System Documentation

For

AHS Reservation System

Document Version: 1.0

Date: 06/05/2022

Student ID: S3741327
Name: Juyeon Kim

Table of Contents

1. SYSTEM DESCRIPTION

This system is designed for AHS to streamline the manual room reservation system. Users of this system could be either Staff, Receptionist, or Manager who has been in charge of manual booking records. (Please note that the term "User" refers to a person who will use the system). This service enables users to do CRUD practices of the customer(guest), staff, room type, room, and booking details.

2. DEVELOPMENT TOOLS

Programming language	Python
Suggested IDEs	Visual Studio Code
DB browser	DB browser for SQLite3
GUI	Tkinter

Figure 1. Development Tools table

3. STANDARD TASKS

This part lists all the tasks required for the delivery of the application including manual intervention.

3.1. Initial setup

Step 1	Download a zip folder named "AHS_src_code"
Step 2	Unzip the folder and locate it on "Desktop"
Step 3	Open the command prompt and run the command line as below: `Command> cd Desktop/AHS_src_code/`
Step 4	Check your current location by running the command: `Command> pwd` and be sure you are in `<Root>/<User>/Desktop/AHS_src_code/`

Figure 2. Steps for initial setup

3.2. Database setup

If you are running a program for the very first time, run the following command in the command prompt to create an initial database, then "ahs_reservation.db" file where all the new records will be saved is created inside of the AHS_source_code file.

```
Command> python create_tables.py
```

If you would like to create a new database, be sure to delete an existing "ahs_reservation.db" file and repeat the above practice.

3.3. Run the program

To run the AHS reservation program, run the following code in the command prompt.

```
Command> python main_menu.py
```

4. CRUD PRACTICES

4.1. Main Menu

Completing Step 1 will display Figure 3 main menu where the user can access all entities: Customer, Staff, Room Type, Room, and Booking. Users can quit the program by clicking the Home > Exit button.

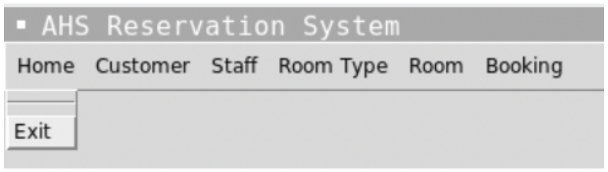


Figure 3. Main menu

4.2. Manipulate entity record

For *Customer & Staff & Room type & Room* table, please read through page 4 Communal Crud practices and rules applied to each field on page 5.

For the *Booking* table, It's of paramount importance to click the "Price" button before creating a booking record as the button will calculate the total price of the booking. e.g., If Room price of Room ID 101: $200/night, Check-in: 01/05/2022, Check out: 03/05/2022, Extra bed: Yes, then the total price will be: (Room price + extra bed price) x stay = (200+80) x 2 = $560. Please read through page 4 Communal Crud practices and rules applied to each field on page 5.

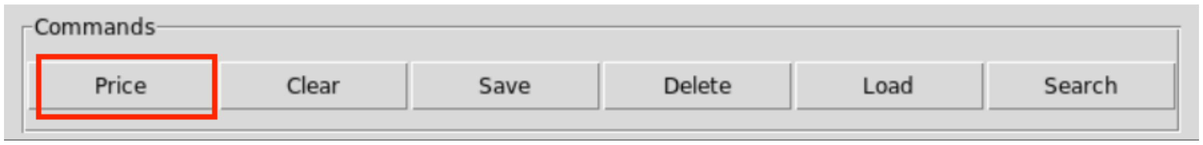


Figure 4. Commands of booking with "Price" button

4.3. Communal CRUD practices

Those CRUD practices apply to all entities in the database.

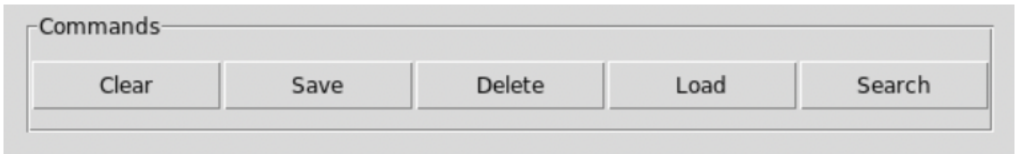


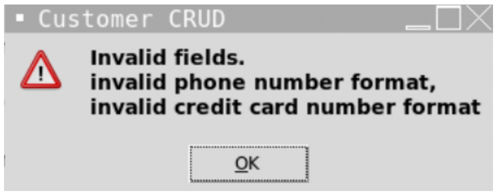
Figure 5. Communal command buttons to all entities

➢ **Clear**

Users can clear a form by clicking the “Clear” button and making all fields empty. Rest assured it won’t manipulate the database.

➢ **Create**

Users can create a new record by clicking the “Save” button within the Commands box. However, different rules are applied to each field by an entity. Breaching the rules will display users an error message. To know in detail, please refers to page 5. E.g.

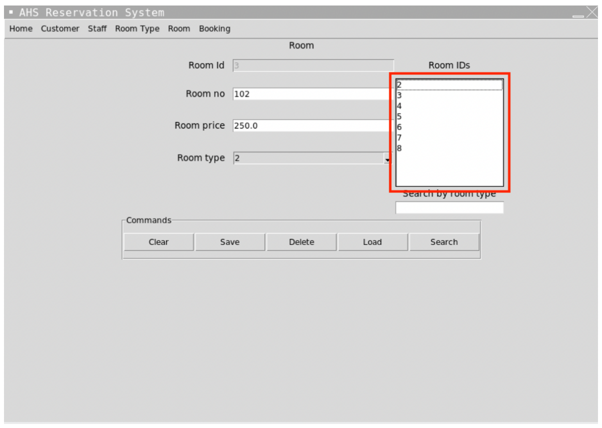


Invalid fields.
invalid phone number format,
invalid credit card number format

Figure 6. Error message misaligning with the rules

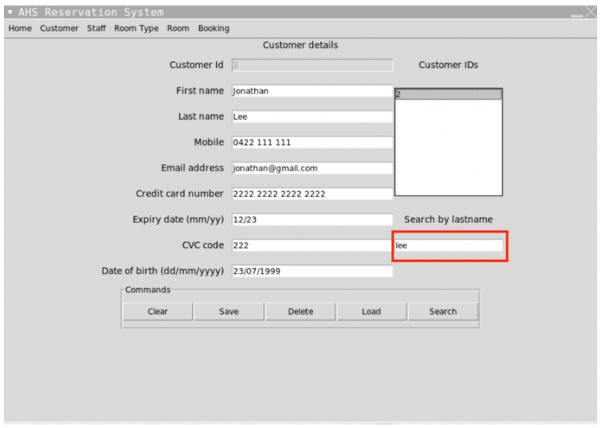
➢ **Read**

Users can view all records by clicking the “Load” button within the Commands box and all the records will be displayed in the list box as in Figure 7. Users can also filter only records with the specific keyword by inputting a keyword in the “Search by __” field and clicking the “Search” button as in Figure 8.



Room Id	3
Room no	102
Room price	250.0
Room type	2

Room IDs: 2, 3, 4, 5, 6, 7, 8



Customer Id	2
First name	Jonathan
Last name	Lee
Mobile	0422 111 111
Email address	jonathan@gmail.com
Credit card number	2222 2222 2222 2222
Expiry date (mm/yy)	12/23
CVC code	222
Date of birth (dd/mm/yyyy)	23/07/1999
Search by lastname	lee

Customer IDs: 2

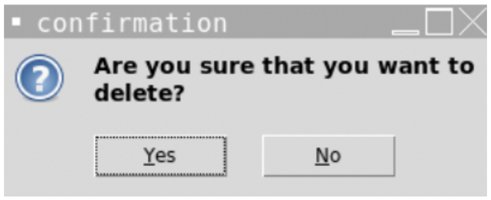
Figure 7. View all records *Figure 8. Filter records*

➢ **Update**

Users can update the customer information by filling out the updated detail and then clicking the “save” button. The system will then show the result message confirming a customer record was successfully updated.

➢ **Delete**

Users can delete a record by clicking the “Delete” button, but then the system will pop the confirmation message to give the user a chance to cancel the operation. To confirm the deletion of a record, click “Yes”.



Are you sure that you want to delete?

Figure 9. Delete confirmation message

4.4. Rules

Each field must follow the specific rules. Unless the system will pop an error message.

- **Customer table**

Field	Rules
First name	Only contains alphabet (No number, special char)
Last name	Only contains alphabet (No number, special char)
Mobile	Show follows an Australian mobile number format: 0000 000 000 or 0000-000-000
Email address	Show be valid email format
Credit card number	Show follows valid credit card number format: 0000 0000 0000 0000 or 0000-0000-0000-0000
Expiry date	Show follows a date format of: mm/yy
CVC code	Only contains number (No alphabet, special char)
Date of birth	Show follows a date format of: dd/mm/yyyy

Figure 10.1. Rules applied to customer fields

- **Staff table**

Field	Rules
First name	Only contains alphabet (No number, special char)
Last name	Only contains alphabet (No number, special char)
Email address	Show be valid email format
Title	Only contains alphabet (No number, special char)

Figure 10.2. Rules applied to staff fields

- **Room Type table**

Field	Rules
Room Type	Only contains alphabet (No number, special char)
King bed counts	Only contains number (No alphabet, special char)
Queen bed counts	Only contains number (No alphabet, special char)
Single bed counts	Only contains number (No alphabet, special char)
Bathroom counts	Only contains number (No alphabet, special char)

Figure 10.3. Rules applied to room type fields

- **Room table**

Field	Rules
Room no	Only contains number (No alphabet, special char)
Room price	Only contains number (No alphabet, special char except "." In$200.5)
Room type	Only contains number (No alphabet, special char)

Figure 10.4. Rules applied to room fields

- **Booking table**

Field	Rules
Customer ID	Only contains number (No alphabet, special char)
Room ID	Only contains number (No alphabet, special char)
Staff ID	Only contains number (No alphabet, special char)
Phone	Show follows an Australian mobile number format: 0000 000 000 or 0000-000-000
Check in	Show follows a date format of: dd/mm/yyyy
Check out	Show follows a date format of: dd/mm/yyyy
Adult counts	Only contains number (No alphabet, special char)
Children counts	Only contains number (No alphabet, special char)
Infant counts	Only contains number (No alphabet, special char)
Total Price	Only contains number (No alphabet, special char except "." In$200.5)

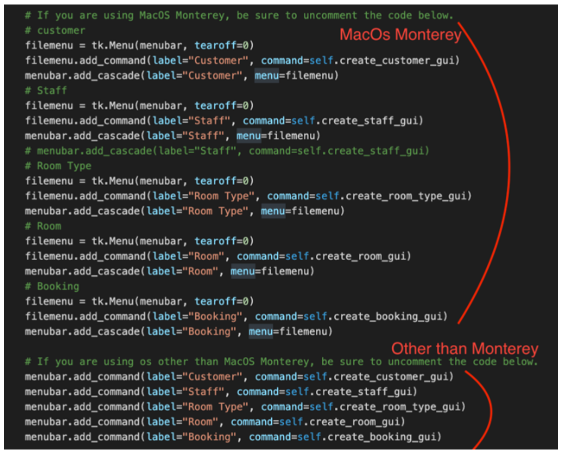
Figure 10.5. Rules applied to booking fields

5. TROUBLESHOOTING GUIDE

5.1. Known issues with the system

a. Accessibility issues

The main menu will not run on macOS Monterey. Be sure to uncomment and comment code in "main_menu.py" by OS you are using.



```
# If you are using MacOS Monterey, be sure to uncomment the code below.
# customer
filemenu = tk.Menu(menubar, tearoff=0)
filemenu.add_command(label="Customer", command=self.create_customer_gui)
menubar.add_cascade(label="Customer", menu=filemenu)
# Staff
filemenu = tk.Menu(menubar, tearoff=0)
filemenu.add_command(label="Staff", command=self.create_staff_gui)
menubar.add_cascade(label="Staff", menu=filemenu)
# menubar.add_cascade(label="Staff", command=self.create_staff_gui)
# Room Type
filemenu = tk.Menu(menubar, tearoff=0)
filemenu.add_command(label="Room Type", command=self.create_room_type_gui)
menubar.add_cascade(label="Room Type", menu=filemenu)
# Room
filemenu = tk.Menu(menubar, tearoff=0)
filemenu.add_command(label="Room", command=self.create_room_gui)
menubar.add_cascade(label="Room", menu=filemenu)
# Booking
filemenu = tk.Menu(menubar, tearoff=0)
filemenu.add_command(label="Booking", command=self.create_booking_gui)
menubar.add_cascade(label="Booking", menu=filemenu)

# If you are using os other than MacOS Monterey, be sure to uncomment the code below.
menubar.add_command(label="Customer", command=self.create_customer_gui)
menubar.add_command(label="Staff", command=self.create_staff_gui)
menubar.add_command(label="Room Type", command=self.create_room_type_gui)
menubar.add_command(label="Room", command=self.create_room_gui)
menubar.add_command(label="Booking", command=self.create_booking_gui)
```

Figure 11. Code to run main menu by different OS

b. Possible errors

An error can occur in the cases below:

- Unique constraint error: If a user creates a record with an existing email or phone number
- Breaching the formatting rules
- Required field is empty

If unknown errors occur for some reason, printing lines on the command line prompt will help to debug practice as it is programmed to print sequence lines.

5.2. Future improvements

a. When deleting a record

When performing a record deletion practice to one and the last record, you might face that an item is still visible in the list box even if the deleting practice was successfully performed.

b. Inputting foreign key

For say, a user wants to create a booking record and to do that, the user must select customer id, room id, and staff id as foreign keys out of the combo box. As the customer id itself does not explicitly represent who the customer is, the user is required to look upon the customer table to find the customer id of a guest who is trying to make a booking, take note of its customer id, and select the id in the field on booking page. To streamline this redundant searching practice, the system will later then implement a search box on all pages where the user can find the user id without needing to move the page back and forth.