

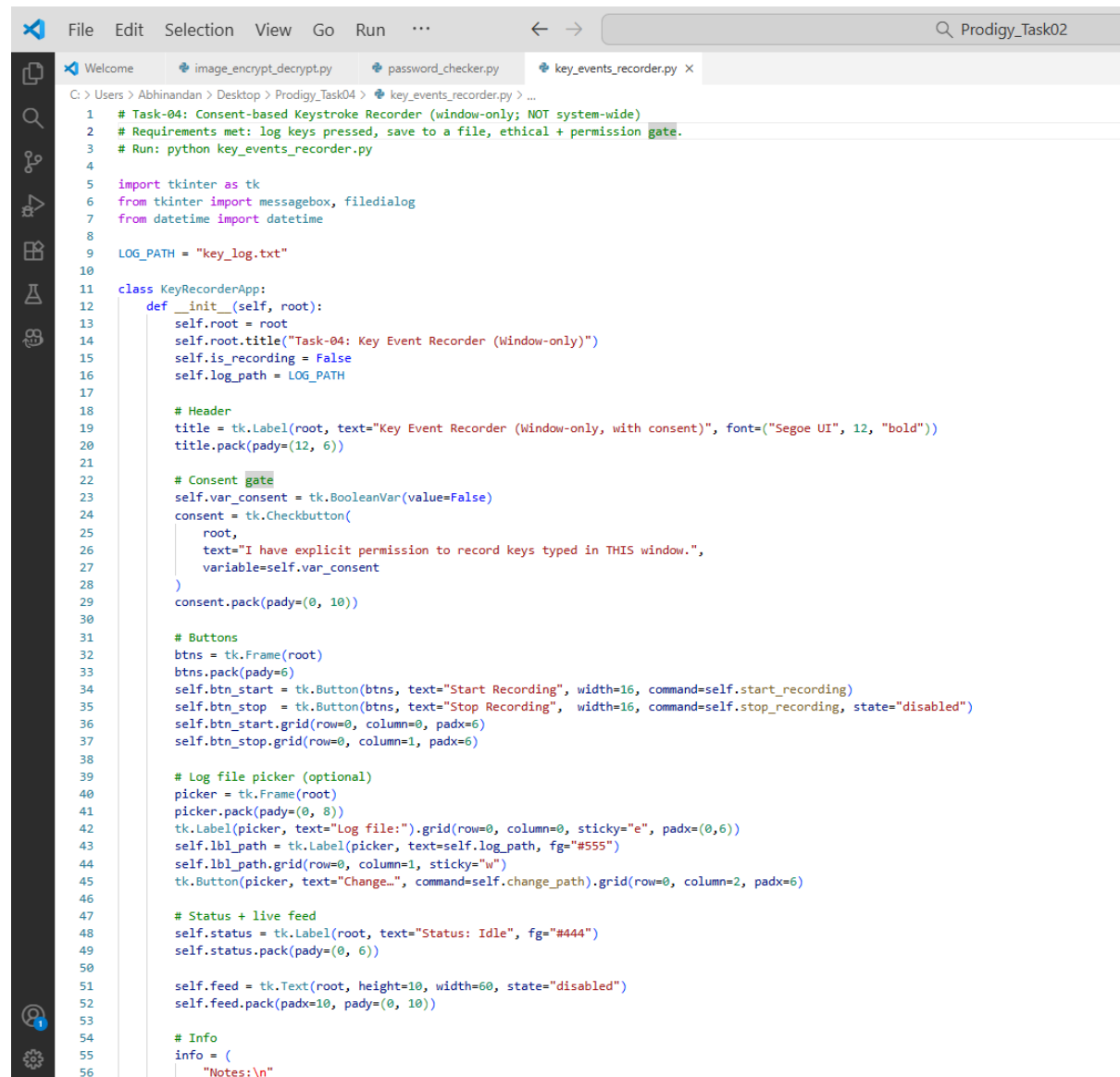
Title: Task-04 — Consent-based Keystroke Recorder (Window-Only)

Objective: Log keys pressed within a focused app window and save to a file with explicit consent and start/stop controls.

Ethics & Scope: Recording is limited to this window; shows consent checkbox and visible status; never record other users or apps without permission.

Implementation: Python tkinter; binds <KeyPress> on start; writes timestamped events to key_log.txt; unbinds on stop.

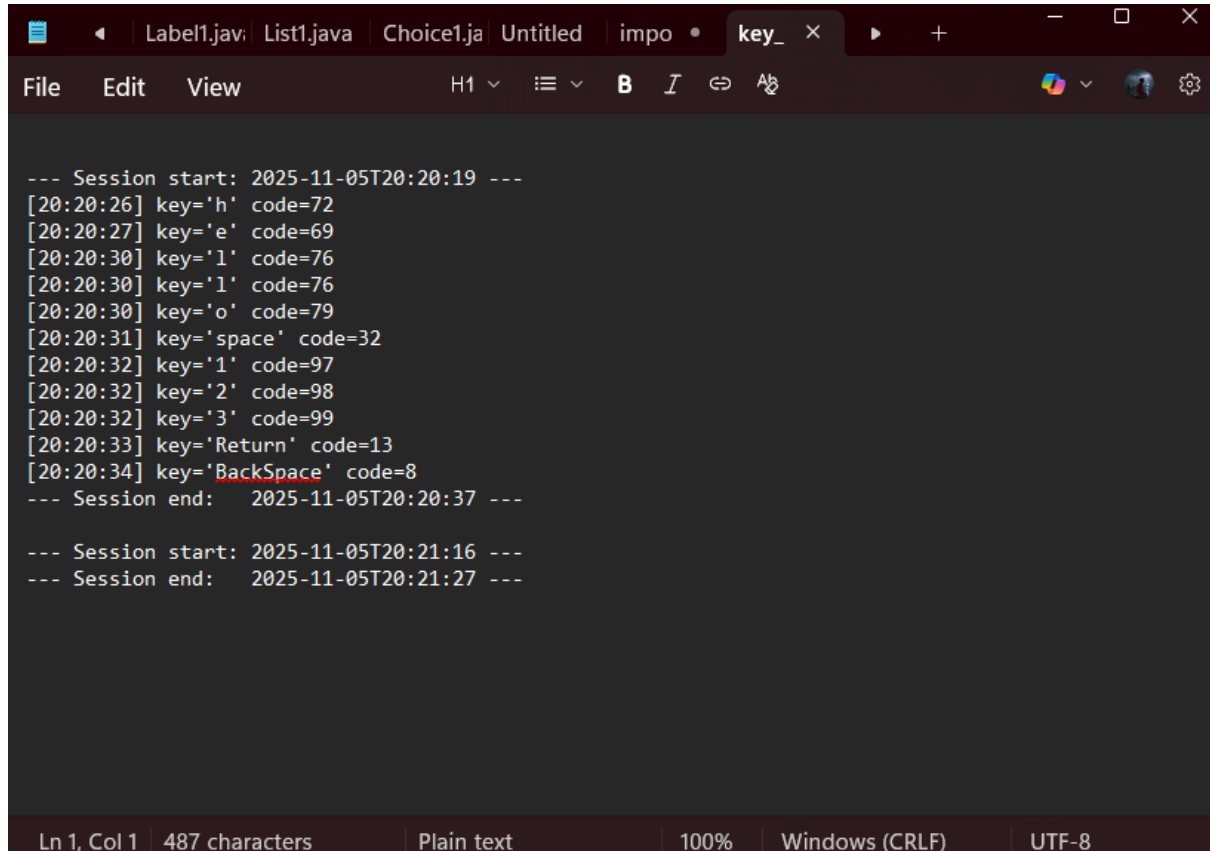
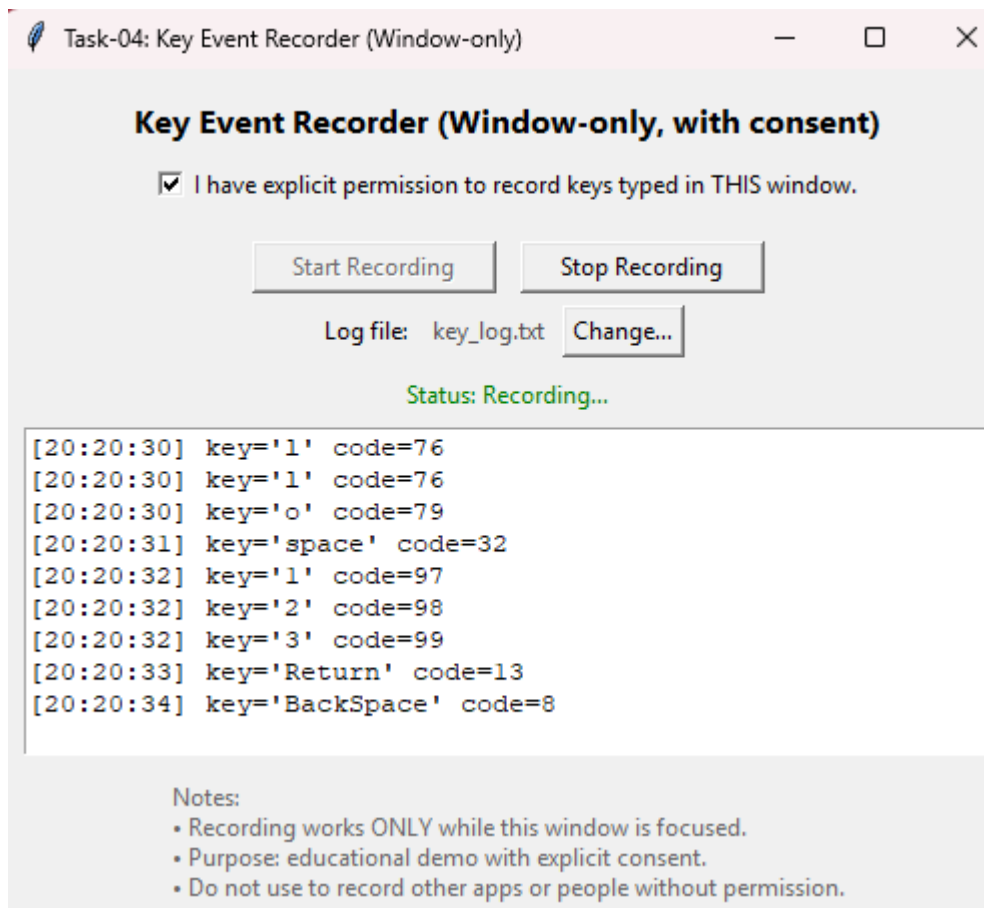
Screenshots:

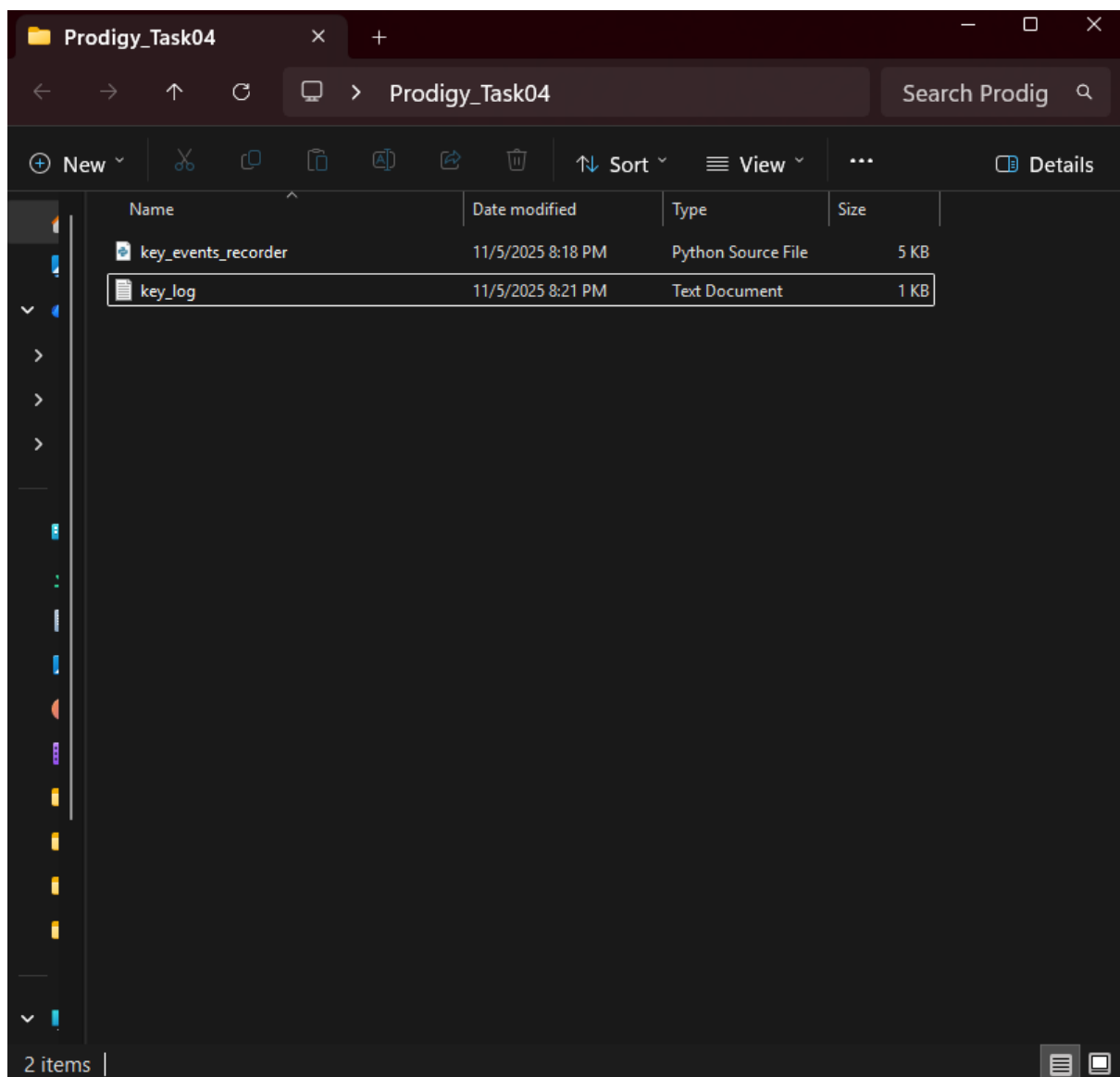


```
File Edit Selection View Go Run ...
Welcome image_encrypt_decrypt.py password_checker.py key_events_recorder.py X
C:\Users\Abhinandan\Desktop> Prodigy_Task04 > key_events_recorder.py > ...
1 # Task-04: Consent-based Keystroke Recorder (window-only; NOT system-wide)
2 # Requirements met: log keys pressed, save to a file, ethical + permission gate.
3 # Run: python key_events_recorder.py
4
5 import tkinter as tk
6 from tkinter import messagebox, filedialog
7 from datetime import datetime
8
9 LOG_PATH = "key_log.txt"
10
11 class KeyRecorderApp:
12     def __init__(self, root):
13         self.root = root
14         self.root.title("Task-04: Key Event Recorder (Window-only)")
15         self.is_recording = False
16         self.log_path = LOG_PATH
17
18         # Header
19         title = tk.Label(root, text="Key Event Recorder (Window-only, with consent)", font=("Segoe UI", 12, "bold"))
20         title.pack(pady=(12, 6))
21
22         # Consent gate
23         self.var_consent = tk.BooleanVar(value=False)
24         consent = tk.Checkbutton(
25             root,
26             text="I have explicit permission to record keys typed in THIS window.",
27             variable=self.var_consent
28         )
29         consent.pack(pady=(0, 10))
30
31         # Buttons
32         btns = tk.Frame(root)
33         btns.pack(pady=6)
34         self.btn_start = tk.Button(btns, text="Start Recording", width=16, command=self.start_recording)
35         self.btn_stop = tk.Button(btns, text="Stop Recording", width=16, command=self.stop_recording, state="disabled")
36         self.btn_start.grid(row=0, column=0, padx=6)
37         self.btn_stop.grid(row=0, column=1, padx=6)
38
39         # Log file picker (optional)
40         picker = tk.Frame(root)
41         picker.pack(pady=(0, 8))
42         tk.Label(picker, text="Log file:").grid(row=0, column=0, sticky="e", padx=(0, 6))
43         self.lbl_path = tk.Label(picker, text=self.log_path, fg="#555")
44         self.lbl_path.grid(row=0, column=1, sticky="w")
45         tk.Button(picker, text="Change...", command=self.change_path).grid(row=0, column=2, padx=6)
46
47         # Status + live feed
48         self.status = tk.Label(root, text="Status: Idle", fg="#444")
49         self.status.pack(pady=(0, 6))
50
51         self.feed = tk.Text(root, height=10, width=60, state="disabled")
52         self.feed.pack(padx=10, pady=(0, 10))
53
54         # Info
55         info = (
56             "Notes:\n"
```

```
File Edit Selection View Go Run ... Prodigy_Task02

Welcome image_encrypt_decrypt.py password_checker.py key_events_recorder.py X
C:\Users\Abhinandan > Desktop > Prodigy_Task04 > key_events_recorder.py > ...
11 class KeyRecorderApp:
12     def __init__(self, root):
13         """
14         * Recording works ONLY while this window is focused.\n"
15         * Purpose: educational demo with explicit consent.\n"
16         * Do not use to record other apps or people without permission."
17         """
18         tk.Label(root, text=info, justify="left", fg="#666").pack(padx=18, pady=(8,18))
19
20         # Key binding (bound when recording)
21         self.root.protocol("WM_DELETE_WINDOW", self.on_close)
22
23     def change_path(self):
24         path = filedialog.asksaveasfilename(
25             title="Choose log file",
26             defaultextension=".txt",
27             filetypes=[("Text files", "*.txt"), ("All files", "*.")]
28         )
29         if path:
30             self.log_path = path
31             self.lbl_path.config(text=path)
32
33     def start_recording(self):
34         if not self.var_consent.get():
35             messagebox.showwarning("Consent required", "Please tick the consent box before recording.")
36             return
37         self.is_recording = True
38         self.btn_start.config(state="disabled")
39         self.btn_stop.config(state="normal")
40         self.status.config(text="Status: Recording-", fg="green")
41         # Bind keypress only now
42         self.root.bind("<KeyPress>", self.on_key)
43
44         # Create/append file header
45         with open(self.log_path, "a", encoding="utf-8") as f:
46             f.write(f"--- Session start: {datetime.now().isoformat(timespec='seconds')} ---\n")
47
48     def stop_recording(self):
49         if self.is_recording:
50             self.is_recording = False
51             self.btn_start.config(state="normal")
52             self.btn_stop.config(state="disabled")
53             self.status.config(text="Status: Stopped", fg="#444")
54             # Unbind
55             self.root.unbind("<KeyPress>")
56             with open(self.log_path, "a", encoding="utf-8") as f:
57                 f.write(f"--- Session end: {datetime.now().isoformat(timespec='seconds')} ---\n")
58
59     def on_key(self, event: tk.Event):
60         if not self.is_recording:
61             return
62         key_sym = event.keysym # e.g., a, A, space, Return
63         key_code = event.keycode # platform-specific code
64         ts = datetime.now().strftime("%H:%M:%S")
65         line = f"[{ts}] key={key_sym!r} code={key_code}\n"
66
67         # Append to file
68         with open(self.log_path, "a", encoding="utf-8") as f:
69             f.write(line)
70
71         # Append to on-screen feed
72         self.feed.config(state="normal")
73         self.feed.insert("end", line)
74         self.feed.see("end")
75         self.feed.config(state="disabled")
76
77     def on_close(self):
78         self.stop_recording()
79         self.root.destroy()
80
81 if __name__ == "__main__":
82     root = tk.Tk()
83     app = KeyRecorderApp(root)
84     root.mainloop()
```





Conclusion: Demonstrated safe, consent-based keystroke logging for educational purposes.