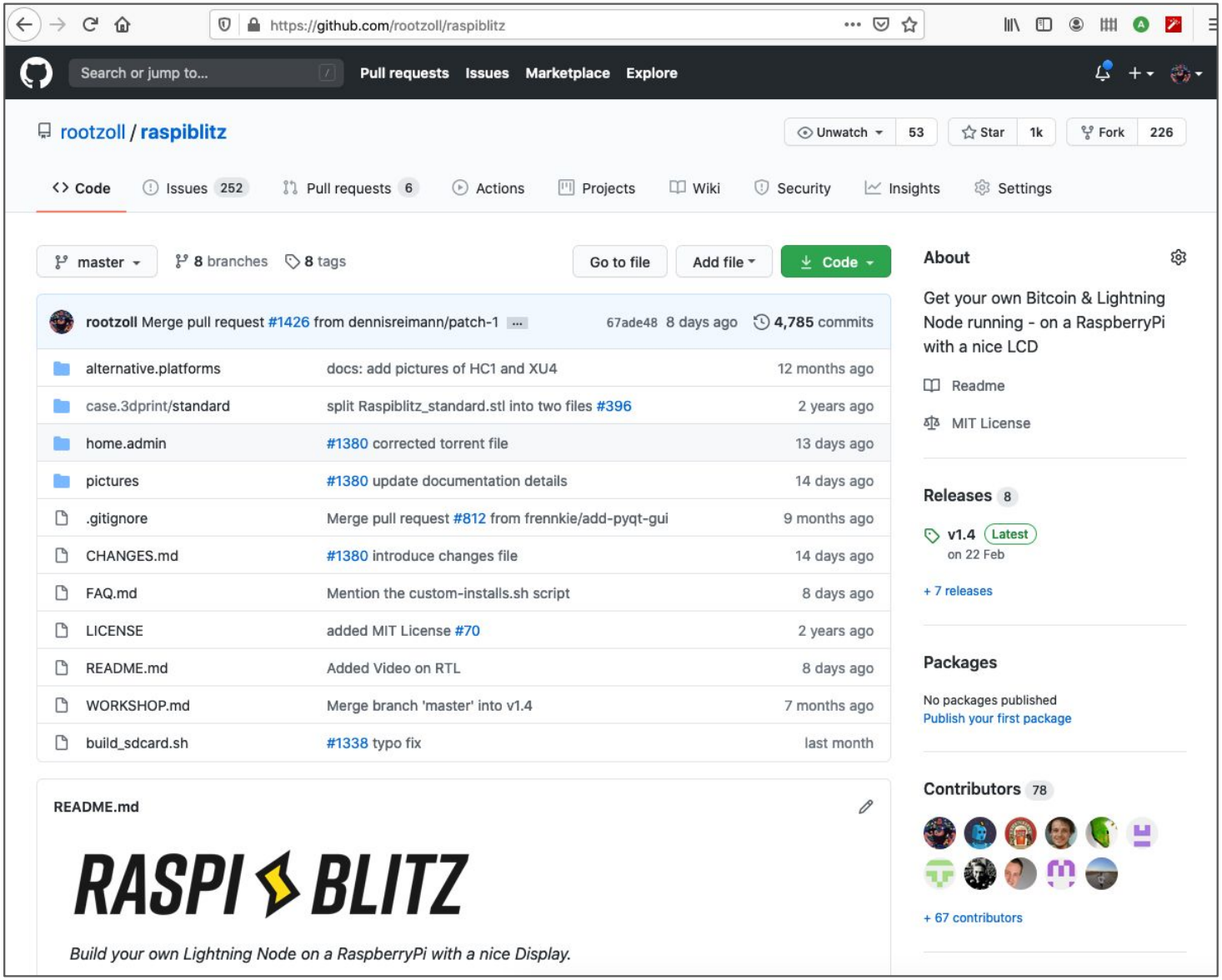
Contributing to RaspiBlitz Development



raspiblitz.org

→ go to GitHub Page

- At least get a GitHub Account to contribute
- **Create an Issue** to get conversation started
- You can even start without any further knowledge on how to use GitHub

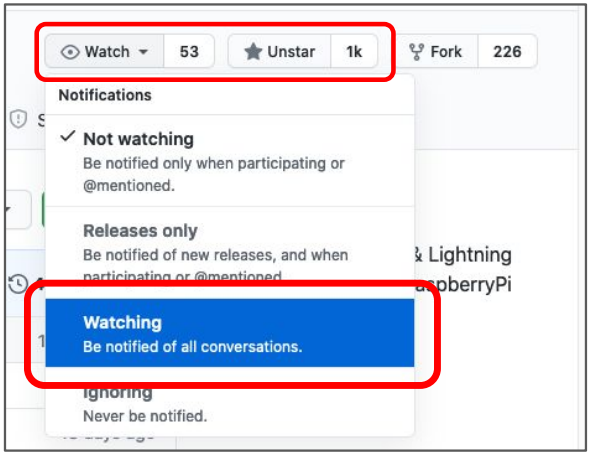
Levels of Contribution to the RaspiBlitz Project

1. **Testing, Report & Discuss**: RaspiBlitz or Additional Apps
2. **Experiment**: Try things out on your own RaspiBlitz
3. **Install-Script**: Turn your Experiment into a basic Shell-Script
4. **Config-Script**: Integrate your Script into the RaspiBlitz Environment
5. **SSH-GUI**: Make it easy for others to use your Feature
6. **Web-GUI**: Have your Feature ready Customer-Ready

Testing, Report & Discuss: RaspiBlitz or Additional Apps

Skills: User-Level (no programming skills needed)

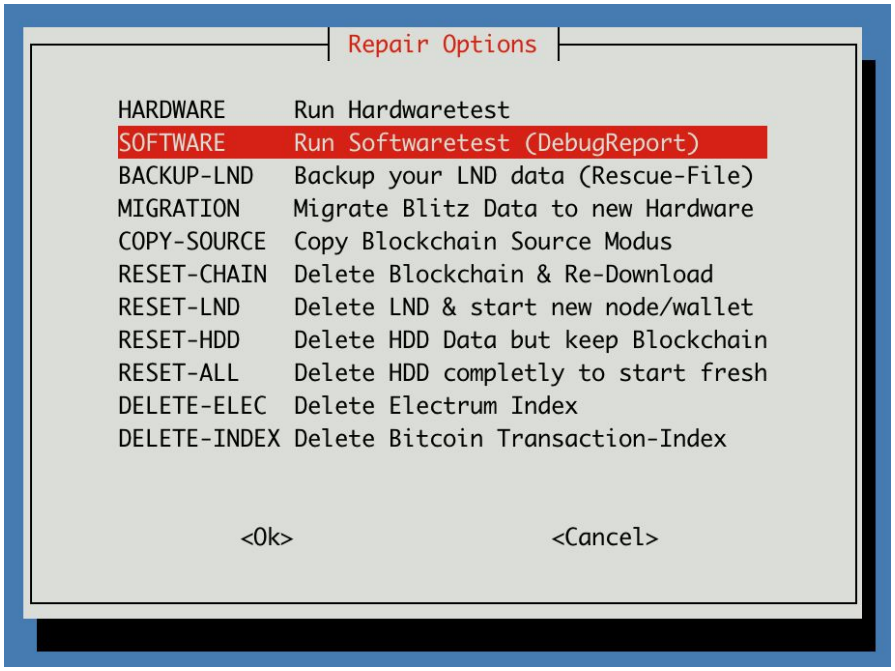
- What can work better? → Create Issue: Feature-Request
- What's not working? → Create Issue: Error-Report → Debug-Report*
- Connect with teams of the "Additional Apps"
- Help other users on Error-Reports
- Correct a Typo → Do it directly on GitHub*

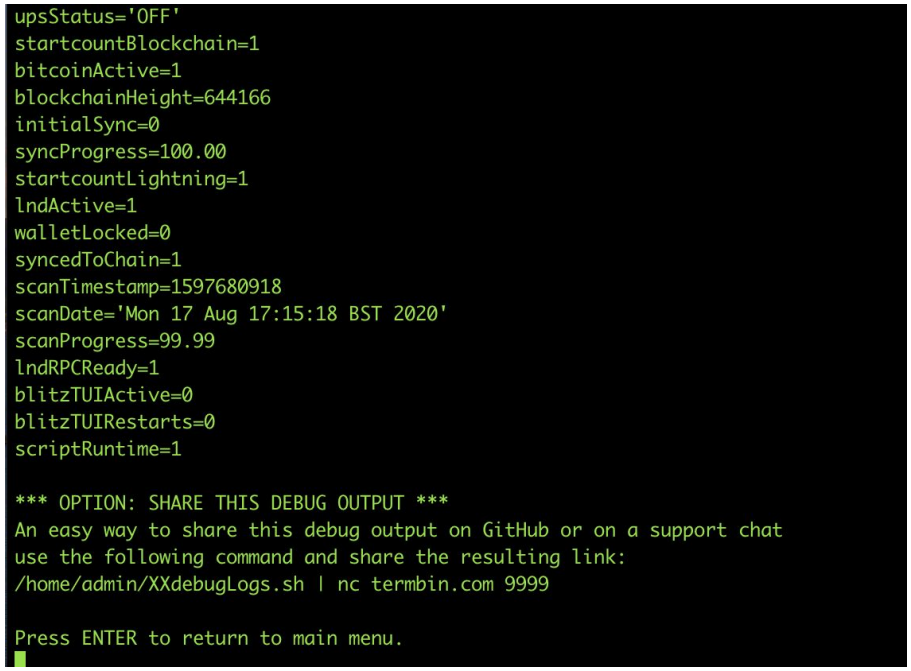


Testing, Report & Discuss: RaspiBlitz or Additional Apps

Skills: User-Level (no programming skills needed)

- What's not working? → Create Issue: Error-Report → Debug-Report



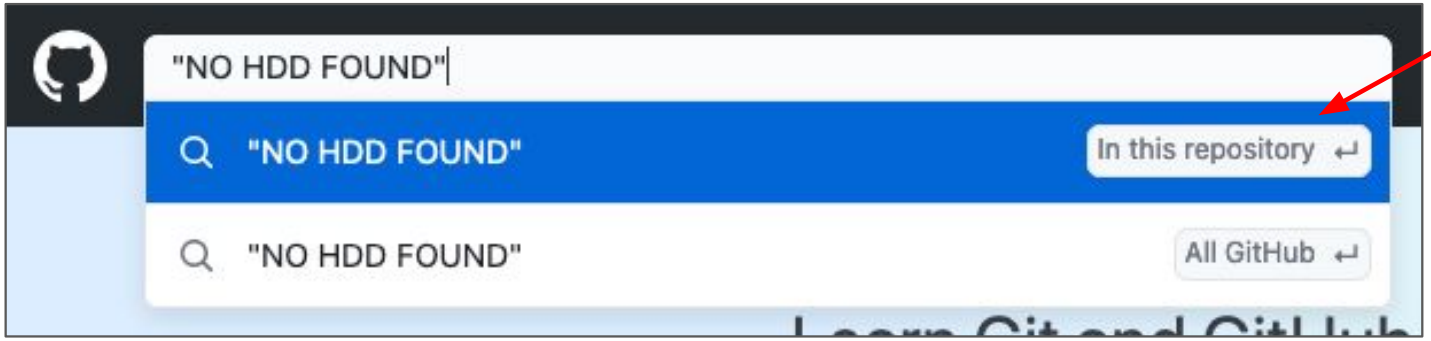


Testing, Report & Discuss: RaspiBlitz or Additional Apps

Skills: User-Level (no programming skills needed)

- Correct a Typo → Do it directly on GitHub

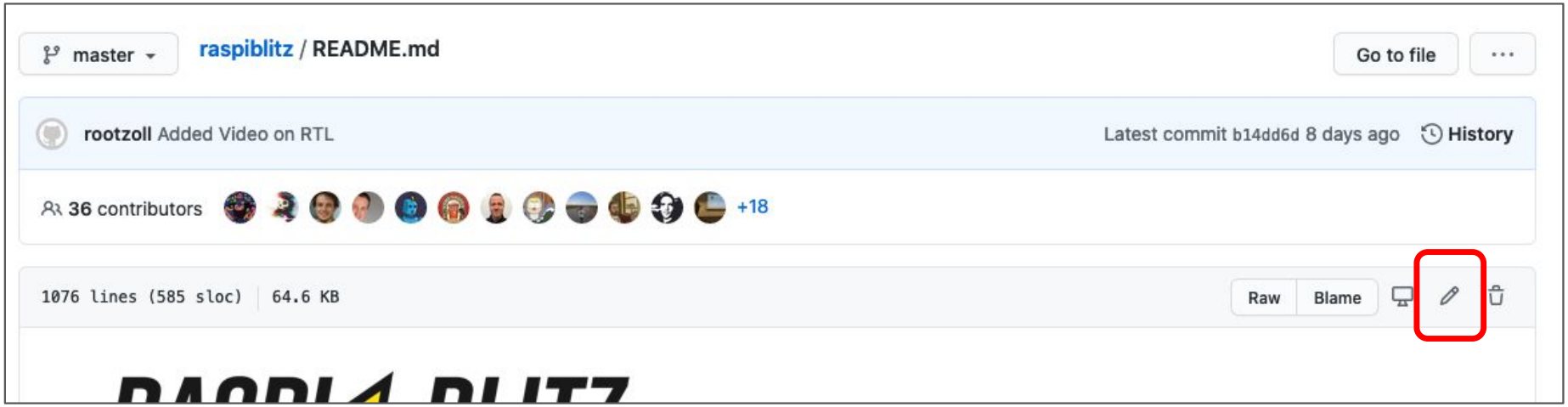
Use GitHub search to find the typo in the code with “In this repository”



Testing, Report & Discuss: RaspiBlitz or Additional Apps

Skills: User-Level (no programming skills needed)

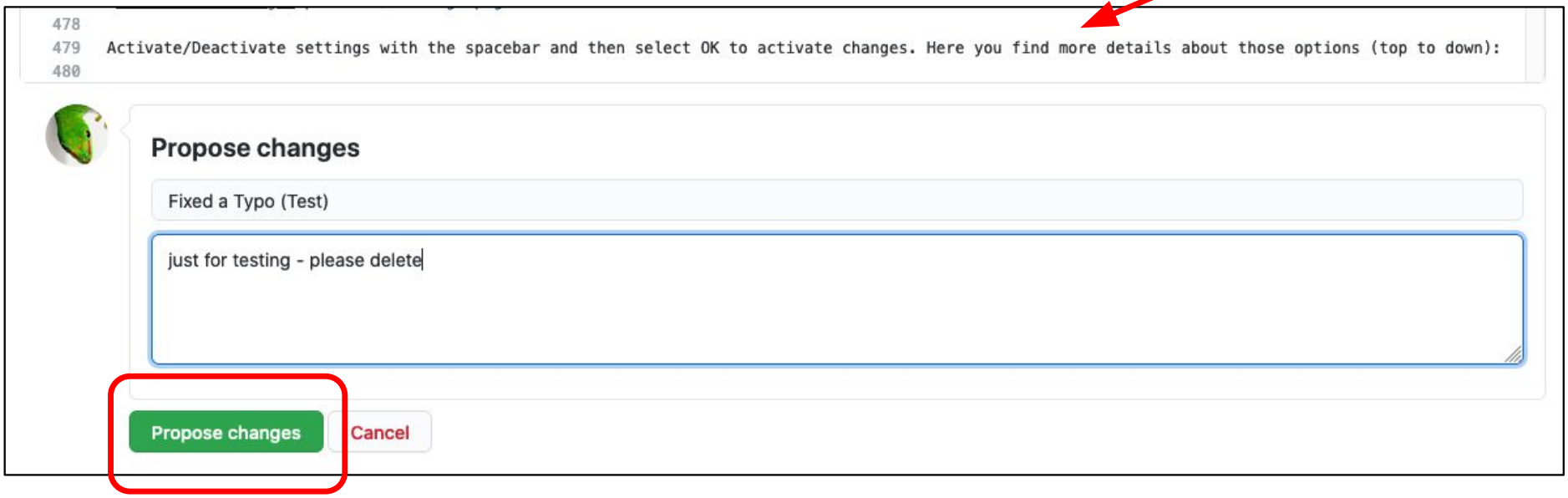
- Correct a Typo → Do it directly on GitHub



Testing, Report & Discuss: RaspiBlitz or Additional Apps

Skills: User-Level (no programming skills needed)

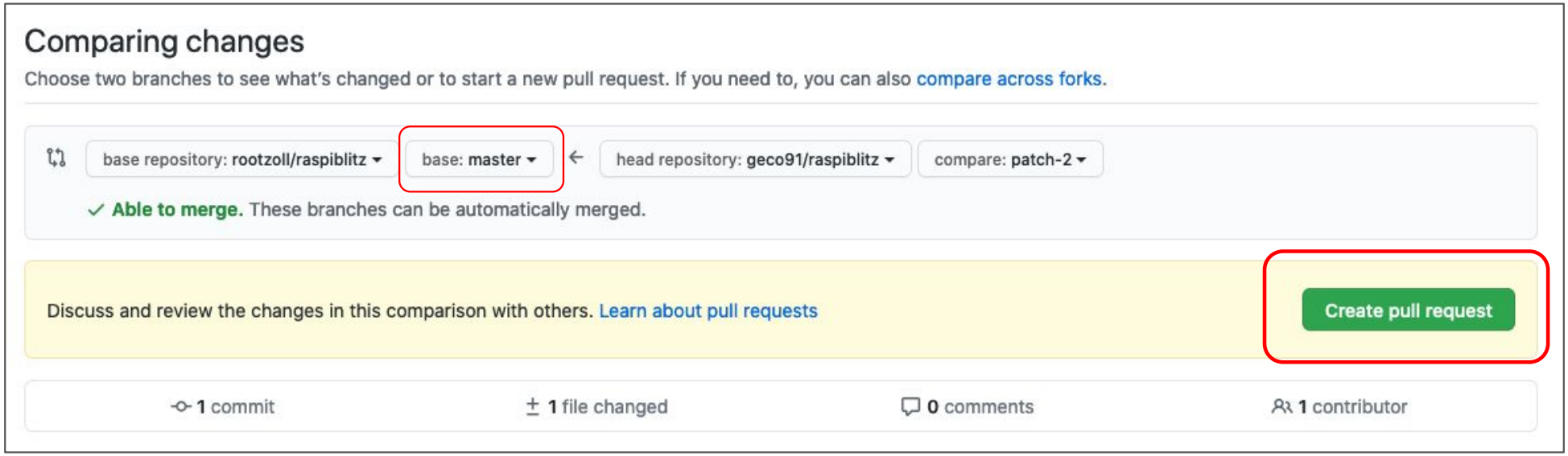
- Correct a Typo → Do it directly on GitHub



Testing, Report & Discuss: RaspiBlitz or Additional Apps

Skills: User-Level (no programming skills needed)

- Correct a Typo → Do it directly on GitHub



Experiment: Try things out on your own RaspiBlitz

Skills: Basic Linux Terminal Knowledge

- Get another project running as an “Additional App” on RaspiBlitz
- Make a change to the Systems configuration
- Connect different Hardware or Hacking Projects

- Built on a Linux distro called Raspberry OS - based on Debian
- `bitcoind` is running under user `bitcoin` in `/mnt/hdd/bitcoin`
- `lnd` running under user `bitcoin` in `/mnt/hdd/lnd`
- `lnd` RPC port: `10009` REST port: `8080`
- get macaroons/TLS under `SSH-MENU > LNDCREDS > EXPORT`

Share your Experience as a Report-Issue

Install-Script: Turn Experiment into a basic Shell-Script

Skills: Basic Bash-Shellscript Knowledge

- RaspiBlitz starts on every Update with a fresh sd card → clean system
- Write your changes into **`/mnt/hdd/app-data/custom-installs.sh`**
- Make sure you script runs without user-interaction
- Test with inserting a fresh sd card, run recovery and check results
- Find debug output of recovery: `/home/admin/raspiblitz.recover.log`

outcome → **Share this shell script within an issue** (without sensitive data)

Config-Script: Integrate into the RaspiBlitz Environment

Skills: GitHub & Advanced Bash-Shellscript

- Work on your own forked Repo & sync it to your RaspiBlitz for testing*
- Add your script to directory: **`/home/admin/config.scripts/`**
- Basic functions should be: `on`, `off` ... including the parameters you need
- Configuration: `/mnt/hdd/raspiblitz.conf` Data: `/mnt/hdd/app-data`
- Part of recovery process: `/home/admin/_bootstrap.provision.sh`

outcome → **Make a Pull-Request from your forked Repo to RaspiBlitz Repo**

Config-Script: Integrate into the RaspiBlitz Environment

Skills: GitHub & Advanced Bash-Shellscript

- Work on your own forked Repo & sync it to your RaspiBlitz for testing
  `MAINMENU > UPDATE > PATCH > PATCH MENU > REPO` and/or `BRANCH`

- Use your Editor to **change scripts** on forked Repo
- **Commit** small changes to that Repo
- On RaspiBlitz terminal: call command **`patch`** to sync changes onto Raspi
- run, test & repeat

good to know

- Temp state info: `/home/admin/raspiblitz.info`
- every additional app should run as own user
- add user to matching groups for access to LND macaroons
- WebUIs should integrate with nginx for SSL & TOR & IP2TOR

SSH-GUI: Make it easy for others to use your Feature

Skills: GitHub & Advanced Bash-Shellscript & Dialog-GUI

- Project should already be community tested & has improved resilience
- Adding `dialog` or `whiptail` SSH menus for install & config
- If its a “Additional App” it should have a `status` page for the MainMenu
- Logs should be included in the `debug` informations
- Commitment for Documentation, Support and Updating the feature

Make a Pull-Request from your forked Repo to RaspiBlitz Repo

Web-GUI: Have your Feature ready Customer-Ready

Skills: GitHub & Advanced Bash-Shellscript & WebFrontend

- Project working fine in SSH world → Make available to WebGUI
- Reduce Complexity to identified “default cases” for users of just WebGUI
- WebGUI JS → calls Endpoint → calls Config-Script → returns JSON Output
- Config-Shellscript outputs usable for Web-Backend-API (JSON or KeyValue)
- Also consider adding a “Infoscreen” on the touchscreen menu (BlitzTUI)

This is still in Development - may see prototypes in RaspiBlitz v1.8