Raspberry Pi Camera with Tactile Controls and Web Gallery

Unix

Aya Kharchafi and Lydia Ayala Hernandez

01 Project Description and Motivation

02 Major Tech Solutions Compared

03 Demo

04 GitHub Repo Tour

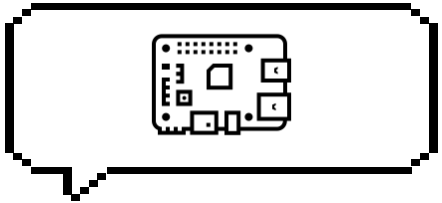
05 Development Journey

06 Major Challenges

07 Major Accomplishments

08 Future Plans

09 Why Should You Buy Our Product?



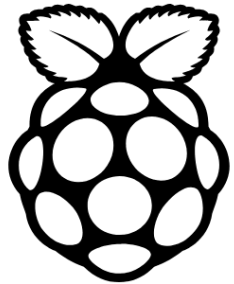
01
Project Description and Motivation

A portable camera system using Raspberry Pi 4 Model B and Camera Module 2 that can take pictures and videos at the press of a button.

Project Description and Goals

Key Features:

- **2 Main Buttons:**
  1. Take pictures
  2. Record videos
- **Media** is saved as files **accessible** via a **web server.**
- **Customization Options:**
  Add filters or crop pictures/videos through command-line tools.



Design and Usability:

- Compact and housed in a custom-built case.

Goals:
To provide an easy-to-use, hands-free solution for capturing moments during activities like hiking or biking.

Target Audience and Motivation

Who Is It For?

- **Primary Users:** Hikers, bikers, and outdoor enthusiasts.
- **Additional Users:** Anyone needing a hands-free camera for recording or taking pictures during daily activities.

Why Build It?

- Enhance user experience by eliminating the need to hold the camera while capturing media.
- Offer a **GoPro-like alternative** that is open-source, affordable, and highly customizable.

Practical Applications:

- Documenting outdoor adventures.
- Creating portable, wearable solutions for general photography or video recording.

Relevance of the Project

Portable:
Lightweight design with a secure case to protect the Raspberry Pi from any external damage
Customizable:

- Open-source software allows modification of features.
- Ability to enhance media files with filters and cropping.

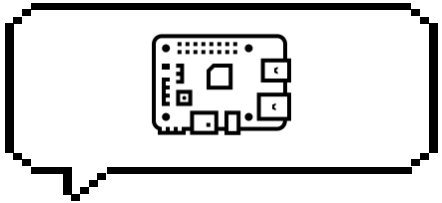
User-Focused:

- Built with ease of use in mind: intuitive button-based controls.
- Sufficient memory to support hours of recording.

Why Raspberry Pi?

- Compact and powerful platform capable of handling the camera module and web server.
- Runs on **Raspberry Pi OS**, enabling flexibility in development and operation.

Impact:
Fills a gap for an affordable, DIY portable camera system, empowering users to capture their adventures with minimal effort.



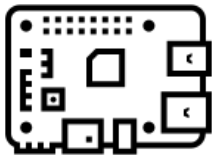
02
Major Tech Solutions Compared

Hardware, Software and Button Setup

Hardware

Chosen Option: Raspberry Pi 4 Model B

- Compact and lightweight for portability.
- Sufficient processing power for camera operation and web server hosting.
- Built-in GPIO pins simplify integration of buttons and peripherals.



Alternative: Arduino

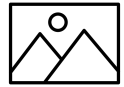
- **Pros**: Excellent for simple input/output tasks, low power consumption.
- **Cons**: Limited processing power; unsuitable for tasks requiring media storage or running a web server.

ARDUINO

Software

Web Server: Nginx

Lightweight and optimized for small devices.
Efficient handling of web traffic with minimal memory usage.Make nginx into a reverse proxy to handle requests from clients to web servers.



Gallery Software: PiGallery2

Lightweight, mobile-friendly, and optimized for Raspberry Pi.
Provides search filters, generates image thumbnails, and integrates seamlessly with Nginx.

Alternatives:

Apache: Robust but resource-heavy; unnecessary for this small-scale project.
Lychee: Simple but lacks advanced tools like search filters.
PhotoPrism: Powerful AI tools but overly complex for project requirements.

Button Setup

	Tactile Buttons	Touch Sensors
Reliable and responsive.	✓	
Easy to integrate with GPIO pins on Raspberry Pi.	✓	
Sleek and modern design.		✓
Cost-effective and durable for outdoor activities.	✓	

03

Hands - on

Show the Raspberry Pi setup, including the breadboard and tactile buttons.

04

GitHub Repo Tour

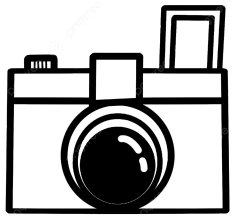
README, license, scripts

https://github.com/ayakharchafi/Unix-Project24

05

Development Journey

Plan VS Actual Experience



First Week

Initial Plan

- Connect the camera to the Raspberry Pi
- Install the tactile buttons and complete the script
- Start the web server

What Actually happened

- Plugged the connectors from the RPi to the breadboard
- Installed the buttons
- Created the GitHub Repository

Problem Encountered: Camera Module Not being Recognized!!

Second Week

Initial Plan

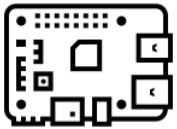
- Finish server
- Debug and test
- Make the camera portable
- Send media files to the web server
- Start creating the case for the completed camera (3D printing)

What Actually happened

- Code to print something when the button is pressed
- Switch SD to Finally Recognize Camera
- Finished Script to Take Pictures

Problem Encountered: Web server not working!!

Third Week



Initial Plan

- Finish the case
- Attach the camera to the harness
- Prepare and finish the content for our presentation
- Go over everything to make sure that everything works

What Actually happened

- Finished the case
- Finished the Web Server
- Finished Script to Record
- Installed the right OS
- Create Systemd processes to run Recording/Picture script when the RPi Boots
- Install Firewall
- Install SSH keys
- Finish Documentation
- Finish GitHub Repos
- Attach the camera to the harness
- Make the camera portable
- Send media files to the web server
- Prepare and finish the content for our presentation

Problem Encountered: Not Running on an RPi !!

06

Major challenges

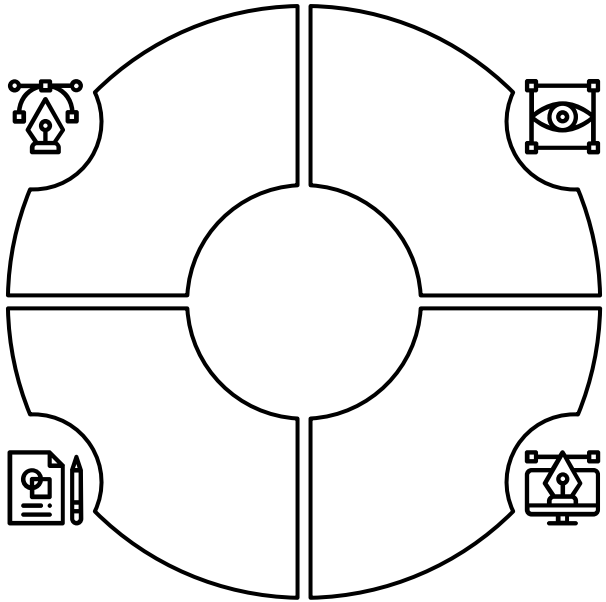
Challenges

Camera not detected

OS version

Button testing

GP/GND and 6 pins switch self locking buttons



Installing the Web Server

Docker vs Node.js

Python Script

“Not running on an RPi!”

07

Major accomplishments

Enabling the Camera

Our Journey

- Installing python3-picamera
- Switching SD
- Edit config.txt → dtoverlay=imx219
- Install Legacy 32-bit RPi OS

Connecting the buttons

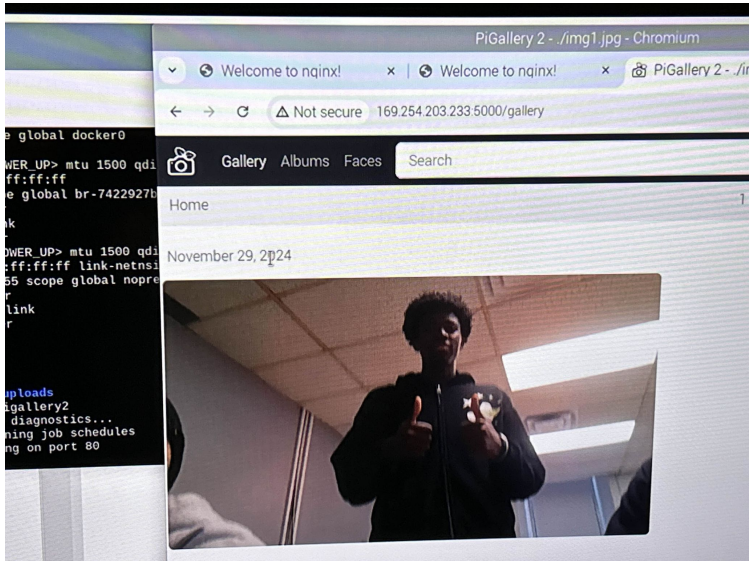
- Material: F:M & F:F + Breadboard + Buttons
- Thonny
- Import gpiozero
- GP2 + GND connected to the button
- Create Code to test is the button is pressed

Python Code to Print A statement When a button is pressed

```
from gpiozero import Button

button = Button(2)

def button_pressed():
    print("Button was pressed!")

button.when_pressed = button_pressed
```

Installing PiGallery2 and Nginx as a Reverse Proxy

- Install docker
- Set Up PiGallery2 Docker Container
- Install Nginx as a reverse proxy
- Configure Nginx
- Create Systemd service



Installing SSH Keys and Firewall

SSH Keys

- Generate SSH keys
- Prepare SSH directory and configure access
- Check network details and attempt connection

Firewall

- Check firewall status
- Install and allow SSH through the firewall
- Verify configuration

Hands-Free Solution for Photography

Prototype

- Hold the camera to take pictures or record, which limits the ability to take pictures from afar.
- The camera can be attached to a durable cord worn around the neck, similar to regular cameras.
- Allows users to take pictures and record hands-free, making it especially useful for capturing moments during activities such as hiking or biking or vlogging.

08

Future plans

Button to Turn on the RPi

Raspberry Pi

- When using an RPi, you won’t be able to turn it on by using a button. You will need to connected to an outlet. An off/on button or other component does not exist.

Solution

- latching power switch circuit
- Uses a momentary button and a transistor or relay to control power delivery to the Raspberry Pi
- Once a button is pressed, the circuit will supply power to the Raspberry Pi, until the button is pressed again

Install PiGallery2 installed natively

Our Project

- PiGallery2 installed and runs in a docker
- Pre-made docker Image
- Web Server depends on the upstream
- Can affect our product
- Easy and Fast Installation (if everything goes well)

Enhancement

- Longer installation
- Native setup
- Manual Updates
- Complete control over your environment (Software and configuration setup)

Screen

Our Project

- When accessing and navigating through the RPi GUI, we need to connect the RPi to a screen (Monitor,TV, etc)

Enhancement

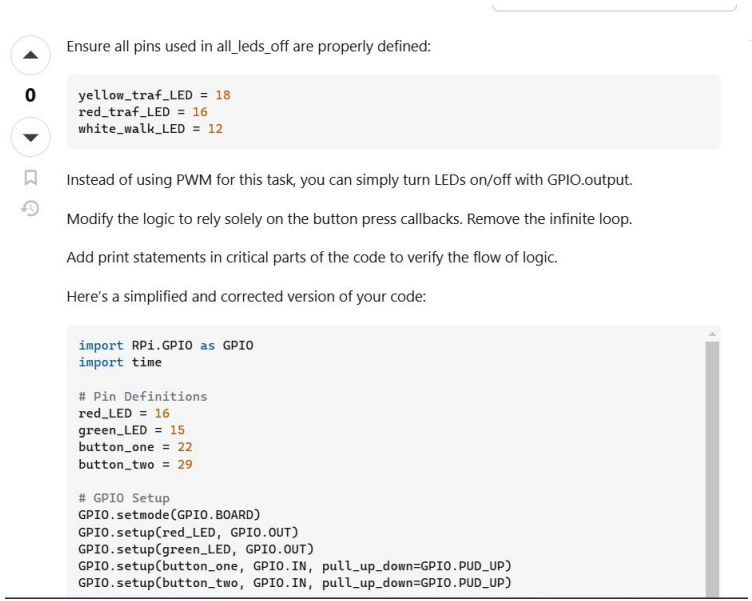
- Install a touch screen to create a more realistic camera where the user will be able to have everything at their disposal using the screen
- Can use it to monitor the web server
- Cleaner look and provides a more compact set up
- Add a cooling system to keep everything at a decent temperature (heatsink/fan)

10

Any contributions to FLOSS?

FLOSS Contribution

We wanted to give back to the open-source community that helped us throughout our project. By browsing Stack Overflow, we found a question about Raspberry Pi buttons and LEDs. Even though our project didn't involve LEDs, we applied what we learned, did additional research, and provided the person with an updated solution to their problem.



0

Ensure all pins used in all_leds_off are properly defined:

```
yellow_traf_LED = 18
red_traf_LED = 16
white_walk_LED = 12
```

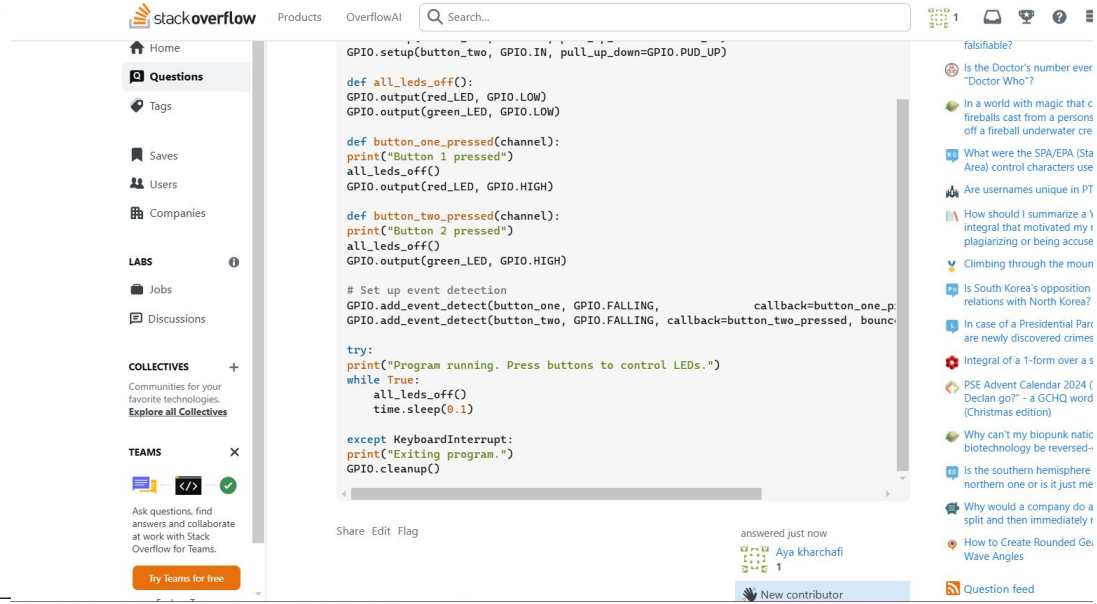
Instead of using PWM for this task, you can simply turn LEDs on/off with GPIO.output.

Modify the logic to rely solely on the button press callbacks. Remove the infinite loop.

Add print statements in critical parts of the code to verify the flow of logic.

Here's a simplified and corrected version of your code:

```
import RPi.GPIO as GPIO
import time

# Pin Definitions
red_LED = 16
green_LED = 15
button_one = 22
button_two = 29

# GPIO Setup
GPIO.setmode(GPIO.BOARD)
GPIO.setup(red_LED, GPIO.OUT)
GPIO.setup(green_LED, GPIO.OUT)
GPIO.setup(button_one, GPIO.IN, pull_up_down=GPIO.PUD_UP)
GPIO.setup(button_two, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```



stackoverflow Products OverflowAI Search...

Home
Questions
Tags
Saves
Users
Companies
LABS
Jobs
Discussions
COLLECTIVES
Communities for your favorite technologies. Explore all Collectives
TEAMS
Ask questions, find answers and collaborate at work with Stack Overflow for Teams.
Try Teams for free

```
GPIO.setup(button_two, GPIO.IN, pull_up_down=GPIO.PUD_UP)

def all_leds_off():
GPIO.output(red_LED, GPIO.LOW)
GPIO.output(green_LED, GPIO.LOW)

def button_one_pressed(channel):
print("Button 1 pressed")
all_leds_off()
GPIO.output(red_LED, GPIO.HIGH)

def button_two_pressed(channel):
print("Button 2 pressed")
all_leds_off()
GPIO.output(green_LED, GPIO.HIGH)

# Set up event detection
GPIO.add_event_detect(button_one, GPIO.FALLING,              callback=button_one_p
GPIO.add_event_detect(button_two, GPIO.FALLING, callback=button_two_pressed, bounc

try:
print("Program running. Press buttons to control LEDs.")
while True:
    all_leds_off()
    time.sleep(0.1)

except KeyboardInterrupt:
print("Exiting program.")
GPIO.cleanup()
```

Share Edit Flag

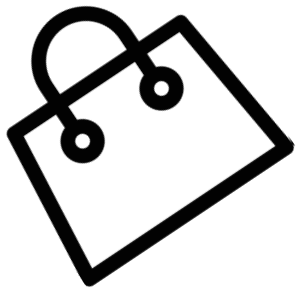
answered just now
Aya kharchafi
1

New contributor

falsifiable?
Is the Doctor's number ever "Doctor Who"?
In a world with magic that c fireballs cast from a persons off a fireball underwater cre
What were the SPA/EPA (Sta Area) control characters use
Are usernames unique in PT
How should I summarize a integral that motivated my plagiarizing or being accuse
Climbing through the moun
Is South Korea's opposition relations with North Korea?
In case of a Presidential Parc are newly discovered crimes
Integral of a 1-form over a s
PSE Advent Calendar 2024 (Declan go?" - a GCHQ word (Christmas edition)
Why can't my biopunk natic biotechnology be reversed-
Is the southern hemisphere northern one or is it just me
Why would a company do a split and then immediately
How to Create Rounded Ge Wave Angles
Question feed

11

Why Should You Buy Our Product?



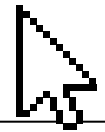
Why Should You Buy Our Product?

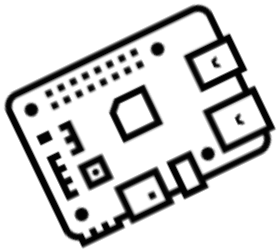
Portability

- **Lightweight and Compact**: Designed for effortless wear during outdoor activities.
- **Custom Harness**: Resizable and stable for hours of comfortable use, even during dynamic activities like hiking or biking.
- **Portable Power Source**: Operates without needing to be tethered to an outlet.

Ease of Use

- **Simple Controls**:
  - Two tactile buttons for capturing photos and recording videos
  - Clear, intuitive functionality—no learning curve required.
- **Accessible Media**: View and manage your pictures and videos directly through a user-friendly web interface.





Thanks!

Do you have any questions?

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**