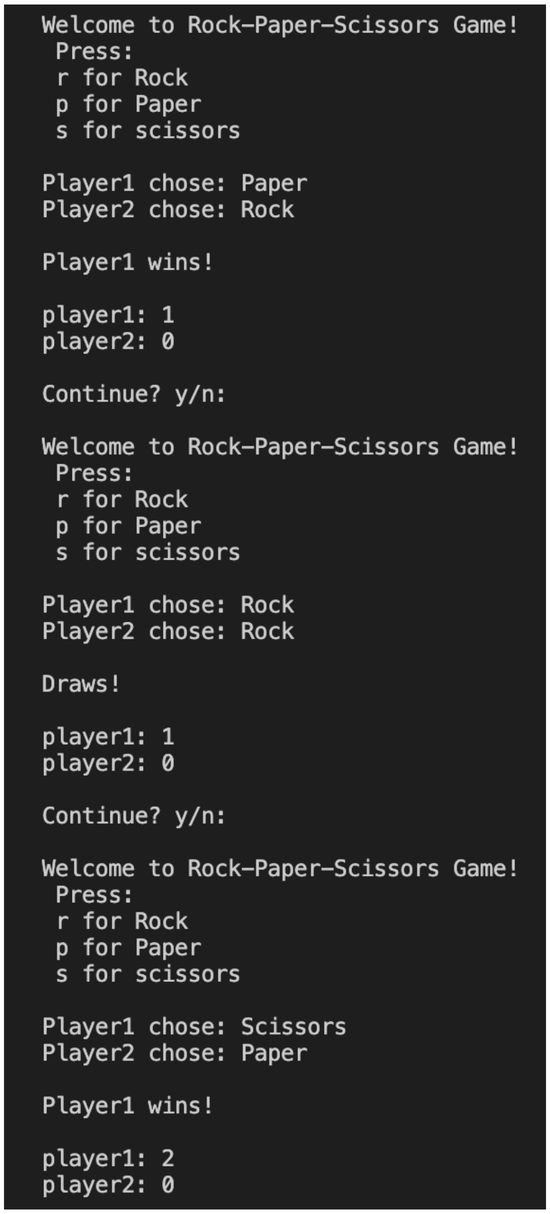
Engkanchanith Tan
005834717

Lab 3: GUI

Basic Command-Line Game: Playing against AI

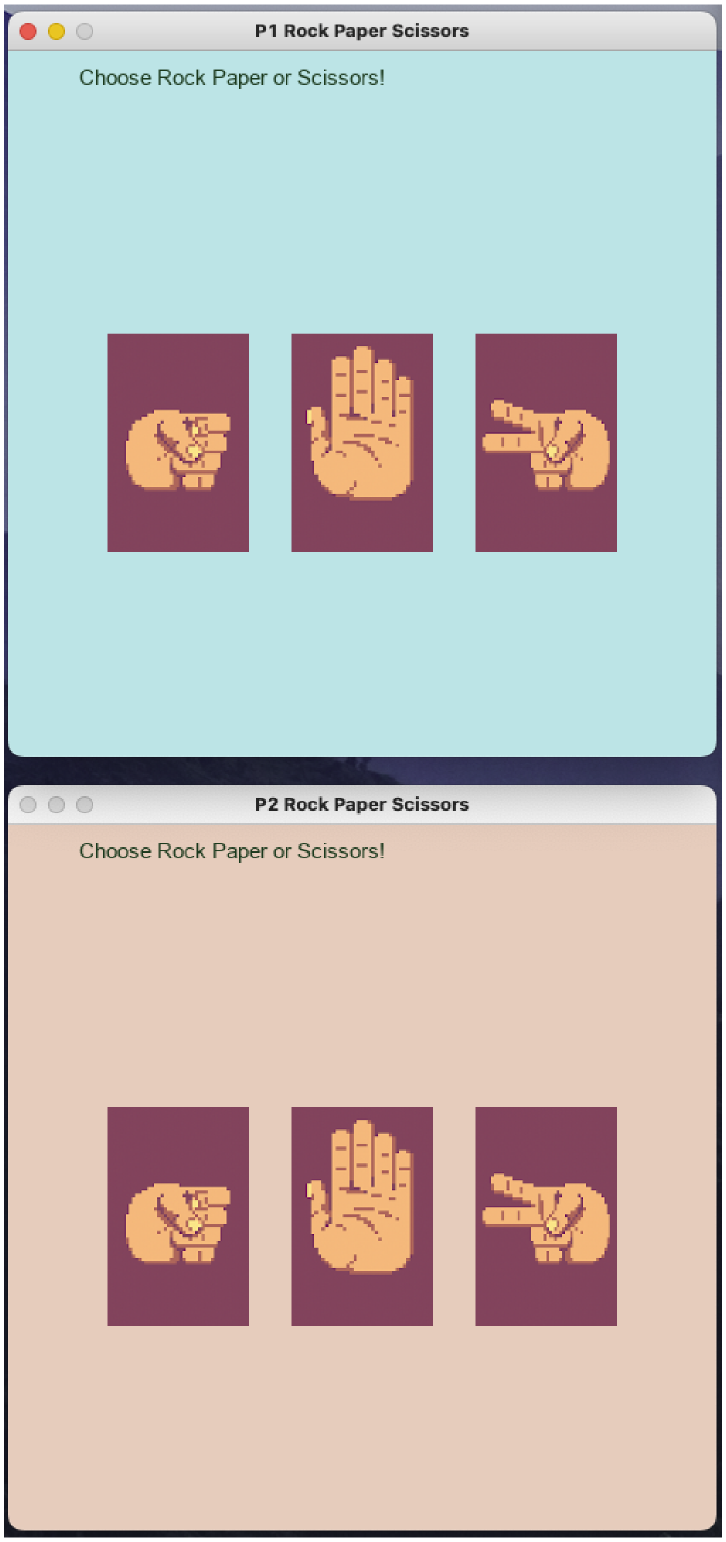
Player’s Terminal:



```
Welcome to Rock-Paper-Scissors Game!
 Press:
 r for Rock
 p for Paper
 s for scissors

Player1 chose: Paper
Player2 chose: Rock

Player1 wins!

player1: 1
player2: 0

Continue? y/n:

Welcome to Rock-Paper-Scissors Game!
 Press:
 r for Rock
 p for Paper
 s for scissors

Player1 chose: Rock
Player2 chose: Rock

Draws!

player1: 1
player2: 0

Continue? y/n:

Welcome to Rock-Paper-Scissors Game!
 Press:
 r for Rock
 p for Paper
 s for scissors

Player1 chose: Scissors
Player2 chose: Paper

Player1 wins!

player1: 2
player2: 0
```

Task 1: Two Players using pygame

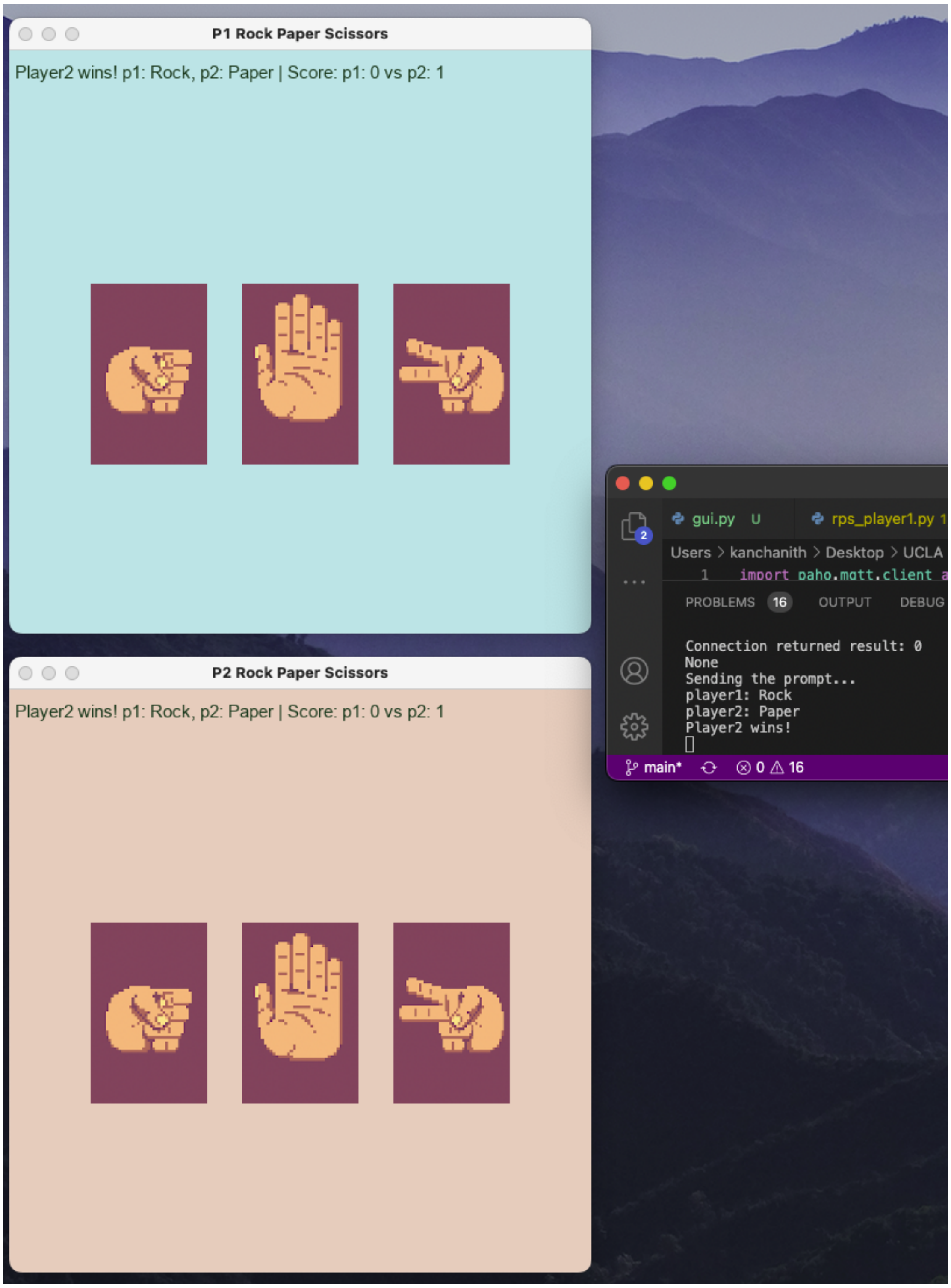
Part 1: Start of the game



Choose Rock Paper or Scissors!

Choose Rock Paper or Scissors!

Part 2: First game. Player2 wins + Central Server Terminal



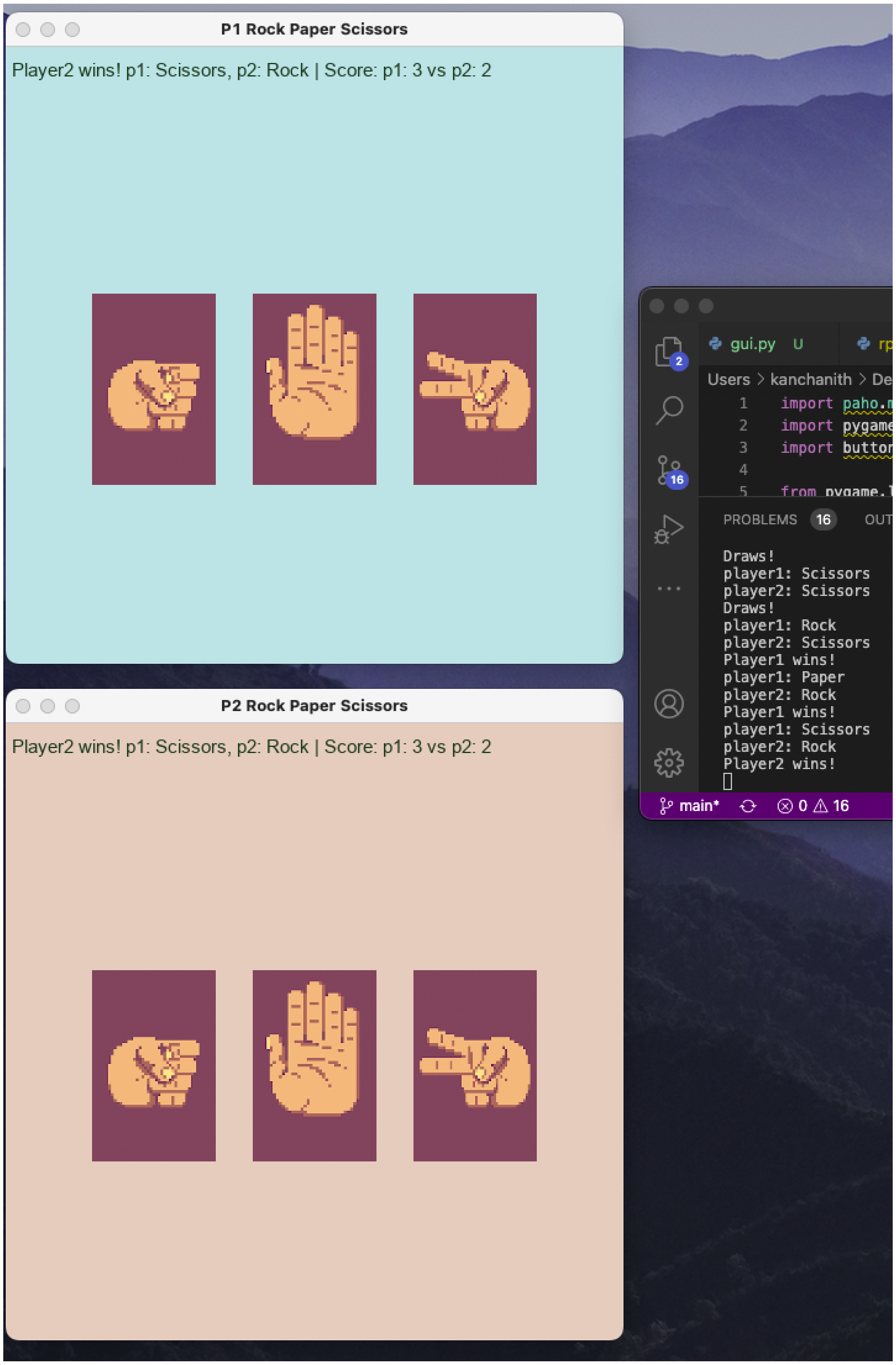
Player2 wins! p1: Rock, p2: Paper | Score: p1: 0 vs p2: 1

Player2 wins! p1: Rock, p2: Paper | Score: p1: 0 vs p2: 1

```
import paho.mqtt.client
```

```
Connection returned result: 0
None
Sending the prompt...
player1: Rock
player2: Paper
Player2 wins!
```

Part 3: After multiple rounds + Central Server Terminal



Player2 wins! p1: Scissors, p2: Rock | Score: p1: 3 vs p2: 2

Player2 wins! p1: Scissors, p2: Rock | Score: p1: 3 vs p2: 2

Users > kanchanith > De

```
import paho.m
import pygame
import button

from pygame.
```

```
Draws!
player1: Scissors
player2: Scissors
Draws!
player1: Rock
player2: Scissors
Player1 wins!
player1: Paper
player2: Rock
Player1 wins!
player1: Scissors
player2: Rock
Player2 wins!
```

Task 2: I chose to look briefly over PyQT.

Task 3:

This lab gave me a clear overview on how central server should work and how to communicate between central server and players. Fortunately, our game's main action is shooting. We can apply the concept of collision detection like most of the pygame beginner's games. For each team, there'll be sprites for a base, the players themselves, the veggie bullets, enemies, etc. User input will initially be keyboard inputs. We can also add sound effects for collision and background music for the game itself.

For communication part, it's more clear that the game logic happens in the central server. The players code are only responsible for taking in inputs, publishing the data, and displaying the updates on the windows after the central server doing all the work.

Task 4:
www.github.com/Kanchanith/180DA-WarmUp/

added pygame_p1.py, pygame_p2.py, rps_central.py, gui.py, button.py, and image files