OTA PROJECT REPORT

Problem Statement: Create an OTA solution to update sketch on a microcontroller (ESP8266) which is situated remotely and connected to a different network.

Solution and Approach:

Approach 1: Upload code using basic OTA

With this approach, it is very simple to upload the code without any physical connection between the computer and the microcontroller. But the main issue with this approach is that it is limited to local area network and the computer through which the code was first uploaded, since in connects the microcontroller to the laptop through a UDP set on the local IP of the ESP8266 microcontroller.

Approach 2: Upload code via web server

With this approach, the problem of inability of local IP to be accessible by a remote network can be solved by a roundabout method, so this method has two main steps.

Step 1: Get the local IP of ESP8266 and send code using AsyncElegantOTA library.

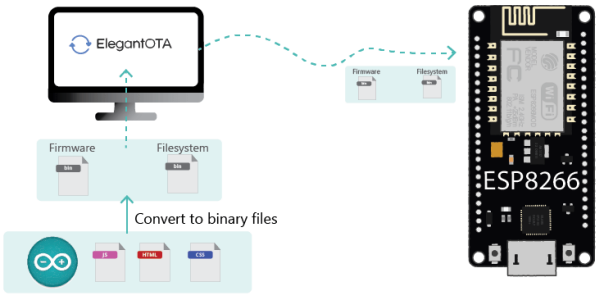
Step 2: Use a third party to host the local IP one web browser through a tunnel.

Implementation Of second approach:

Step 1: Enabling AsyncElegantOTA

OTA (Over-the-Air) update is the process of loading new firmware to the ESP8266 NodeMCU board using a Wi-Fi connection rather than serial communication. This functionality is extremely useful in case of no physical access to the ESP8266 board.

There are different ways to perform OTA updates. In this tutorial, we'll cover how to do that using the AsyncElegantOTA library. In our opinion, this is one of the best and easiest ways to perform OTA updates.
The AsyncElegantOTA library creates a web server that you can access on your local network to upload new firmware or files to the filesystem (LittleFS). The files you upload should be in *.bin* format. We'll show you later in the tutorial how to convert your files to *.bin* format.



Warning:
The only disadvantage of OTA programming is that you need to add the code for OTA in every sketch you upload so that you're able to use OTA in the future. In the case of the AsyncElegantOTA library, it consists of just three lines of code.

To add OTA capabilities to your projects using the AsyncElegantOTA library, follow these steps:

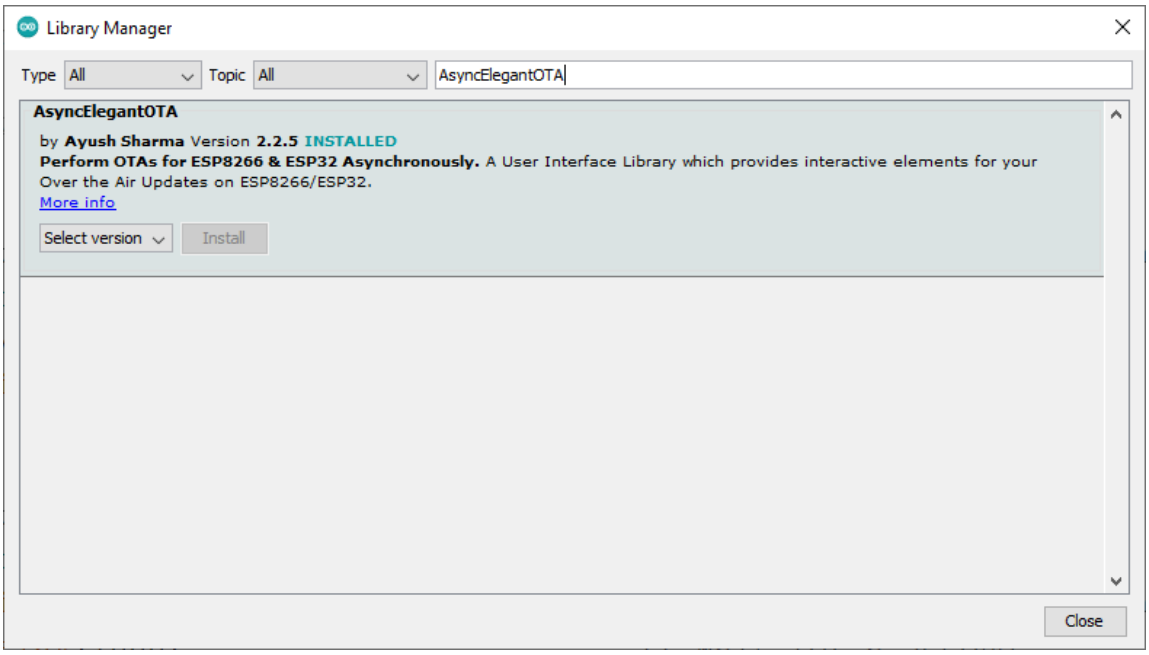
1. Install AsyncElegantOTA, ESPAsyncTCP and ESPAsyncWebServer libraries;
2. Include AsyncElegantOTA library at the top of the Arduino sketch: `#include <AsyncElegantOTA.h>;`
3. Add this line `AsyncElegantOTA.begin(&server);` before `server.begin();`
4. Open your browser and go to `http://<IPAddress>/update`, where `<IPAddress>` is your ESP8266 IP address.

How does OTA Web Updater Work?

- The first sketch should be uploaded via the serial port. This sketch should contain the code to create the OTA Web Updater so that you can upload code later using your browser.
- The OTA Web Updater sketch creates a web server you can access to upload a new sketch via a web browser.
- Then, you need to implement OTA routines in every sketch you upload so that you're able to do the next updates/uploads over-the-air.
- If you upload a code without an OTA routine, you'll no longer be able to access the web server and upload a new sketch over-the-air.

Install AsyncElegantOTA Library

You can install the AsyncElegantOTA library using the Arduino Library Manager. In your Arduino IDE, go to **Sketch** > **Include Library** > **Manage Libraries…** Search for "**AsyncElegantOTA**" and install it.



Install ESPAsyncWebServer and ESPAsyncTCP Libraries

You also need to install the ESPAsyncTCP and the ESPAsyncWebServer libraries. Click the links below to download the libraries.

- ESPAsyncWebServer
- ESPAsyncTCP

These libraries aren't available to install through the Arduino Library Manager, so you need to copy the library files to the Arduino Installation Libraries folder. Alternatively, in your Arduino IDE, you can go to **Sketch** > **Include Library** > **Add .zip Library** and select the libraries you've just downloaded.

AsyncElegantOTA ESP8266 Basic Example

Let's start with the basic example provided by the library. This example creates a simple web server with the ESP8266. The root URL displays some text, and the `/update` URL displays the interface to update firmware and filesystem.
Copy the following code to your Arduino IDE.

```
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <ESPAsyncTCP.h>
#include <ESPAsyncWebServer.h>
#include <AsyncElegantOTA.h>

const char* ssid = "REPLACE_WITH_YOUR_SSID";
const char* password = "REPLACE_WITH_YOUR_PASSWORD";

AsyncWebServer server(80);

void setup(void) {
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  Serial.println("");

  // Wait for connection
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
```

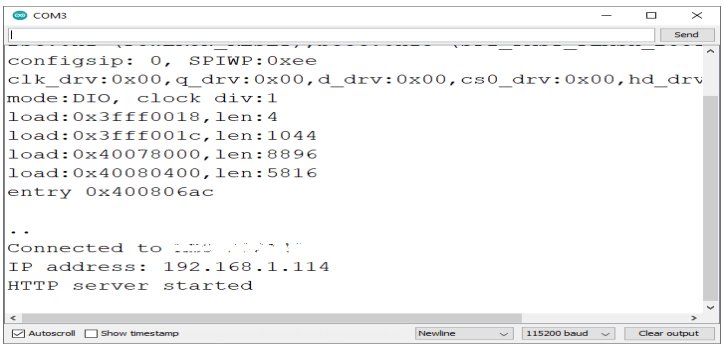
```
  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  server.on("/", HTTP_GET, [](AsyncWebServerRequest *request) {
    request->send(200, "text/plain", "ESP 8266 up and
running");
  });

  AsyncElegantOTA.begin(&server);    // Start ElegantOTA
  server.begin();
  Serial.println("HTTP server started");
}

void loop(void) {

}
```

Insert your network credentials and the code should work straight away:

```
const char* ssid = "REPLACE_WITH_YOUR_SSID";
const char* password = "REPLACE_WITH_YOUR_PASSWORD";
```

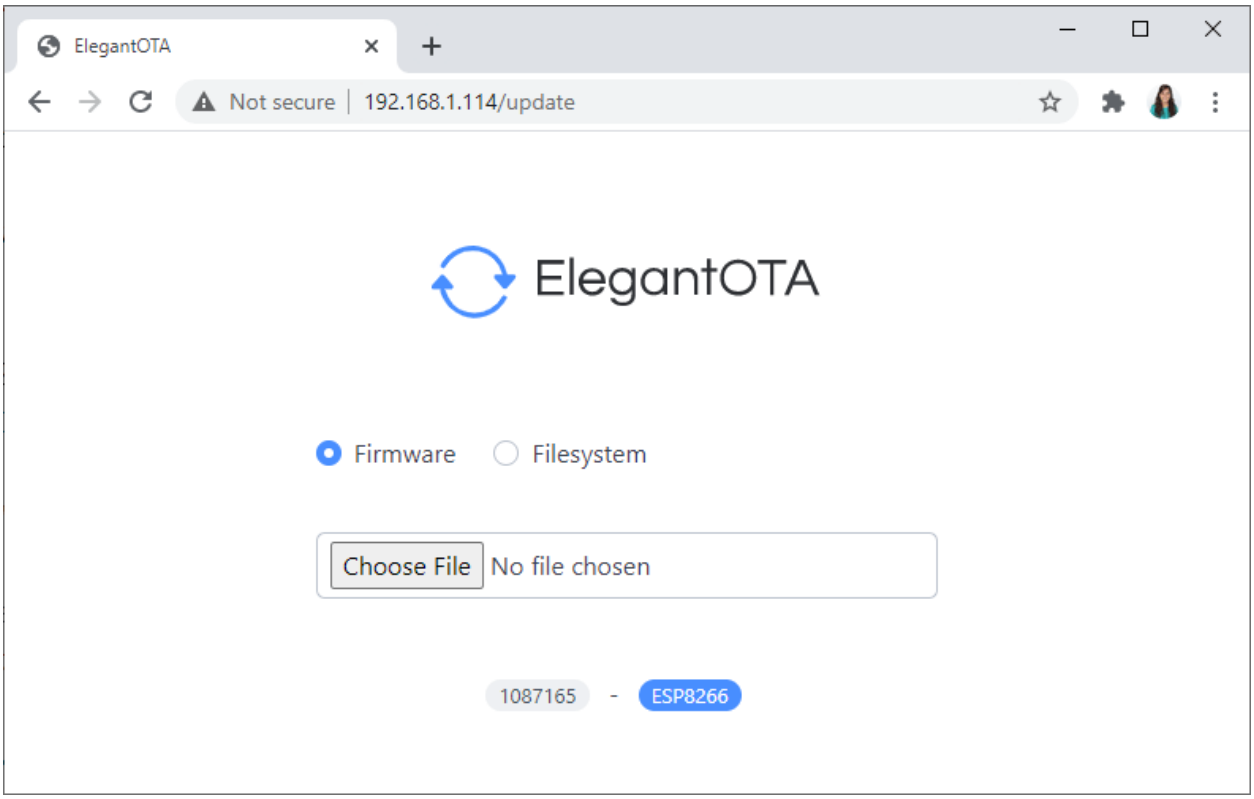
Access the Web Server

After uploading code to the board, open the Serial Monitor at a baud rate of 115200. Press the ESP8266 on-board RST button. It should display the ESP IP address as follows:



In your local network, open your browser and type the ESP8266 IP address. You should get access the root (/) web page with some text displayed.

Now, imagine that you want to modify your web server code. To do that via OTA, go to the ESP IP address followed by /update. The following web page should load.



Upload New Firmware OTA (Over-the-Air) Updates – ESP8266

Every file that you upload via OTA should be in *.bin* format. You can generate a *.bin* file from your sketch using the Arduino IDE.

With your sketch opened, you need to go to **Sketch** > **Export Compiled Binary**. A *.bin* file will be generated from your sketch. The generated file will be saved under your project folder.

That's that *.bin* file you should upload using the AsyncElegantOTA web page if you want to upload new firmware.

1. Don't forget to insert your network credentials.
2. Save your sketch: **File** > **Save** and give it a name. For example: *Web_Server_LED_OTA_ESP8266*.
3. Generate a *.bin* file from your sketch. Go to **Sketch** > **Export Compiled Binary**. A new *.bin* file should be created under the project folder.
4. Now you just need to upload that file using the ElegantOTA page. Go to your ESP IP address followed by /update. Make sure you have the **firmware** option selected. Click on **Choose File** and select the *.bin* file you've just generated.

5. When it's finished, click on the **Back** button.
6. Then, you can go to the root (/) URL to access the new web server. This is the page that you should see when you access the ESP IP address on the root (/) URL.

Step 2: Hosting the IP using Ngrok

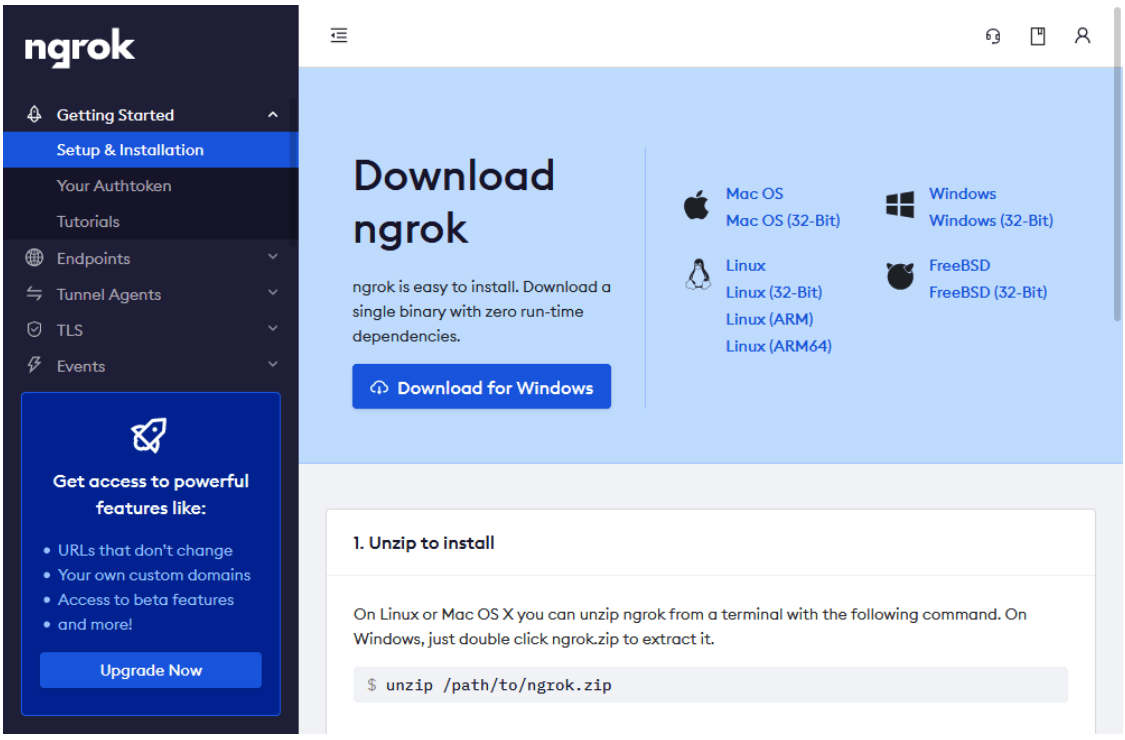
What is ngrok?

Your development machine may be connected to a secure network behind a firewall. To work around access restrictions, ngrok runs a small client process on your machine which creates a private connection tunnel to the cloud service. Your localhost development server is mapped to an ngrok.io sub-domain, which a remote user can then access. There's no need to expose ports, set up forwarding, or make other network changes.

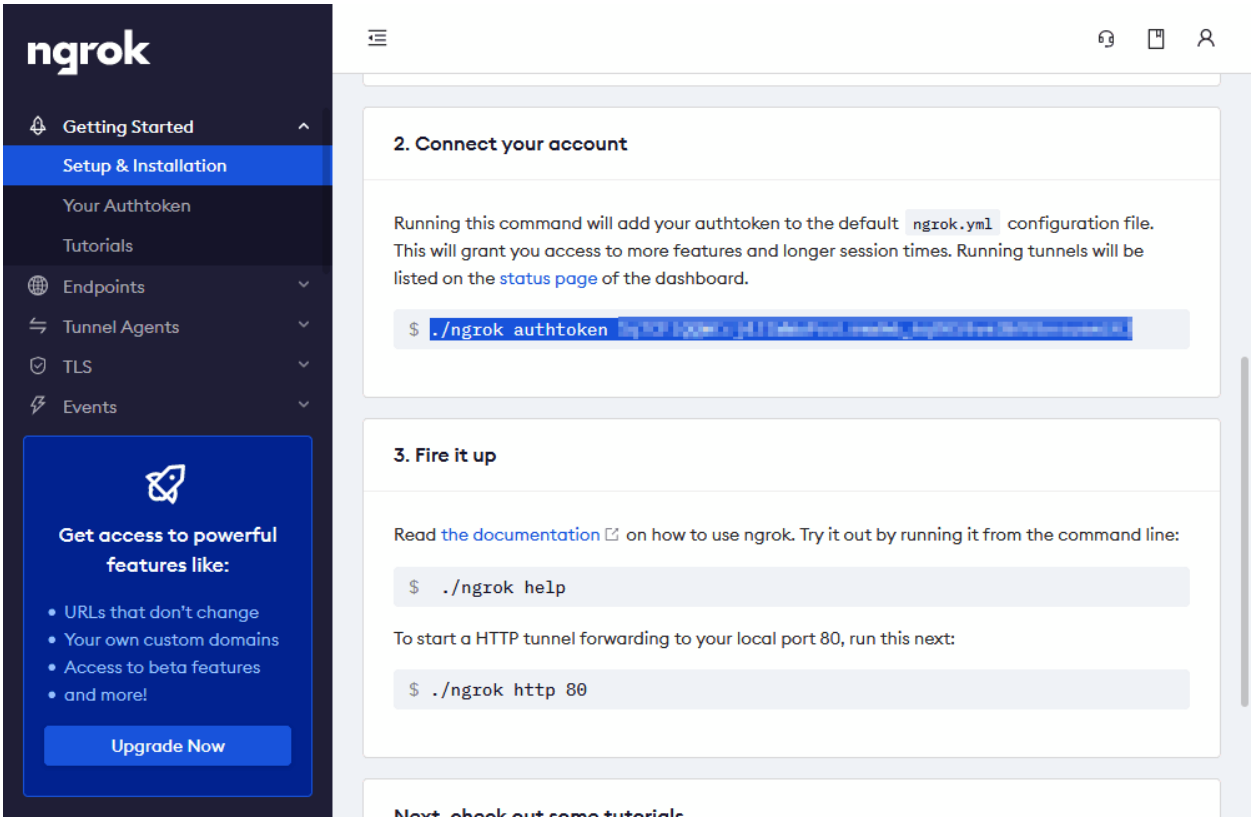
Get the ngrok Download

To start, open ngrok.com in your browser and click **Sign up** to register. A Google or GitHub account is easiest, but you can choose standard registration with an email address and password. An email verification link will be sent to you.

After login, you'll be directed to the ngrok dashboard where you can download the client for your operating system



Download and extract the file, following any specific instructions for your OS. It's then necessary to add your authentication token by running the command shown in the **Connect your account** section a little further down the page.



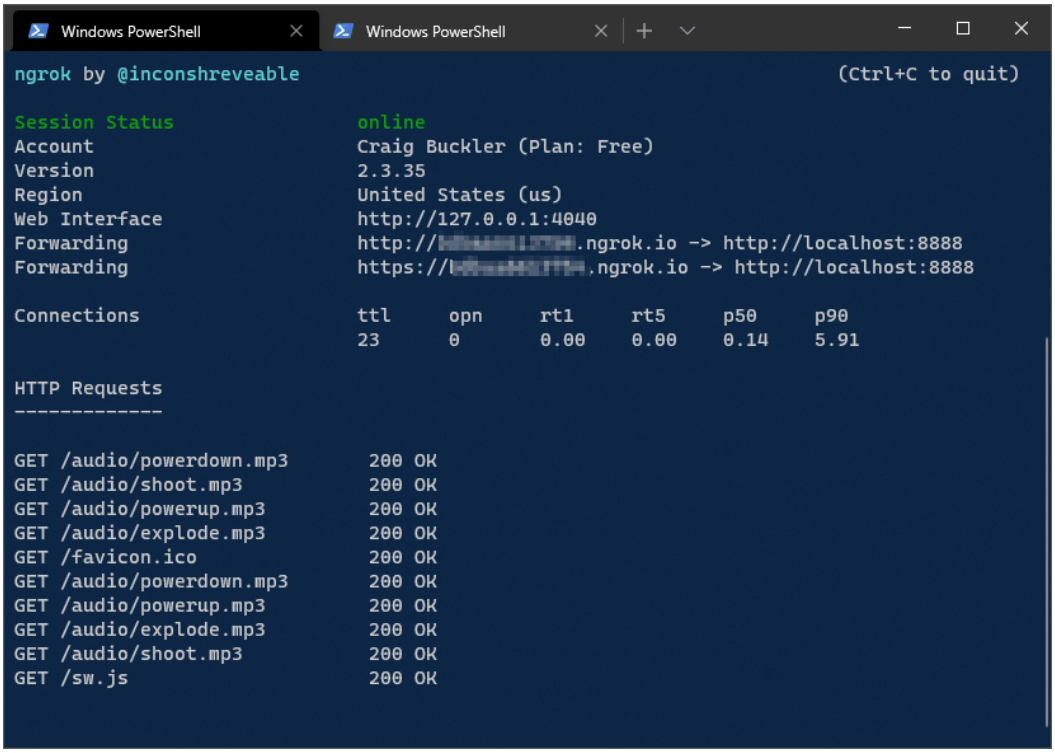
Now run this command in your terminal or the terminal provided by ngrok to authenticate.

```
./ngrok authtoken <token>
```

Run this command to start the server on a url. Enter the IP of microcontroller. Instead of 192.168.0.27

```
ngrok http 192.168.0.27:80
```

The terminal will clear and show the status with two **Forwarding** http and https addresses, such as http://123456789.ngrok.io/. You can pass either URL to another person so they can access your application from anywhere. The terminal shows a log of requests while ngrok is active.



The ngrok status panel panel at *dashboard.ngrok.com/endpoints/status* also shows a list of currently active URLs and client IP addresses. (You may need to refresh the browser to update it.)