MEDICINE FINDER

Real-time Medicine finder and reminder checker

NAME	ID NUMBER
YEABSIRA SISAY	ETS 1386/15
YISEHAK ALELIGN	ETS 1415/15
YEABSIRA DESALEGN	ETS 1391/15
YEABKAB TIBEBU	ETS 1385/15
YARED MIHRET	ETS 1380/15
YARED GETACHEW	ETS 1383/15
TEBIBU SOLOMON	ETS 1296/15

Explore Now

TABLE OF CONTENT

Problem Statement

Problem 01

Many individuals forget to take their medications on time

Problem 02

struggle to locate medicines across various pharmacies

Problem 03

lack of centralized prescription tracking

INTRODUCTION

This project presents a Smart Medicine Reminder and Finder App that helps users manage prescriptions, receive timely reminders, and search for medicines in nearby pharmacies.

The app is accessible via both a Streamlit web interface and a Command Line Interface (CLI) for maximum flexibility and usability.

Solutions

Easy to use prescription manager

Automated reminder and tracker for mediction

Pharmacy search simulation with medicine availability

Auto-suggestion during medicine search for ease

Notification system

interactive streamlit UI with a user-friendly navigation experience

Objectives

- build a simple, functional app
- help users manage their medications effectively
- use real JSON data for storing and accessing medicine info
- showcase practical implementation of python
- provide clear UI with a clean, navigable structure
- interactive streamlit UI with a user-friendly navigation experience

methodology

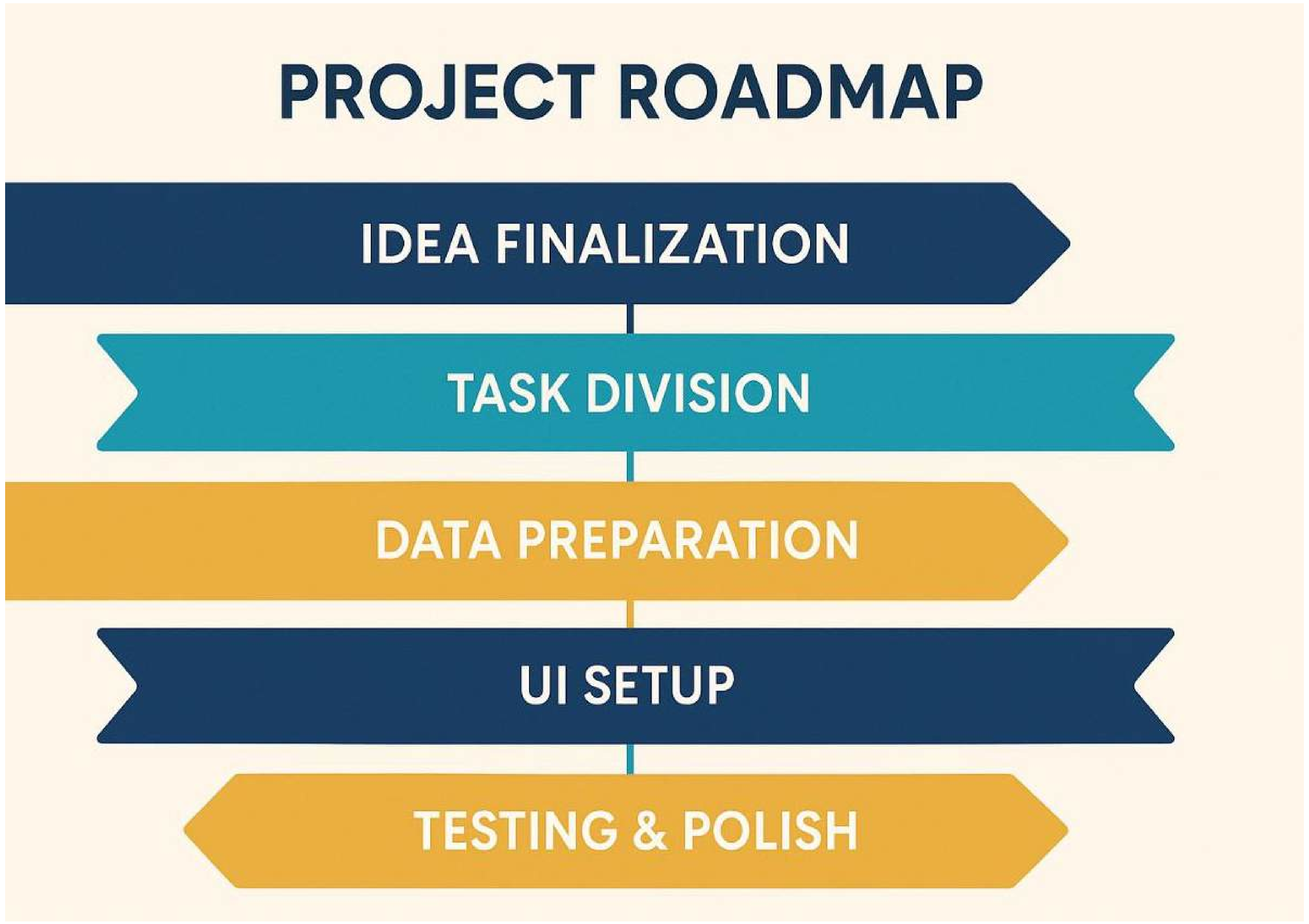
step 1 :-
data collection using mock JSON files

step 2:-
UI creation using streamlit

step 3:-
Backend logic using python module and function

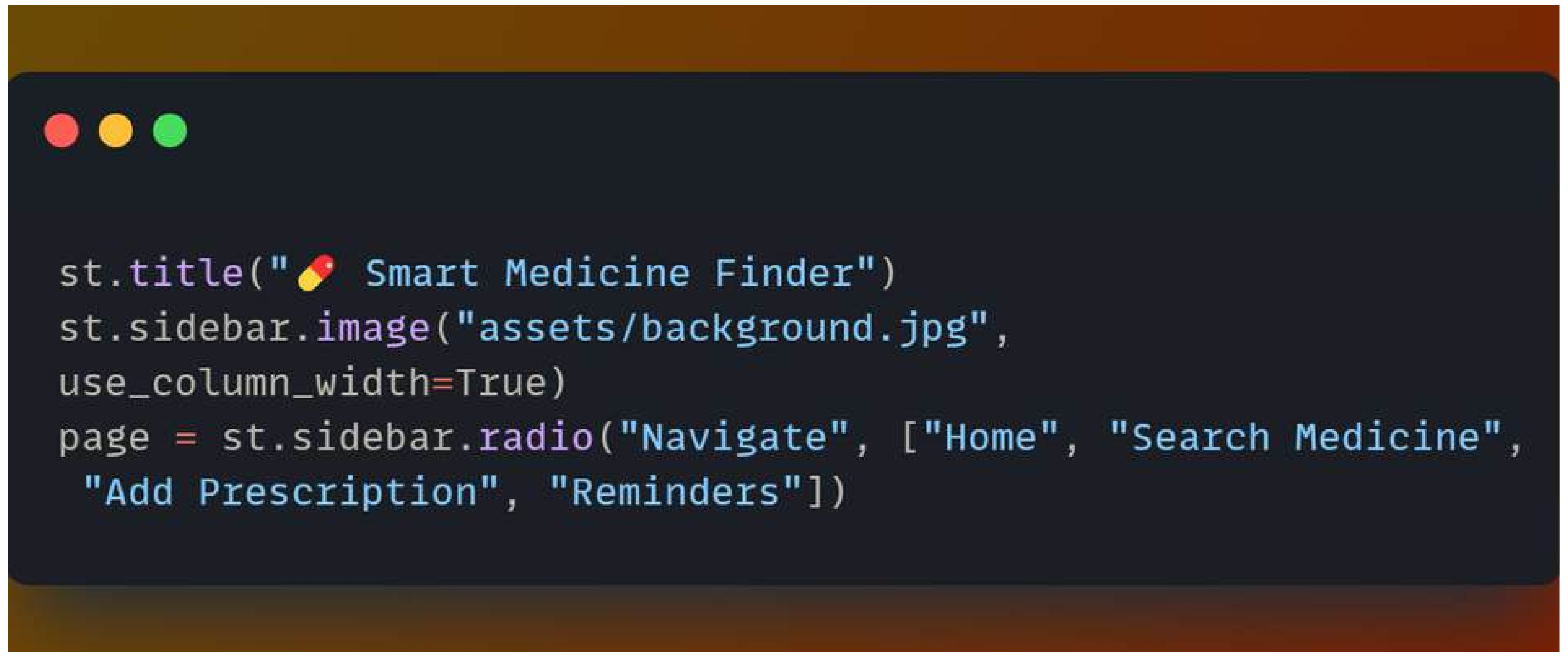
step 4:
integration of search, reminders, and simulation

step 5:
Testing, debugging, and polish for presentation


PROJECT ROADMAP
IDEA FINALIZATION
TASK DIVISION
DATA PREPARATION
UI SETUP
TESTING & POLISH

library/module	purpose
streamlit	To create the interactive web app interface
datetime	To handle and display time for reminders
json	To store and read prescription data in .json format
os	To check if prescription file exists before loading

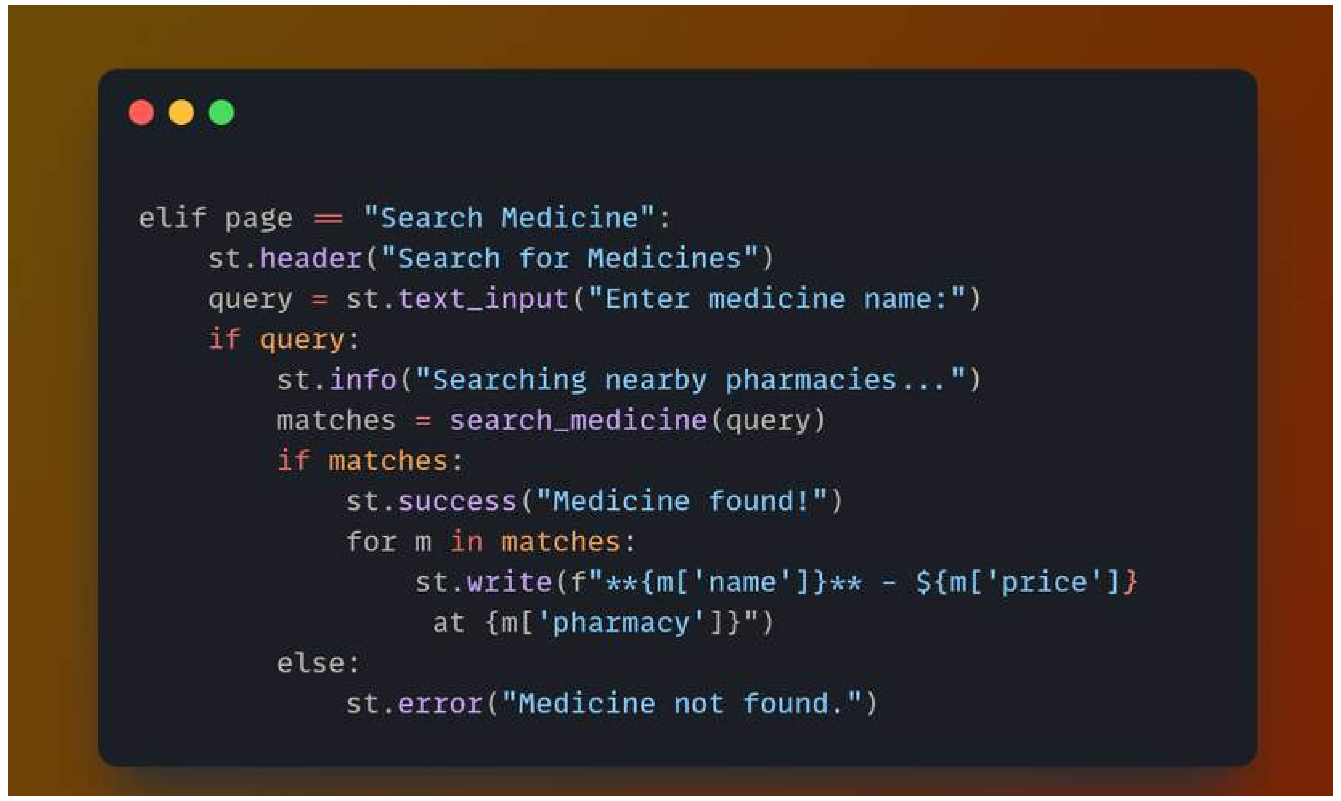
Technical Presentation



```
st.title("💊 Smart Medicine Finder")
st.sidebar.image("assets/background.jpg",
use_column_width=True)
page = st.sidebar.radio("Navigate", ["Home", "Search Medicine",
 "Add Prescription", "Reminders"])
```

- Sets the main title of the app.
- Adds a background image to the sidebar for design.
- Creates a navigation menu using radio() that lets users switch between different pages of the app.

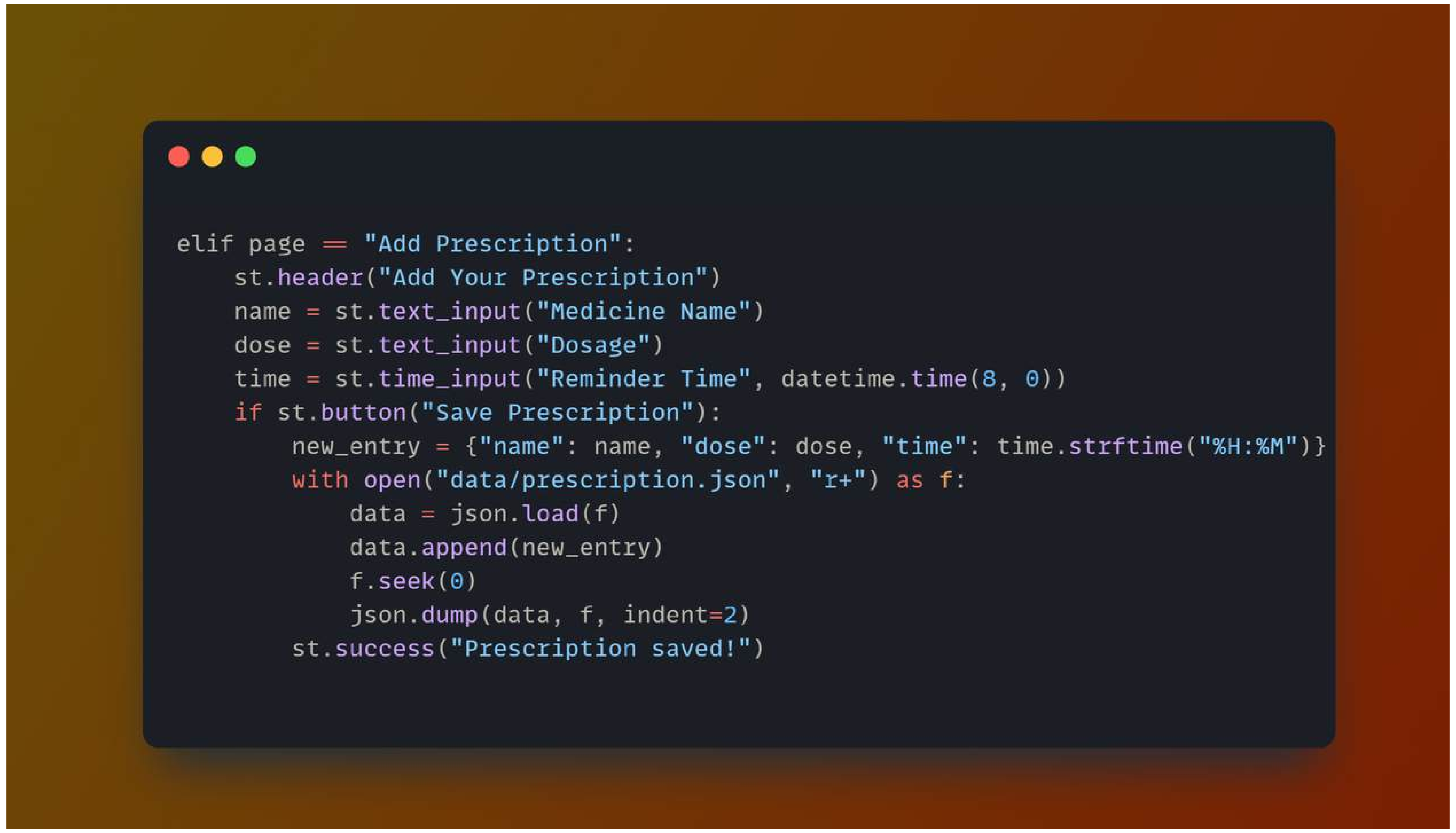
Technical Presentation



```
elif page == "Search Medicine":
    st.header("Search for Medicines")
    query = st.text_input("Enter medicine name:")
    if query:
        st.info("Searching nearby pharmacies...")
        matches = search_medicine(query)
        if matches:
            st.success("Medicine found!")
            for m in matches:
                st.write(f"**{m['name']}** - ${m['price']}
                 at {m['pharmacy']}")
        else:
            st.error("Medicine not found.")
```

- Users can type a medicine name.
- Calls search_medicine() function to simulate a pharmacy search.
- Shows search results if found, or displays an error if not.

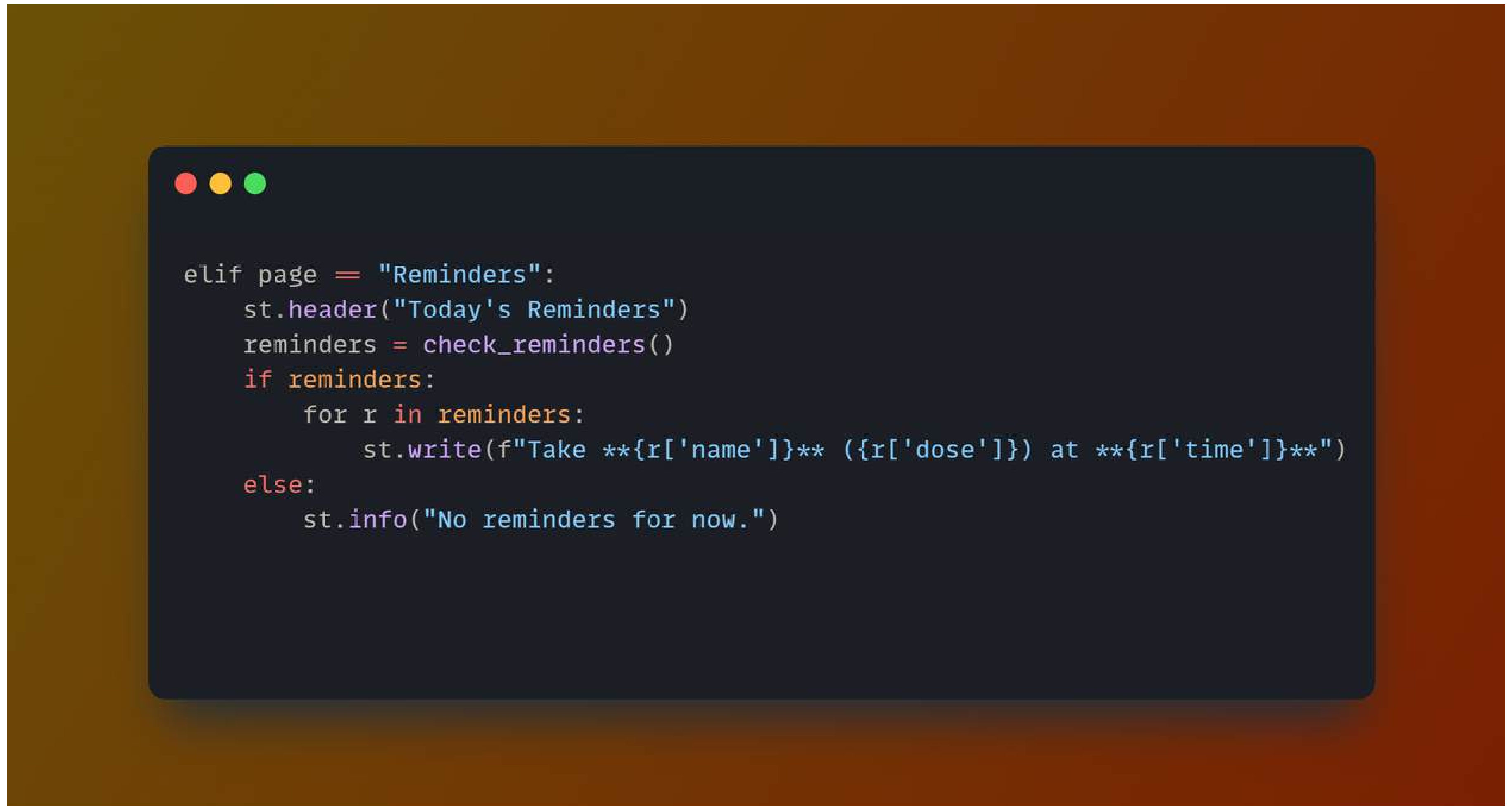
Technical Presentation



```python
elif page == "Add Prescription":
    st.header("Add Your Prescription")
    name = st.text_input("Medicine Name")
    dose = st.text_input("Dosage")
    time = st.time_input("Reminder Time", datetime.time(8, 0))
    if st.button("Save Prescription"):
        new_entry = {"name": name, "dose": dose, "time": time.strftime("%H:%M")}
        with open("data/prescription.json", "r+") as f:
            data = json.load(f)
            data.append(new_entry)
            f.seek(0)
            json.dump(data, f, indent=2)
        st.success("Prescription saved!")
```

- Lets the user enter medicine name, dosage, and reminder time.
- Saves the info to a local JSON file (prescription.json).
- Confirmation shown after saving.

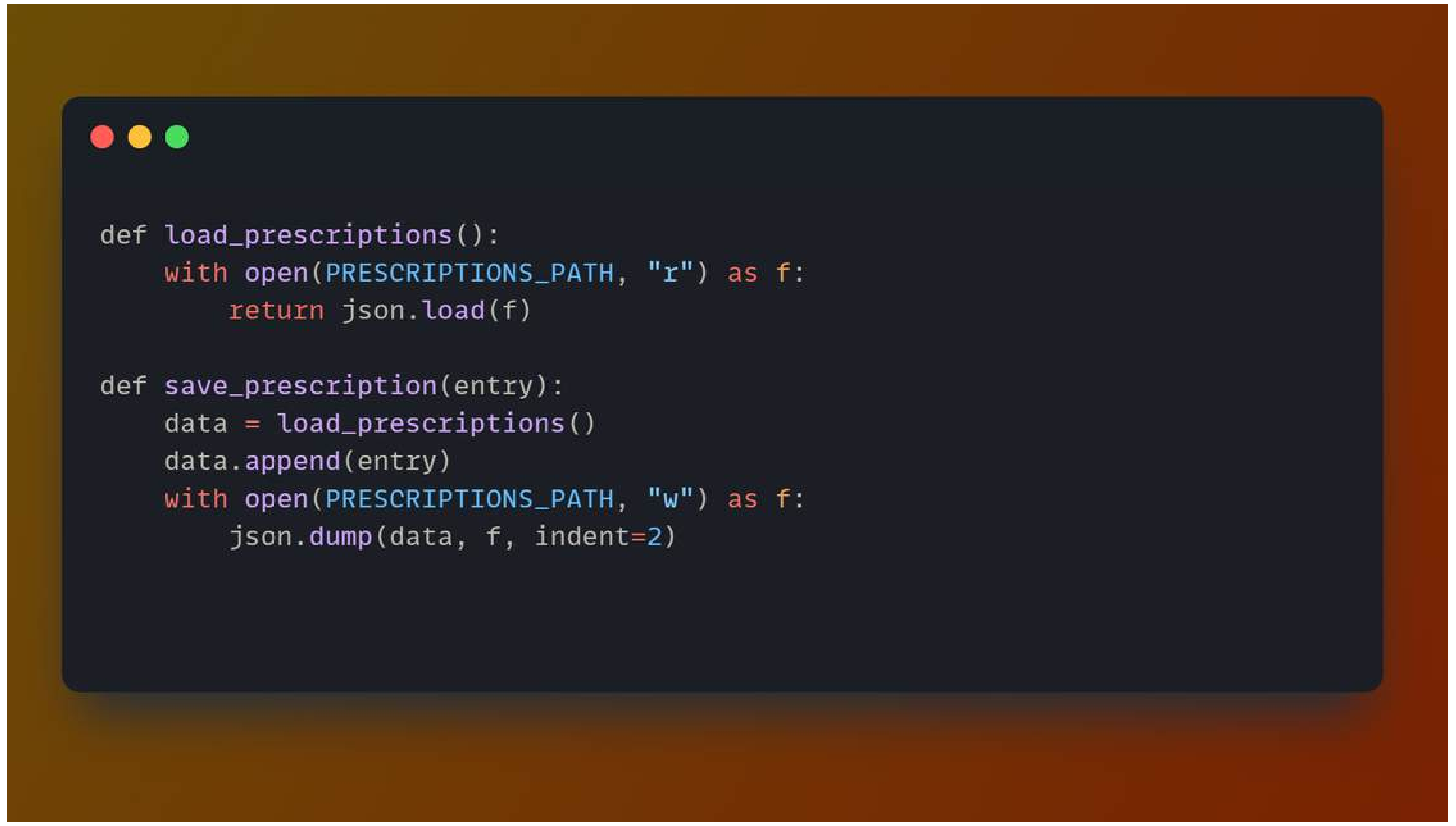
Technical Presentation



```
elif page == "Reminders":
    st.header("Today's Reminders")
    reminders = check_reminders()
    if reminders:
        for r in reminders:
            st.write(f"Take **{r['name']}** ({r['dose']}) at **{r['time']}**")
    else:
        st.info("No reminders for now.")
```

- Displays reminders for the day based on saved prescription data.
- If no reminders are due, it shows a message.

Technical Presentation

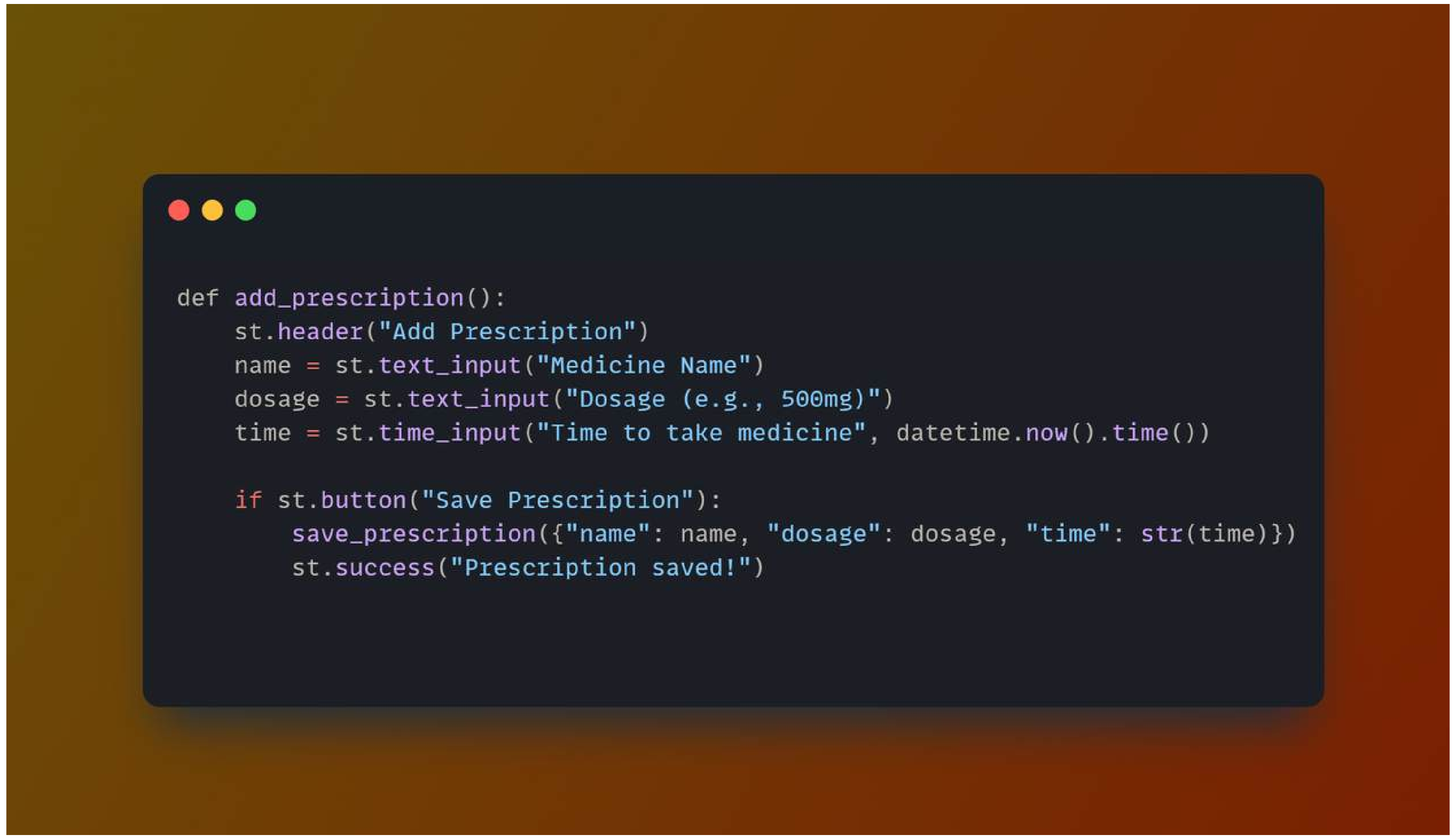


```python
def load_prescriptions():
    with open(PRESCRIPTIONS_PATH, "r") as f:
        return json.load(f)

def save_prescription(entry):
    data = load_prescriptions()
    data.append(entry)
    with open(PRESCRIPTIONS_PATH, "w") as f:
        json.dump(data, f, indent=2)
```

- load_prescriptions(): Reads data from the JSON file.
- save_prescription(entry): Adds new data and saves it back.
- Used by both CLI and Streamlit to keep data in sync.

Technical Presentation

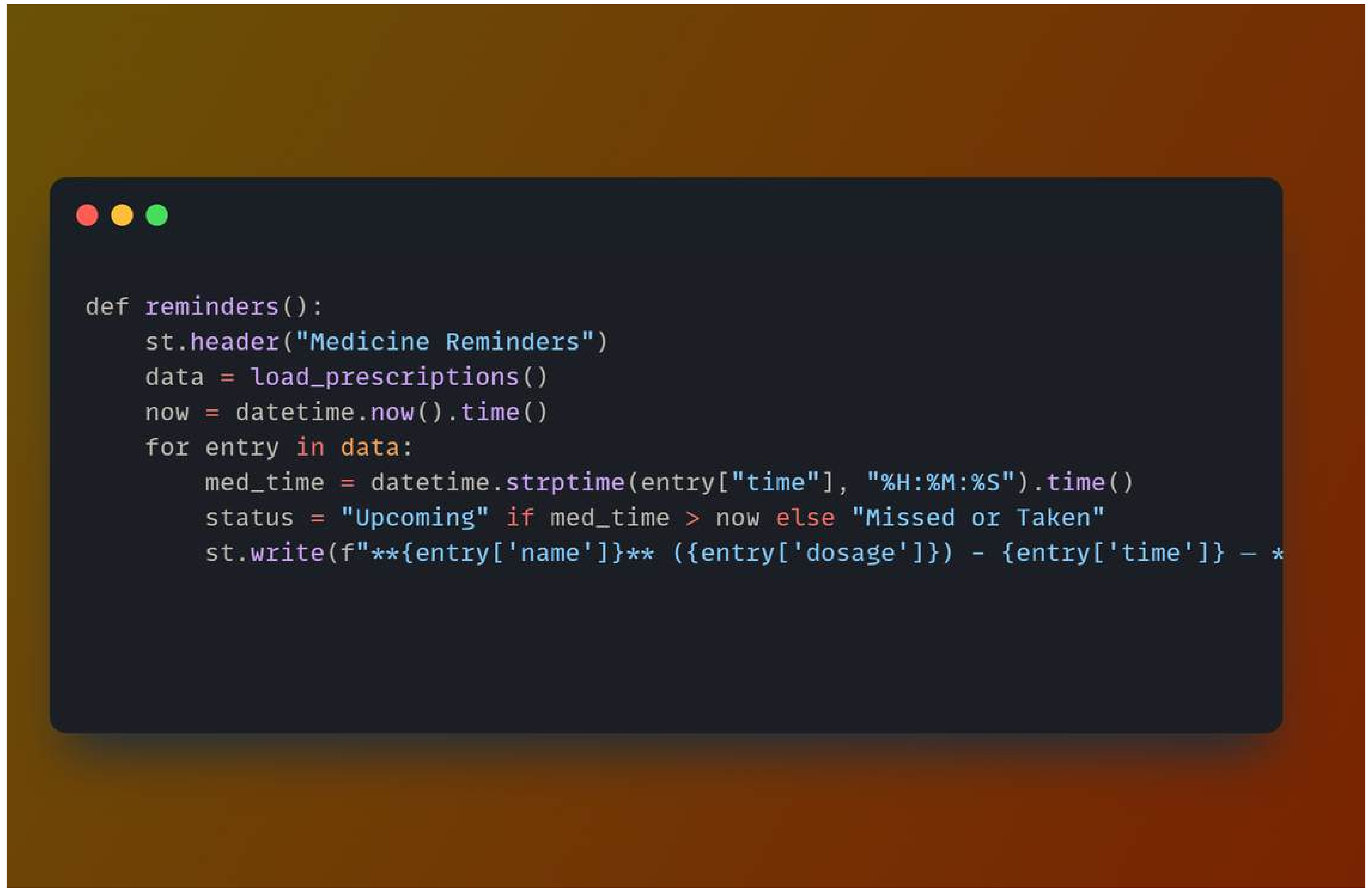


```python
def add_prescription():
    st.header("Add Prescription")
    name = st.text_input("Medicine Name")
    dosage = st.text_input("Dosage (e.g., 500mg)")
    time = st.time_input("Time to take medicine", datetime.now().time())

    if st.button("Save Prescription"):
        save_prescription({"name": name, "dosage": dosage, "time": str(time)})
        st.success("Prescription saved!")
```

- This version of add_prescription() is modular and used inside Streamlit.
- Handles form input and saves it using save_prescription()

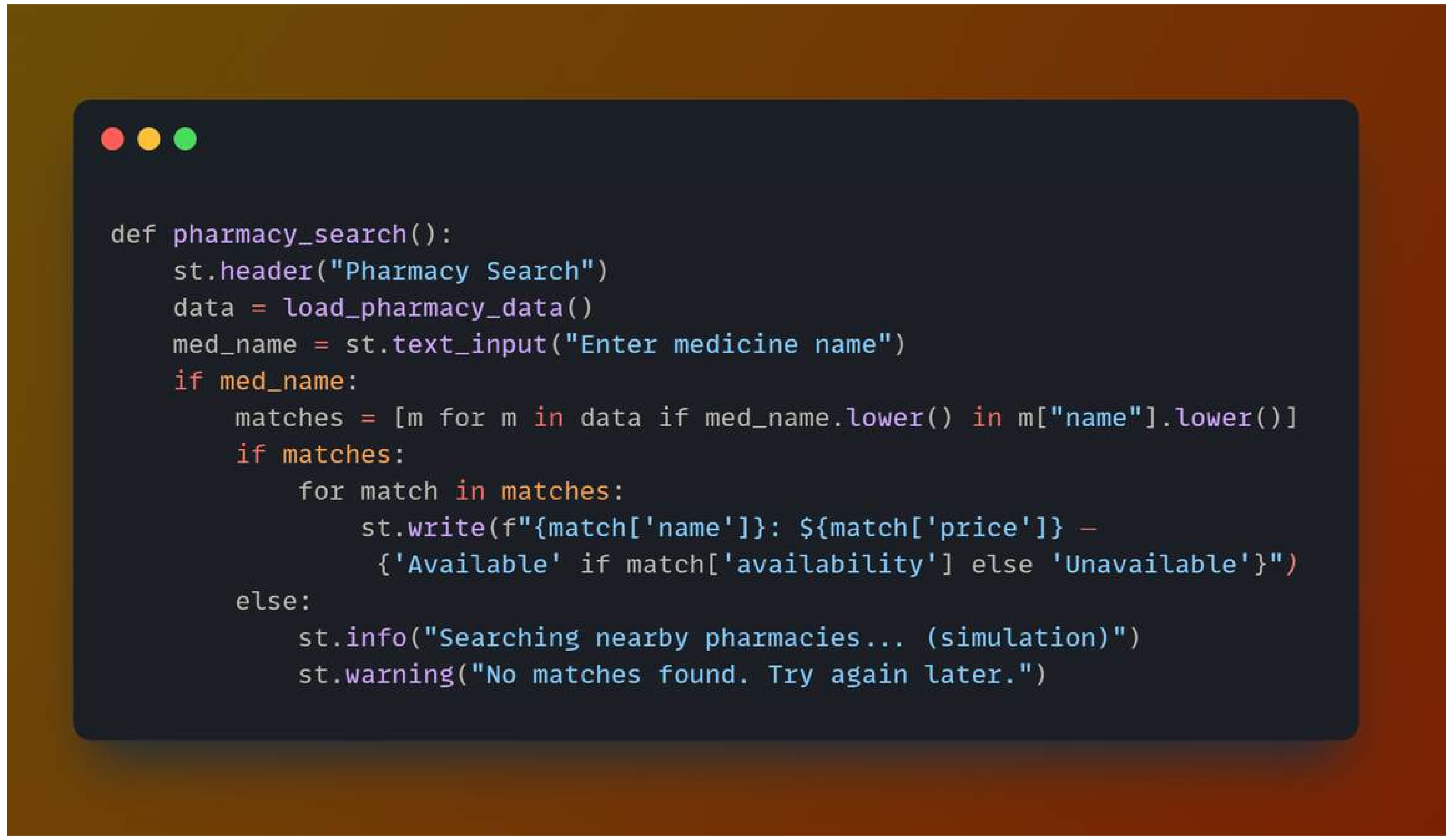
Technical Presentation



```
def reminders():
    st.header("Medicine Reminders")
    data = load_prescriptions()
    now = datetime.now().time()
    for entry in data:
        med_time = datetime.strptime(entry["time"], "%H:%M:%S").time()
        status = "Upcoming" if med_time > now else "Missed or Taken"
        st.write(f"**{entry['name']}** ({entry['dosage']}) - {entry['time']} — *
```

- Reads all prescriptions.
- Compares current time to reminder time.
- Displays each medicine with a status like Upcoming or Missed.

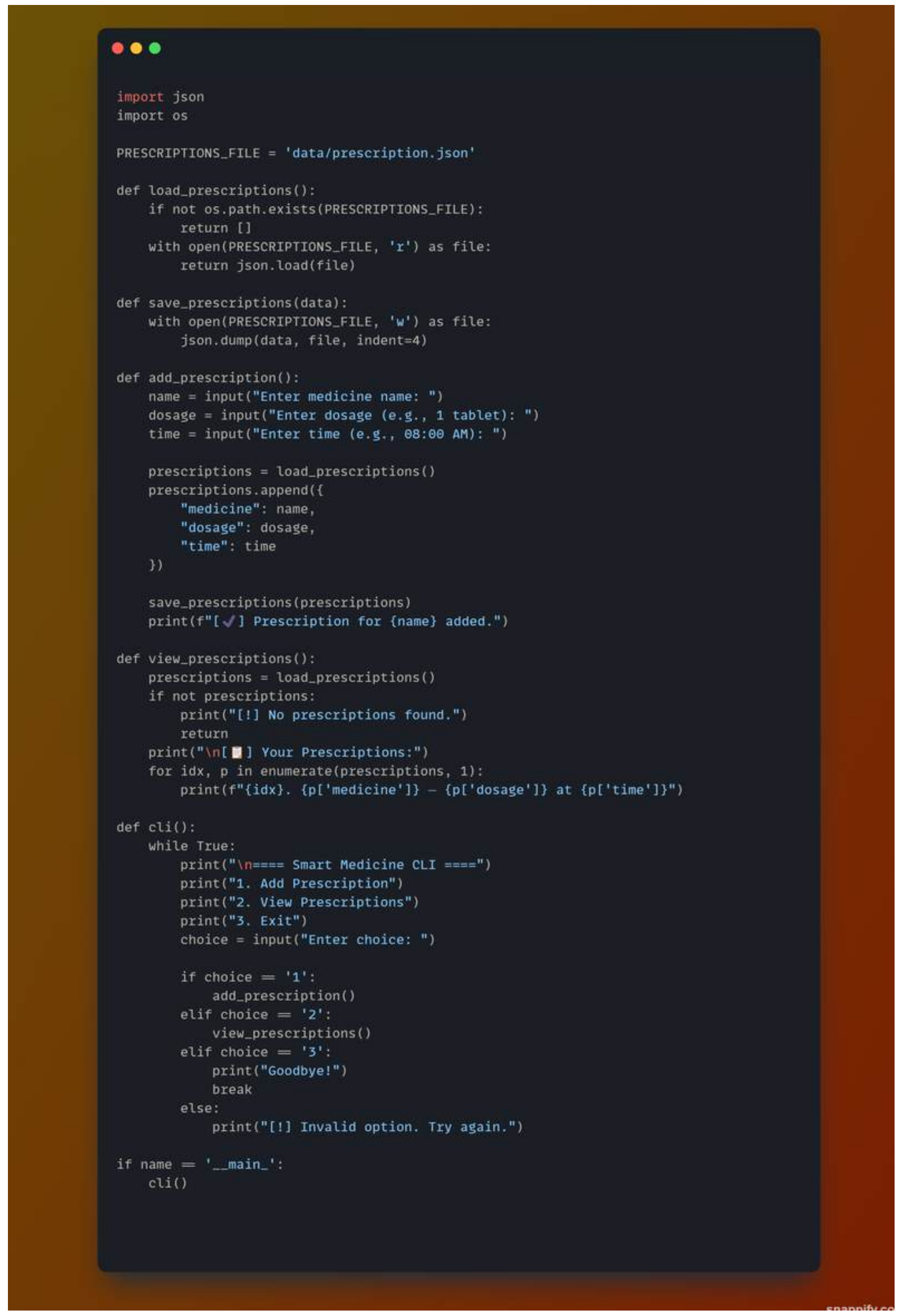
Technical Presentation



```python
def pharmacy_search():
    st.header("Pharmacy Search")
    data = load_pharmacy_data()
    med_name = st.text_input("Enter medicine name")
    if med_name:
        matches = [m for m in data if med_name.lower() in m["name"].lower()]
        if matches:
            for match in matches:
                st.write(f"{match['name']}: ${match['price']} —
                 {'Available' if match['availability'] else 'Unavailable'}")
        else:
            st.info("Searching nearby pharmacies... (simulation)")
            st.warning("No matches found. Try again later.")
```

- Searches local pharmacy dataset for medicine name.
- Shows availability and price info.
- Simulated real-world pharmacy search behavior.

Technical Presentation



```python
import json
import os

PRESCRIPTIONS_FILE = 'data/prescription.json'

def load_prescriptions():
    if not os.path.exists(PRESCRIPTIONS_FILE):
        return []
    with open(PRESCRIPTIONS_FILE, 'r') as file:
        return json.load(file)

def save_prescriptions(data):
    with open(PRESCRIPTIONS_FILE, 'w') as file:
        json.dump(data, file, indent=4)

def add_prescription():
    name = input("Enter medicine name: ")
    dosage = input("Enter dosage (e.g., 1 tablet): ")
    time = input("Enter time (e.g., 08:00 AM): ")

    prescriptions = load_prescriptions()
    prescriptions.append({
        "medicine": name,
        "dosage": dosage,
        "time": time
    })

    save_prescriptions(prescriptions)
    print(f"[✔] Prescription for {name} added.")

def view_prescriptions():
    prescriptions = load_prescriptions()
    if not prescriptions:
        print("[!] No prescriptions found.")
        return
    print("\n[📋] Your Prescriptions:")
    for idx, p in enumerate(prescriptions, 1):
        print(f"{idx}. {p['medicine']} – {p['dosage']} at {p['time']}")

def cli():
    while True:
        print("\n==== Smart Medicine CLI ====")
        print("1. Add Prescription")
        print("2. View Prescriptions")
        print("3. Exit")
        choice = input("Enter choice: ")

        if choice == '1':
            add_prescription()
        elif choice == '2':
            view_prescriptions()
        elif choice == '3':
            print("Goodbye!")
            break
        else:
            print("[!] Invalid option. Try again.")

if name == '__main_':
    cli()
```

- This code enables adding and viewing prescriptions via the terminal.
- Uses JSON for storage, same as Streamlit.
- Friendly messages help guide the user.

RESULT

- A fully working app with a clean UI, background visuals, and separate sections.
- Easy prescription management and tracking.
- Realistic medicine search with autosuggestions.
- Beginner-friendly Python implementation using real-world logic.

Reference

- Streamlit Documentation: https://docs.streamlit.io/
- Python Official Docs: https://docs.python.org/3/
- JSON Module Reference: https://docs.python.org/3/library/json.html
- Playsound GitHub: https://github.com/TaylorSMarks/playsound