TECHNOLOGY · RASPBERRY PI VILLAGESERVER

A library that fits in your hand

The Raspberry Pi broadcasts a local Wi-Fi hotspot so nearby phones can browse an offline Bible library — no internet, no data plan. This guide covers what the interface looks like, how to use it in the field, how to build and configure the server, and how to manage it remotely via WireGuard VPN.

English

OVERVIEW

What is the VillageServer Pi?

A Raspberry Pi running a local web server. Plug in microSD cards loaded with language libraries, power it on, and it broadcasts a Wi-Fi network. Anyone within range can join that network and open the library in a phone browser — like having a small offline internet made entirely of Scripture and gospel resources.

The complete kit: carry case, Raspberry Pi, power bank, and USB readers holding the language library cards.

Hardware you need

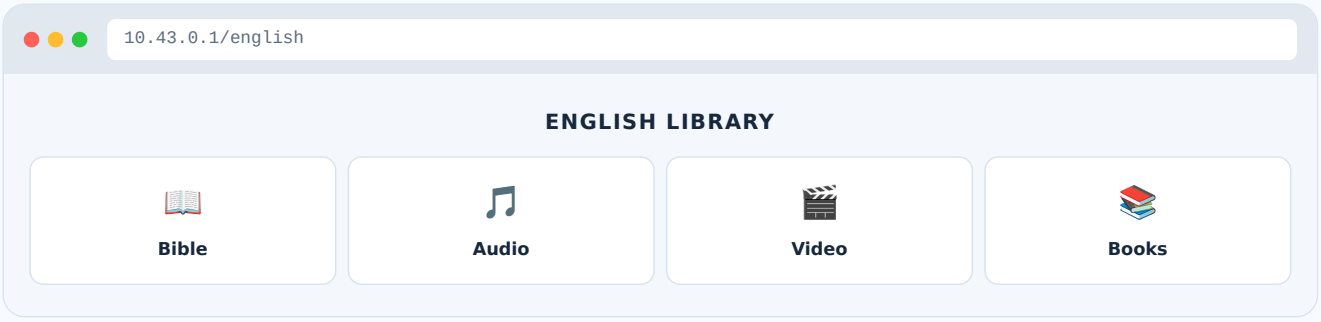
ITEM	DETAILS
Raspberry Pi	**Pi 4 (2 GB+ RAM recommended) or Pi 3B+**
System microSD card	**16 GB min · 32 GB recommended (for the Pi OS)**
Language library cards	**One microSD per language, in USB readers/adapters**
Power supply	**5V USB-C (Pi 4) or USB Micro-B (Pi 3) · power bank works**
Setup laptop	**Any OS · needs Raspberry Pi Imager (free)**
Carry case (optional)	**Small case keeps everything together for field use**

THE INTERFACE

What you see on a phone

When a phone joins the VillageServer Wi-Fi and opens `10.43.0.1` in a browser, this is what appears — a clean offline library home. No login, no app install, no data required.

The home page shows every loaded language. Tap a card to open that library.



Each language has four content sections — tap any tile to browse.

Bible — full Scripture text, chapter by chapter.
Audio — dramatised audio Bible recordings, playable directly in the browser.
Video — LUMO gospel films (Matthew, Mark, Luke, John) and other gospel media.
Books — commentaries, discipleship materials, and reference PDFs.

> **Updating Libraries banner:** when content is being refreshed remotely, the interface shows "Updating Libraries — please wait." The server continues to serve existing content while the update runs in the background.

FOR THE FIELD

Using the library — step by step

This is all a person in the field needs to know. The server handles everything else automatically.

1

Power on the VillageServer

Plug in the power bank or power supply. Wait about 60 seconds for the Pi to finish starting up. The Wi-Fi network will appear automatically.

2

Join the Wi-Fi network

On your phone, open **Settings → Wi-Fi** and select `VillageServer`. Enter the password `village123` when prompted.

> **Note:** your phone may warn that this network has no internet — that's expected. Tap "Stay connected" or "Use anyway."

3

Open the library in a browser

Open any browser (Chrome, Safari, Firefox) and type `10.43.0.1` into the address bar. Hit Go. The library home page loads instantly.

4

Choose a language and browse

Tap a language card, then tap Bible, Audio, Video, or Books. You can read, listen, or watch directly in the browser without saving anything first.

5

Download to share later

Hold any file link to save it to your phone. Once saved, share it to other phones via AirDrop, Quick Share, LocalSend, Bluetooth, or USB without any Wi-Fi needed.

文

SETUP GUIDE · PART 1

Building the server

These steps take 1–2 hours on first run. Once done the Pi is fully self-contained and requires no laptop to operate in the field.

Workshop assembly before a field deployment.

Step 1 — Flash the operating system

Download and open **Raspberry Pi Imager** on your laptop (free from raspberrypi.com). Choose the device, then select:

OS: Raspberry Pi OS Lite (64-bit) — no desktop, runs faster as a server.
Storage: your system microSD card (not the language cards).

Before clicking Write, open the settings gear ⚙ Edit Settings and fill in:

- **Hostname:** `villageserver`
- **Enable SSH** with password authentication
- **Username/password:** `pi` and a strong password — write it down
- **Wi-Fi:** your home/office network so the Pi can reach the internet on first boot for updates

> Click **Save** then **Yes** to apply these settings, then write the image. This pre-configures the Pi so you don't need a monitor or keyboard.

Step 2 — Install the server software

Insert the freshly-written card into the Pi and power it on. Wait 60 seconds, then find the Pi on your router's device list (look for hostname `villageserver`) and SSH in:

```
# Replace the IP with your Pi's actual address from the router
ssh pi@192.168.x.x
```

Accept the fingerprint, enter your password, then update and install the required packages:

```
# Update system packages first
sudo apt update && sudo apt upgrade -y

# Install Kiwix (offline library engine) and nginx (web server)
sudo apt install -y kiwix-tools nginx

# Create the mount point for language cards
sudo mkdir -p /mnt/library
```

Plug the USB readers (with language cards) into the Pi's USB ports, then mount them:

```
# See what cards are connected
lsblk -o NAME,SIZE,LABEL,MOUNTPOINT

# Mount each card (repeat for each language)
sudo mkdir -p /mnt/library/english
sudo mount /dev/sda1 /mnt/library/english

sudo mkdir -p /mnt/library/spanish
sudo mount /dev/sdb1 /mnt/library/spanish
```

Create the Kiwix service so it starts automatically on every boot:

```
sudo nano /etc/systemd/system/kiwix.service
```

Paste this content, save with **Ctrl+O**, then exit with **Ctrl+X**:

```
[Unit]
Description=Kiwix VillageServer
After=network.target

[Service]
ExecStart=/usr/bin/kiwix-serve --library --port=8080 /mnt/library/**/*.zim
Restart=on-failure
User=pi

[Install]
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload
sudo systemctl enable kiwix
sudo systemctl start kiwix
```

Configure nginx to forward requests on port 80 to Kiwix:

```
sudo nano /etc/nginx/sites-available/default
```

Replace the file contents with:

```
server {
    listen 80 default_server;
    server_name _;
    location / {
        proxy_pass http://127.0.0.1:8080;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }
}
```

```
sudo nginx -t && sudo systemctl reload nginx
```

Step 3 — Configure the Wi-Fi hotspot

```
sudo apt install -y hostapd dnsmasq
sudo systemctl unmask hostapd
```

Set the hotspot name and password:

```
sudo nano /etc/hostapd/hostapd.conf
```

```
interface=wlan0
driver=nl80211
ssid=VillageServer
hw_mode=g
channel=7
wpa=2
wpa_passphrase=village123
wpa_key_mgmt=WPA-PSK
rsn_pairwise=CCMP
```

Tell dnsmasq to hand out addresses in the `10.43.0.x` range:

```
sudo nano /etc/dnsmasq.conf
```

```
interface=wlan0
dhcp-range=10.43.0.2,10.43.0.50,255.255.255.0,24h
address=/#/10.43.0.1
```

Give the wireless interface a fixed IP address:

```
sudo nano /etc/dhcpcd.conf
```

Add to the end of the file:

```
interface wlan0
    static ip_address=10.43.0.1/24
    nohook wpa_supplicant
```

```
sudo systemctl enable hostapd dnsmasq
sudo systemctl restart dhcpcd
sudo systemctl start hostapd dnsmasq
```

Step 4 — Test from a phone

Testing phone connections to the VillageServer before field deployment.

Pick up a phone and follow the field-use steps above: join `VillageServer` Wi-Fi → open browser → go to `10.43.0.1`. Verify language tiles appear, tap into a content section, and confirm audio or video plays without any mobile data.

> **All good?** The server is ready for field deployment. Power it off, pack it in the case with the language cards seated in the USB readers, and it's field-ready.

SETUP GUIDE · PART 2

WireGuard — remote access tunnel

Once a Pi is deployed, it is typically behind a cellular NAT with a dynamic IP — there's no way to connect to it directly. WireGuard fixes this: the Pi connects outward to a relay server (VPS) you control. Your laptop connects to the same VPS. Both devices are on one private network, so you can SSH into any field Pi from anywhere in the world.

> ***What you need:*** *a VPS with a public IP (any provider — DigitalOcean, Linode, Vultr, AWS — the cheapest $4–6/month tier is enough). Ubuntu 22.04 or Debian 12 recommended. WireGuard will run on the VPS, the Pi, and your admin laptop.*

How the addresses work: each device gets a private WireGuard IP on the `10.99.0.0/24` subnet:

- **VPS (relay):** `10.99.0.1`
- **Raspberry Pi (field):** `10.99.0.2`
- **Your laptop (admin):** `10.99.0.3`

VPS

Set up the VPS relay

SSH into your VPS, install WireGuard, and generate its keypair:

```
sudo apt update && sudo apt install -y wireguard
cd /etc/wireguard
wg genkey | tee vps_private.key | wg pubkey > vps_public.key
chmod 600 vps_private.key
```

Create the VPS config at `/etc/wireguard/wg0.conf` :

```
[Interface]
Address = 10.99.0.1/24
ListenPort = 51820
PrivateKey = <contents of vps_private.key>
PostUp   = iptables -A FORWARD -i wg0 -j ACCEPT; iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
PostDown = iptables -D FORWARD -i wg0 -j ACCEPT; iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE

# Pi peer — fill in Pi public key after generating it below
[Peer]
PublicKey = <Pi public key>
AllowedIPs = 10.99.0.2/32

# Admin laptop peer — fill in laptop public key after generating it
[Peer]
PublicKey = <Laptop public key>
AllowedIPs = 10.99.0.3/32
```

```
# Enable IP forwarding
echo "net.ipv4.ip_forward=1" | sudo tee -a /etc/sysctl.conf
sudo sysctl -p

# Open the WireGuard port
sudo ufw allow 51820/udp

# Start the tunnel
sudo systemctl enable wg-quick@wg0
sudo systemctl start wg-quick@wg0
```

文

Pi

Configure the Pi to connect to the VPS

SSH into the Pi (local network for now), install WireGuard, and generate its keypair:

```
sudo apt install -y wireguard
cd /etc/wireguard
wg genkey | tee pi_private.key | wg pubkey > pi_public.key
chmod 600 pi_private.key

# Print the public key — copy it into the VPS wg0.conf above
cat pi_public.key
```

Create the Pi's config. Replace `VPS_PUBLIC_IP` with your VPS's real IP address:

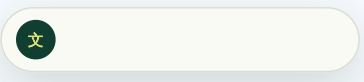
```
sudo nano /etc/wireguard/wg0.conf
```

```
[Interface]
Address = 10.99.0.2/24
PrivateKey = <contents of pi_private.key>
DNS = 1.1.1.1

[Peer]
PublicKey = <contents of VPS vps_public.key>
Endpoint = VPS_PUBLIC_IP:51820
AllowedIPs = 10.99.0.0/24
PersistentKeepalive = 25
```

`PersistentKeepalive = 25` sends a heartbeat every 25 seconds — this keeps the tunnel alive through cellular NAT even when the Pi isn't actively communicating.

```
sudo systemctl enable wg-quick@wg0
sudo systemctl start wg-quick@wg0

# Verify — you should see "latest handshake" for the VPS peer
sudo wg show
```

Paste the Pi's public key back into the VPS `wg0.conf` under the Pi peer block, then restart the VPS tunnel: `sudo systemctl restart wg-quick@wg0`



YOU

Connect your admin laptop

On **macOS**: install the WireGuard app from the Mac App Store, then create a new tunnel and paste the config below.

On **Linux**: `sudo apt install wireguard`

Generate a keypair on your laptop:

```
wg genkey | tee laptop_private.key | wg pubkey > laptop_public.key
```

Your laptop WireGuard config:

```
[Interface]
Address = 10.99.0.3/24
PrivateKey = <contents of laptop_private.key>
DNS = 1.1.1.1

[Peer]
PublicKey = <contents of VPS vps_public.key>
Endpoint = VPS_PUBLIC_IP:51820
AllowedIPs = 10.99.0.0/24
PersistentKeepalive = 25
```

Add your laptop public key to the VPS `wg0.conf` (the laptop peer block), restart the VPS tunnel, then activate your tunnel. Now SSH directly into the field Pi:

```
ssh pi@10.99.0.2
```

You're in — from anywhere in the world, with no port-forwarding or public IP on the Pi.

OPERATIONS

Updating content remotely

With the WireGuard tunnel running, you can push new language libraries to any field Pi without touching the device.

1

Copy the new library file to the Pi

Run this from your laptop. `scp` sends the file over the WireGuard tunnel:

```
# Run on your laptop — not on the Pi
scp new-library-english.zim pi@10.99.0.2:/home/pi/incoming/
```

2

Move the file and restart Kiwix

```
# SSH into the Pi first
ssh pi@10.99.0.2

# Move the file into the library mount
sudo mv /home/pi/incoming/new-library-english.zim /mnt/library/english/

# Restart Kiwix to load the new content
sudo systemctl restart kiwix
```

Connected phones briefly see the "Updating Libraries" banner, then the new content is live within about 30 seconds.

3

Verify from a connected phone

On a phone joined to the VillageServer Wi-Fi, reload `10.43.0.1`. The new or updated library tile should appear. Tap through to confirm the content opens correctly.

TROUBLESHOOTING

Common issues

Phone joins Wi-Fi but can't reach 10.43.0.1

Check nginx and Kiwix are both running:

```
sudo systemctl status nginx kiwix
```

Confirm `wlan0` has the right IP:

```
ip addr show wlan0
```

You should see `10.43.0.1/24`. If not, restart dhcpcd and hostapd.

WireGuard shows no handshake

Run `sudo wg show` on both the Pi and VPS. Common causes: VPS firewall blocking port 51820/udp, or a typo in a public key. Verify keys were copy-pasted correctly with no extra spaces.

Kiwix returns no libraries

```
# Check if cards are mounted
df -h
ls /mnt/library/
```

If the mount point is empty, the USB card may have ejected or the device path changed. Re-plug the reader and re-mount.

Hotspot disappears after a Pi update

A system update can overwrite dhcpcd config. Re-check `/etc/dhcpcd.conf` for the `wlan0` static IP block, then restart: `sudo systemctl restart hostapd dhcpcd`

Download the Raspberry Pi system overview as a printable PDF for field briefings or board packets.

Download Raspberry Pi PDF