What I have to do to book with clinic?
I am thinking to call clinic and write down my appointment in my notebook, do you have any other idea?

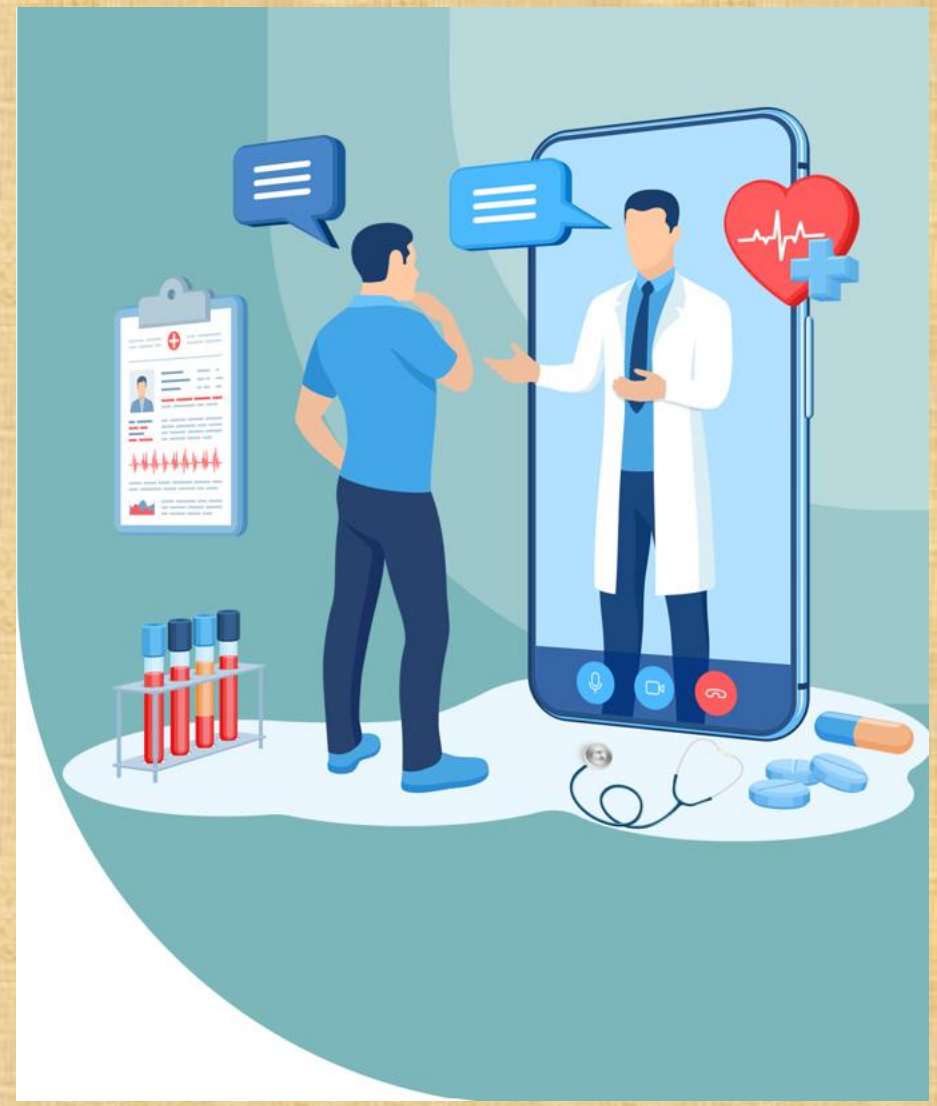
52.4% of patients forget about their appointment or forget to cancel

If you're looking for an alternative way to book appointments with the clinic, rather than manually calling and writing down the details, I have a suggestion for you. You can use my application to simplify the booking process!

Clinic Appointment Booking and Reminder

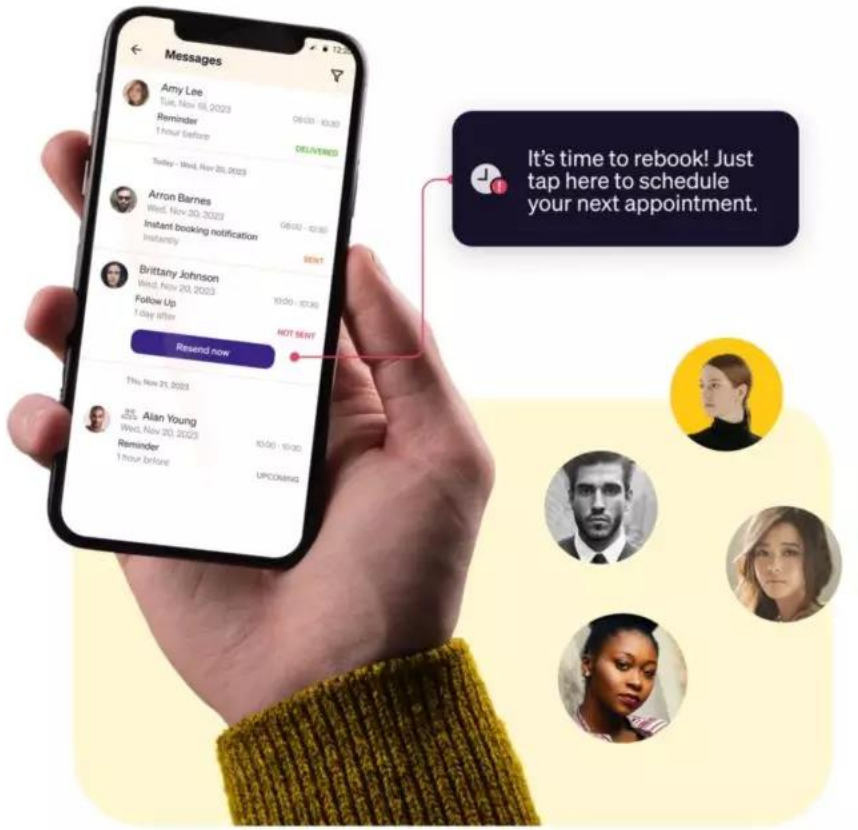
Python Terminal Application

By. Ramiz Saeed



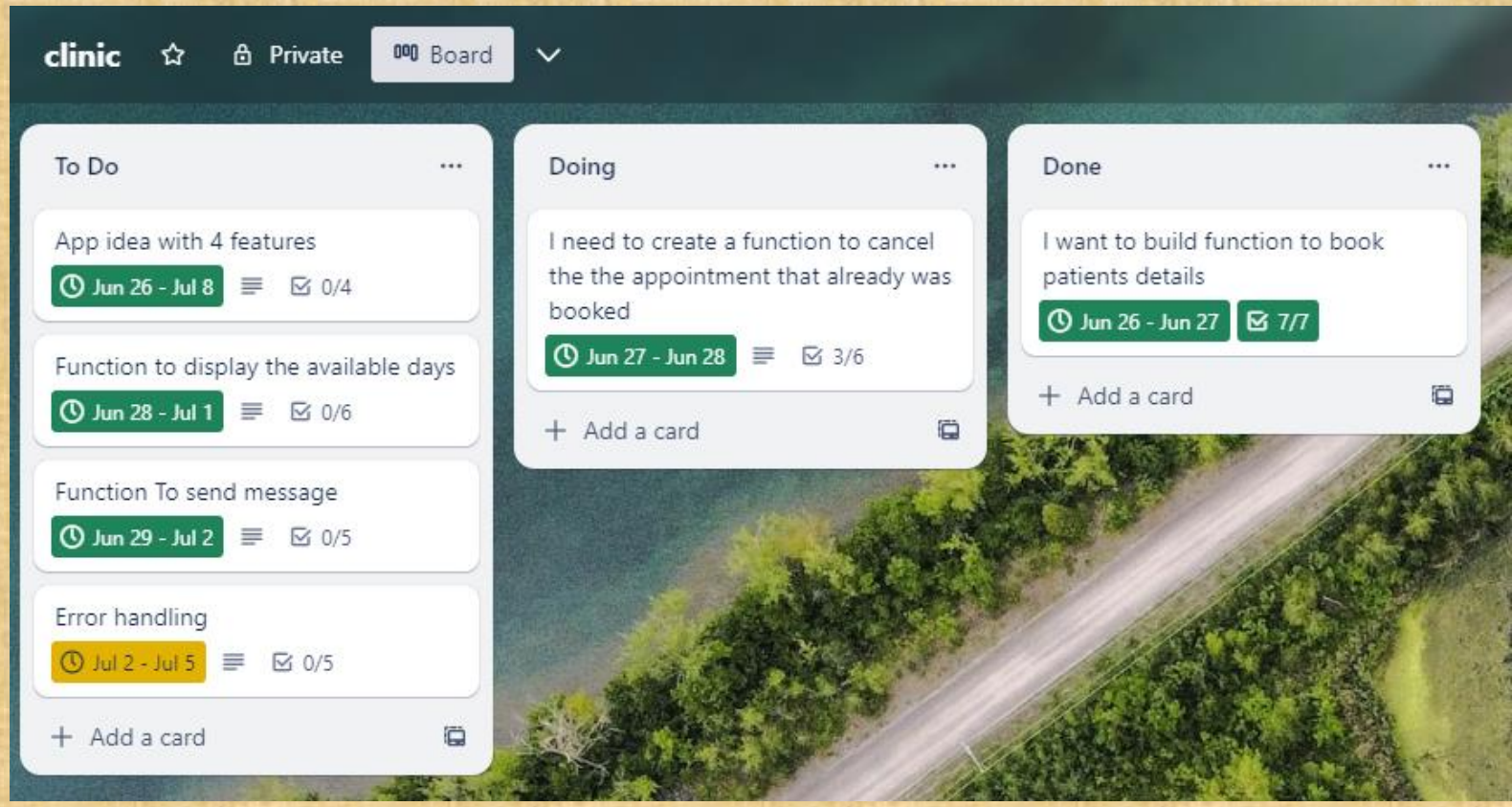
Purpose

To provide a simple and efficient way for patients to book appointments at a clinic, manage their appointments, and receive appointment reminders.



Project Management

- Listing the requirements
- Prioritizing requirements based on dependencies
- Starting with the main and simple elements then gradually incorporating accurate details:



App idea with 4 features

in list To Do

Notifications: Watch

Dates: Jun 26 - Jul 8 at 12:09 PM Complete

Description

Edit

Planning to create app allows patients to view the available days in certain clinic, book appointments at a clinic, and sends appointment reminders. The application will store patient information and appointment details in a single CSV file.

Checklist of Features

Delete

0%

- [ ] - Book Appointment: Patients can enter their name, contact number, email (optional), and desired appointment date. The application will save this information to the CSV file.
- [ ] - Delete Appointment: Patients can delete their appointment by entering their name. The application will remove the corresponding entry from the CSV file.
- [ ] - Send Reminders: The application will automatically send appointment reminders to patients on the day of their appointment. Reminders will be sent via email (if provided) using SMTP.
- [ ] before booking we need to include view appointment to see if the day/date available or not

Add an item

Activity

Show details

RS Write a comment...

Add to card: Members, Labels, Checklist, Dates, Attachment, Cover, Custom Fields

Power-Ups: Add Power-Ups

Automation: Add button

Actions: Move, Copy, Make template, Archive, Share

I want to build function to book patients details

in list Done

Notifications: Watch

Dates: Jun 26 - Jun 27 at 12:09 PM Complete

Description

Edit

Book Appointment: Patients can enter their name, contact number, email (optional), and desired appointment date. The application will save this information to the CSV file.

- we need csv module in Python to read and write data to a CSV file.
- The CSV file will have the following columns: "name", "contact_number", "email", and "appointment_date".
- When a patient books an appointment, the application will prompt the patient to enter their details and save the appointment information to the CSV file.

Checklist

Hide checked items | Delete

100%

- [x] ~~define function for patients~~
- [x] ~~set input details as name, contact, email, appointment days~~
- [x] ~~prepare CSV file and store 1 or 2 rows with some details~~
- [x] ~~store the file name in the path file to facilitate use it with another function in the future~~
- [x] ~~using import feature to make function know the CSV file~~
- [x] ~~write on the file new details of booking~~
- [x] ~~prepare display menu to control the flow of the project, provide the choice to enable the user how to use this app~~

Add an item

Activity

Show details

Add to card: Members, Labels, Checklist, Dates, Attachment, Cover, Custom Fields

Power-Ups: Add Power-Ups

Automation: Add button

Actions: Move, Copy, Make template, Archive, Share

Function to display the available days

in list To Do

Notifications: Watch

Dates: Jun 28 - Jul 1 at 12:09 PM Complete

Description

Edit

I would like to include this function because it is so important here. the responsibility of this feature to check and display the availability of days to enable the user choose available day to book

Checklist

Delete

0%

- [ ] prepare display function
- [ ] import datetime and define the date of today
- [ ] set the period of showing availability days
- [ ] thinking to use nested loops to iterate in patients list file within the indicated period
- [ ] after iteration we need to add if the day is free from appointment to the available list
- [ ] check the function and make some modification if needed

Add an item

Activity

Show details

RS Write a comment...

Add to card: Members, Labels, Checklist, Dates, Attachment, Cover, Custom Fields

Power-Ups: Add Power-Ups

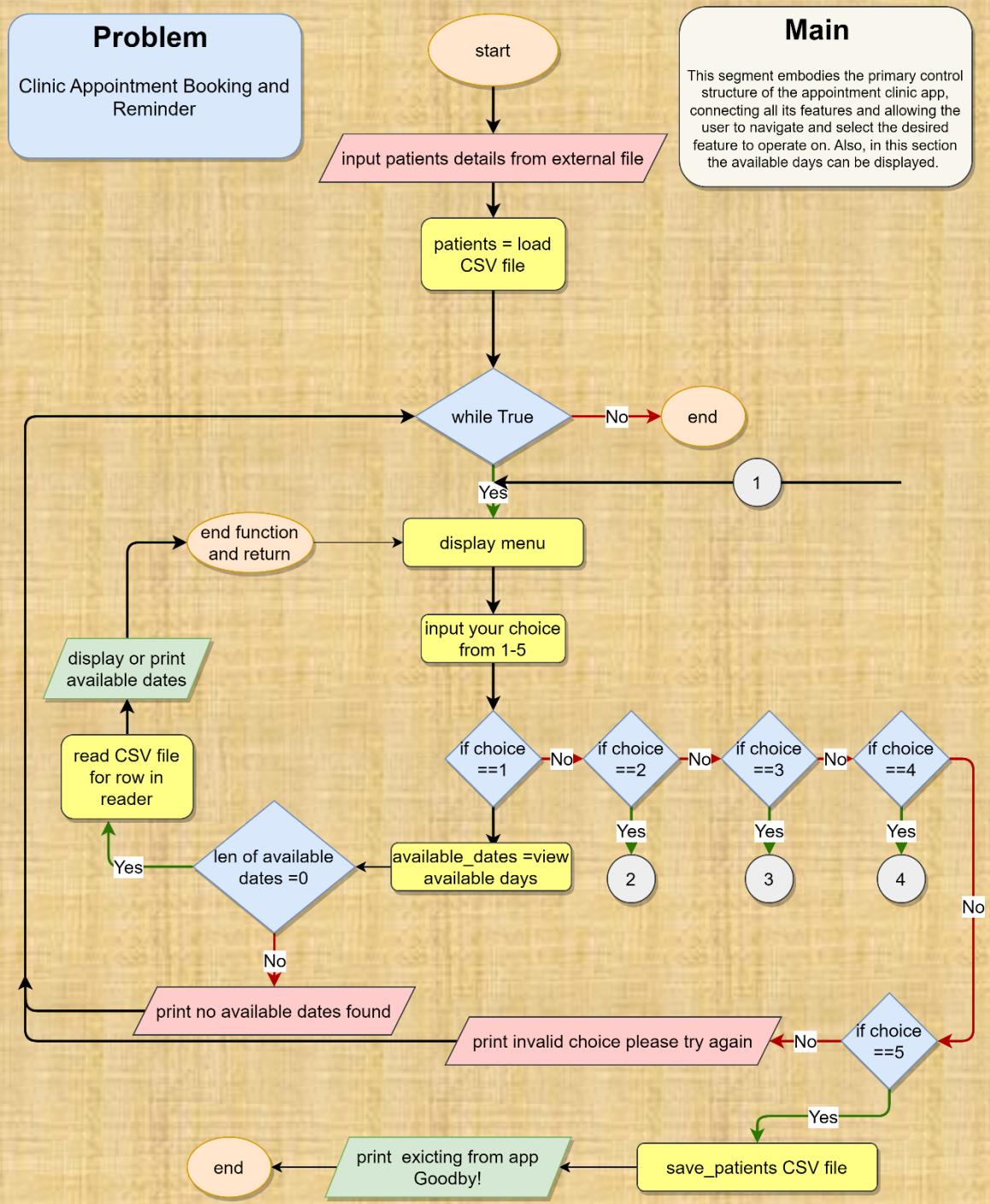
Automation: Add button

Actions: Move, Copy, Make template, Archive, Share

Features

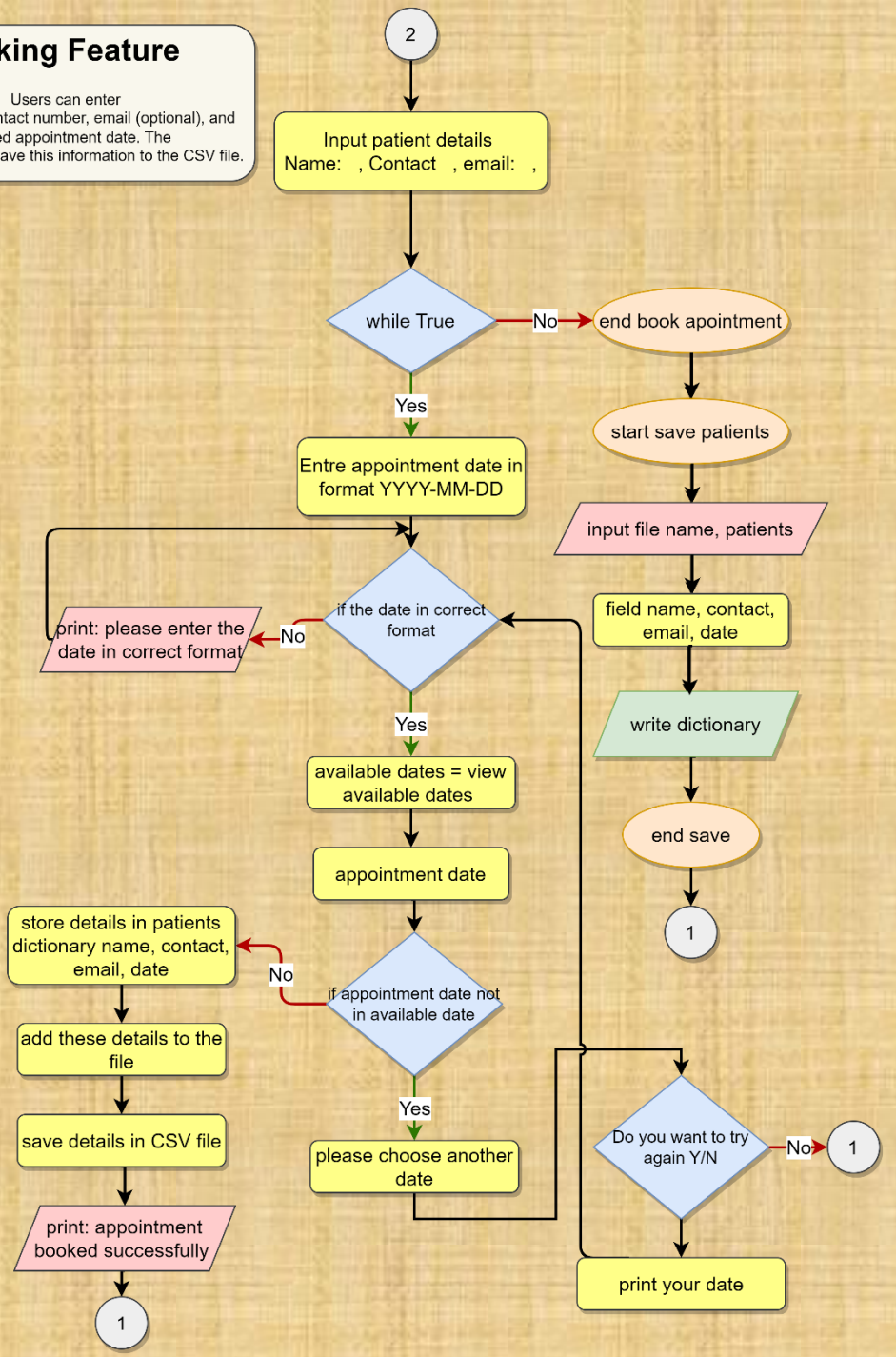
1. View Available Dates: Patients can access a calendar view to see which days are available for booking appointments. This feature helps patients plan their appointments conveniently by checking the availability in advance.
2. Booking Appointment: Patients can enter their name, contact number, email (optional), and desired appointment date. The application will save this information to the CSV file.
3. Cancelation Appointment: Patients can delete their appointment by entering their name. The application will remove the corresponding entry from the CSV file.
4. Send Reminders: The application will automatically send appointment reminders to patients on the day of their appointment. Reminders will be sent via email (if provided) using SMTP.

Flow Chart


Problem
Clinic Appointment Booking and Reminder
start
Main
This segment embodies the primary control structure of the appointment clinic app, connecting all its features and allowing the user to navigate and select the desired feature to operate on. Also, in this section the available days can be displayed.
input patients details from external file
patients = load CSV file
while True
No
end
1
Yes
end function and return
display menu
input your choice from 1-5
display or print available dates
read CSV file for row in reader
if choice ==1
No
if choice ==2
No
if choice ==3
No
if choice ==4
Yes
Yes
Yes
Yes
2
3
4
No
Yes
len of available dates =0
available_dates =view available days
No
print no available dates found
print invalid choice please try again
No
if choice ==5
Yes
save_patients CSV file
print exicting from app Goodby!
end

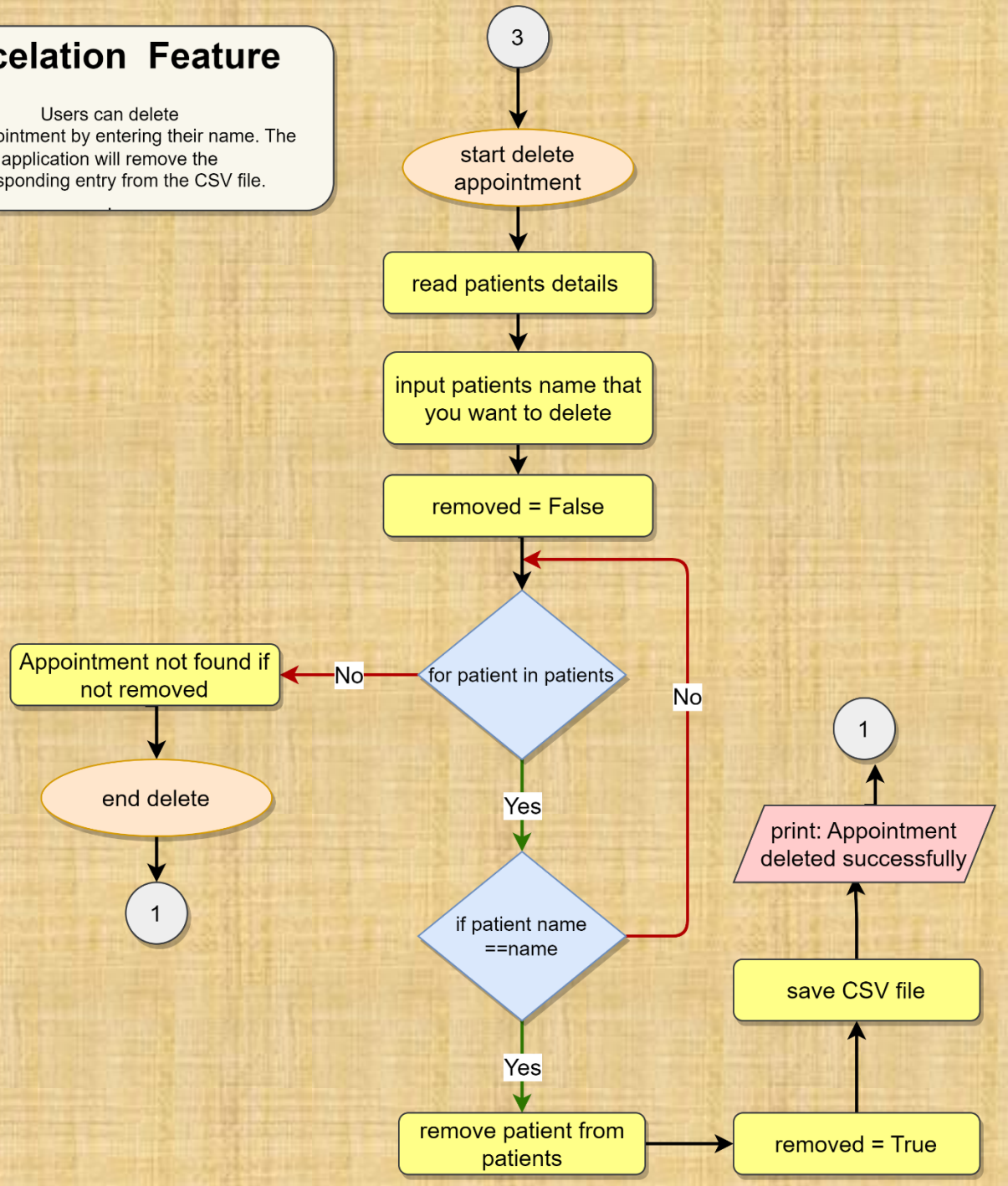
Booking Feature

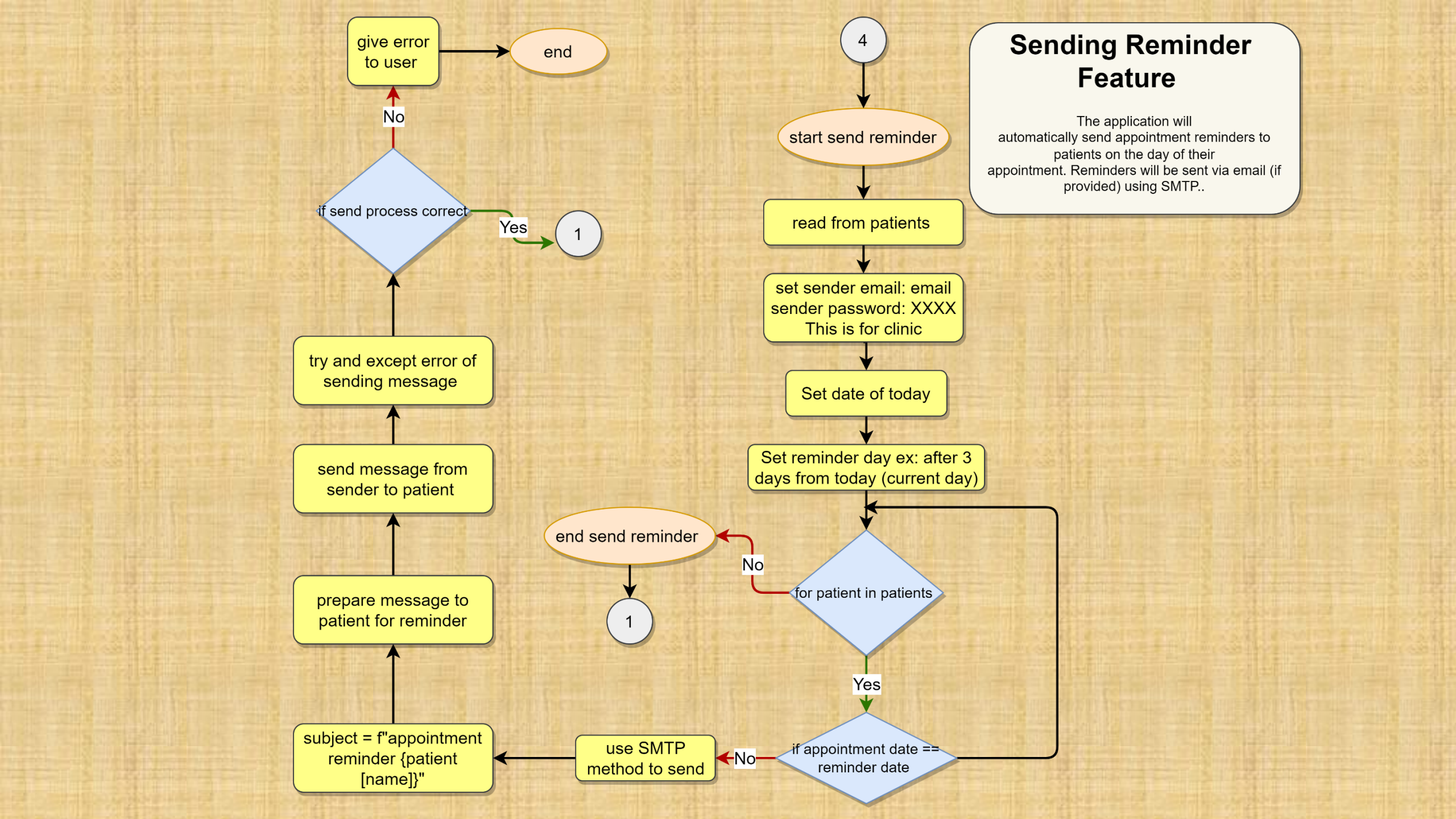
Users can enter
their name, contact number, email (optional), and desired appointment date. The application will save this information to the CSV file.



Cancelation Feature

Users can delete
their appointment by entering their name. The application will remove the corresponding entry from the CSV file.




Sending Reminder Feature
The application will automatically send appointment reminders to patients on the day of their appointment. Reminders will be sent via email (if provided) using SMTP..
4
start send reminder
read from patients
set sender email: email
sender password: XXXX
This is for clinic
Set date of today
Set reminder day ex: after 3 days from today (current day)
for patient in patients
No
end send reminder
1
Yes
if appointment date == reminder date
No
use SMTP method to send
subject = f"appointment reminder {patient [name]}"
prepare message to patient for reminder
send message from sender to patient
try and except error of sending message
if send process correct
No
give error to user
end
Yes
1

Over view

- Main driver for the clinic
- Entry point for the program
- Continuously display the menu options

main.py > ...

```python
from assistant import save_patients,view_available_days, book_appointment, send_reminders, delete_appointment
from load_data import load_patients
from display import display_menu
#Define the CSV file path here
CSV_FILE_PATH = 'clinic.csv'
#  Main function to run the program
def main():

    patients = load_patients(CSV_FILE_PATH)
    #patients = load_patients(CSV_FILE_PATH)

    while True:
        display_menu()
        choice = input("Enter your choice (1-5): ")

        if choice == '1':
            available_dates = view_available_days(patients)
            if len(available_dates) == 0:
                print("No available dates found.")
        elif choice == '2':
            book_appointment(patients)
        elif choice == '3':
            delete_appointment(patients)
        elif choice == '4':
            send_reminders(patients)
        elif choice == '5':
            save_patients(CSV_FILE_PATH, patients)
            print("Exiting Clinic Appointment Booking and Reminder. Goodbye!")
            break
        else:
            print("Invalid choice. Please try again.")

if __name__ == '__main__':
    main()
```

Display menu Function

- Display banner or logo
- Display "Clinic Appointment Booking and Reminder."
- View Available Days
- Book Appointment
- Delete Appointment
- Reminder:
- Quit

display.py > ...

```python
#Function to display the menu options

def display_menu():
    print("                                                                    ")
    print("      ___   _  _              _                _     ___   ___     ")
    print("     / __| | |(_) _ __   (_)  __          / \   | _ \ | _ \    ")
    print("    | |    | | | | '_ \  | | / _| ____   / _ \  | |_) | | |_) |   ")
    print("    | |__  | | | | | | | | | (__ |____| / ___ \ | __/  | __/    ")
    print("     \___| |_| |_| |_| |_| |_|  \__|    /_/   \_\|_|    |_|      ")

    print("                                                                    ")
    print("Clinic Appointment Booking and Reminder")
    print("1. View Available Days")
    print("2. Book Appointment")
    print("3. Delete Appointment")
    print("4. Reminder")
    print("5. Quit")
```

```
 ____  _  _              _                _     ____   ____
/ ___| | |(_) _ __   (_)  __          / \   |  _ \ |  _ \
| |    | | | | '_ \  | | / _| _____  / _ \  | |_) | | |_) |
| |__  | | | | | | | | | (__ |_____| / ___ \ |  __/  |  __/
\____| |_| |_| |_| |_| |_|  \__|    /_/   \_\|_|     |_|

Clinic Appointment Booking and Reminder
1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5):
```

Load Patients Function

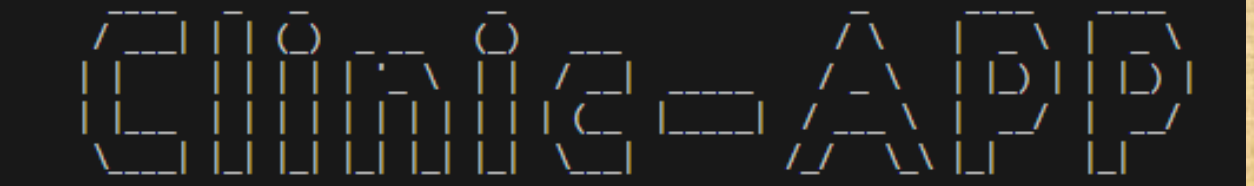
load_data.py > ...

```python
import csv
# Importing the csv module to work with CSV files

# Function definition for load_patients, which takes a filename as input CSV file
def load_patients(filename):
    # Initializing an empty list to store the patient records
    patients = []
    with open(filename, 'r') as file:
        # Opening the specified CSV file in read mode
        reader = csv.DictReader(file)
        # Creating a csv.DictReader object to read the contents of the file as a dictionary
        for row in reader:
            patients.append(row)
    return patients
    # Returning the list of patient records after reading and processing the CSV file

```

- **Load patient data from a CSV file**
- **DictReader class to read the CSV file**
- **iterates through each row and appends it to the patients**

View Available Days Feature

```
Available Days:
2023-07-04
2023-07-05
2023-07-10
2023-07-12
2023-07-13
2023-07-14
2023-07-15
2023-07-16
```



```
1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5):
```

```python
# Function to display the available days
def view_available_days(patients):
    # store available days in list
    available_dates = []
    # Get the start and end dates for the week
    today = datetime.date.today()
    next_week_start = today + datetime.timedelta(days=(7 - today.weekday()))
    next_week_end = next_week_start + datetime.timedelta(days=6)

    print("Available Days:")
    # here I use nested loops to iterate in patients list file within period from today to next week end
    current_date = today

    while current_date <= next_week_end:


        has_appointment = False
        for patient in patients:
            appointment_date = datetime.datetime.strptime(str(patient['appointment_date']), '%Y-%m-%d').date()
            if appointment_date == current_date:
                has_appointment = True
                break
        if not has_appointment:
            # here we can add if the day is free from apointment to the available list
            # this will aid us to use it in booking function
            available_dates.append(current_date)
            print(current_date.strftime("%Y-%m-%d"))
        current_date += datetime.timedelta(days=1)

    return available_dates
```

Booking Feature

```python
# Function for booking an appointment
def book_appointment(patients):
    # Gather patient information
    name = input("Enter patient's name: ")
    contact_number = input("Enter patient's contact number: ")
    email = input("Enter patient's email address (optional): ")
    # I wrped those sentences to repeat until user put correct date
    while True:
        appointment_date = input("Enter appointment date (YYYY-MM-DD): ")
        try:
            #appointment_date = datetime.datetime.strptime(appointment_date, '%Y-%m-%d').date()
            datetime.datetime.strptime(appointment_date, '%Y-%m-%d')
        except ValueError:
            print("Invalid date format. Please enter the date in the format YYYY-MM-DD.")
            continue
        # Check if the appointment date is within the available days
        available_dates = view_available_days(patients)
        appointment_date = datetime.datetime.strptime(appointment_date, '%Y-%m-%d').date()
        if appointment_date not in available_dates:
            print("Please choose another day. This day is not available.")
            #here we can give the user to choose continue or go to menu again
            choice = input("Do you want to try again? (Y/N): ")
            while choice.lower() not in ['y', 'n']:
                print("Please choose 'Y' for Yes or 'N' for No.")
                choice = input("Do you want to try again? (Y/N): ")
            if choice.lower() == 'n':
                return
            #return
            # here we can skip because the day is not available
            continue
        # Create a patient dictionary
        patient = {
            'name': name,
            'contact_number': contact_number,
            'email': email,
            'appointment_date': appointment_date
        }
        # Add the patient to the list and save to file
        patients.append(patient)
        save_patients(CSV_FILE_PATH, patients)
        print("Appointment booked successfully!")
        # Send a reminder to the patient
        #send_reminder_email(patient)
        break
```

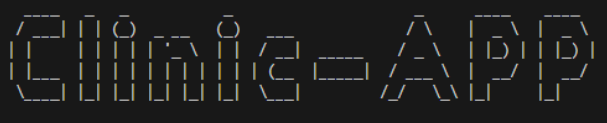
assistant.py | display.py | clinic.csv | main.py

clinic.csv

```
name,contact_number,email,appointment_date
Jhon,0780898989,ramizibraheem20@gmail.com,2023-07-06
Peter,07757575,ramizibraheem20@gmail.com,2023-07-07
Sara,07757575,ramizibraheem20@gmail.com,2023-07-08
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
→ src git:(main) X ./script.sh
```



```
1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5): 2
Enter patient's name: Reem
Enter patient's contact number: 87878787
Enter patient's email address (optional): ramizibraheem20@gmail.com
Enter appointment date (YYYY-MM-DD): 2023-07-09
```

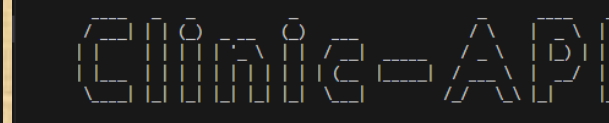
assistant.py | display.py | clinic.csv | main.py

clinic.csv

```
name,contact_number,email,appointment_date
Jhon,0780898989,ramizibraheem20@gmail.com,2023-07-06
Peter,07757575,ramizibraheem20@gmail.com,2023-07-07
Sara,07757575,ramizibraheem20@gmail.com,2023-07-08
Reem,87878787,ramizibraheem20@gmail.com,2023-07-09
```

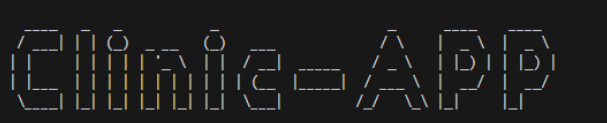
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
→ src git:(main) X ./script.sh
```



```
1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5): 2
Enter patient's name: Paul
Enter patient's contact number: 7676767
Enter patient's email address (optional): ramizibraheem20@gmail.com
Enter appointment date (YYYY-MM-DD): 2023-07-09
```

```
Appointment booked successfully!
```



```
1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5):
```

```
Enter your choice (1-5): 2
Enter patient's name: Paul
Enter patient's contact number: 7676767
Enter patient's email address (optional): ramizibraheem20@gmail.com
Enter appointment date (YYYY-MM-DD): 2023-07-09
Available Days:
2023-07-04
2023-07-05
2023-07-10
2023-07-11
2023-07-12
2023-07-13
2023-07-14
2023-07-15
2023-07-16
Please choose another day. This day is not available.
Do you want to try again? (Y/N):
```

Cancel Feature

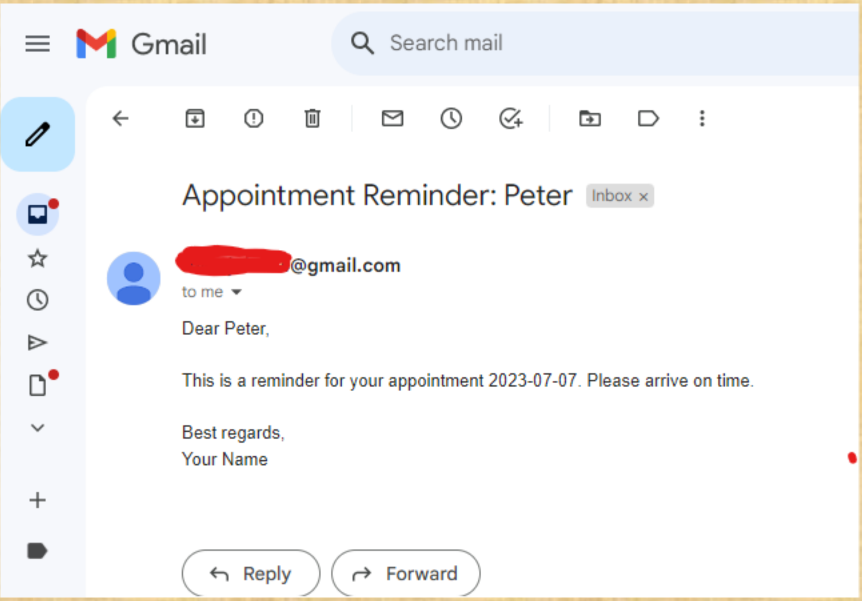
```python
# Function for canceling an appointment
def delete_appointment(patients):
    # Get the name of the patient to delete
    name = input("Enter patient's name to delete the appointment: ")
    removed = False
    for patient in patients:
        if patient['name'] == name:
            patients.remove(patient)
            removed = True
            save_patients(CSV_FILE_PATH, patients)
            print("Appointment deleted successfully!")
            break
    if not removed:
        print("Appointment not found!")


```

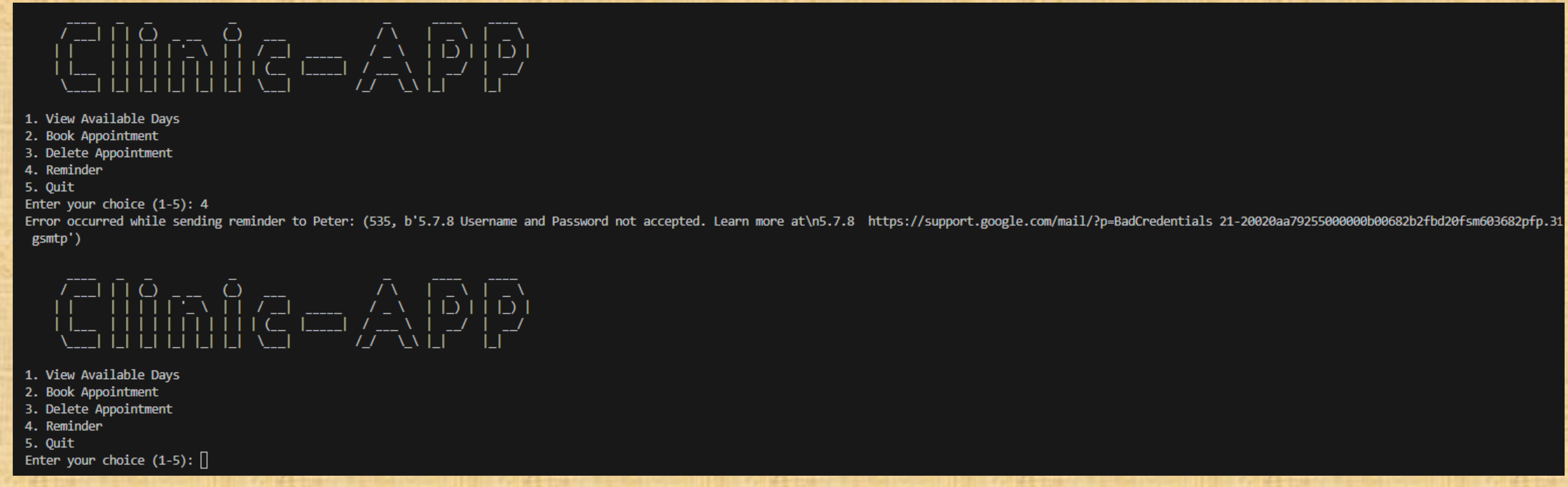
```
1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5): 3
Enter patient's name to delete the appointment: John
Appointment deleted successfully!
```

Reminder Feature

```python
# Function to remind the patient
def send_reminders(patients):
    # sender_email = 'your_email@example.com'
    # sender_password = 'your_password'
    sender_email = 'your_email@example.com'
    sender_password = 'your_password'
    #sender_password=print("Enter your password, ")
    today = datetime.date.today()
    # here you can use suitable rminder after 2, 3, or 4 days from current day
    reminder_date = today + datetime.timedelta(days=3)  # Calculate the reminder date
    for patient in patients:
        appointment_date = datetime.datetime.strptime(str(patient['appointment_date']), '%Y-%m-%d').date()
        if appointment_date == reminder_date:
        #if appointment_date == today:
            subject = f"Appointment Reminder: {patient['name']}"
            message = f"Dear {patient['name']},\n\nThis is a reminder for your appointment {reminder_date}. Please arrive on time.\n\nBest regards,\nYour Name"

            msg = MIMEText(message)
            msg['Subject'] = subject
            msg['From'] = sender_email
            msg['To'] = patient['email']

            try:
                with smtplib.SMTP('smtp.gmail.com', 587) as server:
                    server.starttls()
                    server.login(sender_email, sender_password)
                    #server.sendmail(sender_email, "example0@gmail.com", msg.as_string("Hello again"))
                    server.sendmail(sender_email, patient['email'], msg.as_string())
                print(f"Reminder sent to {patient['name']}!")
            except Exception as e:
                print(f"Error occurred while sending reminder to {patient['name']}: {str(e)}")
```


1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5): 4
Reminder sent to Peter!
1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5):


Gmail
Search mail
Appointment Reminder: Peter
Inbox
[redacted]@gmail.com
to me
Dear Peter,
This is a reminder for your appointment 2023-07-07. Please arrive on time.
Best regards,
Your Name
Reply
Forward


1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5): 4
Error occurred while sending reminder to Peter: (535, b'5.7.8 Username and Password not accepted. Learn more at\n5.7.8 https://support.google.com/mail/?p=BadCredentials 21-20020aa79255000000b00682b2fbd20fsm603682pfp.31
gsmtp')
1. View Available Days
2. Book Appointment
3. Delete Appointment
4. Reminder
5. Quit
Enter your choice (1-5):

Testing

- **Manual testing was performed for 3 Features**
- **three test cases for each feature**
- **Purpose is to verify the expected behavior of the features**
- **For more information on the test cases and their results, please see Clinic App Test spreadsheet available at the following link: https://docs.google.com/spreadsheets/d/1ZjEnXHz5Oj8te3-MvndX3ExZW5dWXd8UU81QsOg-K2Q/edit#gid=1645781014**

Challenges:

- **Dealing with different types of data, especially handling string and time formats correctly.**
- **Refactoring the code to enhance simplicity and readability. Ensuring that the code is well-commented and logically organized.**
- **Employing a modular approach for efficient project management, separating functionality into distinct modules or functions.**
- **Exploring methods to send messages from the application to email addresses using the SMTP protocol, ensuring successful email delivery.**

Favorite part
Sending reminder for appointment