Lydia Ayala Hernandez 2264230

Journal #01

Weekly Journal Report

Expected Tasks for this week:

Week #1 (Sunday 10 - Friday 15) :

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

Monday 11th: Connecting the camera module and Testing the camera's functionalities

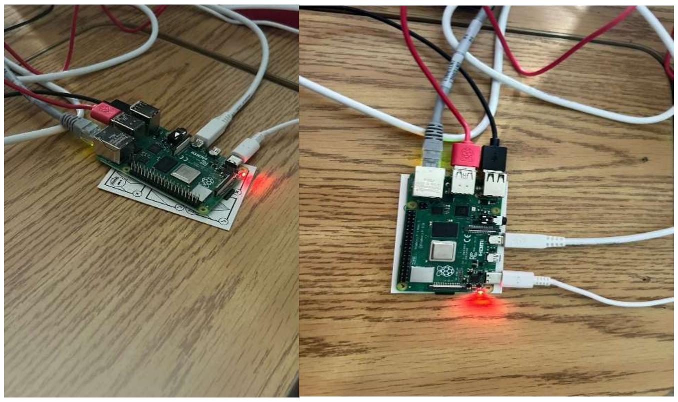
TEAMMATE:

- Installed the Raspberry Pi Imager
- Found a micro SD card and used it to install the OS
- Installed the OS, which will be used to make the Raspberry Pi Work

AT SCHOOL:

Connect and turn on the Raspberry Pi:

- To turn on the Pi, we used a lab room and plugged the **ethernet cable, mouse, keyboard and HDMI cable into the Raspberry Pi and the computer** (using a monitor to navigate the OS and run/install softwares, using the command line).
- We made the **fatal error** of **connecting the camera module after turning the Raspberry Pi on**, which we later found out that it can it can damage the camera permanently. Fortunately, we did not ruin the camera module and the camera worked as it should.
- Insert the flex cable of the camera module into the CSI camera port (next to the HDMI ports)



Turning on the Raspberry Pi and connecting all the necessary cables except the camera module (mistake)

<u>Commands to activate the camera by configuring it:</u>

- In the terminal: **sudo raspi-config**
  - Used to configure the Raspberry Pi. Advanced features are accessed using the terminal, however, you can also **access its configuration graphically** (GUI) but without the advanced features.
- Select: **Interface options > Camera** and enable it
  - **Interface option:** Configure connections to peripherals
- **Reboot** the Raspberry Pi to apply to enable the changed feature

<u>Commands to take pictures and videos:</u>

- Take a picture (still image): **raspistill -o image1.jpg**
- Take a 10-second video: **raspivid -o video1.h264 -t 10000**

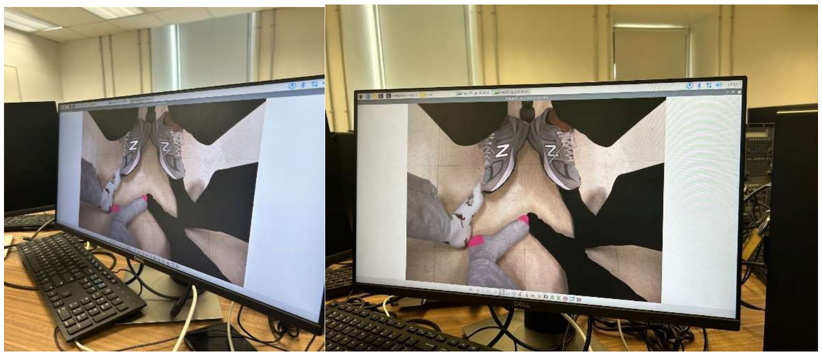
- **-o**: output

- **video1.h264**: specifies the filename where the video will be stored and the format of the video, which in this case is going to be a compressed video file without any audio

- **-t:** specifies the duration of the video in milliseconds

- **10000**: 10000 milliseconds which will record a 10-second-long video

- After running the command, the Raspberry takes a picture and the jpg file will be situated in the file explorer with the filename "image1"
- Our camera module works and can take videos and pictures



First time testing the camera module: Very impressed by the quality ("high quality 8 megapixel Sony IMX219")

Tuesday 12th: Searching/Picking the web server and the gallery software

TEAMMATE:

- Researched about raspberry pi buttons and connected them to the breadboard
- 3 buttons:1- taking pictures (1-click) / 2- recording (press once and it holds) / 3- On/Off button(1-click)

HOME:

- While my teammate was researching the physical components, I was researching the possible servers that could be compatible with our Raspberry Pi and our project description. Since we did a pretty deep research about this during our project proposal, the process of picking the right softwares was fast and we ended up picking Nginx.

- For the gallery software we ended choosing PiGallery2 for it's features an compatibility. Since it is a Node.js application, we will also need to install Node.js (version 16)

Links that we've viewed before picking the right one (Chat gpt also helped)

These articles basically compare different webservers but mostly apache2 and nginx which exceeds more in some cases. They mention that it really all depends on your personal project so I asked chat gpt about it considering our project:

- https://www.tomshardware.com/news/raspberry-pi-web-server,40174.html
    - How to install web servers in the Raspberry Pi: We would finally have an idea of the process and the idea of having a running web server
- https://www.instructables.com/Turning-your-Raspberry-Pi-into-a-personal-web-serv/
    - More details about running a web server in your Raspberry Pi.
- https://www.hostinger.com/tutorials/nginx-vs-apache-what-to-use/#:~:text=NGINX%20has%20an%20event%2Ddriven,and%20high%2Dperformance%20than%20Apache.
    - Differences in performance and size of 2 different common web servers
- https://www.quora.com/Is-Nginx-the-best-web-server-for-a-Raspberry-Pi-given-that-it-has-a-smaller-footprint-than-Apache
    - A lot of articles bring up the fact that Nginx is better but than we found some forums where people would say that it all depends on what you need it for, what are the requirements that we need, etc.
- https://www.digitalocean.com/community/tutorials/apache-vs-nginx-practical-considerations
    - I remember hearing about DigitalOcean in our lecture so i decided to skim through the article and it explained perfectly the differences between nginx and Apache and what we should consider in their performances.

Reverse proxy or Web server

- Since PiGallery2 will act as the picture and video manager, Chat GPT has requested to make Nginx a reverse proxy.
- Nginx web server: if we make Nginx directly as a web server, it can either enhance or replace PiGallery2's functionalities.
- Nginx verse proxy: Will forward a request to the PiGallery which will basically handle the interactions between the user and application (viewing and editing). This option will enhance the security and scalability. We will use this version and create a reverse proxy

Additional sources for the reverse proxy (installation commands and their explanation):

- https://engineerworkshop.com/blog/setup-an-nginx-reverse-proxy-on-a-raspberry-pi-or-any-other-debian-os/
- https://nginx.org/en/
- https://medium.com/globant/understanding-nginx-as-a-reverse-proxy-564f76e856b2

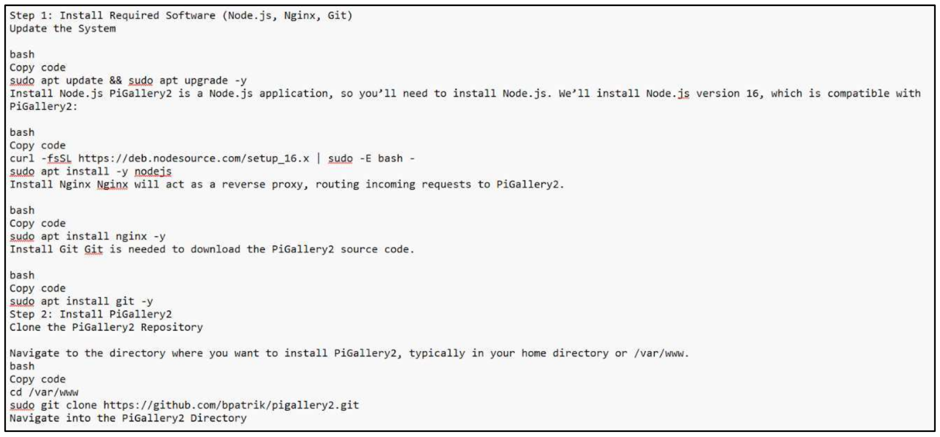
Installing PiGallery2 sources (installation):

- https://github.com/bpatrik/pigallery2

Editing images from the command line:

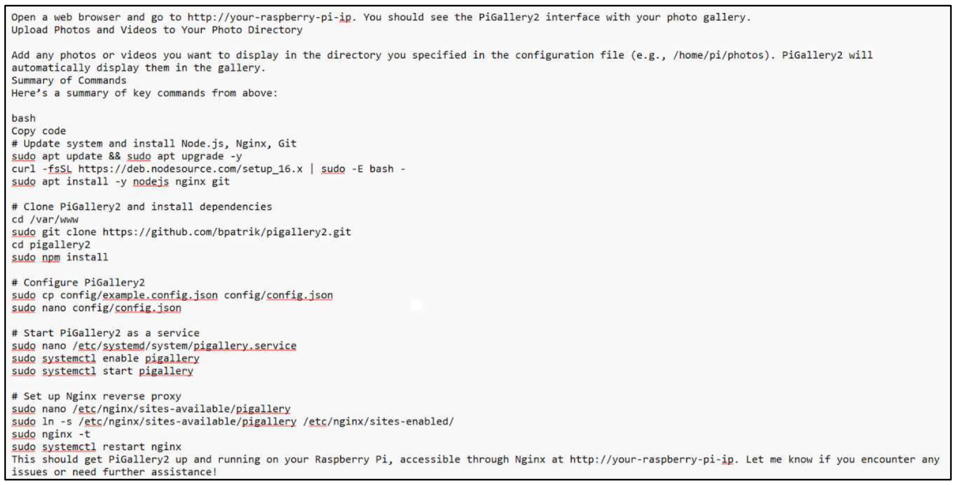
- On the project proposal we chose to edit our images on the command line using ImageMagick
    - https://www.theurbanpenguin.com/image-manipulation-on-the-raspberry-pi-using-imagemagick/
    - https://imagemagick.org/index.php

Prepared the commands using the links and Chat GPT:



```
Step 1: Install Required Software (Node.js, Nginx, Git)
Update the System

bash
Copy code
sudo apt update && sudo apt upgrade -y
Install Node.js PiGallery2 is a Node.js application, so you’ll need to install Node.js. We’ll install Node.js version 16, which is compatible with
PiGallery2:

bash
Copy code
curl -fsSL https://deb.nodesource.com/setup_16.x | sudo -E bash -
sudo apt install -y nodejs
Install Nginx Nginx will act as a reverse proxy, routing incoming requests to PiGallery2.

bash
Copy code
sudo apt install nginx -y
Install Git Git is needed to download the PiGallery2 source code.

bash
Copy code
sudo apt install git -y
Step 2: Install PiGallery2
Clone the PiGallery2 Repository

Navigate to the directory where you want to install PiGallery2, typically in your home directory or /var/www.
bash
Copy code
cd /var/www
sudo git clone https://github.com/bpatrik/pigallery2.git
Navigate into the PiGallery2 Directory
```

*Code snippet from Chat GPT **Beginning of the code***



```
Open a web browser and go to http://your-raspberry-pi-ip. You should see the PiGallery2 interface with your photo gallery.
Upload Photos and Videos to Your Photo Directory

Add any photos or videos you want to display in the directory you specified in the configuration file (e.g., /home/pi/photos). PiGallery2 will
automatically display them in the gallery.
Summary of Commands
Here’s a summary of key commands from above:

bash
Copy code
# Update system and install Node.js, Nginx, Git
sudo apt update && sudo apt upgrade -y
curl -fsSL https://deb.nodesource.com/setup_16.x | sudo -E bash -
sudo apt install -y nodejs nginx git

# Clone PiGallery2 and install dependencies
cd /var/www
sudo git clone https://github.com/bpatrik/pigallery2.git
cd pigallery2
sudo npm install

# Configure PiGallery2
sudo cp config/example.config.json config/config.json
sudo nano config/config.json

# Start PiGallery2 as a service
sudo nano /etc/systemd/system/pigallery.service
sudo systemctl enable pigallery
sudo systemctl start pigallery

# Set up Nginx reverse proxy
sudo nano /etc/nginx/sites-available/pigallery
sudo ln -s /etc/nginx/sites-available/pigallery /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl restart nginx
This should get PiGallery2 up and running on your Raspberry Pi, accessible through Nginx at http://your-raspberry-pi-ip. Let me know if you encounter any
issues or need further assistance!
```

*Code snippet from Chat GPT **End of the code***

Wednesday 13th: Button Integration

LAB WITH MY TEAMMATE:

- we continued to work with the buttons and we were prepared to finish the scripts and begin the installation of the reverse proxy and softwares.

Setting Up a New Micro SD Card:

- My teammate prepared another SD card and installed the OS to turn on the Raspberry Pi

Re-enable the camera interface on this new card:

- **sudo raspi-config**
- **Interface options > No camera option available**
- The system was not detecting the camera, even after checking that everything was perfectly connected.

Trying to find the issue by troubleshooting

- Rebooted the Raspberry Pi multiple times to see if the camera option would finally appear.
- Running the commands: sudo apt update AND sudo apt upgrade
- Ran diagnostic commands in the terminal to check for device recognition: **vcgencmd get_camera**

  Problem Outcome
- Weirdly enough, the system could not recognize the camera so we believed that somehow we broke it, but it could still take pictures and videos and store them in a file. This problem took us hours and we still do not understand what could have happened, we believed that it could be the new micro SD since the old one worked
- Fortunately: When we finish it, we will already have the scripts and other installation commands ready so we can rapidly catch up

---

Thursday 14th: Continuation of Wednesday's research

Raspberry Pi buttons:

- Due to exams and project deliveries that were due for Friday, we decided to postpone the button installation for next week.

Research

- I continued to research a bit more on the installations and with the help of external sources and AI.

Team meeting:

- We tried to organize ourselves, for next week, to finalize any physical installation since we are late to the installation of the buttons. The camera detection problem was not supposed to happen but we need to allocate time for unexpected events

Sunday 17th: Updating the git

TEAMMATE (on saturday):

- My teammate created the git and sent me an invite.
- Created the READ ME and the License (MIT License)

HOME:

- After finalizing the journal along with my teammate, i added the changes that we did on the README.md file:
    - Timeline: Since we had some complications: the button installation will get pushed back to next week
    - We finalized our webserver/reverse proxy and the softwares that we will be installing
    - Arranged the structure since there were some repetitions that could be grouped up
    - Additional modifications that we could do if we finish in advance
        - We thought of adding a screen since it could also come in a case, so we wouldn't need to create one with a 3D printer, but these are just possibilities.

Lydia Ayala Hernandez 2264230

Final results from this week:

Week #1 (Sunday 10 – Sunday 17) :

- Connect the camera to the Raspberry Pi
- Start the web server
- Installed the necessary connectors to the breadboard and the buttons
- Created the GitHub repository

Github project repository:

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