```
import tkinter as tk
from tkinter import ttk, messagebox,
Simpledialog
import json
import hashlib
import os
\#----------------------------
\#User Authentication System
\#----------------------------
USER\_FILE = "users.json"
def load\_users():
"""Load existing users from file."""
if not os.path.exists(USER\_FILE):
return {}
with open(USER\_FILE, "r") as f:
return json.load(f)
def save\_users(users):
"""Save users to file."""
with open(USER\_FILE, "w") as f:
json.dump(users, f, indent=4)
def hash\_password(password):
"""Return hashed password."""
return
hashlib.sha256(password.encode()).hexdigest()
def signup():
"""Sign up a new user."""
username = simpledialog.askstring("Sign Up", "Enter a username:")
if not username:
return
password = simpledialog.askstring("Sign Up", "Enter a password:", show="*")
if not password:
return
users = load\_users()
if username in users:
messagebox.showerror("Error", "Username already exists!")
return
users\[username] = hash\_password(password)
save\_users(users)
messagebox.showinfo("Success", "Signup successful! Please log in.")
def login():
"""Login existing user."""
username = simpledialog.askstring("Login", "Enter username:")
if not username:
return None
password = simpledialog.askstring("Login", "Enter password:", show="*")
if not password:
return None
users = load\_users()
```

```
if username in users and users\[username] == hash\_password(password):
messagebox.showinfo("Success", "Login successful!")
return username
else:
messagebox.showerror("Error", "Invalid credentials!")
return None
\#Flight Data and Core Functions
\#----------------------------
flights = \[ {"flight\_id": 1, "airline": "Airline A", "from": "New York", "to": "London", "price": 500,
"available\_seats": 10}, {"flight\_id": 2, "airline": "Airline B", "from": "Los Angeles", "to":
"Tokyo",
"price": 800, "available\_seats": 5}, {"flight\_id": 3, "airline": "Airline C", "from": "San
Francisco", "to":
"Paris", "price": 650, "available\_seats": 3}, ]
def search\_flights(from\_city, to\_city):
"""Return a list of flights matching the departure and destination cities."""
return \[flight for flight in flights if flight\["from"].lower() == from\_city.lower() and
flight\["to"].lower() ==
to\_city.lower()]
def book\_ticket(flight\_id, num\_tickets):
"""Attempt to book the specified number of tickets."""
for flight in flights:
if flight\["flight\_id"] == flight\_id:
if flight\["available\_seats"] >= num\_tickets:
flight\["available\_seats"] -= num\_tickets return f"Successfully booked {num\_tickets}
ticket(s)
on {flight\['airline']} flight {flight\_id}."
else:
return "Not enough available seats." return "Invalid flight ID."
\#----------------------------
\#Payment Processing
\#----------------------------
def payment\_process(amount):
"""Simulate a payment process."""" card\_number =
simpledialog.askstring("Payment", "Enter your card number (16 digits):", show="*")
if not card\_number or len(card\_number) != 16: messagebox.showerror("Error", "Invalid card
number!") return False
cvv = simpledialog.askstring("Payment", "Enter CVV (3 digits):", show="*")
if not cvv or len(cvv) != 3:
messagebox.showerror("Error", "Invalid CVV!")
return False
messagebox.showinfo("Payment Successful", f"Payment of \${amount} completed
successfully!")
return True
\#----------------------------
\#Tkinter GUI Application
\#----------------------------
def main():
```

```
root = tk.Tk()
root.title("Flight Booking System")

# --- User Authentication ---

logged\_in\_user = None
while not logged\_in\_user:
choice = messagebox.askquestion("Login or Signup", "Do you have an
account?")
if choice == "yes":
logged\_in\_user = login()
else:
signup()

# --- Search Frame ---

search\_frame = tk.Frame(root, padx=10, pady=10)
search\_frame.pack(fill=tk.X)
tk.Label(search\_frame, text="From:").grid(row=0, column=0, padx=5, pady=5,
sticky=tk.W)
from\_entry = tk.Entry(search\_frame)
from\_entry.grid(row=0, column=1, padx=5, pady=5)
tk.Label(search\_frame, text="To:").grid(row=0, column=2, padx=5, pady=5,
sticky=tk.W)
to\_entry = tk.Entry(search\_frame)
to\_entry.grid(row=0, column=3, padx=5, pady=5)
def perform\_search():
"""Search flights based on user input."""
from\_city = from\_entry.get().strip()
to\_city = to\_entry.get().strip()
if not from\_city or not to\_city:
messagebox.showerror("Error", "Enter both cities!")
results = search\_flights(from\_city, to\_city)
for i in tree.get\_children():
tree.delete(i)
if not results:
messagebox.showinfo("No Flights", "No flights found for this
route.")
else:
for flight in results:
tree.insert("", tk.END, values=(
flight\["flight\_id"], flight\["airline"],
flight\["from"], flight\["to"], f"\${flight\['price']}", flight\["available\_seats"]
))
search\_button = tk.Button(search\_frame, text="Search Flights",
command=perform\_search)
search\_button.grid(row=0, column=4, padx=5, pady=5)
```

```
# --- Results Frame ---

results\_frame = tk.Frame(root, padx=10, pady=10)
results\_frame.pack(fill=tk.BOTH, expand=True)
columns = ("flight\_id", "airline", "from", "to", "price",
"available\_seats")
tree = ttk.Treeview(results\_frame, columns=columns, show="headings")
for col in columns:
tree.heading(col, text=col.capitalize())
tree.column(col, anchor=tk.CENTER)
tree.pack(fill=tk.BOTH, expand=True)

# --- Booking Button ---

def book\_selected\_ticket():
"""Book tickets and process payment."""
selected = tree.selection()
if not selected:
messagebox.showerror("Error", "Select a flight to
book!")
return
flight\_values = tree.item(selected\[0])\["values"]
flight\_id = flight\_values\[0]
price = int(flight\_values\[4]\[1:])
available\_seats = flight\_values\[5]
num\_tickets = simpledialog.askinteger("Book Ticket", f"Enter tickets
(Available: {available\_seats}):", minvalue=1, maxvalue=available\_seats)
if num\_tickets is None:
return
total\_amount = price \* num\_tickets
result = book\_ticket(flight\_id, num\_tickets)
messagebox.showinfo("Booking", result)

perform\_search()

book\_button = tk.Button(root, text="Book Selected Flight",

command=book\_selected\_ticket)

book\_button.pack(pady=10)

root.mainloop()

if \_\_name \_\_== "**main**":

main()
if payment\_process(total\_amount):
return
```