Pi SSD Media Server
Created by Ruiz Brothers

https://learn.adafruit.com/pi-ssd-media-server

Last updated on 2023-08-29 04:50:41 PM EDT
# Table of Contents

## Overview
- SSD Media Server
- 3D Printed Case
- TFT Display
- Extra Features
- SSD Attachment
- Prerequisite Guides
- Parts

## Circuit Diagram
- Adafruit Library for Fritzing

## CAD Files
- CAD Parts List
- Build Volume
- Design Source Files

## Software Setup
- Choose your OS
- USB Boot with SSD
- Install OS with Raspberry Pi Imager
- Install CircuitPython Libraries
- TFT Display Info
- TFT Display GIF images
- Autostart TFT python scripts
- Enable Safe Shutdown
- Install Plex Media Server
- Autostart Chromium in Full Screen Kiosk Mode

## Wiring
- Pi Hat
- Wiring TFT
- Wiring Power Button
- Wiring Fan
- Wiring Pi Hat
- Wired Pi Hat

## Assembly
- Install Pi Standoffs
- Add Heat Inserts to Case
- Install Handle
- Secured Handle
- Button Plate Install
- Button Plate Secure
- Installed Buttons
- Install Pi
- Secure Pi to Case
- Install microSD card extender
- Secure Power Button
- Hardware for TFT Display
- Install TFT Display
- Secured TFT Display
• Install Pi Hat
• Connect LED for Power Button
• Connect Power Button
• Connect Top Buttons
• Hardware for Fan
• Secure Fan to Panel
• Connect Fan
• Secure Panel to Case
• Hardware for SSD mount
• Secure SSD Panel to Case
• Secure SSD to mount
• Connect SSD to USB
Overview

Build your own media server using Raspberry Pi 4 and an external SSD. Stream live TV and movies with Plex using the Chromium web browser, or install any other network drive / streaming software to run on this all-in-one lunchbox sized Pi-based server

SSD Media Server

Use an SSD to store all of your media and get blazing fast read and write speeds using a SATA to USB 3.0 cable.

Learn how to create a bootable USB drive for your Raspberry Pi.

3D Printed Case

Build a PC tower inspired 3D printed case. Secure the Pi and extra accessories such as a 1.9in TFT display, 5V fan, 2x buttons and a power switch with an LED.

Side panels attach to the frame making this a modular design for easy access. The case features a handle with a print-in-place hinge for transport.

TFT Display

Use the 1.9in color TFT screen to display your media servers IP address, CPU temperature or memory/disk usage using CircuitPython libraries for Raspberry Pi OS.

Also play looping GIF images on the display and cycle through them using the built-in buttons.
Extra Features
Customize the side panels to add more features and accessories.

A 5V fan can be mounted to one side of the panels to keep the Raspberry Pi nice and cool.

A 16mm metal pushbutton with built-in LED is mounted to the back side of the case, right above the microSD card slot.

SSD Attachment
Secure the 2.5in SSD with standard mounting holes to the 3D printed bracket.

Attach the SSD bracket to the side panel for a secure and reliable external drive.
Prerequisite Guides

Take a moment to browse through the following guides.

- How to make bootable Raspberry Pi from USB SSD
- Install CircuitPython on Raspberry Pi
- 1.9in TFT display python usage
- GIF player on TFT displays with Raspberry Pi
- Heat-set rig for Soldering Irons

Parts

Raspberry Pi 4 Model B - 2 GB RAM
The Raspberry Pi 4 Model B is the newest Raspberry Pi computer made, and the Pi Foundation knows you can always make a good thing
https://www.adafruit.com/product/4292
Adafruit 1.9" 320x170 Color IPS TFT Display
This lovely little display breakout is the best way to add a small, colorful, and very bright display to any project. Since the display uses 4-wire SPI to communicate and has its own...
https://www.adafruit.com/product/5394

STEMMA Wired Tactile Push-Button Pack - 5 Color Pack
Little clicky switches are standard input "buttons" on electronic projects. These are just like our Colorful Round...
https://www.adafruit.com/product/4431

Micro SD Card PCB Extender
Stop fiddling with micro SD card sockets in the back of your latest gadget during development, and give yourself a little more elbow room with this Small Micro SD...
https://www.adafruit.com/product/4395

Miniature 5V Cooling Fan with Molex PicoBlade Connector
Looking for another way to keep your project cool? Hook up this 5V Mini Cooling Fan and prevent from overheating! Of course, it's also...
https://www.adafruit.com/product/4468
<table>
<thead>
<tr>
<th>Quantity</th>
<th>Item Description</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 x</td>
<td>Pimoroni Nano HAT Hacker</td>
<td><a href="https://www.adafruit.com/product/2876">https://www.adafruit.com/product/2876</a></td>
</tr>
<tr>
<td></td>
<td>Raspberry Pi Accessory</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>GPIO Header for Raspberry Pi</td>
<td><a href="https://www.adafruit.com/product/3577">https://www.adafruit.com/product/3577</a></td>
</tr>
<tr>
<td></td>
<td>2x20 female headers</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>16mm Metal Pushbutton with LED</td>
<td><a href="https://www.adafruit.com/product/559">https://www.adafruit.com/product/559</a></td>
</tr>
<tr>
<td></td>
<td>Momentary switch with red LED</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>16GB microSD Card</td>
<td><a href="https://www.adafruit.com/product/2693">https://www.adafruit.com/product/2693</a></td>
</tr>
<tr>
<td></td>
<td>Memory card for Pi</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>SATA to USB 3 cable</td>
<td><a href="https://www.adafruit.com/product/5622">https://www.adafruit.com/product/5622</a></td>
</tr>
<tr>
<td></td>
<td>Hard Driver Adapter</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>2.5in SATA SSD Drive</td>
<td><a href="https://www.amazon.com/SAMSUNG-Inch-Internal-MZ-77E1T0B-AM/dp/B08QBJ2YMG/">https://www.amazon.com/SAMSUNG-Inch-Internal-MZ-77E1T0B-AM/dp/B08QBJ2YMG/</a></td>
</tr>
<tr>
<td></td>
<td>Samsung 1TB SSD drive</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>10-wire Silicone Cover Ribbon Cable</td>
<td><a href="https://www.adafruit.com/product/3890">https://www.adafruit.com/product/3890</a></td>
</tr>
<tr>
<td></td>
<td>28AWG stranded core wire</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>Quick-Connect Wire Pairs</td>
<td><a href="https://www.adafruit.com/product/1152">https://www.adafruit.com/product/1152</a></td>
</tr>
<tr>
<td></td>
<td>0.11” (10 pack) for Arcade buttons</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>2-pin Cable Molex Connector</td>
<td><a href="https://www.adafruit.com/product/4720">https://www.adafruit.com/product/4720</a></td>
</tr>
<tr>
<td></td>
<td>1.25mm Pitch 40cm long - Molex PicoBlade Compatible</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>M3 Hardware Kit</td>
<td><a href="https://www.adafruit.com/product/4685">https://www.adafruit.com/product/4685</a></td>
</tr>
<tr>
<td></td>
<td>Black Nylon Hardware</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>M2.5 Hardware Kit</td>
<td><a href="https://www.adafruit.com/product/3299">https://www.adafruit.com/product/3299</a></td>
</tr>
<tr>
<td></td>
<td>Black Nylon Hardware</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>M3 Heat-Set Inserts</td>
<td><a href="https://www.adafruit.com/product/4255">https://www.adafruit.com/product/4255</a></td>
</tr>
<tr>
<td></td>
<td>Brass Heat-Set Inserts for Plastic - M3 x 4mm - 50 pack</td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>M3 Heat-Set Tip</td>
<td><a href="https://www.adafruit.com/product/4239">https://www.adafruit.com/product/4239</a></td>
</tr>
<tr>
<td></td>
<td>Heat-Set Insert For Soldering Irons - #4-40 / M3 Inserts</td>
<td></td>
</tr>
</tbody>
</table>
Circuit Diagram

The diagram below provides a visual reference for wiring of the components. This diagram was created using the software package [Fritzing](https://fritzing.org/).

Adafruit Library for Fritzing

Use Adafruit's Fritzing parts library to create circuit diagrams for your projects. Download the library or just grab individual parts. Get the library and parts from [GitHub - Adafruit Fritzing Parts](https://github.com/adafruit/Adafruit_Python_Fritzing).

TFT Display

- Vin from TFT to 3V3 on Pi
- Gnd from TFT to Gnd on Pi
- SCK from TFT to GPIO11 on Pi
- MISO from TFT to GPIO9 on Pi
- MOSI from TFT to GPIO10 on Pi
- TSC from TFT to GPIO8 on Pi
- RST from TFT to GPIO24 on Pi
- DC from TFT to GPIO25 on Pi

Momentary Button Switch with LED

- Anode(+) from LED to GPIO14 on Pi
• Cathode(–) from LED to GND on Pi
• Normally Open pin from switch to GPIO3 and GPIO27 on Pi
• Common pin from switch to GND on Pi

5V Fan

• Red wire from fan to 5V pin on Pi
• Black wire from fan to GND pin on Pi

Momentary Button Switches

• Button 1 to GND and GPIO12 on Pi
• Button 2 to GND and GPIO16 on Pi

CAD Files

CAD Parts List
STL files for 3D printing are oriented to print "as-is" on FDM style machines. Parts are designed to 3D print without any support material. Original design source may be downloaded using the links below:

pi-ssd-button-mount.stl
pi-ssd-case.stl
pi-ssd-fan-panel.stl
pi-ssd-handle.stl
pi-ssd-mount.stl
pi-ssd-port-panel.stl

Download STLs_V2.zip
Download CAD source
Build Volume
The parts require a 3D printer with a minimum build volume.

100mm (X) x 114mm (Y) x 64mm (Z)

Design Source Files
The project assembly was designed in Fusion 360. This can be downloaded in different formats like STEP, STL and more. Electronic components like Adafruit's boards, displays, connectors and more can be downloaded from the Adafruit CAD parts GitHub Repo.

Software Setup

Choose your OS
Take a moment to decide which operating system you'd like to use. Consider the following limitations.

Raspberry Pi OS has support for CircuitPython libraries and works with the 1.9in TFT screen for displaying GIFs and information like the Pi’s IP address, CPU temperature, etc. Ubuntu variants but does not have support for CircuitPython libraries.

USB Boot with SSD
If you’d like to boot your Pi using an external SSD, you can follow along with the guide from Tom’s Hardware. This process utilizes the full storage and features faster.
performance when reading/writing. This can be a good option for running a media server.

Ensure only the USB SSD is connected on the initial boot. Pi OS WILL NOT start if any other USB devices such as keyboard/mouse are connected.

Install OS with Raspberry Pi Imager

Download the Raspberry Pi Imager and choose your preferred OS and method of booting.

If you're using Pi OS, click the gear icon to open advanced options. There you can enable SSH and configure wireless LAN.

Install CircuitPython Libraries

If you'd like to use the 1.9in TFT display, follow the installation guide to get setup.

TFT Display Info

Use the Python script below to display the IP address, CPU temperature and disk/memory usage. Make sure you have the fonts directory installed in the same directory as the Python script.

```python
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import subprocess
import board
import displayio
from adafruit_display_text import label
from adafruit_bitmap_font import bitmap_font
from adafruit_st7789 import ST7789

BORDER_WIDTH = 4
TEXT_SCALE = 1

font = bitmap_font.load_font("/home/pi/fonts/Arial-18.bdf")
font_small = bitmap_font.load_font("/home/pi/fonts/Arial-14.bdf")
font_bold = bitmap_font.load_font("/home/pi/fonts/Arial-Bold-24.bdf")

# Release any resources currently in use for the displays
```
displayio.release_displays()

spi = board.SPI()
tft_cs = board.CE0
tft_dc = board.D25
tft_rst = board.D24

display_bus = displayio.FourWire(spi, command=tft_dc, chip_select=tft_cs,
reset=tft_rst)

display = ST7789(display_bus, width=320, height=170, colstart=35, rotation=270)

# Make the display context
splash = displayio.Group()
display.show(splash)

color_bitmap = displayio.Bitmap(display.width, display.height, 1)
color_palette = displayio.Palette(1)
color_palette[0] = 0xAA0088  # Purple
bg_sprite = displayio.TileGrid(color_bitmap, pixel_shader=color_palette, x=0, y=0)
splash.append(bg_sprite)

# Draw a smaller inner rectangle
inner_bitmap = displayio.Bitmap(display.width - (BORDER_WIDTH * 2), display.height - (BORDER_WIDTH * 2), 1
)
inner_palette = displayio.Palette(1)
inner_palette[0] = 0x222222  # Dark Gray
inner_sprite = displayio.TileGrid(
    inner_bitmap, pixel_shader=inner_palette, x=BORDER_WIDTH, y=BORDER_WIDTH
)
splash.append(inner_sprite)

# Display IP, CPU and memory usage
cmd = "hostname -I | cut -d' ' -f1"
IP = subprocess.check_output(cmd, shell=True).decode("utf-8")

# Display CPU
CPU = subprocess.check_output(cmd, shell=True).decode("utf-8")

# Display Memory Usage
MemUsage = subprocess.check_output(cmd, shell=True).decode("utf-8")
cpu_temp = subprocess.check_output(cmd, shell=True).decode("utf-8")

text_ip = label.Label(
    font_bold,
    text="IP: " + IP,
    color=0x1BF702,
    scale=TEXT_SCALE,
)
text_cpu = label.Label(
    font,
    text="CPU: " + cpu_temp,
    color=0xFFFFFF,
    scale=TEXT_SCALE,
)
text_mem = label.Label(
    font_small,
    text=MemUsage,
    color=0xCCCCCC,
    scale=TEXT_SCALE,
)
text_ip.x = 12
text_cpu.x = 12
text_mem.x = 12

text_ip.y = 30
text_cpu.y = 70
text_mem.y = 150
splash.append(text_ip)
splash.append(text_cpu)
splash.append(text_mem)

while True:
    text_ip.text = "IP: " + IP
    text_cpu.text = "CPU: " + cpu_temp
    text_mem.text = MemUsage
    display.refresh()

Download and copy the Python file to the home folder on your Raspberry Pi. Use the command in terminal to run the Python script: `sudo nano python3 display-info.py`

**TFT Display GIF images**

Use the Python script below to display GIF images on the TFT display. You can use buttons to cycle through the images. Make sure you have the GIF images in a directory named GIFs. Ensure the python script is in a directory just outside the GIFs directory.

```python
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import os
import time
import digitalio
import board
from PIL import Image, ImageOps
import numpy  # pylint: disable=unused-import
from adafruit_rgb_display import st7789  # pylint: disable=unused-import

# Change to match your display
BUTTON_NEXT = board.D17
BUTTON_PREVIOUS = board.D22

# Configuration for CS and DC pins (these are PiTFT defaults):
cs_pin = digitalio.DigitalInOut(board.CE0)
dc_pin = digitalio.DigitalInOut(board.D25)
reset_pin = digitalio.DigitalInOut(board.D24)

def init_button(pin):
    button = digitalio.DigitalInOut(pin)
    button.switch_to_input()
    button.pull = digitalio.Pull.UP
    return button

# pylint: disable=too-few-public-methods
class Frame:
    def __init__(self, duration=0):
        self.duration = duration
        self.image = None

# pylint: enable=too-few-public-methods
class AnimatedGif:
    def __init__(self, display, width=None, height=None, folder=None):
        self._frame_count = 0
        self._loop = 0

        # Change to match your display
        BUTTON_NEXT = board.D17
        BUTTON_PREVIOUS = board.D22

        # Configuration for CS and DC pins (these are PiTFT defaults):
cs_pin = digitalio.DigitalInOut(board.CE0)
dc_pin = digitalio.DigitalInOut(board.D25)
reset_pin = digitalio.DigitalInOut(board.D24)

def init_button(pin):
    button = digitalio.DigitalInOut(pin)
    button.switch_to_input()
    button.pull = digitalio.Pull.UP
    return button

# pylint: disable=too-few-public-methods
class Frame:
    def __init__(self, duration=0):
        self.duration = duration
        self.image = None

# pylint: enable=too-few-public-methods
class AnimatedGif:
    def __init__(self, display, width=None, height=None, folder=None):
        self._frame_count = 0
        self._loop = 0
```

© Adafruit Industries
self._index = 0
self._duration = 0
self._gif_files = []
self._frames = []

if width is not None:
    self._width = width
else:
    self._width = display.width
if height is not None:
    self._height = height
else:
    self._height = display.height
self.display = display
self.advance_button = init_button(BUTTON_NEXT)
self.back_button = init_button(BUTTON_PREVIOUS)
if folder is not None:
    self.load_files(folder)
self.run()

def advance(self):
    self._index = (self._index + 1) % len(self._gif_files)

def back(self):
    self._index = (self._index - 1 + len(self._gif_files)) % len(self._gif_files)

def load_files(self, folder):
    gif_files = [f for f in os.listdir(folder) if f.endswith(".gif")]
    for gif_file in gif_files:
        gif_file = os.path.join(folder, gif_file)
        image = Image.open(gif_file)
        # Only add animated Gifs
        if image.is_animated:
            self._gif_files.append(gif_file)

print("Found", self._gif_files)
if not self._gif_files:
    print("No Gif files found in current folder")
    exit()  # pylint: disable=consider-using-sys-exit

def preload(self):
    image = Image.open(self._gif_files[self._index])
    print("Loading {}...".format(self._gif_files[self._index]))
    if "duration" in image.info:
        self._duration = image.info["duration"]
    else:
        self._duration = 0
    if "loop" in image.info:
        self._loop = image.info["loop"]
    else:
        self._loop = 1
    self._frame_count = image.n_frames
    self._frames.clear()
    for frame in range(self._frame_count):
        image.seek(frame)
        frame_object = Frame(duration=self._duration)
        if "duration" in image.info:
            frame_object.duration = image.info["duration"]
        frame_object.image = ImageOps.pad(  # pylint: disable=no-member
            image.convert("RGB"),
            (self._width, self._height),
            method=Image.Resampling.NEAREST,
            color=(0, 0, 0),
            centering=(0.5, 0.5),
        )
    self._frames.append(frame_object)
def play(self):
    self.preload()

    _prev_advance_btn_val = self.advance_button.value
    _prev_back_btn_val = self.back_button.value
    if not self._gif_files:
        print("There are no Gif Images loaded to Play")
        return False
    while True:
        for frame_object in self._frames:
            start_time = time.monotonic()
            self.display.image(frame_object.image)
            _cur_advance_btn_val = self.advance_button.value
            _cur_back_btn_val = self.back_button.value
            if not _cur_advance_btn_val and _prev_advance_btn_val:
                self.advance()
                return False
            if not _cur_back_btn_val and _prev_back_btn_val:
                self.back()
                return False
            _prev_back_btn_val = _cur_back_btn_val
            _prev_advance_btn_val = _cur_advance_btn_val
            while time.monotonic() < (start_time + frame_object.duration / 1000):
                pass
            if self._loop == 1:
                return True
            if self._loop > 0:
                self._loop -= 1
    def run(self):
        while True:
            auto_advance = self.play()
            if auto_advance:
                self.advance()

    # Config for display baudrate (default max is 64mhz):
    BAUDRATE = 64000000
    # Setup SPI bus using hardware SPI:
    spi = board.SPI()
    disp = st7789.ST7789(
        spi,
        rotation=270,
        width=170,
        height=320,
        x_offset=35,
        cs=cs_pin,
        dc=dc_pin,
        rst=reset_pin,
        baudrate=BAUDRATE,
    )
    # pylint: enable=line-too-long
    if disp.rotation % 180 == 90:
        disp_height = disp.width  # we swap height/width to rotate it to landscape!
        disp_width = disp.height
    else:
        disp_width = disp.width
        disp_height = disp.height
    gif_player = AnimatedGif(disp, width=disp_width, height=disp_height, folder="gifs")
Download and copy the Python file to the home folder on your Raspberry Pi. Use the command in terminal to run the Python script: `sudo nano python3 gif-player.py`

**Autostart TFT python scripts**

Create an autostart script to enable the TFT display automatically on boot. Start by creating a new file with this command.

```
sudo nano /lib/systemd/system/tftdisplay.service
```

Copy the text below and paste it into the new file. Check the .py file is in the correct directory. You may need to modify the path if you've changed the Pi's hostname or title of Python script.

```
[Unit]
Description=TFT Display Service
After=multi-user.target

[Service]
Type=idle
ExecStart=/usr/bin/python /home/pi/tft-display.py

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

Save this file and exit the editor. A few commands then enable this new service and will launch it on startup:

```
sudo systemctl daemon-reload
sudo systemctl enable tftdisplay.service
sudo reboot
```

To change which program is loaded on startup, just edit `/lib/systemd/system/tftdisplay.service` and change the `ExecStart` line, and reboot. No need to repeat the systemctl commands.

**Enable Safe Shutdown**

You can use the metal push button to safely shutdown your Pi. Add a line to the config.txt file using the `sudo nano /boot/config.txt` command in terminal.

In the circuit diagram, the metal push button is wired to both GPIO3 and GPIO27. By default, ground and GPIO3 will power up the Pi. GPIO27 is used to safely shutdown the Pi.

```
dtoverlay=gpio-shutdown,gpio_pin=27,active_low=0
```
Save the config.txt file and exit the editor. Reboot the Raspberry Pi and the pressing the power button should run the safe shutdown service.

**Install Plex Media Server**

If you’d like to use the Pi as a media center, you can install Plex on the Raspberry Pi. This will use the Chromium web browser to display the Plex interface where you can play and stream media.

Head over to the [plex website](https://www.plex.tv) and choose Linux as the platform and then click Choose Distribution. Select the Ubuntu ARM (v7 for 32-bit Pi OS) and download.

Once downloaded, double-click the file to install the software. When complete, Head over to localhost:32400/web/ where you should see the Plex log in screen. There you can configure your settings and add media.

Follow along with the [Basic Setup Wizard guide](https://plex.tv) from Plex for more documentation on configuring your settings.

**Autostart Chromium in Full Screen Kiosk Mode**

You can launch Chromium in full screen mode by adding a line to the autostart file on the LXDE desktop. Run the following command in terminal to edit the configuration file.

```
sudo nano /etc/xdg/lxsession/LXDE-pi/autostart
```

Modify the autostart file by adding the line below and save with key combination Ctrl+X then Y.

```
@lxpanel --profile LXDE-pi
@pcmanfm --desktop --profile LXDE-pi
@xscreensaver -no-splash
/usr/bin/chromium-browser --kiosk --disable-restore-session-state http://localhost:32400/web/
```
Reboot the Raspberry Pi to test the updated autostart file. Chromium will automatically open and display the Plex interface by loading the [http://localhost:32400/web/](http://localhost:32400/web/) website.

---

**Wiring**

---

**Pi Hat**
The 2x20 GPIO header will be soldered to the Pi Nano hacker hat. Place the header into one of the set of pins like shown in the photo.

Solder all 40 pins in place and remove the excess pins using flush snips.

---

**Wiring TFT**
The 1.9in TFT will need 8 wired connections to the Raspberry Pi. Use a piece of ribbon cable to make the connections.

Measure and cut a piece of ribbon cable so it's about 4in(10cm) in length. Solder the individual wires to 8 of the 11 pins on the TFT display.
Wiring Power Button
Connect two sets of quick connect cables to the four leads on the power button.

Connect one cable set to the LED, labeled + and – on the side of the button switch.

Connect the second cable set to the switch, labeled C(common) and NO(normally open)

Wiring Fan
The wires from the 5V fan can either be soldered directly to the pins on the Pi hat or use a set of connecting JST cables.

This 3-wire fan uses a 3-pin molex connector and has been shortened to better fit inside the case.

Wiring Pi Hat
Solder all of the cables and wires to the various pins on the Pi hat.

Cables for the two buttons.
Cables for the power button and LED.
Wires for the 5V fan (optional cable)
Wires for the 1.9in TFT display
Wired Pi Hat
Double check all of the wires and cables have been soldered to the correct pins on the Pi Hat.

Assembly

Install Pi Standoffs
Use the following hardware for mounting the Raspberry Pi to the 3D printed case.

- Four M2.5 x 10mm long standoffs
- Four M2.5 x 6mm long screws

M3 sized hardware can be used but mounting holes on the Pi will need to be tapped in order to fit properly.
Add Heat Inserts to Case
Install 8x M3 inserts to the case. Inserts are heat-set into place using the tip of a soldering iron.

Check out our learn guide for creating a heat-set rig to make this process more precise and easier.

Install Handle
Secure the handle to the 3D printed case using the following hardware.

Four M3 x 6mm long screws
Four M3 hex nuts
Place the tabs from the handle over the mounting holes on the top side of the case. Insert screws and fasten hex nuts while holding in place.
Secured Handle
The handle features a hinge that is printed in place and allows it to be rotated 180 degrees.
Button Plate Install

Use the following hardware to secure the buttons to the 3D printed plate.

- Four M2.5 x 10mm standoffs
- Eight M2.5 x 6mm long screws
- Four M2.5 hex nuts

Secure the two buttons to the plate using Four M2.5 x 6mm screws and hex nuts.
Button Plate Secure
Install and fasten the standoffs to the button plate in the correct orientation.

Place the button plate inside the case with the button caps properly fitted through the cutouts.

Secure the button plate to the 3D printed plate using the remaining M2.5 screws.

Installed Buttons
Press the buttons to verify correct placement.
Install Pi
Get the Raspberry Pi ready to install into the 3D printed case.

Optionally use thumbscrews to double as feet risers for the Raspberry Pi.

Fit the Raspberry Pi into the case with the various ports fitting through the cutouts.

Secure Pi to Case
Insert and fasten the hardware to secure the standoffs from the Raspberry Pi to the case.
Install microSD card extender
Carefully insert the micro SD card extender to the slot in the Raspberry Pi.

Secure Power Button
Insert the power button through the hole on the side of the case with cables going in first.
Slip the hex nut over the cables and fasten onto the threading of the button.
Fit the hex nut into the recess inside the case and tighten the power button to secure in place.
Hardware for TFT Display
Use the following hardware to secure the TFT display to the 3D printed case.

Four M2.5 x 10mm long screws
Four M2.5 hex nuts

Install TFT Display
Orient the TFT display with the case and match the photo for correct placement.

Insert screws through the case and mounting tabs. Use hex nuts to secure the TFT display to the 3D printed case.
Secured TFT Display  
Double check the protective film has been removed from the TFT display that it has been properly secured to the 3D printed case.

Install Pi Hat  
Carefully press fit the Pi hat onto the GPIO headers of the Raspberry Pi.

Ensure the all of the pins are correctly fitting through the Pi hat headers.

Connect LED for Power Button  
Plug in the cable from the LED of the power button to the corresponding cable on the Pi hat.
Connect Power Button
Plug in the cable from the power button to the corresponding cable on the Pi hat.

Connect Top Buttons
Plug in the cable from the corresponding cables on the Pi hat to the JST ports on the two buttons.

Hardware for Fan
Use the following hardware to secure the 5V fan to the 3D printed side panel.

Four M3 x 16mm long screws
Four M3 hex nuts
Secure Fan to Panel
Place the 5V fan over the side panel and line up the mounting holes. Insert and fasten hex nuts to secure fan to the side panel.

Connect Fan
Plug in the corresponding cable from the Pi hat to the 5V fan's cable.

Secure Panel to Case
Place the side panel over the mounting holes on the case.

Insert and fasten four M3 x 6mm long screws to secure the panel to the case.
Hardware for SSD mount
Use the following hardware to attach the SSD mount to the 3D printed case.

Four M3 x 6mm long standoffs
Eight M3 x 6mm long screws
Secure the standoffs to the mounting holes on the SSD mount. Then, secure the SSD mount to the SSD side panel.

Secure SSD Panel to Case
Place the panel over the mounting holes on the 3D printed case.

Secure the panel to the case using 4x M3 x 6mm long screws.
Secure SSD to mount
Orient the SSD with the connector facing the ports on the Raspberry Pi.

Slide the SSD into the mount and line up the tabs with the mounting holes on the side.

Secure the SSD to the mount using 4x M3 x 6mm long screws.

Connect SSD to USB
Plug in the SATA to USB cable from the SSD to one of the USB 3 (blue) ports on the Raspberry Pi.

If you plug the SSD into one of the USB 2 ports (the ones without the blue color), the project may work but you’re not transferring data at USB 3 speeds. Use a blue USB 3 port for best results.