Here's a simplified, step-by-step guide to building a personal firewall using Python:

Step 1: Install Required Tools

- Install Python (if not already installed)
- Install scapy using pip: pip install scapy
- Install Tkinter (for GUI, if desired): sudo apt-get install python3-tk
- Install iptables (on Linux): sudo apt-get install iptables

Step 2: Understand Scapy

- Scapy is a powerful packet sniffer and manipulator
- It allows us to capture, analyze, and modify network packets

Step 3: Define Rule Sets

- Create a dictionary to store rules:

```
rules = {
    'allow': ['192.168.1.100', '80'],  # allow incoming traffic from IP 192.168.1.100 on port 80
    'block': ['192.168.1.200', '22']  # block incoming traffic from IP 192.168.1.200 on port 22
}
```

Step 4: Sniff Packets using Scapy

- Use scapy to sniff incoming and outgoing packets:

```
from scapy.all import sniff, IP, TCP

def packet_sniffer(packet):
    if packet.haslayer(IP) and packet.haslayer(TCP):
        # process packet
        pass
```

```
sniff(prn=packet_sniffer)
```

Step 5: Process Packets

- Check if packet matches any rule:

```
def process_packet(packet):
    src_ip = packet[IP].src
    dst_port = packet[TCP].dport
    for rule in rules['allow']:
        if src_ip == rule[0] and dst_port == int(rule[1]):
            # allow packet
            return
    for rule in rules['block']:
        if src_ip == rule[0] and dst_port == int(rule[1]):
            # block packet
            return
```

Step 6: Log Suspicious Packets

- Log packets that don't match any rule:

```
import logging

logging.basicConfig(filename='firewall.log', level=logging.INFO)

def log_packet(packet):
    logging.info(f"Suspicious packet: {packet.summary()}")
```

Step 7: Enforce Rules using Iptables

- Use iptables to block or allow traffic:

```
import subprocess

def enforce_rule(rule):
    if rule['action'] == 'allow':
        subprocess.run(['iptables', '-A', 'INPUT', '-s', rule['src_ip'], '-p', 'tcp', '--dport', rule['dst_port'], '-j', 'ACCEPT'])
    elif rule['action'] == 'block':
        subprocess.run(['iptables', '-A', 'INPUT', '-s', rule['src_ip'], '-p', 'tcp', '--dport', rule['dst_port'], '-j', 'DROP'])
```

Step 8: Create GUI (Optional)

- Use Tkinter to create a simple GUI for live monitoring:

```
import tkinter as tk

class FirewallGUI:
    def __init__(self):
        self.root = tk.Tk()
        self.root.title("Firewall")
        self.log_text = tk.Text(self.root)
        self.log_text.pack()

    def update_log(self, packet):
        self.log_text.insert(tk.END, packet.summary() + "\n")
```

Step 9: Integrate Components

- Combine packet sniffing, processing, logging, and rule enforcement:

```
def main():
    # start packet sniffer
    sniff(prn=packet_sniffer)
    # start GUI (if using)
```

```
gui = FirewallGUI()
gui.root.mainloop()
```

Step 10: Test Firewall

- Test firewall rules and logging:
  - Send test packets using tools like nc or telnet
  - Verify logs and rule enforcement

Step 11: Refine and Improve

- Refine rules and logging
- Improve GUI (if using)

Step 12: Document Project

- Write a 1-2 page report in PDF format:
  - Introduction
  - Abstract
  - Tools Used
  - Steps Involved in Building the Project
  - Conclusion

This guide provides a basic outline for building a personal firewall using Python. You can improve and customize it according to your needs.