serve.py — User Guide

Spreadsheet-to-Live-Webpage script

Before you start

1. Ensure Python is installed on your computer

```
Python --version
Python 3.14.5
Ensure you have at least python version 3.8 or higher
If you do go to "Setting up the Python Script"
```

2.Install Python to your computer

Go to python.org/downloads

Download the latest version for your OS

Windows: run the installer and check "Add Python to PATH" before clicking Install

Note: You may have to close and reopen any terminals to apply PATH variables.

Setting up the Python script:

1. Copy Serve.py to your desired directory

2. Set up the virtual environment (one-time)

```
Go into the same directory as serve .py and run the following:
python -m venv venv
venv\Scripts\activate          # Windows
source venv/bin/activate        # Linux / Mac
pip install openpyxl
```

3. Run the script

```
py .\serve.py .\your-file.csv
```

Note: The .csv or excel file does not need to be in the same directory as serve.py. simply pass the file path of the target file as a parameter

The terminal will print the local and network URLs:

```
Serving 'your-file.csv'
  Local:   http://localhost:8080
  Network: http://192.168.x.x:8080
  Poll interval: 2000ms
```

```
Press Ctrl+C to stop.


Ensure this script continuously runs on a machine with the same. If not
the display will not function.
Document the Network Ip address including the port as it will be used in
the future
```

3. Open in a browser

- Same machine: open http://localhost:8080
- Another device on the same network: open http://192.168.x.x:8080
- Embed in an iframe (e.g. Vidimax HTML editor):

```
<iframe src="http://<Ip_Addr>:8080"
style="width:100%;height:100%;border:none;"></iframe>
```

Supported File Types

Extension	Format	Notes
.csv	CSV	Works with file open in any editor
.xlsx	Excel (modern)	Works with file open in Excel
.xls	Excel (legacy)	Requires openpyxl
.xlsm	Excel (macro-enabled)	Macros are ignored; data only

Flags

Flag	Default	Description
--port <int>	8080	Port the server listens on
--poll <ms>	2000	How often (milliseconds) clients refresh the table

Examples

Default settings:

```
py .\serve.py .\data.xlsx
```

Default port : 8080

Default Interval : 2 Secs

Custom port:

```
py .\serve.py .\data.xlsx --port 9000
```

Custom poll rate (10 seconds — recommended for 100+ clients):

```
py .\serve.py .\data.csv --poll 10000
```

Both flags together:

```
py .\serve.py .\data.xlsx --port 9000 --poll 10000
```

Configuring Vidimax to Display your live spreadsheet

Note: This section assumes your Vidimax Display System has been properly configured and set up. It will guide you on creating the slate that displays the spreadsheet data. For instructions on rotating displays/rule based displays see the Full Vidimax User manual.

Login to your Vidimax system and navigate to the Slates tab.

Create a new slate template with the following values:

- **Name: Desired Name**
- **Slate Type: Message**
- **Html Editor: Enabled**

Once the HTML Editor is enabled you should be prompted with a HTML Editor.

Paste the following into the editor replacing <IP_ADDR> and <PORT> with the Network IP address and Port documented in earlier steps:

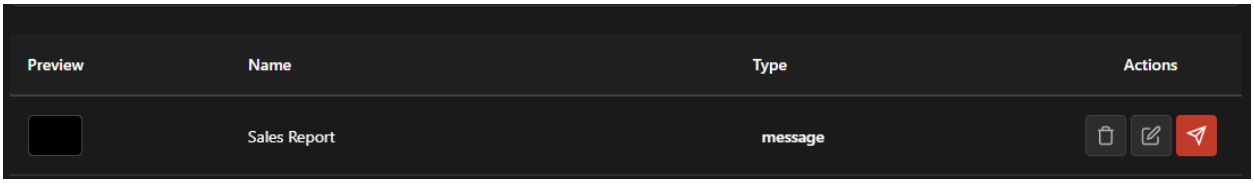
```
<iframe src="http://<IP_ADDR>:<PORT>"  style="width:100%;height:100%;border:none;"></iframe>
```

Don't worry about adding any head or body elements Vidimax handles that for you.

Click the save Icon in the top right.

Your new slate should now exist. Test it by clicking the red "send" icon and send the spreadsheet to your desired display.

Edit the targeted spread sheet and see the changes in real time.



Suggested Use Cases

- **Live dashboards —** Live dashboards:
    - Display real-time KPIs, finance data, or schedules on dedicated screens fed by a Raspberry Pi or similar device.
- **Office signage —** Office signage:
    - Show room bookings, shift rosters, or stock levels on wall-mounted displays. Update the spreadsheet and the screen refreshes automatically.
- **Meeting room displays —** Meeting room displays:
    - Keep a shared .xlsx on a network drive; anyone with Excel access can update the display in real time.
- **Multi-site monitoring —** Multi-site monitoring:

    - Run serve.py on a central machine visible to all sites; each site's display device iframes the same URL.

Notes & Tips

- Windows Firewall: on first run Windows may prompt to allow Python through the firewall — click Allow so other devices can reach the server.
- Static IP: pin a static IP to the machine running serve.py so the iframe URL never changes after a reboot.
- High client counts: use --poll 10000 or higher when serving 100+ clients to reduce CPU load.
- Stop the server: press Ctrl+C in the terminal, then run deactivate to exit the virtual environment.
- Linux/Mac activation: use source venv/bin/activate instead of venv\Scripts\activate.