* **Live Server** (the VS Code extension you’re using) is basically a **local development server** that makes web development easier. Here’s what it does:

**🔹 1. Runs a Local Web Server**

Normally, when you open an HTML file directly in the browser (file:///...), the browser just loads it as a static file.  
But Live Server starts a **local HTTP server** (like http://127.0.0.1:5500/) so your files are served as if they’re hosted on a real website.  
This helps because:

* It mimics how a real server works.
* Features like AJAX, APIs, fetch requests, and relative paths work properly.

**🔹 2. Auto Reload (Live Reload)**

Whenever you save your HTML, CSS, or JavaScript file:

* Live Server automatically refreshes the browser tab.
* No need to press Ctrl + R manually.

This makes development faster 🚀.

**🔹 3. Network Access**

Since it runs on your machine’s local IP, you can:

* Open the page on your laptop (http://localhost:5500/index.html).
* Also open it on another device (like your phone) using your machine’s **LAN IP** (e.g., http://192.168.0.142:5500/index.html).

**🔹 4. Simulates Deployment Environment**

Live Server helps you test your project **as if it’s deployed on the web**, but without needing to actually host it online.

👉 In short:  
**Live Server = Mini web server + Auto refresh tool** for faster, real-like web development.

Let’s go step by step into what happens **internally** when you click **“Go Live”** in VS Code with the Live Server extension:

**⚙️ Step-by-Step Process**

**1. Extension Starts a Local Server**

* When you click **Go Live**, the Live Server extension launches a **Node.js-based lightweight HTTP server**.
* This server listens on a **port** (default: 5500, but it can be changed).
* Example: http://127.0.0.1:5500/

**2. Finds Your Project Root**

* Live Server sets your project’s **root folder** (the one opened in VS Code).
* When you open index.html, it serves that file via the HTTP server.
* So instead of file:///C:/Users/Vikas/Desktop/project/index.html, you get http://127.0.0.1:5500/index.html.

**3. Serves Files Over HTTP**

* Any request from the browser (/index.html, /style.css, /script.js) is handled by the server.
* The server responds with the correct file, just like how a real web server (e.g., Apache, Nginx) would.

**4. Injects Live Reload Script**

* Live Server injects a **WebSocket script** into your page.
* This script connects your browser to the server.
* When you save a file:
  + The server detects the change.
  + It sends a message through WebSocket to the browser.
  + Browser **auto-refreshes** the page.

**5. Allows LAN Access**

* If you use your **system’s IP address** (like 192.168.0.142:5500), other devices on the same Wi-Fi can also access it.
* This is why you could test your site on your phone.

**6. Keeps Running Until You Stop It**

* The server keeps listening until you click **“Stop Live Server”**.
* At that moment, it releases the port and shuts down the process.

**📊 In Terms of Computer Networks**

* **Live Server = A mini HTTP server (Application Layer)**
* It uses:
  + **Transport Layer (TCP)** → to deliver data reliably.
  + **Network Layer (IP)** → to decide where to send data (localhost or LAN IP).
  + **Link Layer (Wi-Fi/Ethernet)** → to physically send bits across your network.

👉 So in summary:  
When you click **Go Live**, you’re spinning up a **local HTTP server** → serving files via **HTTP over TCP/IP** → with **WebSocket auto-refresh** → accessible on both localhost and LAN IP.

Explore and **compare Live Server vs Netlify** (local vs deployed server) so you see how it moves from your PC to being public?

How to Go Public:-

<https://chatgpt.com/share/68b7cbf5-d9dc-8009-a63e-e6b7d690c561>