Group project
Hotel reservation system

Course: Introduction to Computer Science (Sec. D)

Lecturer : Gagik Khalafyan

Presenters: Hakob Janesian
Hayk Shahsuvaryan
Elya Hovsepyan
Mari Tadevosyan

Overview

- Console-based application for **hotel room reservation** .
- Developed entirely in **Python**.
- Utilizes four key datasets: **booking**, **country**, **guest**, and **hotel** data.
- Features include user **authentication**, **viewing**, **creating**, and **canceling** bookings.

Data (tables)

hotel_data.csv

- hotel_id
- hotel_name
- hotel_address
- hotel_url
- hotel_email
- hotel_phone_number
- hotel_star
- country_code

guest_data

- guest_id
- guest_first_name
- guest_last_name
- guest_email
- guest_age
- password

Data (tables)

booking_data.csv

- booking_id
- guest_email
- country_name
- hotel_name
- adults
- children
- status
- check_in
- check_out
- room_type

country_data

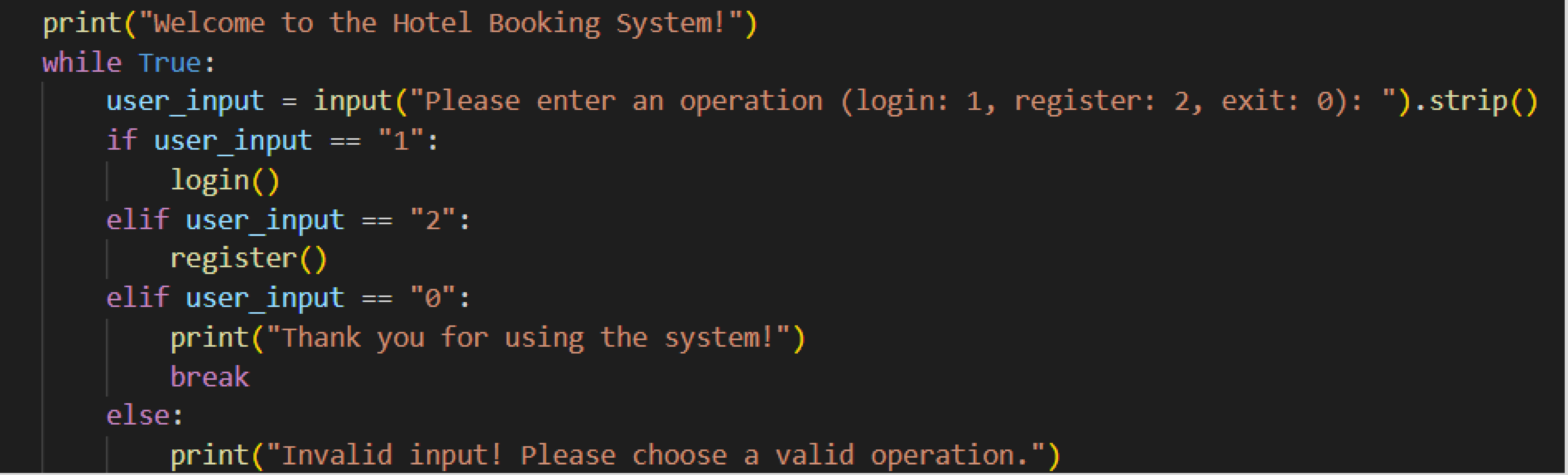
- country_code
- country_name

List of the countries

Greece, Iran, Italy, Japan, Monaco, Oman, Netherlands, Peru, Qatar, Slovenia.

System architecture

- **Entry Point:**
  - ***main.py*** *serves as the application's starting point.*
  - *User choices for login, registration, or exit are handled here.*
- **Operations Folder:**
  - ***Contains key functionalities split into four scripts:***
    - **authentication.py** for user login and registration.
    - **view.py** for displaying menus and handling user choices.
    - **crud.py** for creating, reading, updating, and canceling bookings.
    - **validation.py** for data validation (room types, dates, etc.).
- **Utils Folder:**
  - ***utils.py*** *provides shared data handling utilities.*
  - *Functions for CSV file operations (read, write, transform) are centralized here.*



```python
print("Welcome to the Hotel Booking System!")
while True:
    user_input = input("Please enter an operation (login: 1, register: 2, exit: 0): ").strip()
    if user_input == "1":
        login()
    elif user_input == "2":
        register()
    elif user_input == "0":
        print("Thank you for using the system!")
        break
    else:
        print("Invalid input! Please choose a valid operation.")
```

- **Main Functionality: Manages user interactions for hotel booking operations like login, registration, and exit.**
- **Use of while True: Ensures continuous user engagement until they choose to exit.**
- **Input Options:**

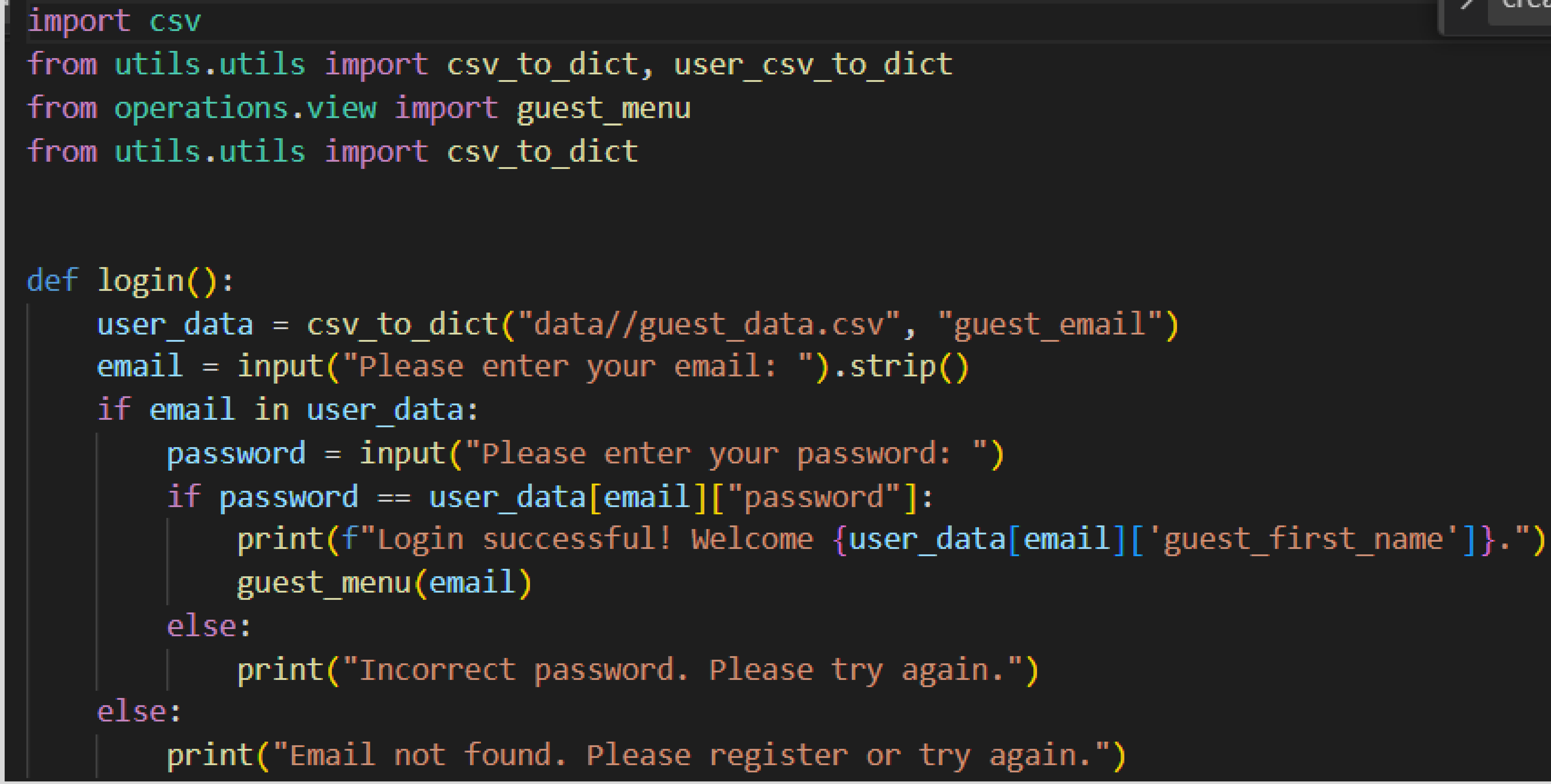
  "1" for Login: Access user-specific booking features.

  "2" for Register: New users can create an account.

  "0" to Exit: Safely exits the application.

main.py

authentication.py (login logic)



```python
import csv
from utils.utils import csv_to_dict, user_csv_to_dict
from operations.view import guest_menu
from utils.utils import csv_to_dict


def login():
    user_data = csv_to_dict("data//guest_data.csv", "guest_email")
    email = input("Please enter your email: ").strip()
    if email in user_data:
        password = input("Please enter your password: ")
        if password == user_data[email]["password"]:
            print(f"Login successful! Welcome {user_data[email]['guest_first_name']}.")
            guest_menu(email)
        else:
            print("Incorrect password. Please try again.")
    else:
        print("Email not found. Please register or try again.")
```

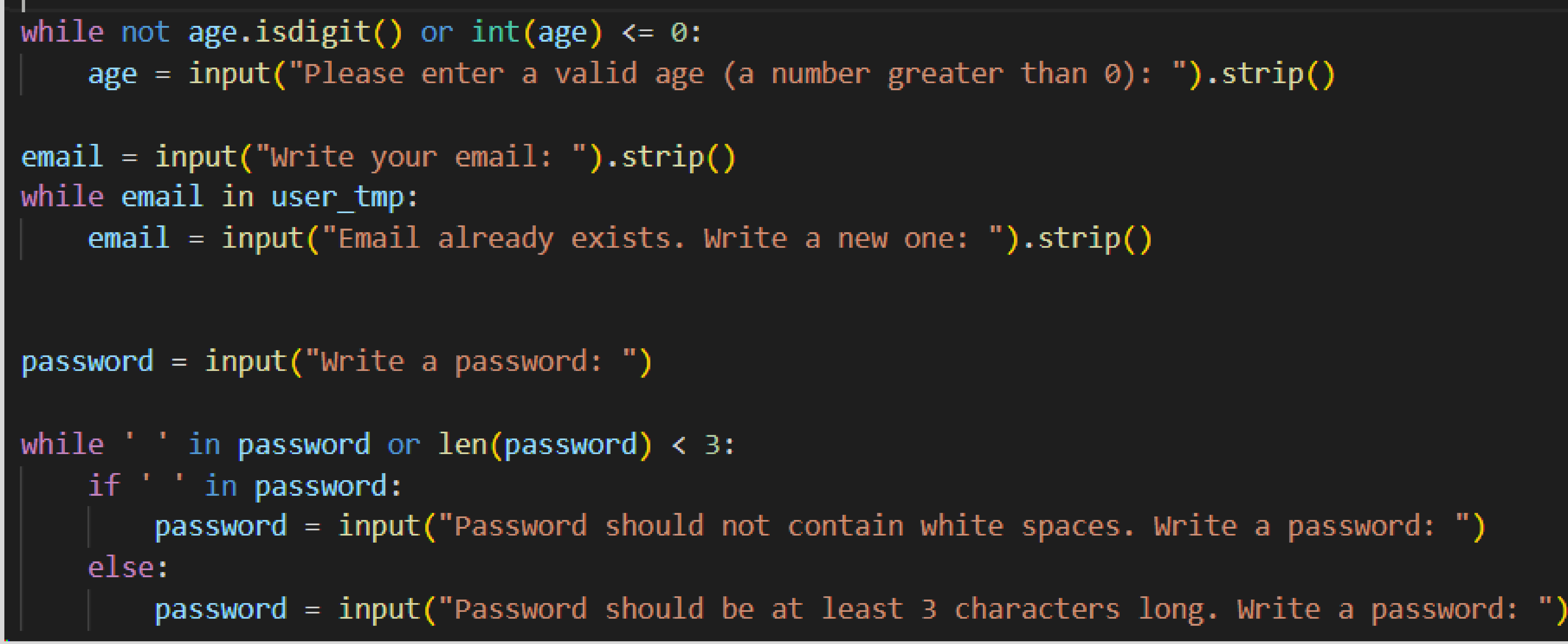
Login Overview

1. Essential for accessing personalized booking features.
2. Validates user's email and password

Login Flow:

1. User inputs email.
2. User inputs password.
3. System checks credentials against the database.

authentication.py (registration logic)



```python
while not age.isdigit() or int(age) <= 0:
    age = input("Please enter a valid age (a number greater than 0): ").strip()

email = input("Write your email: ").strip()
while email in user_tmp:
    email = input("Email already exists. Write a new one: ").strip()


password = input("Write a password: ")

while ' ' in password or len(password) < 3:
    if ' ' in password:
        password = input("Password should not contain white spaces. Write a password: ")
    else:
        password = input("Password should be at least 3 characters long. Write a password: ")
```

Registration Steps:

- Users input personal details: **name**, **surname**, **age**, **email and password**.

Password Requirements:

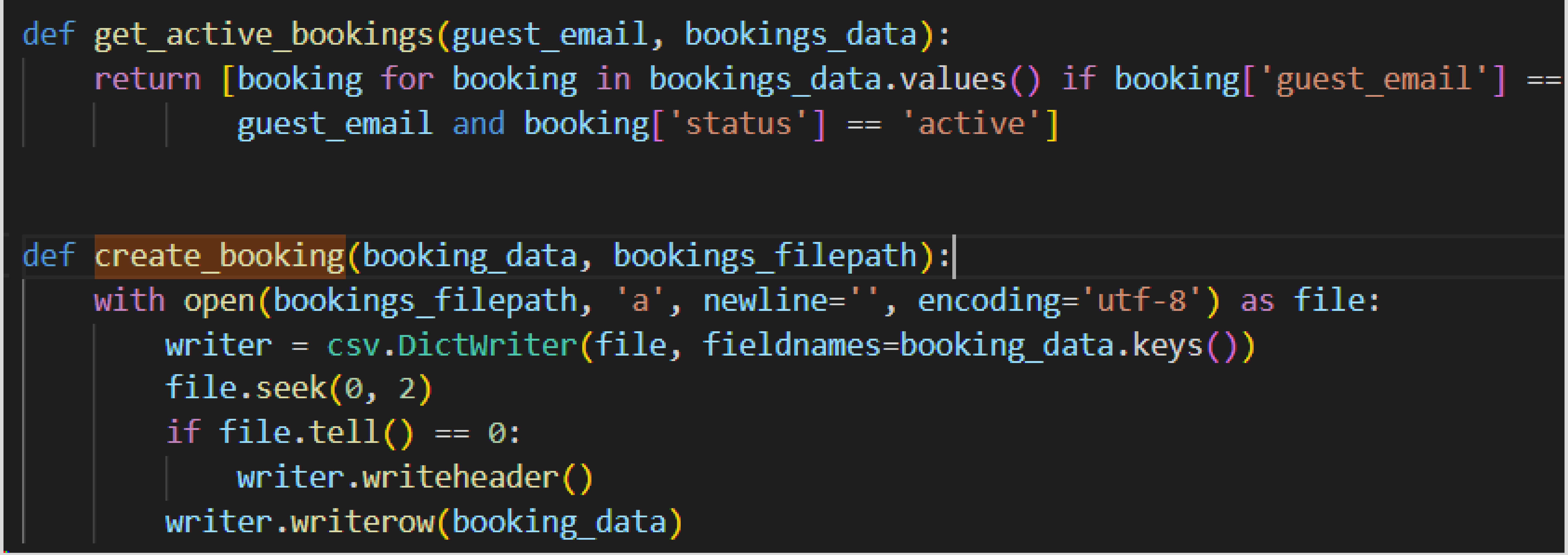
- Must be at **least 3 characters long**.
- Cannot contain white spaces.
- Prompts user to re-enter if requirements are not met.

Age and Email Validation:

- Age must be a positive number.
- Email must be unique within the system.

Code Deep-Dive:

- user_csv_to_dict checks existing emails for uniqueness.
- Loops for age and password validation.



```python
def get_active_bookings(guest_email, bookings_data):
    return [booking for booking in bookings_data.values() if booking['guest_email'] ==
            guest_email and booking['status'] == 'active']


def create_booking(booking_data, bookings_filepath):
    with open(bookings_filepath, 'a', newline='', encoding='utf-8') as file:
        writer = csv.DictWriter(file, fieldnames=booking_data.keys())
        file.seek(0, 2)
        if file.tell() == 0:
            writer.writeheader()
        writer.writerow(booking_data)
```

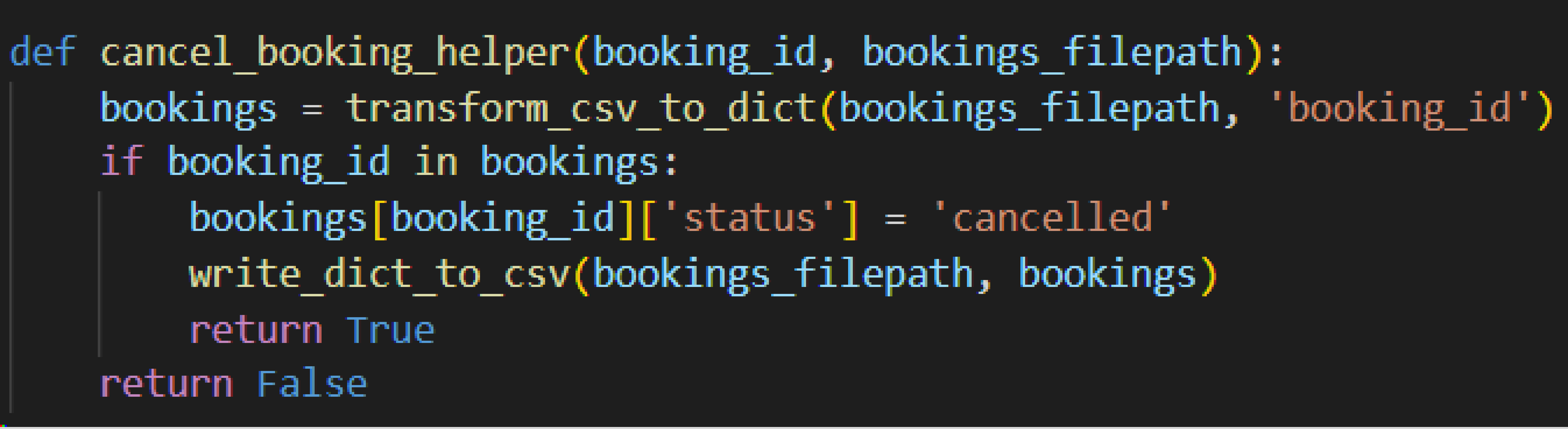
crud.py (create booking & active bookings retrieval)

Creating a Booking:

- Function: **create_booking**.
- Adds **new bookings** to the system.
- Utilizes **csv.DictWriter** for data integrity.

Active Bookings Retrieval:

- Function: **get_active_bookings**.
- Fetches user's active bookings.
- Filters by **guest_email** and '**active**' status.



```python
def cancel_booking_helper(booking_id, bookings_filepath):
    bookings = transform_csv_to_dict(bookings_filepath, 'booking_id')
    if booking_id in bookings:
        bookings[booking_id]['status'] = 'cancelled'
        write_dict_to_csv(bookings_filepath, bookings)
        return True
    return False
```

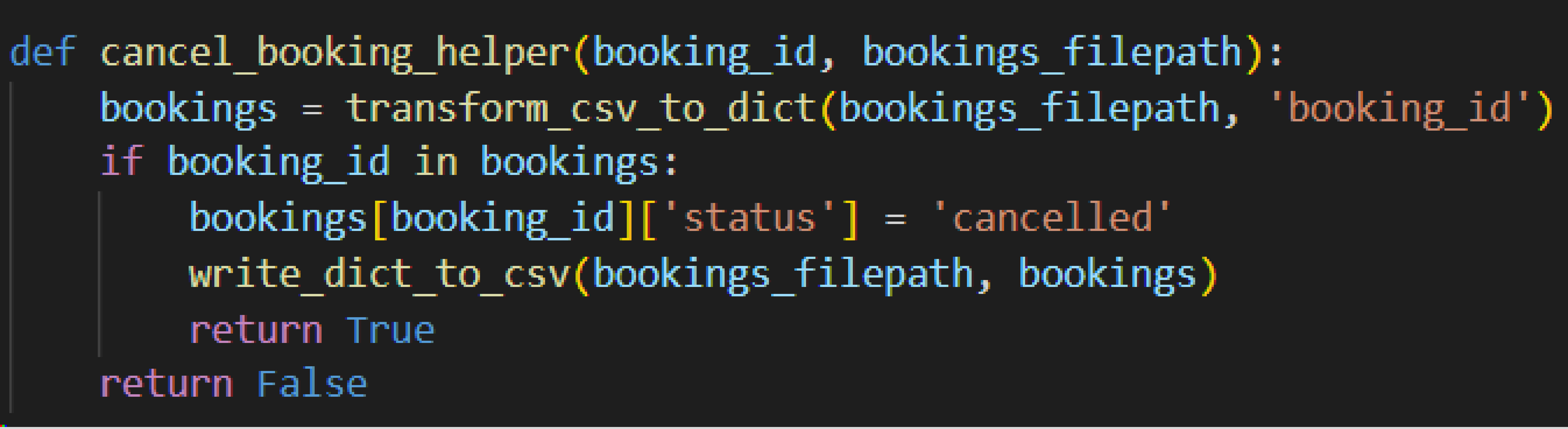
crud.py (cancelling bookings)

Process:

- Identifies the booking by **booking_id**.
- Changes the status of the booking to '**cancelled**'.
- Updates the booking data in the system.

Code Mechanics:

- Retrieves all bookings and locates the specific booking using the **booking_id**.
- Modifies the booking's status in the booking dictionary.
- Writes the updated data back to the CSV file using **write_dict_to_csv**.



```
def cancel_booking_helper(booking_id, bookings_filepath):
    bookings = transform_csv_to_dict(bookings_filepath, 'booking_id')
    if booking_id in bookings:
        bookings[booking_id]['status'] = 'cancelled'
        write_dict_to_csv(bookings_filepath, bookings)
        return True
    return False
```

crud.py (cancelling bookings)

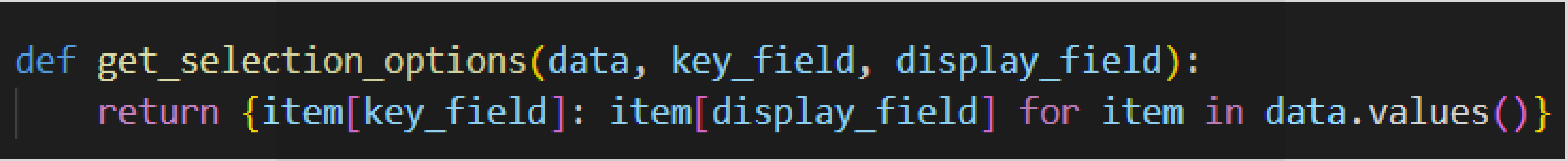
Process:

- Identifies the booking by **booking_id**.
- Changes the status of the booking to '**cancelled**'.
- Updates the booking data in the system.

Code Mechanics:

- Retrieves all bookings and locates the specific booking using the **booking_id**.
- Modifies the booking's status in the booking dictionary.
- Writes the updated data back to the CSV file using **write_dict_to_csv**.

crud.py (selecting booking options)



```
def get_selection_options(data, key_field, display_field):
    return {item[key_field]: item[display_field] for item in data.values()}
```

- **Usage: Facilitates user interaction by presenting selectable options (e.g., hotel selection) in a user-friendly format.**

- **How It Works:**
  - **Takes structured data (like hotel listings) and creates a key-value pair for easier selection.**
  - **Enhances user experience by streamlining the selection process during booking creation.**

validation.py (validating room and country Details)

Room Type Validation:

*Function: check_room_type***.**

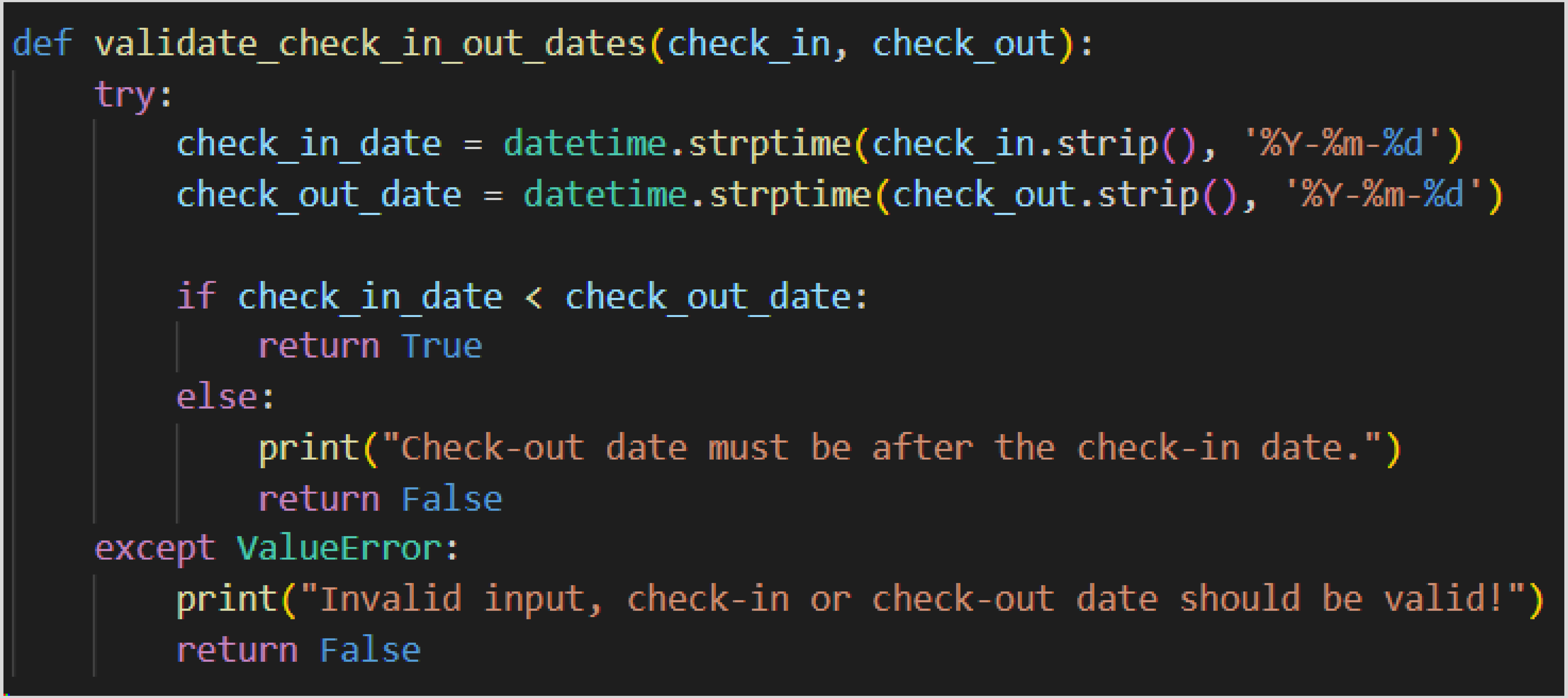
- Validates against types like **"Single", "Double", "Triple", "Quad", "Queen", "King", "Twin"**
- Handles single and list inputs.

Country Code and Name Validation:

Functions: check_country_code and check_country_name.

- Ensures country details are from **predefined lists**.

validation.py (validating check-in & check-out dates)

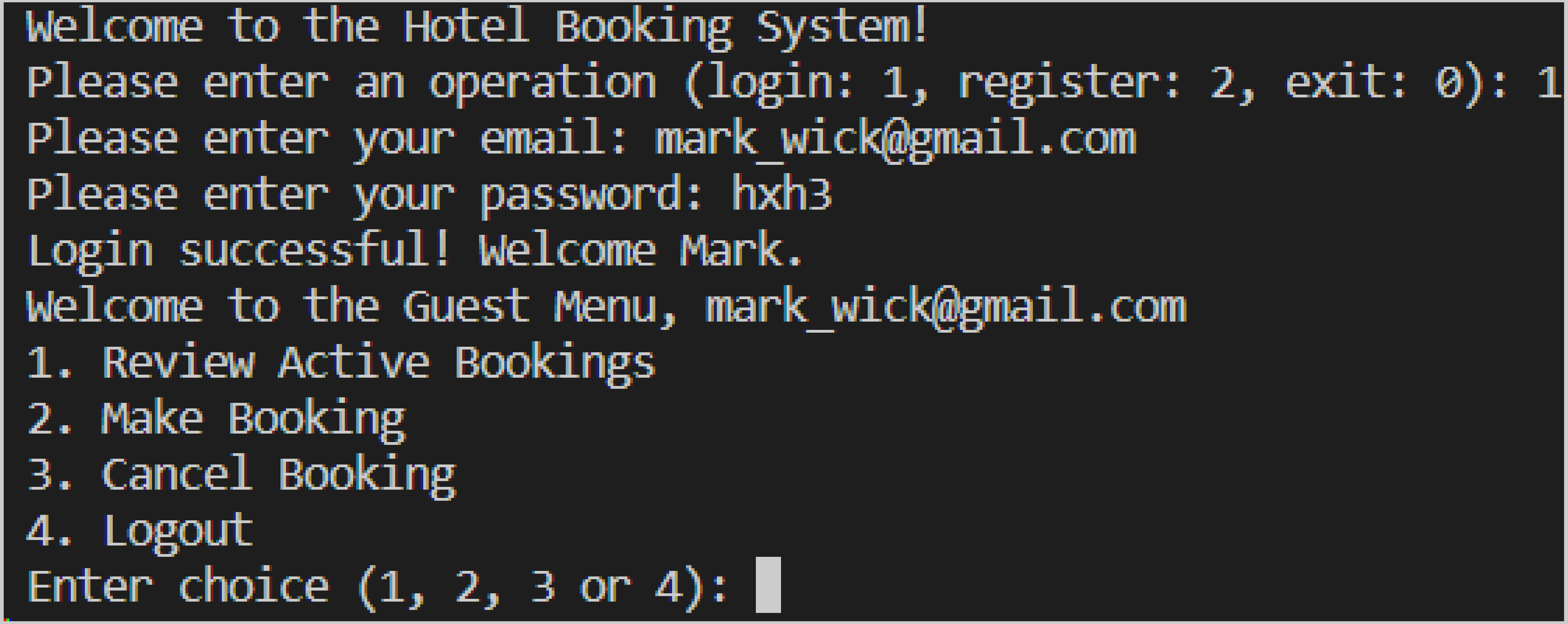


```python
def validate_check_in_out_dates(check_in, check_out):
    try:
        check_in_date = datetime.strptime(check_in.strip(), '%Y-%m-%d')
        check_out_date = datetime.strptime(check_out.strip(), '%Y-%m-%d')

        if check_in_date < check_out_date:
            return True
        else:
            print("Check-out date must be after the check-in date.")
            return False
    except ValueError:
        print("Invalid input, check-in or check-out date should be valid!")
        return False
```

- Ensures check-in date is before check-out date.
- Validates proper date format (YYYY-MM-DD).

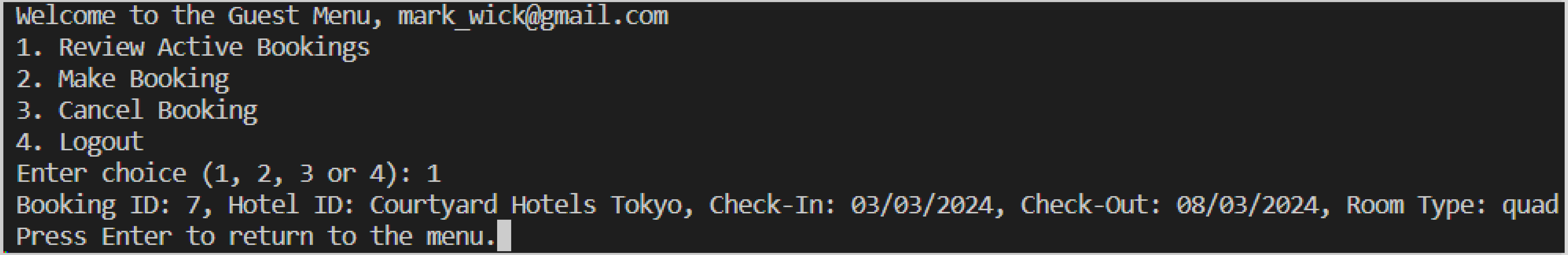
view.py



```
Welcome to the Hotel Booking System!
Please enter an operation (login: 1, register: 2, exit: 0): 1
Please enter your email: mark_wick@gmail.com
Please enter your password: hxh3
Login successful! Welcome Mark.
Welcome to the Guest Menu, mark_wick@gmail.com
1. Review Active Bookings
2. Make Booking
3. Cancel Booking
4. Logout
Enter choice (1, 2, 3 or 4): 
```

- **Entry point for users post-login.**
- **Offers options: review bookings, make a booking, cancel a booking, logout.**

view.py

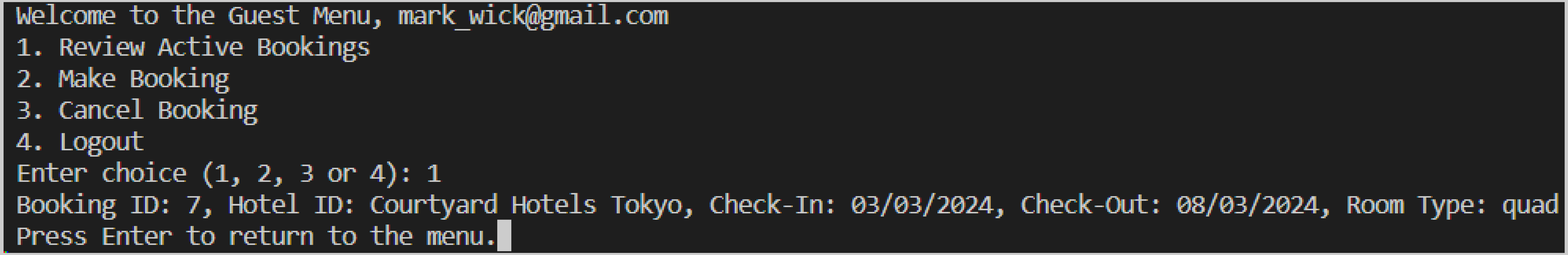


```
Welcome to the Guest Menu, mark_wick@gmail.com
1. Review Active Bookings
2. Make Booking
3. Cancel Booking
4. Logout
Enter choice (1, 2, 3 or 4): 1
Booking ID: 7, Hotel ID: Courtyard Hotels Tokyo, Check-In: 03/03/2024, Check-Out: 08/03/2024, Room Type: quad
Press Enter to return to the menu.
```

Reviewing Active Bookings:

- **Function: review_active_bookings.**
- **Displays all active bookings linked to the user's email.**
- **Formats and presents booking details.**

view.py



```
Welcome to the Guest Menu, mark_wick@gmail.com
1. Review Active Bookings
2. Make Booking
3. Cancel Booking
4. Logout
Enter choice (1, 2, 3 or 4): 1
Booking ID: 7, Hotel ID: Courtyard Hotels Tokyo, Check-In: 03/03/2024, Check-Out: 08/03/2024, Room Type: quad
Press Enter to return to the menu.
```

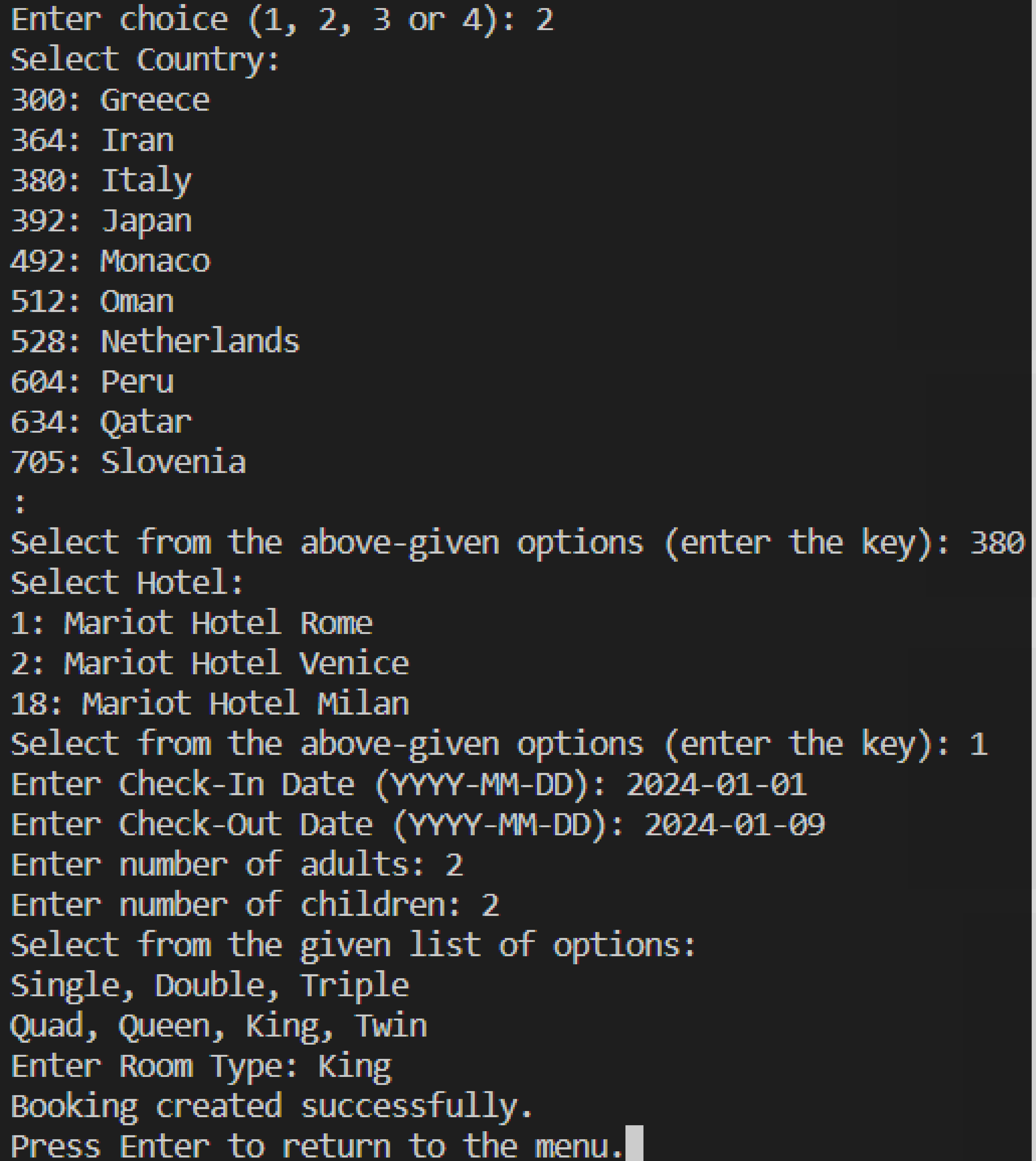
Reviewing Active Bookings:

Function: review_active_bookings.

- Displays all **active** bookings linked to the user's email.
- Formats and presents booking details.

Menu Interaction:

- Uses a loop for continuous interaction.
- Handles user input for selecting menu options.



```
Enter choice (1, 2, 3 or 4): 2
Select Country:
300: Greece
364: Iran
380: Italy
392: Japan
492: Monaco
512: Oman
528: Netherlands
604: Peru
634: Qatar
705: Slovenia
:
Select from the above-given options (enter the key): 380
Select Hotel:
1: Mariot Hotel Rome
2: Mariot Hotel Venice
18: Mariot Hotel Milan
Select from the above-given options (enter the key): 1
Enter Check-In Date (YYYY-MM-DD): 2024-01-01
Enter Check-Out Date (YYYY-MM-DD): 2024-01-09
Enter number of adults: 2
Enter number of children: 2
Select from the given list of options:
Single, Double, Triple
Quad, Queen, King, Twin
Enter Room Type: King
Booking created successfully.
Press Enter to return to the menu.
```

view.py (making a new book)

Booking Process Initiation:

- Function: make_booking.
- Guides user through the booking creation steps.

Selecting Hotel and Room:

- Retrieves and presents country, hotel, and room options.
- Uses select_option for user-friendly selections.

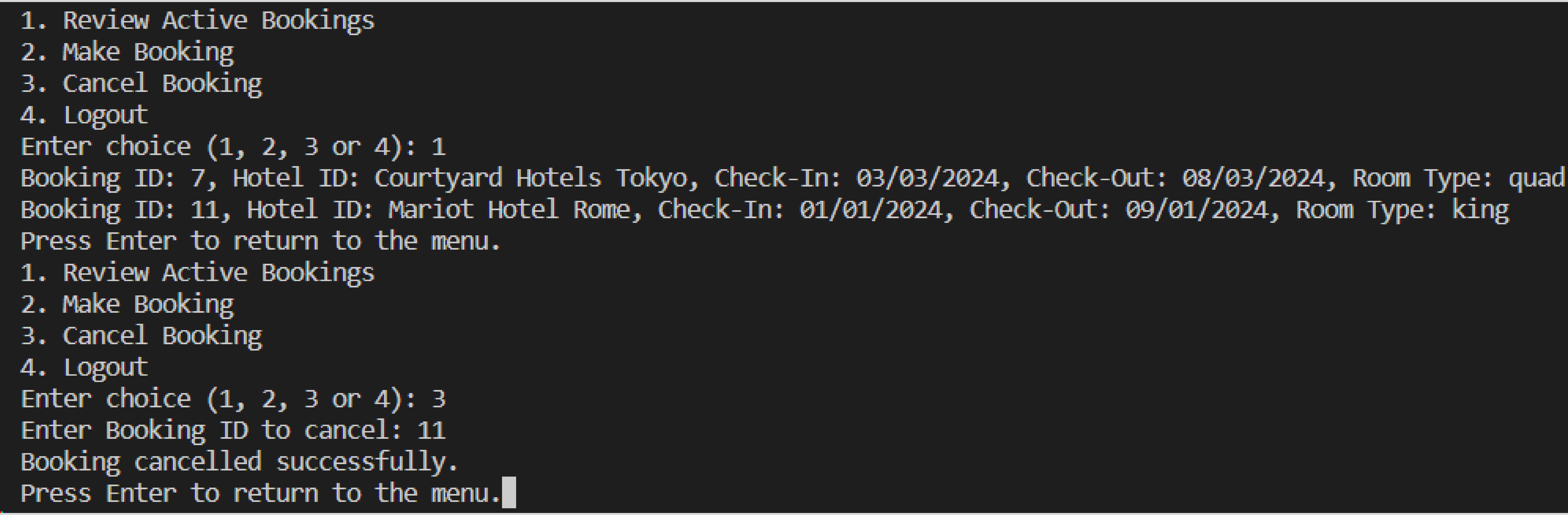
Input Validation:

- Validates check-in and check-out dates, room type, and guest count.
- Leverages functions from validation.py.

Booking Confirmation:

- Finalizes booking details.
- Adds new booking to the system via create_booking from crud.py.

view.py (cancel the booking)



```
1. Review Active Bookings
2. Make Booking
3. Cancel Booking
4. Logout
Enter choice (1, 2, 3 or 4): 1
Booking ID: 7, Hotel ID: Courtyard Hotels Tokyo, Check-In: 03/03/2024, Check-Out: 08/03/2024, Room Type: quad
Booking ID: 11, Hotel ID: Mariot Hotel Rome, Check-In: 01/01/2024, Check-Out: 09/01/2024, Room Type: king
Press Enter to return to the menu.
1. Review Active Bookings
2. Make Booking
3. Cancel Booking
4. Logout
Enter choice (1, 2, 3 or 4): 3
Enter Booking ID to cancel: 11
Booking cancelled successfully.
Press Enter to return to the menu.
```

Booking Identification:

- Users input the booking ID for cancellation.
- System verifies ownership and status of the booking

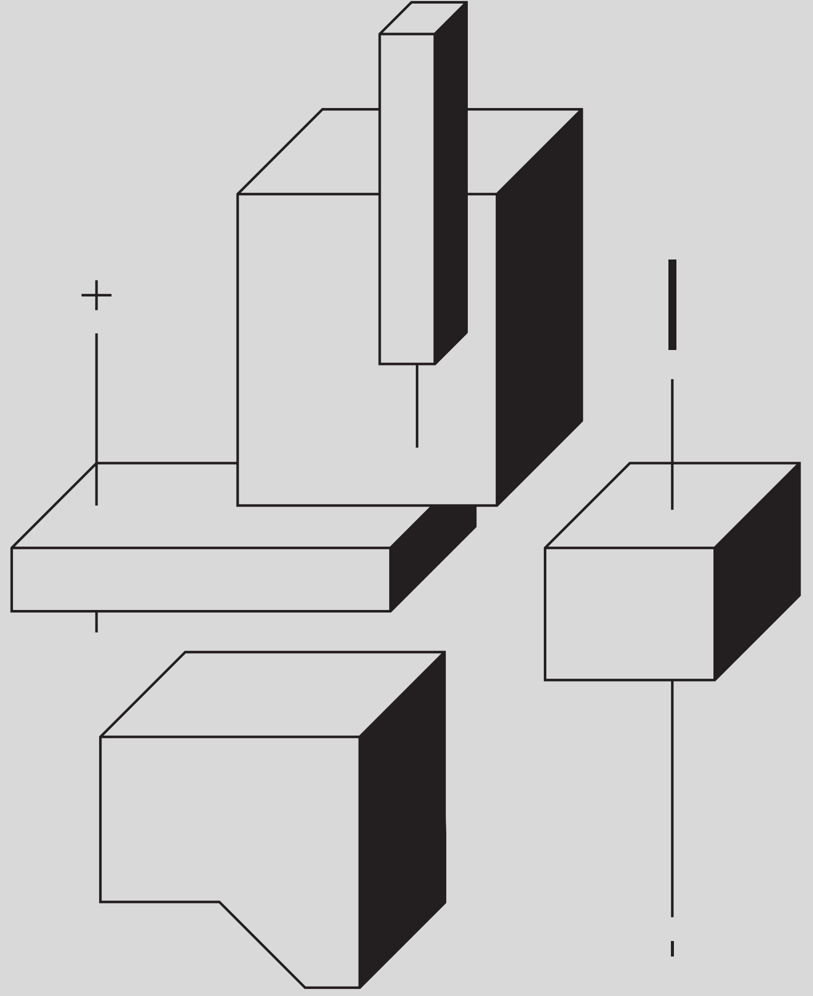
Cancellation Execution:

- Utilizes **cancel_booking_helper** from crud.py.
- Updates the booking's status to 'cancelled'.

Cancellation Execution:

- Confirms the successful cancellation.
- Handles cases of invalid booking IDs or already cancelled bookings..

utils.py(data handling)



Overview of utils.py:

- Central module for handling CSV file operations.
- Key in managing and accessing structured data.

Reading CSV Data:

- Functions: csv_to_dict and user_csv_to_dict.
- Convert CSV files into Python dictionaries for easy data manipulation.
- csv_to_dict: Maps a specified field as the key for each entry.
- user_csv_to_dict: Tailored for user-specific data extraction.

Error Handling:

- Implements checks for key presence and file existence.
- Ensures robustness in data operations.

utils.py (writing data and transforming CSV)

Writing Data to CSV:

- Function: **write_dict_to_csv.**
- Saves and updates data in CSV format.
- Automatically manages data structure and headings.

Transforming CSV Data:

- Function: **transform_csv_to_dict.**
- Converts CSV files into easy-to-use dictionaries.
- Useful for various data access and manipulation tasks.

Practical Use in the System:

- Essential for keeping booking and user data up-to-date.
- Core component for reading and writing data in **crud.py and authentication.py**

The END

Thank You