningún archivo seleccionado

Actualizar

Imagen 2.1.E: Actualización de firmware

Si por el contrario el usuario le da click al botón de versión, se ejecuta el endpoint /version del dispositivo el cual devuelve la versión actual y se muestra en una nueva pantalla, esto se puede observar en la siguiente imagen.



Imagen 2.1.F: Consulta de versión

Para concluir con esta sección, el usuario dispone de un botón en la barra de navegación para cerrar sesión, de esta forma si se desea volver a acceder al sistema deberá volver a loguearse. Cabe destacar que los endpoint del sistema web cuentan con autenticación y autorización, por lo que si el usuario no se logea no podrá acceder a estas funciones.

2.2- Interacción con hardware

La interacción con el hardware se realiza mediante peticiones realizadas por el servidor web y son atendidas por el servidor local levantado en el Hardware, este servidor consta de dos endpoints (/update y /version) de tipo HTTP GET, el endpoint /version se encarga de devolver la versión actual del archivo .cpp y el endpoint /update realizar la búsqueda y actualización del firmware por OTA.

El endpoint /update está declarado como HTTP GET ya que para actualizar el firmware se realiza mediante una búsqueda del archivo al webserver, y es el webserver el que le indica al ESP en qué momento realizar la búsqueda del archivo que se hace por medio de la librería HTTPUpdate.

2.3- Interacción con software externo

Para llevar registro de la información sobre las actualizaciones del dispositivo y los distintos usuarios que se van generando se utiliza MongoDB, una base de datos no relacional. Su configuración se encuentra dentro de la aplicación del servidor web, donde se define su string de conexión y sus colecciones. Para utilizar el sistema se debe tener instalado este motor de bases de datos y estar ejecutando el respectivo servicio. Respecto a las colecciones, no es necesario que el usuario las genere manualmente, la aplicación las va generando automáticamente a medida que es necesario.

Para dar a conocer un poco su estructura, se trata de una base de datos de nombre *otadata* que posee cuatro colecciones: *ota_users* (Contiene la información de los usuarios registrados para el login del sistema), *ota_transactions* (Contiene el detalle de las actualizaciones, archivo, version, usuario, fecha y hora), *fs.files* y *fs.chunks* (Son dos colecciones que se generan al guardar el archivo binario en la base de datos, *files* posee información del id relacionado de *transactions* y *chunks* almacena el archivo en cuestión).

Otra herramienta importante a mencionar es el entorno virtual de python utilizado para instalar todas las dependencias necesarias para levantar la aplicación. Si el usuario desea ejecutar el sistema web OTA en otra computadora, es necesario que instale Python 3.9.5 o superior. Además debe configurar un entorno virtual e instalar las dependencias via *pip install*, las mismas se detallan a continuación:

- Flask, redirect, render_template, request, Session, url_for. (*flask*)
- PyMongo. (*flask_pymongo*)

El resto de librerías utilizadas son heredadas del kit de desarrollo Python, por lo cual no son necesarias instalar otras dependencias externas.

3.- Esquema Gráfico del Proyecto

A continuación se presenta el esquema general del proyecto con sus componentes.

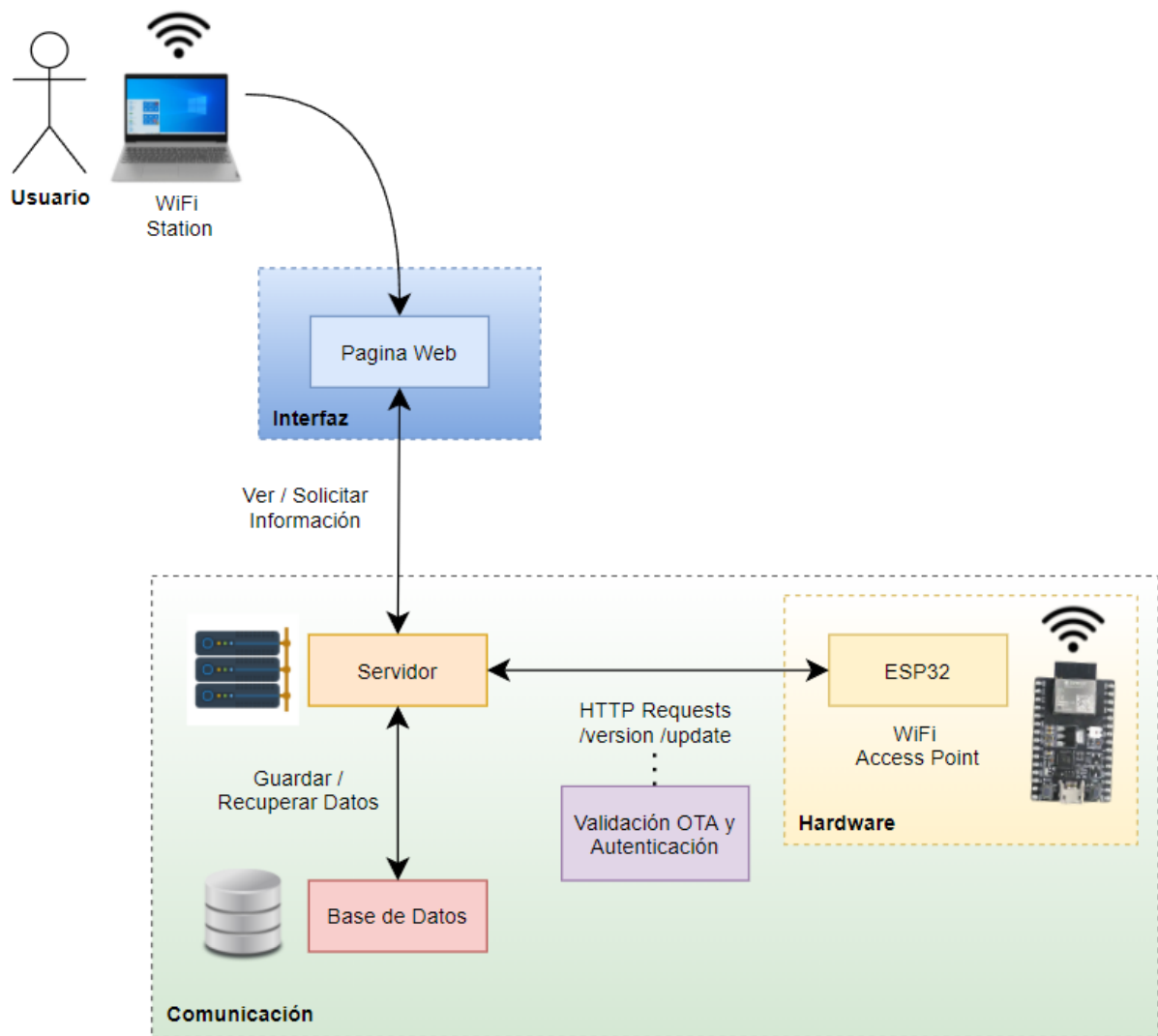


Imagen 3.A: Esquema general del proyecto.

Para comprender mejor el recorrido y opciones del sistema se presenta en la **Imagen 3.B** un diagrama de flujo del sistema con sus funciones.

3.1- Diagrama de Flujo

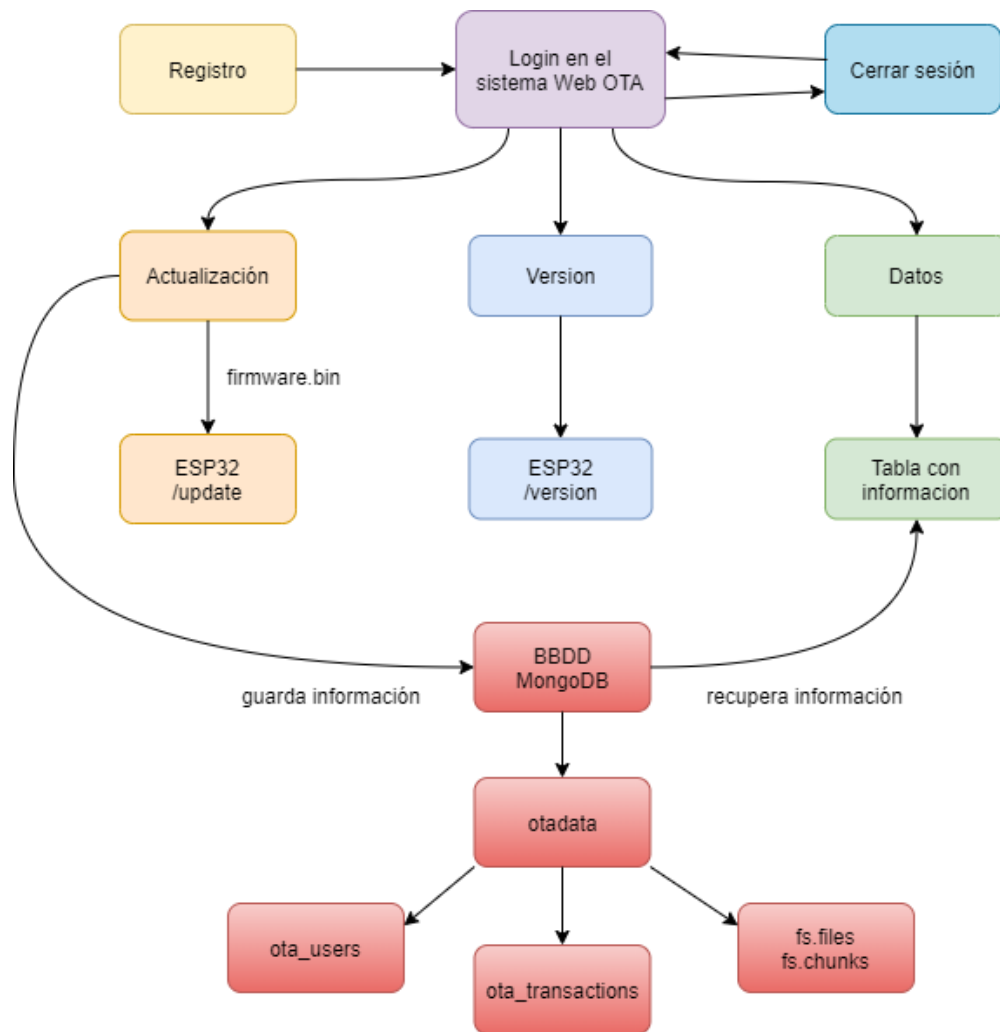


Imagen 3.C: Diagrama de flujo.

En la **Tabla 1** a continuación se detalla el presupuesto requerido del ESP32.

Componente	Precio
ESP32	\$1195 (Mercado Libre)

Tabla 1: Presupuesto de componentes de hardware

4.- Documentación del Software del Proyecto

En cuanto a los archivos y ficheros del proyecto, este tiene dos carpetas: Por un lado **Platformio_code/OTAWEB_AP** contiene todo lo relacionado al proyecto en platformio entre los que se incluyen un archivo con extensión *.ini* para la configuración del proceso de build del proyecto en platformio, una carpeta llamada *src* con el archivo *.cpp* con el código del ESP, un archivo *pre_extra_script.py* con el script para la verificación del código OTA y tres carpetas llamadas *.vscode*, *include* y *lib* que fueron creadas con el proyecto de platformio y contienen el funcionamiento de cada una en archivos *README* con la documentación necesaria del funcionamiento de cada carpeta; Por otro lado está la carpeta **Web** todo lo relacionado al servidor e interfaz web, esta carpeta contiene un archivo llamado *app.py* el cual se encarga de levantar el server web python y del ruteo y definición de los endpoints, una carpeta *templates* con los html/vistas de la interfaz y una carpeta *static* que contiene todos aquellos archivos de configuración y diseño como Bootstrap y jQuery.

4.1- platfomio.ini

Este archivo contiene las configuraciones del enviroment del proyecto de platformio con el que se realiza el proceso de build (compilado) del proyecto establecido como esp32dev, en esta se especificaron la plataforma con la cual se ejecuta el proyecto, el framework, board y un script que se ejecutan antes del proceso de build (definido con pre:).

4.2- pre_extra_script.py

Este archivo contiene un script hecho en python que se encarga de la verificación del código que contiene las funciones OTA, que se encargan de la actualización del firmware, para realizar la verificación, el script que se ejecuta antes del proceso de build buscará la función indicada en el archivo *.cpp* que se encargaría de llamar a las funciones correspondientes al proceso de actualización de firmware, para realizar la búsqueda de la función se abre el archivo *.cpp* como archivo de lectura y analiza línea por línea la coincidencia con el string de invocación a la función especificada, el script se ejecutará antes de realizar el build y en caso de no encontrar alguna coincidencia con la instanciación de la función especificada, generará una excepción y el proceso de build no llegará a generar el archivo binario necesario para la actualización del firmware.

4.3- OTAWEB.cpp

Este archivo contiene el código que será cargado al ESP, contiene las siguientes librerías importadas:

- **WiFi.h**
- **WebServer.h**
- **HTTPClient.h**
- **HTTPUpdate.h**

Se declaran las variables necesarias para la conexión a la versión access point del ESP32 y una variable donde se inicializa la versión del archivo, un objeto de tipo WebServer para levantar un servidor local en el ESP, entre las funciones declaradas en el archivo se encuentran:

- **UpdateFile**

Esta función se encarga de la actualización del archivo para esto se declara una variable de tipo WiFiClient y se setea la librería HTTPUpdate para que realice las redirecciones pasados por los headers al momento de ir a buscar el archivo al endpoint del webserver y luego se indica a la librería HTTPUpdate que no reinicie el microcontrolador al realizar la actualización del firmware, luego se realiza la búsqueda y actualización del firmware por medio de la función update, El funcionamiento de la función update de la librería HTTPUpdate se encarga de realizar la actualización del firmware mediante OTA y es el siguiente, se intenta buscar el archivo estableciendo una conexión el web server mediante la librería HTTPClient, en caso de no poder realizar la conexión devuelve un error de tipo HTTP_UPDATE_FAILED, en caso contrario devuelve un mensaje de ejecutar la función handleUpdate, en esta función se añaden los headers necesarios para la búsqueda http a través de una variable de tipo HTTPClient, luego por medio de esta se realiza la búsqueda del código al servidor, si se logró encontrar correctamente el archivo de la dirección indicada se realiza la actualización del firmware por medio de la función runUpdate, en caso contrario se instancia el error correspondiente. La función runUpdate ejecuta las funciones encargadas de realizar OTA de la librería Update.h que son Update.begin para la inicialización de la carga del archivo al ESP, Update.writeStream para escribir el archivo en el ESP y por último Update.end para finalizar la carga del archivo en el ESP.

- **SetupServer**

Esta función se encarga de levantar los endpoints del servidor local del ESP, estos se utilizan para responder a las peticiones del servidor web.

- **setup**

La función setup se ejecuta antes de ejecutar el loop del programa y se encarga de inicializar el modo access point y llamar a la función SetupServer para levantar el servidor local

4.4- app.py

Como indicamos al principio de esta sección, app.py se encarga de levantar y configurar la funcionalidad del servidor web, lo cual involucra endpoints, rutas y conexión con BBDD. El servidor web está implementado en Python con el framework Flask y su integración con MongoDB gracias a PyMongo. A continuación se detallan los endpoints más importantes del sistema web:

- **/login:** Vista de sistema de login, si no coinciden las credenciales con las almacenadas en la base de datos no podrá iniciar sesión. El resto de rutas están protegidas y no son accesibles si el usuario no se loguea anteriormente.
- **/home:** Vista del menú principal, si el usuario está logueado podrá acceder a las funcionalidades del sistema.
- **/upload-file:** Vista de la función para seleccionar un archivo y luego ejecutar la actualización. Si el archivo no posee extensión .bin no dejará realizar la actualización.
- **/update:** Toma el archivo cargado en el endpoint anterior y realiza tres acciones importantes, primero guarda una copia en *static/uploads/* para que luego el ESP lo recupere y realice la actualización del firmware. En segundo lugar realiza un request de tipo GET para ejecutar el /update del ESP32. Por último recopila la información del usuario logueado y del archivo .bin y lo almacena en la base de datos.
- **/show-version:** Realiza un request de tipo GET al ESP32 el cual le devuelve la versión del firmware que se está ejecutando y la muestra en una nueva vista.
- **/show-data:** Primero recupera los datos de la BBDD y los muestra vista con tabla.
- **/add-new-user:** Vista para agregar un nuevo usuario a la base de datos.
- **/register:** Verifica que el usuario no exista y que las contraseñas ingresadas coincidan, acto seguido da el alta en la base de datos.
- **/logout:** Cierra la sesión del usuario logueado.

El código fuente del proyecto se encuentra en este [REPOSITORIO GIT](#). En el mismo también encontrarán las diversas *branches* a las que se hace referencia en la bitácora.

5.- Documentación en Formato Gráfico y Video

5.1- Imágenes



5.1.A: Módulo ESP32

Bienvenido

Para acceder al sistema OTA Web ingrese sus datos

Usuario

Contraseña

OTA Web App

5.1.B: Vista de login.

ESP32 OTA WEB

Actualizar

Version

OTA Web App

5.1.C: Vista del menú principal.

Actualizar Firmware

Seleccionar archivo

Ningún archivo seleccionado

Actualizar

OTA Web App

5.1.D: Vista de función actualizar.

Version

La version actual del dispositivo es: v1.0.1

OTA Web App

5.1.E: Vista de consulta versión.

Datos

Fecha	Usuario	Archivo	Version
Dec 05 2021 15:13:10	Felipe	test.bin	v1.0.0
Dec 05 2021 15:15:34	Julian	ota.bin	v1.0.1
Dec 05 2021 19:00:15	Kenneth	ota-1.0.3.bin	v1.0.3
Dec 05 2021 19:06:29	Kenneth	ota-1.0.4.bin	v1.0.4

OTA Web App

5.1.F: Vista de datos registrados.

5.2 - Video

A continuación se presenta un video donde se puede observar el ingreso al sistema web y sus diversas funcionalidades, incluyendo la actualización OTA del firmware del ESP32 y la visualización de los datos en la tabla. El [VÍDEO](#) se encuentra compartido en YouTube.

5.3 - Bitácora

Una herramienta de registro utilizada durante el desarrollo del proyecto es la bitácora. Aquí se detallan los desarrollos, avances, errores, bloqueos y alternativas que fueron teniendo lugar a lo largo de estos meses. La misma puede encontrarse en el siguiente [ARCHIVO PDF](#).